



uni 的量化日记

机器学习在量化交易中的应用

Yiming Wang
2025 / 12 / 6~7



微信 uni

机器学习	数据获取	数据分析
自定义轴	数据标签	特征工程
数据降维	模型训练	模型组合
模型推理	实盘交易	扩展阅读

全 生 产 线



实现完全独立自主的进行模型训练与实盘交易

机器学习的基础知识

机器学习模型的类别

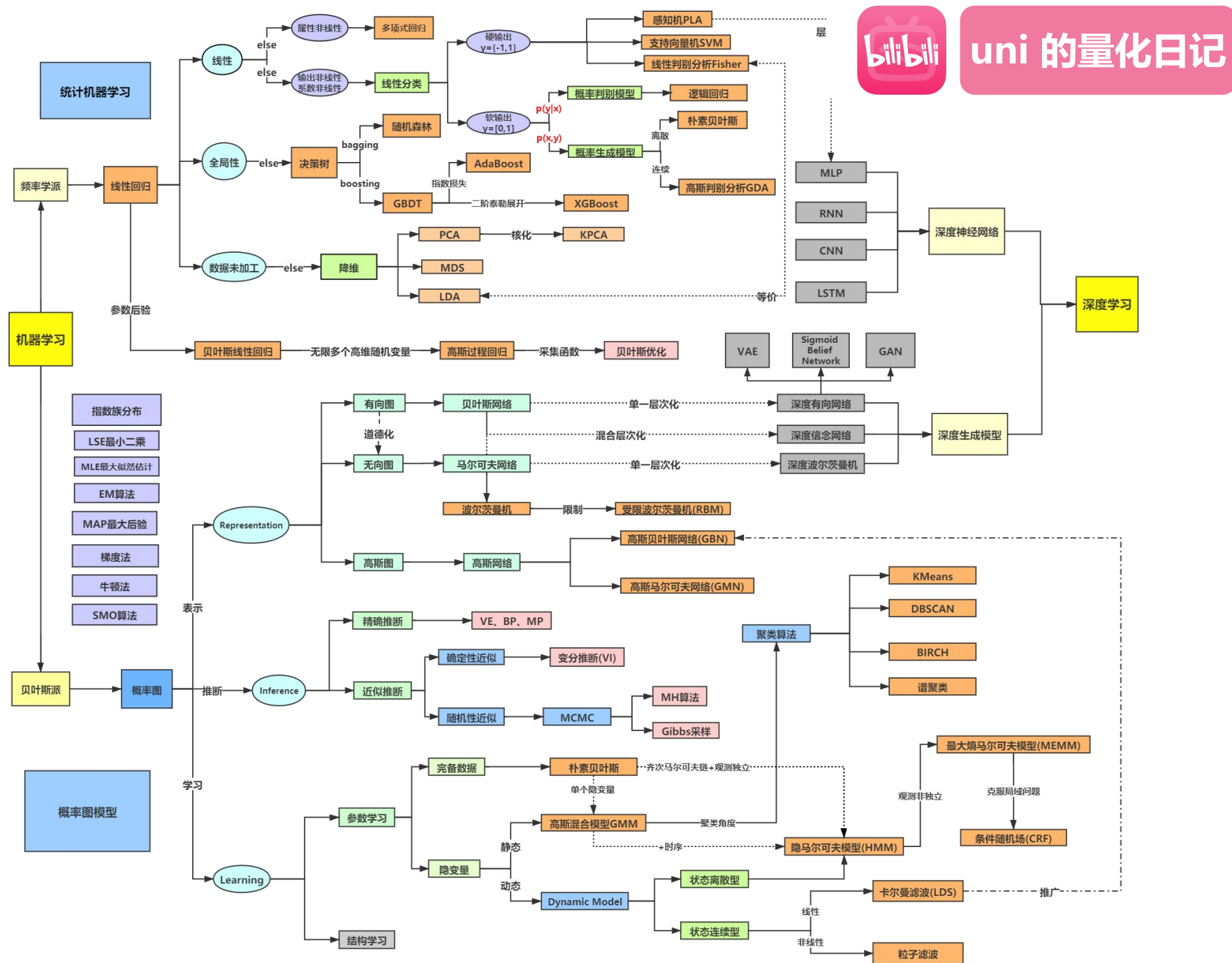
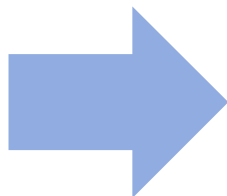
机器学习的应用场景

机器学习在量化交易中的应用

机器学习

判别式模型

生成式模型



任务场景： 分类

回归

降维

聚类

任务目标： 有监督

无监督

半监督

强化学习

重点：不同的应用场景，不同的目标任务，需要做不同的选择与组合

金融数据获取

国内金融数据与海外金融数据的区别

金融数据的差分概率分布与自相关性解析

金融数据时间轴的隐性风险

国内金融数据与海外金融数据的区别



数据：RB指数
数据源：米筐
时间：1m

成交量					成交额			持仓量	
order_book_id	datetime	volume	high	open	trading_date	total_turnover	close	open_interest	low
RB99	2019/1/2 9:01	81956	3391.5891	3378.4159	2019/1/2	2781279160	3381.0018	2840918	3366.3018
RB99	2019/1/2 9:02	93736	3418.7589	3381.8016	2019/1/2	3195969560	3390.1908	2829106	3381.8016
RB99	2019/1/2 9:03	23552	3394.8393	3391.028	2019/1/2	801347580	3393.7451	2830470	3389.9339
RB99	2019/1/2 9:04	18952	3393.8737	3392.8853	2019/1/2	643751680	3385.1353	2830664	3383.432
RB99	2019/1/2 9:05	15840	3389.3211	3384.2978	2019/1/2	539033480	3386.4772	2832178	3384.2887
RB99	2019/1/2 9:06	22302	3386.9179	3385.6663	2019/1/2	755713360	3382.6883	2834610	3380.0005
RB99	2019/1/2 9:07	13822	3383.9572	3382.7145	2019/1/2	469190880	3380.9031	2838468	3380.7417
RB99	2019/1/2 9:08	12168	3386.7293	3381.7576	2019/1/2	413790900	3386.2691	2840892	3380.4303
RB99	2019/1/2 9:09	19082	3393.1925	3386.2418	2019/1/2	650067440	3393.1925	2840014	3385.2727
RB99	2019/1/2 9:10	18814	3395.0443	3393.1865	2019/1/2	641448780	3394.0415	2842704	3391.107
RB99	2019/1/2 9:11	15574	3395.957	3394.8608	2019/1/2	531070440	3394.0373	2842518	3391.0897
RB99	2019/1/2 9:12	12220	3394.9002	3394.8624	2019/1/2	415655520	3392.9521	2843722	3391.1005

数据：BTCUSDT
数据源：Binance
时间：1m

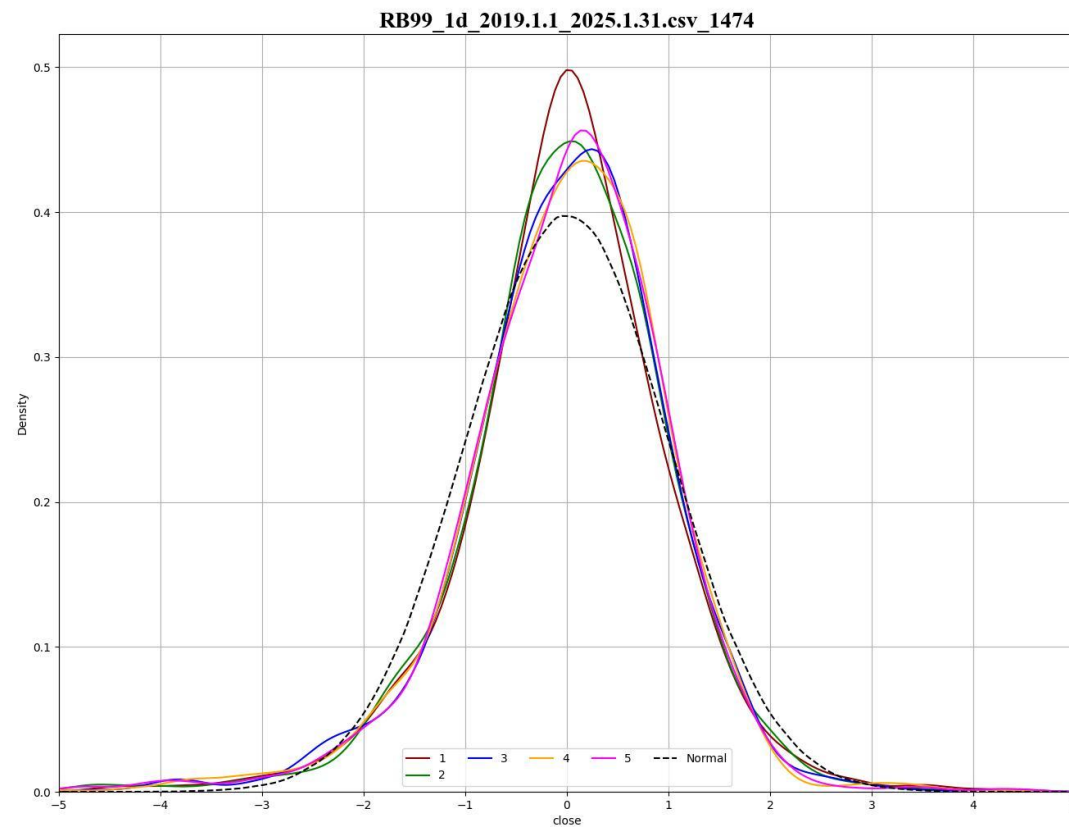
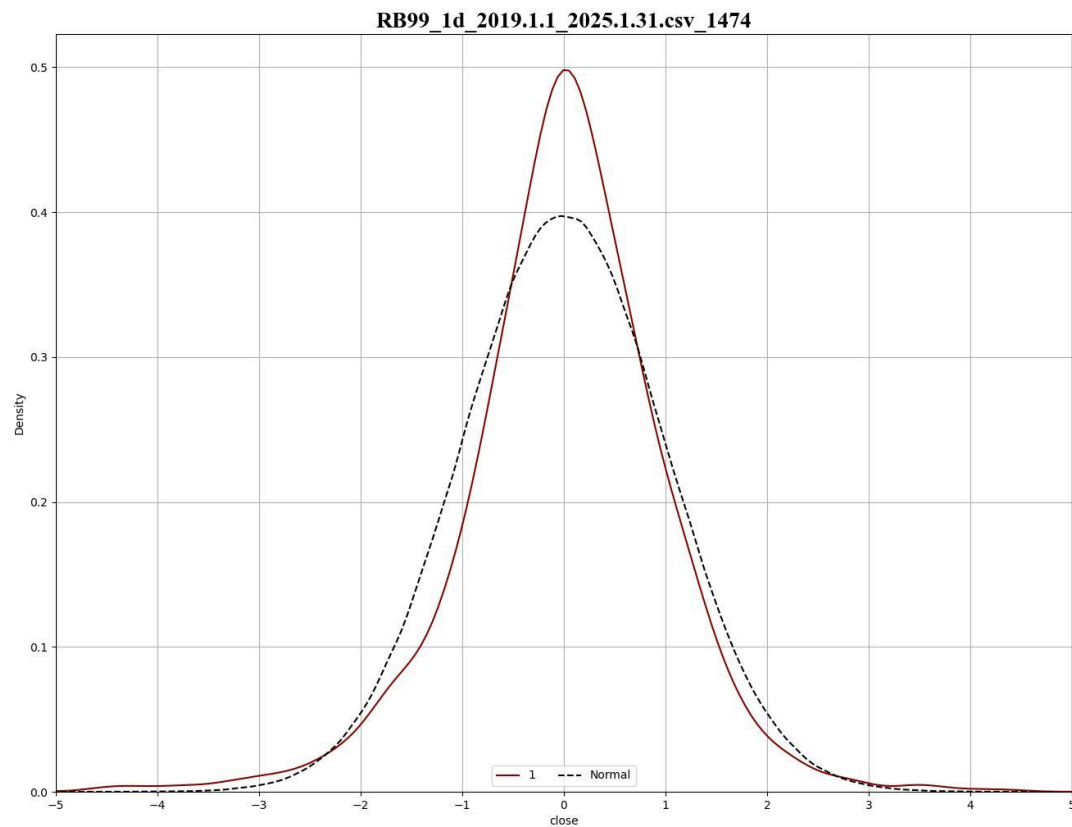
成交量						成交额		交易笔数	主动买入成交量	主动买入成交额
Open time	Open	High	Low	Close	Volume	Close time	Quote asset volume	Number of trades	Taker buy base asset volume	Taker buy quote asset volume
1.57784E+12	7195.24	7196.25	7183.14	7186.68	51.642812	1.57784E+12	371233.5184	493	19.59823	140888.4143
1.57784E+12	7187.67	7188.06	7182.2	7184.03	7.248148	1.57784E+12	52080.12779	135	2.031772	14599.21192
1.57784E+12	7184.41	7184.71	7180.26	7182.43	11.681677	1.57784E+12	83903.74164	202	5.479244	39357.08178
1.57784E+12	7183.83	7188.94	7182.49	7185.94	10.025391	1.57784E+12	72033.22665	136	3.294966	23680.57192
1.57784E+12	7185.54	7185.54	7178.64	7179.78	14.911105	1.57784E+12	107066.5218	161	2.369033	17012.01514
1.57784E+12	7179.76	7182.51	7178.2	7179.99	12.463243	1.57784E+12	89484.36354	143	3.652785	26225.06962
1.57784E+12	7180	7182	7179.99	7182	3.573774	1.57784E+12	25662.34791	72	1.678855	12055.51106
1.57784E+12	7181.7	7183.77	7180.91	7183.66	14.470782	1.57784E+12	103928.8704	147	7.166713	51474.22382
1.57784E+12	7183.9	7187.74	7183.45	7187.68	12.842443	1.57784E+12	92277.91447	134	4.996467	35903.39794
1.57784E+12	7187.68	7191.77	7186.02	7191.07	16.014983	1.57784E+12	115127.764	135	7.271693	52277.41577
1.57784E+12	7193.15	7193.53	7186.25	7187.36	12.60237	1.57784E+12	90624.18672	135	6.353845	45690.58423
1.57784E+12	7187.36	7191.08	7186.82	7188.71	10.263525	1.57784E+12	73783.00919	152	5.343899	38415.3155

重点：金融数据中的各类信息对于整个生产线的每个步骤都至关重要，涉及到轴的定义、标签类别、特征工程

时间轴数据的问题



uni 的量化日记



突出问题:

统计特性差 (增加了模型的收敛难度, 损失函数的值波动较大)

平稳性低 (增加了特征工程的复杂性, 数据漂移问题严重)

非时间轴的优势与重采样的技术体系

复现成交量与成交额重采样自定义轴

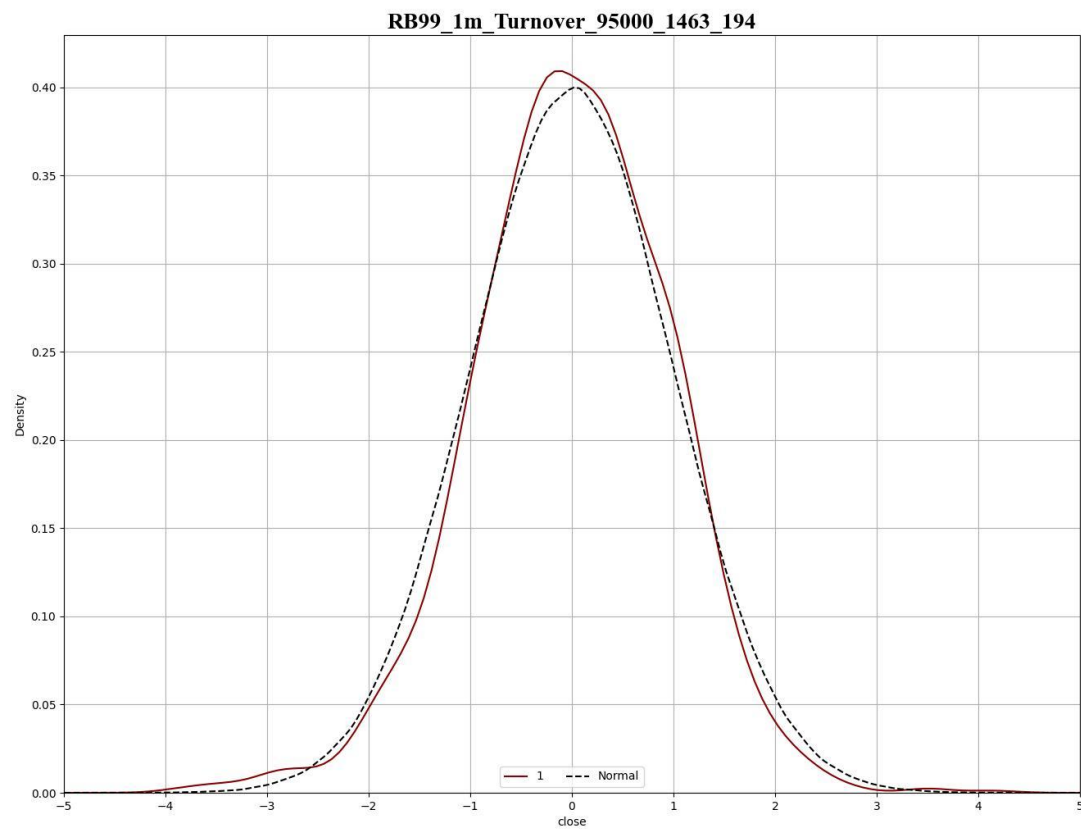
复现价格行为与波动率重采样自定义轴

实现一阶差分接近正态分布的其它自定义轴

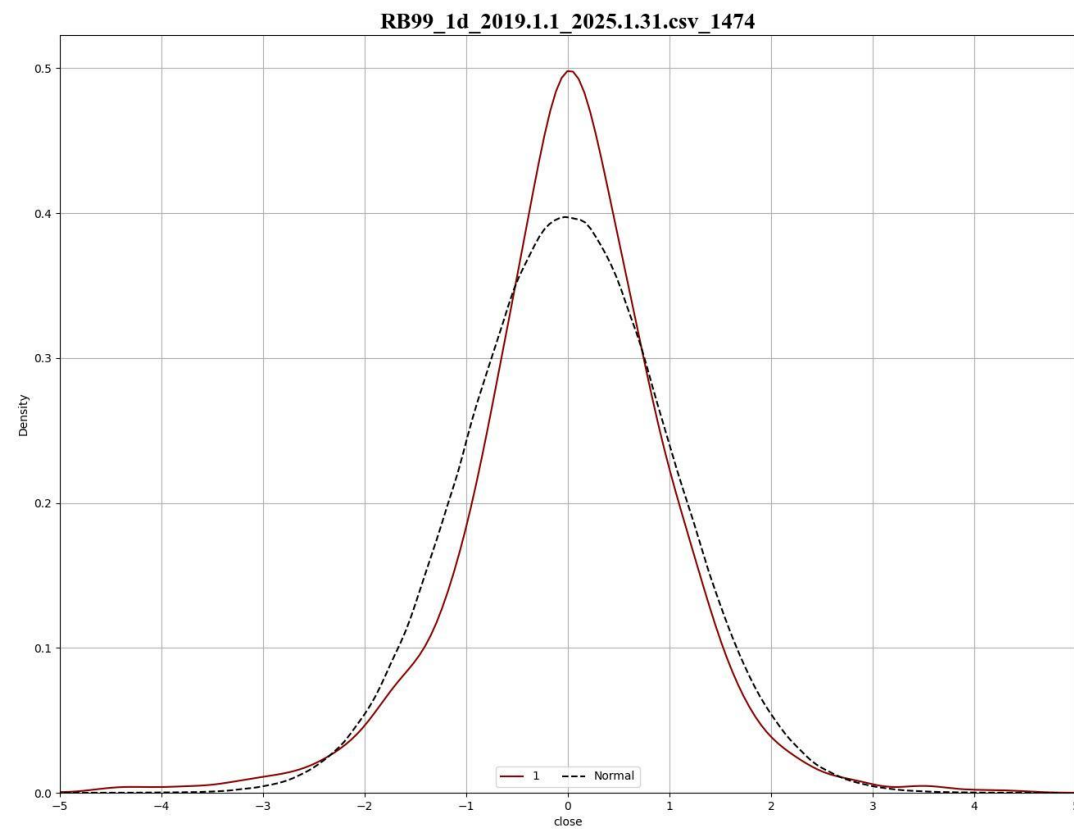
RB 成交额轴的一阶概率分布 (对标 1d 时间轴)



uni 的量化日记



成交额轴



时间轴

ADF平稳性检验



uni 的量化日记

ADF检验结果:

	Period	ADF Statistic	p-value	1% Critical	5% Critical	Stationary
0	13619	-22.796	0	-3.431	-2.862	True
1	13618	-16.8899	0	-3.431	-2.862	True
2	13617	-17.2426	0	-3.431	-2.862	True
3	13616	-16.5102	0	-3.431	-2.862	True
4	13615	-16.2431	0	-3.431	-2.862	True

ADF 检验结果:

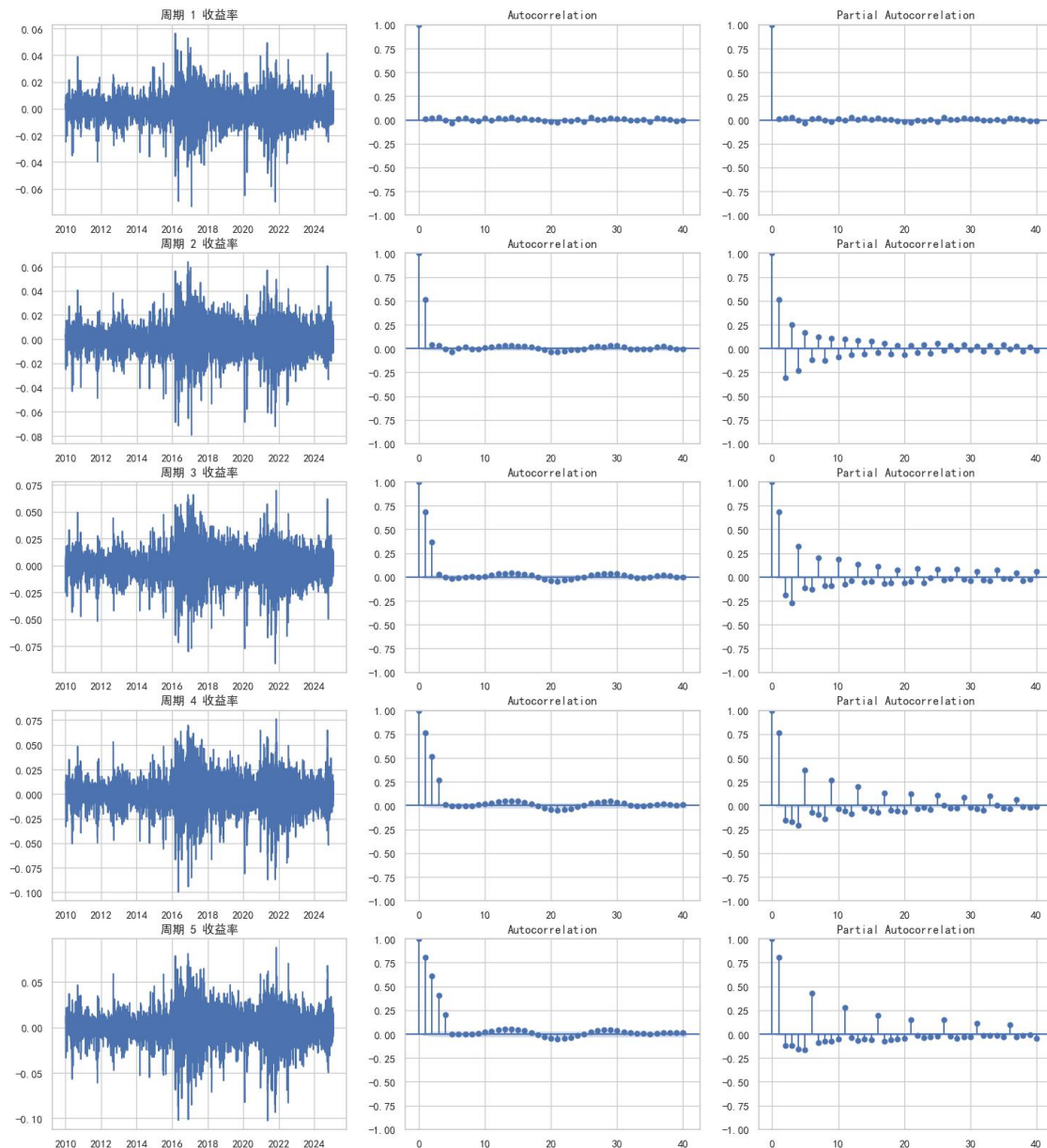
p 值 < 0.05: 拒绝原假设, 数据平稳 (无单位根)

ADF 统计量: 越负越平稳, 通常需小于 1% 临界值

ACF自相关 / PACF偏自相关



uni 的量化日记



ACF/PACF 图分析

ACF (对数收益率) :
反映历史收益对当前收益的综合线性影响

若滞后 1 阶 ACF 显著为正, 说明存在动量效应,
(前 1 Bar 涨, 当下 Bar 涨的概率高, 包含前 1 Bar 直接
影响 + 更早历史的间接传递)

若滞后多阶 ACF 拖尾, 说明收益关联有持续传递性
(如趋势延续多日, 受历史多期收益共同影响)

PACF (对数收益率) :
反映历史收益对当前收益的直接线性依赖

若滞后 1 阶 PACF 显著, 说明 $t-1$ 阶收益对 t 阶收益有
直接影响 (未被中间阶干扰) ;

若滞后 p 阶后 PACF 截尾 (突然降为 0) , 说明仅前
 p 阶收益对当前有直接影响



重点：参考可以使用的金融数据中的各类基础数据与衍生的数据

不同数据的自定义轴与后续的数据标签和特征工程是紧密联系的

用机器学习模型来为数据做趋势性标签

根据趋势行为自定义标签的密度

标签与特征工程的关系

不同种类的标签与模型训练之间的关系

用概率模型来为数据做趋势性标签



uni 的量化日记



使用高斯混合模型（GMM）来对数据进行自动标注，
通过不同参数的设置来获取不同任务的趋势性标签

重点：趋势性标签具有通用性，通过GMM可以获得标签的全局最优解

1. GMM可以为每个隐藏状态定义观测值的概率分布
 2. $n_mix=3$ 表示每个隐藏状态的观测数据由3个独立的高斯分布混合而成，即每个状态下的观测概率密度函数是3个高斯分布加权组合的结果
 3. 由于数据通常是非线性和多模态的，所以单一高斯分布只能描述对称的单峰分布，而3个高斯分量组合可有效拟合双峰、偏态或更复杂的分布形态
- 例如：“温和”（分量1）、“快速”（分量2）、“回调”（分量3）

整数阶数据特征与自定义窗口期的应用

非时间轴因子作为特征的方法

金融思维与数学思维的特征工程各自的优势与区别

特征工程与自定义轴之间的关系

熵在特征工程中的实现与应用

因子的使用

主观：时序、截面、混合
量价、基本面、另类
高频、中频、低频
Alpha、风险、流动性

手动：GPlearn, Deap, KAN, DL, HRL

自动：Featuretools, TSFresh,
TSFEL、OpenFE

非时间轴因子作为特征

可将时间轴上的因子迁移到非时间轴上

时间轴因子作为特征

时间戳对齐

其它非时间轴因子作为特征

时间戳对齐

重点：由于自定义轴的统计特性与数据的平稳性更优，所以减少了后期模型的收敛难度，同时也降低了特征工程的复杂性

高维视角下的特征工程的分类框架



uni 的量化日记

统计特征

描述性统计（熵、均值、方差、偏度/峰度）
时间序列统计（ACF、波动率聚类、协整性）
极值统计（VaR、ES）

领域知识特征

经济理论驱动（Fama-French因子、CAPM）
市场实务驱动（隐含波动率曲面、变量分箱）

关系型特征

横截面关系（跨资产相关性）
网络结构特征（网络中心性、Granger因果）
图嵌入特征（Node2Vec、GNN输出）

数学变换特征

线性变换（标准化、降维）
非线性变换（Box-Cox、分位数变换）
信号处理（小波变换、傅里叶分析）

时序与事件特征

滞后/滚动特征
季节性分解
事件窗口标记（突发事件、政策发布）

自动化与非结构化特征

自动特征合成（机器学习、符号回归、TSFresh）
文本/图像特征（情感分析、卷积特征）
高频微观结构特征（订单簿不平衡、VPIN）

整数阶数据特征



uni 的量化日记

80个特征 X 5个窗口期

400个特征

```
logopen1_1 = (np.log(np.array(open1[1:]))-np.log(np.array(open1[:-1])))[4:]
logopen1_2 = (np.log(np.array(open1[1:]))-np.log(np.array(close[:-1])))[4:]
logopen1_3 = (np.log(np.array(open1[1:]))-np.log(np.array(high[:-1])))[4:]
logopen1_4 = (np.log(np.array(open1[1:]))-np.log(np.array(low[:-1])))[4:]
```

```
logopen2_1 = np.log(np.array(open1[2:]))-np.log(np.array(open1[:-2]))
logopen2_2 = np.log(np.array(open1[2:]))-np.log(np.array(close[:-2]))
logopen2_3 = np.log(np.array(open1[2:]))-np.log(np.array(high[:-2]))
logopen2_4 = np.log(np.array(open1[2:]))-np.log(np.array(low[:-2]))
```

```
logopen3_1 = np.log(np.array(open1[3:]))-np.log(np.array(open1[:-3]))
logopen3_2 = np.log(np.array(open1[3:]))-np.log(np.array(close[:-3]))
logopen3_3 = np.log(np.array(open1[3:]))-np.log(np.array(high[:-3]))
logopen3_4 = np.log(np.array(open1[3:]))-np.log(np.array(low[:-3]))
```

```
logopen4_1 = np.log(np.array(open1[4:]))-np.log(np.array(open1[:-4]))
logopen4_2 = np.log(np.array(open1[4:]))-np.log(np.array(close[:-4]))
logopen4_3 = np.log(np.array(open1[4:]))-np.log(np.array(high[:-4]))
logopen4_4 = np.log(np.array(open1[4:]))-np.log(np.array(low[:-4]))
```

```
logopen5_1 = np.log(np.array(open1[5:]))-np.log(np.array(open1[:-5]))
logopen5_2 = np.log(np.array(open1[5:]))-np.log(np.array(close[:-5]))
logopen5_3 = np.log(np.array(open1[5:]))-np.log(np.array(high[:-5]))
logopen5_4 = np.log(np.array(open1[5:]))-np.log(np.array(low[:-5]))
```

```
loglow1_1 = (np.log(np.array(low[1:]))-np.log(np.array(open1[:-1])))[4:]
loglow1_2 = (np.log(np.array(low[1:]))-np.log(np.array(close[:-1])))[4:]
loglow1_3 = (np.log(np.array(low[1:]))-np.log(np.array(high[:-1])))[4:]
loglow1_4 = (np.log(np.array(low[1:]))-np.log(np.array(low[:-1])))[4:]
```

```
loglow2_1 = np.log(np.array(low[2:]))-np.log(np.array(open1[:-2]))
loglow2_2 = np.log(np.array(low[2:]))-np.log(np.array(close[:-2]))
loglow2_3 = np.log(np.array(low[2:]))-np.log(np.array(high[:-2]))
loglow2_4 = np.log(np.array(low[2:]))-np.log(np.array(low[:-2]))
```

```
loglow3_1 = np.log(np.array(low[3:]))-np.log(np.array(open1[:-3]))
loglow3_2 = np.log(np.array(low[3:]))-np.log(np.array(close[:-3]))
loglow3_3 = np.log(np.array(low[3:]))-np.log(np.array(high[:-3]))
loglow3_4 = np.log(np.array(low[3:]))-np.log(np.array(low[:-3]))
```

```
loglow4_1 = np.log(np.array(low[4:]))-np.log(np.array(open1[:-4]))
loglow4_2 = np.log(np.array(low[4:]))-np.log(np.array(close[:-4]))
loglow4_3 = np.log(np.array(low[4:]))-np.log(np.array(high[:-4]))
loglow4_4 = np.log(np.array(low[4:]))-np.log(np.array(low[:-4]))
```

```
loglow5_1 = np.log(np.array(low[5:]))-np.log(np.array(open1[:-5]))
loglow5_2 = np.log(np.array(low[5:]))-np.log(np.array(close[:-5]))
loglow5_3 = np.log(np.array(low[5:]))-np.log(np.array(high[:-5]))
loglow5_4 = np.log(np.array(low[5:]))-np.log(np.array(low[:-5]))
```

```
logclose1_1 = (np.log(np.array(close[1:]))-np.log(np.array(open1[:-1])))[4:]
logclose1_2 = (np.log(np.array(close[1:]))-np.log(np.array(close[:-1])))[4:]
logclose1_3 = (np.log(np.array(close[1:]))-np.log(np.array(high[:-1])))[4:]
logclose1_4 = (np.log(np.array(close[1:]))-np.log(np.array(low[:-1])))[4:]
```

```
logclose2_1 = np.log(np.array(close[2:]))-np.log(np.array(open1[:-2]))
logclose2_2 = np.log(np.array(close[2:]))-np.log(np.array(close[:-2]))
logclose2_3 = np.log(np.array(close[2:]))-np.log(np.array(high[:-2]))
logclose2_4 = np.log(np.array(close[2:]))-np.log(np.array(low[:-2]))
```

```
logclose3_1 = np.log(np.array(close[3:]))-np.log(np.array(open1[:-3]))
logclose3_2 = np.log(np.array(close[3:]))-np.log(np.array(close[:-3]))
logclose3_3 = np.log(np.array(close[3:]))-np.log(np.array(high[:-3]))
logclose3_4 = np.log(np.array(close[3:]))-np.log(np.array(low[:-3]))
```

```
logclose4_1 = np.log(np.array(close[4:]))-np.log(np.array(open1[:-4]))
logclose4_2 = np.log(np.array(close[4:]))-np.log(np.array(close[:-4]))
logclose4_3 = np.log(np.array(close[4:]))-np.log(np.array(high[:-4]))
logclose4_4 = np.log(np.array(close[4:]))-np.log(np.array(low[:-4]))
```

```
logclose5_1 = np.log(np.array(close[5:]))-np.log(np.array(open1[:-5]))
logclose5_2 = np.log(np.array(close[5:]))-np.log(np.array(close[:-5]))
logclose5_3 = np.log(np.array(close[5:]))-np.log(np.array(high[:-5]))
logclose5_4 = np.log(np.array(close[5:]))-np.log(np.array(low[:-5]))
```

```
loghigh1_1 = (np.log(np.array(high[1:]))-np.log(np.array(open1[:-1])))[4:]
loghigh1_2 = (np.log(np.array(high[1:]))-np.log(np.array(close[:-1])))[4:]
loghigh1_3 = (np.log(np.array(high[1:]))-np.log(np.array(high[:-1])))[4:]
loghigh1_4 = (np.log(np.array(high[1:]))-np.log(np.array(low[:-1])))[4:]
```

```
loghigh2_1 = np.log(np.array(high[2:]))-np.log(np.array(open1[:-2]))
loghigh2_2 = np.log(np.array(high[2:]))-np.log(np.array(close[:-2]))
loghigh2_3 = np.log(np.array(high[2:]))-np.log(np.array(high[:-2]))
loghigh2_4 = np.log(np.array(high[2:]))-np.log(np.array(low[:-2]))
```

```
loghigh3_1 = np.log(np.array(high[3:]))-np.log(np.array(open1[:-3]))
loghigh3_2 = np.log(np.array(high[3:]))-np.log(np.array(close[:-3]))
loghigh3_3 = np.log(np.array(high[3:]))-np.log(np.array(high[:-3]))
loghigh3_4 = np.log(np.array(high[3:]))-np.log(np.array(low[:-3]))
```

```
loghigh4_1 = np.log(np.array(high[4:]))-np.log(np.array(open1[:-4]))
loghigh4_2 = np.log(np.array(high[4:]))-np.log(np.array(close[:-4]))
loghigh4_3 = np.log(np.array(high[4:]))-np.log(np.array(high[:-4]))
loghigh4_4 = np.log(np.array(high[4:]))-np.log(np.array(low[:-4]))
```

```
loghigh5_1 = np.log(np.array(high[5:]))-np.log(np.array(open1[:-5]))
loghigh5_2 = np.log(np.array(high[5:]))-np.log(np.array(close[:-5]))
loghigh5_3 = np.log(np.array(high[5:]))-np.log(np.array(high[:-5]))
loghigh5_4 = np.log(np.array(high[5:]))-np.log(np.array(low[:-5]))
```

对比	样本熵	信息熵
理论基础	无分布假设 直接基于数据相似性计算	依赖概率分布 需数据满足统计假设
数据适应性	适合连续型时间序列 如：价格序列、高频交易数据	适合离散型、分类数据 如：分类标签、交易状态
动态性支持	窗口滚动计算 适应动态市场	静态概率 难以捕捉实时变化
抗噪声能力	容差阈值可过滤噪声 抗干扰性强	对异常值敏感 需数据清洗

计算样本熵用标准差与极差的对比



对比	极差	标准差
理论基础	数据的最大值和最小值之间的差值	衡量数据离散程度的统计量
容差阈值解释性	极差作为容差阈值，反映了数据点的范围变化，样本熵在这种情况下衡量了数据点在指定范围内的变化复杂性	标准差作为容差阈值，反映了数据点相对于均值的离散程度，样本熵在这种情况下衡量了数据点在均值附近的变化复杂性
稳定性	极差对异常值的敏感性较低 计算结果相对稳定	标准差对异常值较为敏感 可能导致样本熵计算结果不稳定
适用场景	适用于数据分布范围较大或标准差过小的情况，能够更好地捕捉数据的变化	适用于数据分布较为均匀且具有一定的离散性的情况

数据降维的在金融场景下的应用优势

实现线性与非线性自编码器数据降维的各类方法

线性数据降维带来的模型实盘推理过程中的问题

如何解决线性数据降维对模型实盘推理时的影响

提高模型效率：通过减少数据的维度，可以降低模型训练过程的复杂度，加快模型的训练速度，提高模型的运行效率。

去除噪声和异常值：高维数据中往往存在一些无关的、冗余的或具有噪声的特征，这些特征可能会干扰模型的学习过程，降低模型的性能。

减少过拟合风险：高维数据容易导致模型过拟合，即模型过于复杂，对训练数据拟合得过于紧密，而在测试数据上表现不佳。降维可以减少模型的复杂度，降低过拟合的风险，提高模型的泛化能力，使其在新的数据上具有更好的预测性能。

增强模型的可解释性：降维可以将数据从高维空间映射到低维空间，使数据的结构和特征更加清晰和直观，便于模型的解释和理解。

提升模型的鲁棒性：降维后的数据更加稳定，对数据中的小波动和变化不那么敏感，从而提高了模型的鲁棒性。通过降维，可以降低模型对数据变化的敏感度，使其在不同的数据环境下都能保持相对稳定的性能。

重点：可以将大量的特征通过降维，进而得到Meta特征（元特征）

uni 的量化日记

模型推理时的降维计算

重点：降维过程是对整体训练集进行的降维，通过计算协方差矩阵的特征值和特征向量，找到数据的主要变化方向（主成分），然后将数据投影到这些主成分上，而推理过程是对单一行的特征进行降维，所以这个过程与最初的计算过程是完全不同的

如何解决数据降维对模型实盘推理时的影响



uni 的量化日记

训练集特征

推理特征

训练集特征

重点：将推理特征融入到训练集特征中，确保协方差矩阵的特征值和特征向量的计算结果与原训练集一致

线性降维方法与非线性降维方法的对比



对比维度	线性降维（如 PCA、LDA）	非线性降维（如自编码器、t-SNE、LLE、核 PCA）
核心原理	通过线性变换（如矩阵乘法）将高维数据投影到低维空间，保持数据的线性结构（如方差最大化、类别可分性）	通过非线性变换（如神经网络、流形学习、核函数）捕捉数据中的非线性关系，适用于数据在低维空间呈非线性流形分布的场景
适用数据结构	数据在低维空间呈线性分布（如全局线性结构、超平面、线性子空间）	数据在低维空间呈非线性分布（如曲线、曲面、流形结构），或存在复杂的局部相关性（如金融时间序列的非线性依赖、图像的局部特征）
降维方式	全局线性投影，仅考虑数据的全局方差或类别边界	局部或全局非线性映射，可能通过局部邻域关系（如 LLE）、核函数（核 PCA）或神经网络（自编码器）学习非线性映射关系
优点	<div>1. 计算高效，复杂度低（如 PCA 时间复杂度为 $O(n \cdot d^2)$，n 为样本数，d 为特征数）</div> <div>2. 可解释性强（如主成分的权重直观反映特征重要性）</div> <div>3. 无参数或参数少，易于实现</div>	<div>1. 能捕捉数据的非线性结构和复杂相关性</div> <div>2. 对非线性流形数据的降维效果更好（如保持局部几何结构）</div> <div>3. 灵活性高（如自编码器可自定义网络结构适应数据特点）</div>
缺点	<div>1. 无法处理非线性关系，可能丢失重要的非线性信息</div> <div>2. 对非高斯分布或复杂流形数据降维效果差</div>	<div>1. 计算复杂度高（如 t-SNE 时间复杂度为 $O(n^2)$，自编码器需迭代训练）</div> <div>2. 可解释性差（如神经网络的中间层难以解释）</div> <div>3. 依赖参数设置（如自编码器的网络深度、t-SNE 的困惑度），易过拟合或欠拟合</div>
典型算法	PCA（主成分分析）、LDA（线性判别分析）、ICA（独立成分分析）	自编码器（Autoencoder）、t-SNE、局部线性嵌入（LLE）、核 PCA（Kernel PCA）、等距映射（Isomap）
应用场景	数据呈线性分布、需要快速降维、对可解释性要求高（如金融数据的初步特征筛选、传统机器学习模型的预处理）	数据呈非线性分布、需要保留复杂结构（如图像 / 视频特征降维、金融时间序列的非线性模式提取、高维生物数据的流形学习）

PCA线性降维方法与自编码器非线性降维方法的对比



对比维度	PCA（线性降维）	自编码器（非线性降维）
降维性质	线性降维，通过正交变换将数据投影到方差最大的线性子空间	非线性降维，通过神经网络学习非线性映射，将高维数据编码到低维隐空间，再解码还原
核心原理	最大化数据在低维空间的方差，最小化重构误差（基于线性代数和特征值分解）	通过编码器（Encoder）将输入映射到低维隐向量，解码器（Decoder）还原输入，通过重构误差（如 MSE）训练网络，学习非线性压缩表示
监督类型	无监督学习，仅利用数据本身的分布信息	通常为无监督学习（仅用输入数据训练），也可引入监督信号（如半监督自编码器，结合标签信息优化隐空间）
灵活性	固定线性变换，无法自定义降维过程，仅通过选择主成分数量控制降维维度	可自定义网络结构（如层数、神经元数量、激活函数），灵活调整非线性映射的复杂度，适应不同数据特征（如增加层数捕捉深层非线性关系）
重构能力	线性重构，重构误差为线性投影后的信息损失（仅保留线性相关信息）	非线性重构，理论上可逼近任意非线性函数（基于 Universal Approximation Theorem），能更准确地重构非线性数据结构
可解释性	高可解释性，主成分是原始特征的线性组合，权重反映特征重要性，可通过载荷矩阵分析降维后的特征含义	低可解释性，隐空间特征是神经网络内部的抽象表示，难以直接对应到原始特征的物理意义（需通过可视化或下游任务间接验证效果）

4种自编码器方法对比



类型	核心优势	适用数据	关键调参点
标准自编码器	简单灵活，基础非线性降维	高维非结构化数据	层数、神经元数、激活函数
变分自编码器	生成新样本，隐空间概率分布	需数据增强或生成的场景	KL 散度权重、先验分布类型
循环自编码器	时序依赖建模	时间序列数据（如股价序列）	RNN 类型（LSTM/GRU）、序列长度
对抗自编码器	隐空间分布可控，生成质量高	需特征解耦或生成的场景	判别器结构、对抗损失权重

4种自编码器模型训练与重构MSE



uni 的量化日记

文件路径: RB99_1m_Train_10877.csv

降维维度: 30

训练轮数: 50

批次大小: 64

☒ 标准自编码器

☒ 变分自编码器

☒ 循环自编码器

☒ 对抗自编码器

▶ 运行降维

读取数据: RB99_1m_Train_10877.csv

数据形状: (10674, 401)

标准化数据...

训练标准自编码器...

Standard AE Epoch [10/50], Loss: 0.0309

Standard AE Epoch [20/50], Loss: 0.0168

Standard AE Epoch [30/50], Loss: 0.0110

Standard AE Epoch [40/50], Loss: 0.0091

Standard AE Epoch [50/50], Loss: 0.0084

标准自编码器重构MSE: 0.008227

训练变分自编码器...

VAE Epoch [10/50], Loss: -10469.2544

VAE Epoch [20/50], Loss: -11500.0669

VAE Epoch [30/50], Loss: -11958.9568

VAE Epoch [40/50], Loss: -12233.3978

VAE Epoch [50/50], Loss: -12433.1920

变分自编码器重构MSE: 0.646083

训练循环自编码器...

Recurrent AE Epoch [10/50], Loss: 0.6627

Recurrent AE Epoch [20/50], Loss: 0.6383

Recurrent AE Epoch [30/50], Loss: 0.6300

Recurrent AE Epoch [40/50], Loss: 0.6237

Recurrent AE Epoch [50/50], Loss: 0.6191

循环自编码器重构MSE: 0.619216

训练对抗自编码器...

Adversarial AE Epoch [10/50], AE Loss: 0.9796, D Loss: 1.4179

Adversarial AE Epoch [20/50], AE Loss: 0.9953, D Loss: 1.3926

Adversarial AE Epoch [30/50], AE Loss: 1.4343, D Loss: 1.3819

Adversarial AE Epoch [40/50], AE Loss: 1.5540, D Loss: 1.4922

Adversarial AE Epoch [50/50], AE Loss: 1.5959, D Loss: 1.3823

对抗自编码器重构MSE: 0.596500

生成评估报告...

重构评估报告已保存。

所有降维方法已完成!

结果已保存至: autoencoder_results

全部完成! 耗时: 508.03秒

重构误差对比:

	方法	重构MSE
0	standard_ae	0.008227
3	adversarial_ae	0.596500
2	recurrent_ae	0.619216
1	vae	0.646102

传统模型与神经网络模型各自的优势与区别

不同模型训练结果的对比与分析

不同模型训练后的回测结果的对比与分析

如何防止过拟合

传统模型与神经网络在模型训练中的区别



uni 的量化日记

xgboost

lightgbm

catboost



CategoryEmbedding

GATE

FT Transformer

TabTransformer

AutoInt

TabNet



ARIMA

SARIMA

GARCH

TBATS

Prophet

LSTM



重点：金融数据为结构化数据，因此可以将金融的时间序列数据通过非时间序列模型进行训练，选择使用对结构化数据更有优势的树模型

特别说明：建议用Inter的13900或14900的CPU进行模型训练

模型回测结果与训练结果的对比



Best 1.csv
回测结果对比

All_no	max_All	CalmarY_no	max_CalmarY	Downarea_no	min_Downarea	WP_no	max_WP	RRR_no	max_RRR	Sharpe_no	max_Sharpe	Sortino_no	max_Sortino
6	70.31	338	3.39	338	-29.00637455	338	54.25	6	55.68	6	9.4318	6	16.8623
8	63.99	6	3.31	62	-29.18909835	337	54.16	8	55.17	8	8.5121	8	15.1128
337	59.39	337	3.28	8	-29.3480363	238	54.16	337	54.8	338	7.7337	338	13.7509
338	58.76	8	2.89	337	-29.4491295	6	54.08	338	54.75	337	7.6265	242	13.5266
242	55.99	242	2.73	99	-30.39413873	257	54.08	242	54.52	242	7.4741	337	12.9361
50	54.18	272	2.65	80	-30.40288645	266	54	50	54.38	50	7.071	43	12.1056
238	53.73	202	2.53	89	-30.96056997	308	53.92	238	54.34	43	6.9686	50	11.9985
43	51.91	62	2.53	85	-32.47550688	89	53.92	43	54.19	238	6.9547	238	11.8162
35	50.46	299	2.48	356	-32.62331577	88	53.92	35	54.08	299	6.6548	35	11.624
272	50.01	43	2.43	119	-32.98730735	366	53.83	272	54.04	35	6.6127	299	11.3989
54	49.92	238	2.41	63	-33.01557118	54	53.83	54	54.03	80	6.5123	80	11.2787
299	49.71	80	2.41	88	-33.62025602	43	53.83	299	54.02	54	6.5091	272	11.1475
80	48.89	54	2.38	107	-33.78358395	340	53.75	80	53.95	272	6.4719	54	11.1148
266	48.71	356	2.32	202	-34.3353852	157	53.75	266	53.94	308	6.4625	11	11.103

Best 2.csv
训练结果对比

Prec_no	max_Prec	Recall_no	max_Recall	F1_no	max_F1
173	0.7903	192	0.8148	173	0.8015
339	0.7901	352	0.8143	339	0.8014
313	0.7893	324	0.8134	313	0.8008
291	0.7886	339	0.8134	374	0.7995
185	0.7882	173	0.8131	192	0.7988
119	0.7881	313	0.8128	291	0.7983
374	0.787	200	0.8128	267	0.7982
375	0.7864	374	0.8126	202	0.7981
253	0.7863	373	0.8126	200	0.798
202	0.7861	362	0.8121	185	0.7979
387	0.7861	267	0.8121	195	0.7979
205	0.7861	195	0.8118	352	0.7978
226	0.7858	366	0.8116	373	0.7973
143	0.7857	320	0.8111	324	0.7971

机器学习中的数据驱动型模型

机器学习中的逻辑驱动型模型

股票的指数化与ETF互相关特征与模型训练

商品期货的指数化互相关特征与模型训练

金融思维下的模型组合方式

数学思维下的模型组合方式

不同模型组合方式的优势与潜在风险

不同自定义轴模型组合的方法

Label	Re 25m	MD	Calmar	DA
5Diff	58.44%	9.67%	2.9	34.04
6Diff	70.31%	10.21%	3.31	34.45
7Diff	62.3%	9.16%	3.27	24.81
8Diff	47.82%	7.18%	3.19	29.92
9Diff	65.3%	11.12%	2.82	30.22
Fusion	67.81%	5.78%	5.63	23.46

Line: 31unit K-Fold: 10 Rolling: 5 DR: PCA
Train: 2010.1.1 ~ 2022.12.31
Backtesting: 2023.1.1 ~ 2025.1.31
Feature: 400 Integer Difference feature

重点：用组合的方式实现自然的风险对冲，间接的实现止盈止损与不持仓，通过多品种，多个轴，多标签，多模型的方式，进而增加整体的管理规模，同时业绩曲线也会更平滑

实盘交易中模型实时推理的方法

模型在实盘交易中实时推理需要注意的问题

实现模型全过程自动完成交易的代码实现

如何提升模型的资金管理规模

实盘交易中模型实时推理的方法与需要注意的问题



uni 的量化日记

CPU问题：推理与训练的CPU需要相同架构

VSCode：环境的选择

库的版本：推理与训练环境的库版本要相同

时间问题：夜盘的数据读取

实时数据：实时数据与训练数据的数据源要相同

轴线对齐：推理轴与自定义轴需要对齐

数据处理：自定义轴的增量累加

PCA降维：将推理特征融入到训练集特征中

实现模型全过程自动完成交易的代码实现



uni 的量化日记

```
+++++
1m Turnover: 2025-03-07 10:31 409.5501
*** Save to file OK ***
New Turnover Sum: 19417.61
Percent: 62.64%
+++++
1m Turnover: 2025-03-07 10:32 301.30448
*** Save to file OK ***
New Turnover Sum: 19718.91
Percent: 63.61%
+++++
1m Turnover: 2025-03-07 10:33 152.00141
*** Save to file OK ***
New Turnover Sum: 19870.91
Percent: 64.1%
+++++
1m Turnover: 2025-03-07 10:34 204.13522
*** Save to file OK ***
New Turnover Sum: 20075.05
Percent: 64.76%
+++++
1m Turnover: 2025-03-07 10:35 104.8475
*** Save to file OK ***
New Turnover Sum: 20179.9
Percent: 65.1%
+++++
1m Turnover: 2025-03-07 10:36 87.57329
*** Save to file OK ***
New Turnover Sum: 20267.47
Percent: 65.38%
+++++
1m Turnover: 2025-03-07 10:37 93.10115
*** Save to file OK ***
New Turnover Sum: 20360.57
Percent: 65.68%
+++++
□
```

模型推理

模型推理结果
1、0、2
文件形式

交易程序

天勤

...

扩展阅读



uni 的量化日记

推荐图书

推荐论文

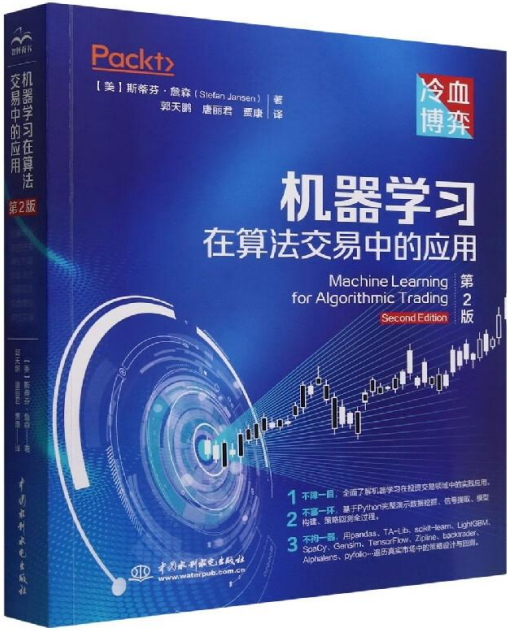
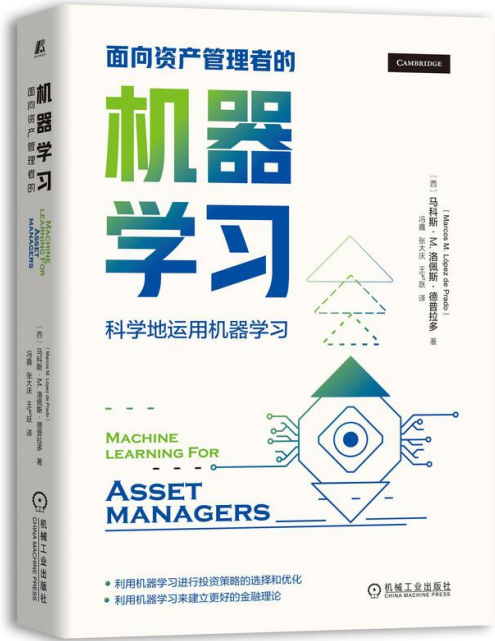
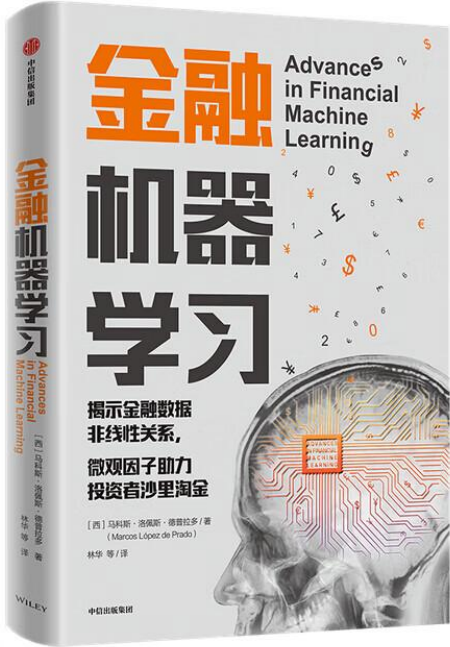
“鸢尾花书”的整体布局



数学 + Python编程 + 可视化 + 机器学习实践



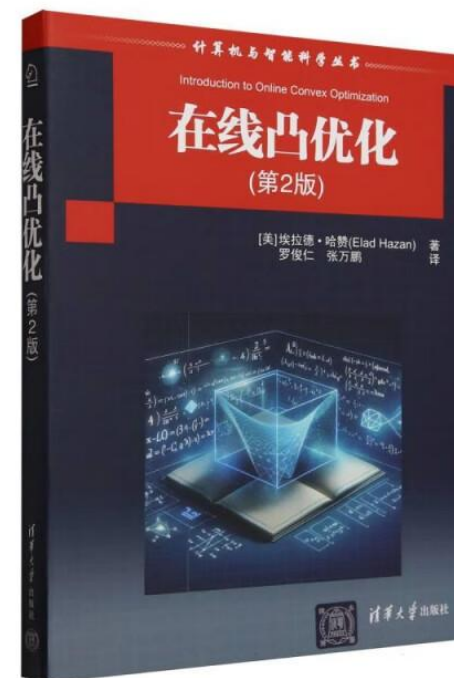
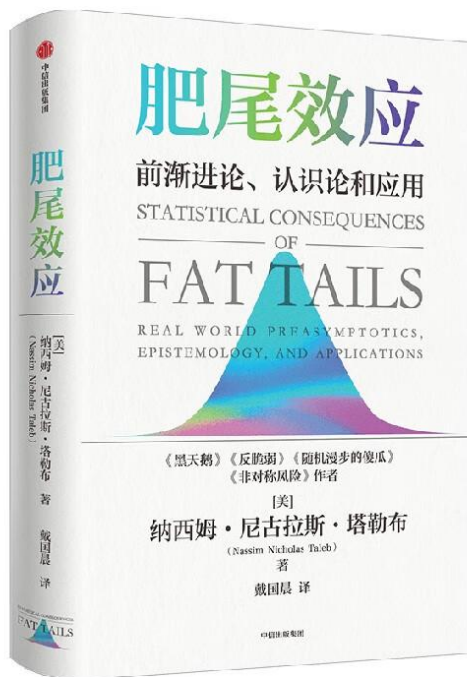
扩展阅读（推荐图书 进阶）



扩展阅读 (推荐图书 高阶)



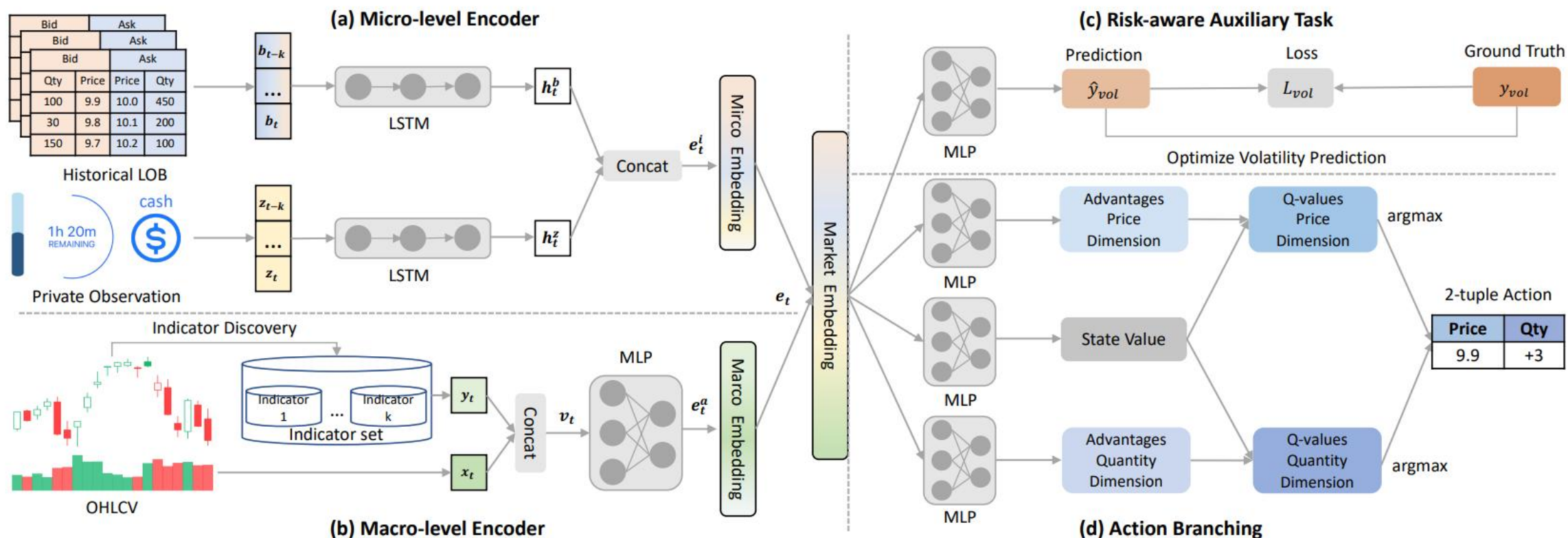
uni 的量化日记



扩展阅读（推荐论文）



uni 的量化日记

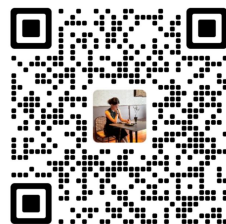


DeepScalper强化学习框架，兼顾微观和宏观市场信息进行策略学习，同时将波动率预测作为辅助任务，以实现策略在实现利润最大化的同时意识到市场风险。

论文: DeepScalper: A Risk-Aware Reinforcement Learning Framework to Capture Fleeting Intraday Trading Opportunities



uni 的量化日记



微信 uni