华东理工大学2021 –2022学年第2学期

《Python与金融计算》实验报告

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验名称 | 中国股市单因子资产定价模型的实证检验 | | | | | | | | | |
| 专 业 | | 金融 | 姓名 | 刘海柏 | 学号 | 20030047 | | 班级 | | 202 |
| 实验时间 | | 3/15-3/22 | | 实验地点 | 信息楼319 | | 指导教师 | | 蒋志强 | |

|  |
| --- |
| 实验目的/要求 |
| 1. 掌握单因子资产定价模型的时间序列估计与检验； 2. 掌握单因子资产定价模型的横截面检验。 |
| 实验内容 |
| 1. 认真阅读3篇文献资料（自己也可以下载相关文献进行阅读），了解资产定价模型的相关研究，重点关注其中的研究思路和方法。 2. 尝试找出以0或者6开头，最后两位数字和自己学号最后两位数字一样的股票代码，并从中任意选取8只股票下载2000-2021年的日收益和月收益数据，进行如下实验：（1）利用股票日收益数据对CAPM模型做单资产时间序列检验；（2）利用股票月收益数据对CAPM模型做多资产时间序列检验。 3. 请利用2000-2021年中国A股月度数据实证检验中国A股市市场是具有**惯性效应**，还是**反转效应**？（数据请从锐思数据库或者Tushare下载）   研究思路（参考课本《金融计量学》P48）：在每个月，计算过去N个月的累积收益率，再根据此累积收益率由低到高排序，构造5个等权重投资组合，并计算持有这些投资组合M个月的累积收益率，追踪投资组合收益率序列，判断中国A股市场存在惯性效应还是反转效应。  N = 1, 3, 6, 12；M = 1, 3, 6, 12  000708 000752 000831 000926 001259 002021 002080 002138 |
| 实验总结  **请提供对本次实验结果的讨论分析，以及实验的心得和体会。包括对知识点的掌握，算法的理解，以及对理论课程和实验课程改进的建议。（不少于500字）** |
| CAPM比较深刻接触应该是上学期的证券投资学，含有很多复杂很长的公式，推导过程也蛮复杂的，印象中有最小方差组合以及许多知识点，在书本上计算的时候虽然复杂但是在写程序的时候又是另一回事了，个人感受在建构金融市场投资组合时，必需有效率前缘概念，就是所有金融资产，多有它的「预期报酬率」与「风险」相对位置。如外汇的位置就是在、「预期报酬率」与「风险」最低的水平；股票则在「预期报酬率」与「风险」较高位置。这种对所有金融资产投资组合，是一般大型寿险公司、或是金控公司投资部门执行长，最基本应有的一种资产配置概念。对于一个全方位金融商品投资组合而言，它的资产是配置在股票、债券、共同基金、及货币等资产之上。它们在效率前缘曲线上，多有它应有的位置，个股或投资组合报酬率的系统风险溢价报酬。程序方面，在验证那些三大检验的时候，其实公式的推导又更为复杂，看了书上不太可能记得住，在检验的时候也大多照着公式去写，大致上能理解代码的运行检验，另外这堂课也学习到如何去mergedata，还挺实用的，对于以后的学术研究应该有很大的帮助，三大检验都有各自的用途,LR检验，既需要估计有约束模型，也需要估计无约束模型；对于Wald检验，只需要估计无约束模型；LM检验的话，只需要估计有约束模型。一般情况下，由于估计有约束模型相对更复杂，而Wald检验好像是最为常用。 |
| 教师批阅： 实验成绩：  教师签名: 日期： |

**实验报告正文：**

（每次实验报告均为一篇小论文，因此，统一按照学术论文的要求完成实验报告正文，应包括：题目、摘要、文献综述、模型和方法、结果和讨论、参考文献、附录，具体格式如下：

**中国股市收益率的分布特征**

金融202 2030047 刘海柏

摘要：资产定价理论作为现代金融理论的组成之一，传统 CAPM 模型指出个股收益只和整个市场的系统性风险有关，越来越多的学者通过实证分析表明，市场中存在着大量实际收益不同于 CAPM 定价的现象，单一系统性风险无法完全衡量股票的平均收益。因此，本文基于 CAPM 模型对8个股进行实证分析，8 只不同上市公司的股票规模各异，样本区间选择 2000～2021 年的日收盘数据进行单资产时间序列检验，和月收盘数据进行多资产联合检验，采用 CAPM 模型分析验证，从而判断 CAPM 能否合理分析资产的价格。

1 文献综述

1.1单因子资产定价：

资本资产的定价问题一直深受金融市场领域乃至整个金融领域的关注。随着世界经济、金融的不断发展,如今,如何有效的确定金融资产的价格仍是很多经济学家所面临的重大问题。 马科威茨通过把收益、风险分别定义为均值和方差,第一次从数量上解决了收益与风险的关系问题,资本资产定价模型就是在这一理论的基础之上提出的。1970年,威廉·夏普率先提出资本资产定价模型:CAPM模型,成为资本资产定价的基础。此次结论为:投资的收益只与风险有关。,但CAPM模型还是存在着很多理论上和实践上的局限性。第一,CAPM的假设前提难以实现，其中包括投资者在投资前能够知道投资收益率的概率分布是正态分布和投资决策的主要因素是期望收益率和风险两项，另外附加假设也包括：投资者是理性的，能从有效边界的某处选择投资组合 ；第二 ，资本市场是完全有效的市场 ，没有任何投资摩擦，然而我们知道，投资者不可能是完全理性，一定存在着许多非理性的投资者，和市场存在变动，投资摩擦现象可能存在，许多假设在现实中较难以实现;第二,CAPM中的β值难以确定;若B=0则表示投资该资产的收益率变动与市场组合收益率变动不存在关系 ②若B=l则表示投资该资产的收益率变动与市场组合收益率的变动一致，投资该资产可获得市场的平均收益水平；最后,与之相关的实证结果比较缺乏检验。因此,金融市场学家不断探求比CAPM更为有效的资本市场理论。

1.2多因子资产定价

MacBeth 的实证结果表明市场贝塔不能完全对投资组合的横截面收益进行解释，承担市场风险不能作为收益来源的唯一解释；通过 CAPM 模型得出它的三个含义：风险与收益的关系是线性的，β 是对系统性风险的完全度量，E（rm）-rf>0。 Ross 提出多因子套利定价理论APT 模型，他指出收益来源于多种因素而非单一因素，总收益可以看作是一系列风险溢价的线性相加。比如假设证券的收益是由一系列特定行业因素、市场因素共同决定,如果当两证券的收益受到相同因素影响,那么这两种证券收益之间就存在相关的关系，另外APT假设比CAPM的假设要少的许多。可应用于资本预算、证券估值或投资基金业绩评估。基本上来说就是CAMP模型的拓展,为投资者理解市场中的风险与收益率间的均衡关系提供了新的思路

最后，CAPM的成立下表示所有投资者均可充分分散投资的极端假设。而在另一个极端，即所有投资者均只能持有一项资产的情况同时，投资者则对其承担的所有风险要求相应的风险补偿，而不仅仅所包含是所持资产的系统风险。Levy[4]提出理论模型表明，投资者所持组合中包含的股票种类越少，组合中个股回报率方差 (即包括系统风险在内的总风险)对组合超额预期回报率的影响越大。

2 模型和方法

…2.1单资产检验

考虑单个资产，在经典线性回归模型假设条件下，我们使用t值来做检验CAPM，在阅读过许多学者的文献，大多也使用这种方法，经典线性回归模型假设为线性性与可加性，扰动项的均值独立于所有解释变量的数据且误差项呈现正态分布，即所有个体的解释变量都不相关。单资产CAPM检验步骤如下：

1. 用OLS估计市场模型，得到*αi* 的估计值
2. 计算 *αi* = 0 的 *t* 检验统计量
3. 确定显著性水平，比较t统计量和计算检验的p值

当无风险资产存在下分析市场有效组合，使用的方法为普通最小二乘法，首先构建所有股票沪深300指数的日收益率序列，再剔除空白值以及异常值后，对股票利用市场模型估计2000/01/01-2021/12/31期间的β系数， Rit = E(Rit )- Rft 为证券 i 在 t 时刻的风险溢价； Rmt = E(Rmt )- Rft 为市场组合的风险溢价；截距项 αi 和斜率项 βi 为模型的待估计参数； εit 则为回归模型的随机误差项。

CAPM说明当市场达成均衡，个别证券收益与风险的配置会落于证券市场线 （Securities Market Line，简称 SML） 上，因此，截距项 αi 可视为个别证券实际收益率与理论收益率的差距。 若 α = 0 时，说明该证券的实际收益率等于理论收益率；若 α > 0 时，说明该证券的价值被低估，使得证券实际收益率大于理论收益率，投资人可以利用套利活动获取超额收益。

2.1.1主要变量定义：

1. 市场指数：沪深300指数是一种价值加权指数，具有成交量大、权重股多以及市值大等特点，是中国股票市场中最具代表性的股票价格指数之一，能较准确地反映整体行情的变化和中国股票市场的发展趋势，符合 CAPM 市场组合指数构造的要求。

2. 无风险利率。根据锐思金融研究数据库，选取一年定期存款利率作为无风险收益率。

3. 个股收益率与市场组合收益率。 Rit =(Pit - Pit - 1)/Pit - 1 其中， Rit 为证券i在t月份的收益率； Pit 为证券 i在t月份的收盘价。当样本股中的收益率数据缺失值太多时，剔除该股票，重新选择样本对象。

经典CAPM模型：E(Rq​)−Rf​=βqm​[E(Rm​)−Rf​] （2-1）

*βim =* Cov(*Ri*, *Rm*) / Var(*Rm*) （2-2）

上述模型中，假定风险因子可以完全解释资产的风险溢价（rm-rf），模型不存在截距项。CAPM衡量期望收益率但一般难以均衡，资产收益率可分解为：期望和未预期部分

文本

描述已自动生成

（2-3）

（2-4）画里面的卡通人物

中度可信度描述已自动生成

Rm ​是市场投资组合的收益率，也就是整个市场所有风险资产的收益情况。它的作用是作为一个基准，评估目标投资组合相对于基准的提升量。在实际应用中，我们通常使用大盘指数

βqm是投资组合q的Beta值，|β|大于1表示我们的投资组合比大盘的波动更剧烈，|β|小于1代表我们的投资组合的波动小于大盘，如果β为负，则说明我们的投资组合与大盘的波动方向相反

​ E(Rq​)−Rf​是风险投资组合q比无风险资产高出的期望收益率，这部分称作风险溢酬。这部分额外的期望收益率是由于我们的投资组合q的风险与大盘不同决定的

​

CAPM模型引入了α参数，CAPM模型可以写成如下形式

Rit​−Rft​=αi​+β​(Rmt​−Rft​)+ϵit （2-5）

考虑单个资产i，在线性回归下的模型假设可用t检验来检验模型

==0 =0 （2-6）

当（2-7）成立时，说明该资产存在超额回报率，CAPM模型不成立

若 α = 0 时，说明该证券的实际收益率等于理论收益率；若 α > 0 时，说明该证券的价值被低估，使得证券实际收益率大于理论收益率，投资人可以利用套利活动获取超额收益。介于47尾号结尾的股票大多存在缺失，较难找出合适的数据本文选取股票代码为000001，000008，000716，000752，000767，000831，000926，000708这8只股票，用其2000年-2021年的日度收益率数据来验证市场模型是否存在超额收益率，CAPM是否成立。

回归方程如下:



(2-7)

最小二乘估计量：

假设条件：自变量与残差的协方差为零验证CAPM模型在实证中是否满足此条件

图示, 示意图

描述已自动生成

文本, 信件

描述已自动生成

(2-8)

检验*H*0: *αi* = 0 的统计量 ( *t* 检验 )

文本

描述已自动生成

文本

描述已自动生成

(2-9)

股票000001的t检验和超额收益和市场超额收益散点图图表

描述已自动生成

图2-1

OLS Regression Results

========================================================================

Dep. Variable: y R-squared: 0.204

Model: OLS Adj. R-squared: 0.203

Method: Least Squares F-statistic: 372.3

Date: Tue, 04 Apr 2023 Prob (F-statistic): 4.73e-74

Time: 23:24:27 Log-Likelihood: 3672.6

No. Observations: 1458 AIC: -7341.

Df Residuals: 1456 BIC: -7331.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

---------------------------------------------------------------------------------------------------------------------------

const 0.0006 0.001 1.255 0.210 -0.000 0.002

x1 0.5378 0.028 19.295 0.000 0.483 0.593

=========================================================================

Omnibus: 785.612 Durbin-Watson: 1.098

Prob(Omnibus): 0.000 Jarque-Bera (JB): 25010.575

Skew: 1.910 Prob(JB): 0.00

Kurtosis: 22.927 Cond. No. 54.6

表2-1股票000001的t检验

股票000008的t检验和超额收益和市场超额收益散点图

图表

描述已自动生成图2-2

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.082

Model: OLS Adj. R-squared: 0.081

Method: Least Squares F-statistic: 129.4

Date: Tue, 04 Apr 2023 Prob (F-statistic): 8.62e-29

Time: 23:28:25 Log-Likelihood: 3754.9

No. Observations: 1458 AIC: -7506.

Df Residuals: 1456 BIC: -7495.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

---------------------------------------------------------------------------------------------------------------------------

const -0.0004 0.000 -0.881 0.378 -0.001 0.001

x1 0.2997 0.026 11.377 0.000 0.248 0.351

=========================================================================

Omnibus: 432.753 Durbin-Watson: 0.926

Prob(Omnibus): 0.000 Jarque-Bera (JB): 6035.295

Skew: 0.983 Prob(JB): 0.00

Kurtosis: 12.771 Cond. No. 54.6

=========================================================================

表2-2股票000008的t检验

图表

描述已自动生成股票000716的t检验和超额收益和市场超额收益散点图

图2-3

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.066

Model: OLS Adj. R-squared: 0.066

Method: Least Squares F-statistic: 103.1

Date: Tue, 04 Apr 2023 Prob (F-statistic): 1.86e-23

Time: 23:36:55 Log-Likelihood: 3203.9

No. Observations: 1450 AIC: -6404.

Df Residuals: 1448 BIC: -6393.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

--------------------------------------------------------------------------------------------------------------------------

const 0.0008 0.001 1.087 0.277 -0.001 0.002

x1 0.3869 0.038 10.156 0.000 0.312 0.462

=========================================================================

Omnibus: 311.799 Durbin-Watson: 1.018

Prob(Omnibus): 0.000 Jarque-Bera (JB): 1579.518

Skew: 0.911 Prob(JB): 0.00

Kurtosis: 7.777 Cond. No. 54.6

=========================================================================

表2-3股票000716的t检验

股票'000752的t检验和超额收益和市场超额收益散点图图表, 折线图

描述已自动生成

图2-4

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.029

Model: OLS Adj. R-squared: 0.028

Method: Least Squares F-statistic: 42.90

Date: Tue, 04 Apr 2023 Prob (F-statistic): 7.97e-11

Time: 23:39:40 Log-Likelihood: 3285.5

No. Observations: 1456 AIC: -6567.

Df Residuals: 1454 BIC: -6556.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

---------------------------------------------------------------------------------------------------------------------------

const -0.0005 0.001 -0.823 0.411 -0.002 0.001

x1 0.2341 0.036 6.550 0.000 0.164 0.304

=========================================================================

Omnibus: 19.362 Durbin-Watson: 0.851

Prob(Omnibus): 0.000 Jarque-Bera (JB): 21.308

Skew: 0.232 Prob(JB): 2.36e-05

Kurtosis: 3.370 Cond. No. 53.8

=========================================================================

表2-4股票000752的t检验

股票000767的t检验和超额收益和市场超额收益散点图

图表, 条形图

描述已自动生成

图2-5

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.034

Model: OLS Adj. R-squared: 0.033

Method: Least Squares F-statistic: 50.91

Date: Tue, 04 Apr 2023 Prob (F-statistic): 1.52e-12

Time: 23:53:05 Log-Likelihood: 3221.6

No. Observations: 1452 AIC: -6439.

Df Residuals: 1450 BIC: -6429.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

---------------------------------------------------------------------------------------------------------------------------

const 0.0010 0.001 1.443 0.149 -0.000 0.002

x1 0.2692 0.038 7.135 0.000 0.195 0.343

=========================================================================

Omnibus: 327.383 Durbin-Watson: 0.981

Prob(Omnibus): 0.000 Jarque-Bera (JB): 1502.323

Skew: 0.991 Prob(JB): 0.00

Kurtosis: 7.572 Cond. No. 54.6

========================================================================

表2-5股票000767的t检验

股票000831的t检验和超额收益和市场超额收益散点图

图表, 直方图

描述已自动生成

图2-6

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.042

Model: OLS Adj. R-squared: 0.042

Method: Least Squares F-statistic: 63.46

Date: Wed, 05 Apr 2023 Prob (F-statistic): 3.30e-15

Time: 14:22:31 Log-Likelihood: 2828.8

No. Observations: 1440 AIC: -5654.

Df Residuals: 1438 BIC: -5643.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

---------------------------------------------------------------------------------------------------------------------------

const 0.0033 0.001 3.742 0.000 0.002 0.005

x1 0.3874 0.049 7.966 0.000 0.292 0.483

=========================================================================

Omnibus: 124.520 Durbin-Watson: 0.990

Prob(Omnibus): 0.000 Jarque-Bera (JB): 186.357

Skew: 0.657 Prob(JB): 3.41e-41

Kurtosis: 4.174 Cond. No. 54.3

=========================================================================

表2-6股票000831的t检验

股票 000926的t检验和超额收益和市场超额收益散点图

图表

描述已自动生成

图2-7

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.165

Model: OLS Adj. R-squared: 0.164

Method: Least Squares F-statistic: 287.8

Date: Tue, 04 Apr 2023 Prob (F-statistic): 4.74e-59

Time: 23:49:12 Log-Likelihood: 4315.6

No. Observations: 1458 AIC: -8627.

Df Residuals: 1456 BIC: -8617.

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

---------------------------------------------------------------------------------------------------------------------------

const -0.0005 0.000 -1.653 0.099 -0.001 0.000

x1 0.3043 0.018 16.966 0.000 0.269 0.339

========================================================================

Omnibus: 472.219 Durbin-Watson: 1.134

Prob(Omnibus): 0.000 Jarque-Bera (JB): 13532.367

Skew: 0.891 Prob(JB): 0.00

Kurtosis: 17.818 Cond. No. 54.6

表2-7股票000926的t检验

股票 000708的t检验和超额收益和市场超额收益散点图

**图表, 散点图

描述已自动生成**

图2-8

OLS Regression Results

=========================================================================

Dep. Variable: y R-squared: 0.465

Model: OLS Adj. R-squared: 0.465

Method: Least Squares F-statistic: 2156.

Date: Tue, 04 Apr 2023 Prob (F-statistic): 0.00

Time: 23:50:53 Log-Likelihood: 6121.1

No. Observations: 2482 AIC: -1.224e+04

Df Residuals: 2480 BIC: -1.223e+04

Df Model: 1

Covariance Type: nonrobust

=========================================================================

coef std err t P>|t| [0.025 0.975]

----------------------------------------------------------------------------------------------------------------------

const 8.401e-05 0.000 0.203 0.839 -0.001 0.001

x1 1.0884 0.023 46.427 0.000 1.042 1.134

=========================================================================

Omnibus: 541.886 Durbin-Watson: 1.876

Prob(Omnibus): 0.000 Jarque-Bera (JB): 2063.321

Skew: 1.033 Prob(JB): 0.00

Kurtosis: 6.961 Cond. No. 56.8

=========================================================================

表2-8股票000708的t检验

2.2多资产检验

2.2.1月收益率情况

图形用户界面, 应用程序

描述已自动生成

**图2-2 月收益率图示从左上至右下分别为股票**000708**股票**000752**股票股票**000831 000926

图形用户界面, 应用程序

描述已自动生成

**图2-3 月收益率图示从左上至右下分别为股票**001259 **股票**002021**股票** 002080 **股票**002138

考虑8个资产，CAPM蕴含的零假设是：

H0： = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 （2-10）

即所有的α 都等于 0,由于数据协方差，联合检验零假设不是单独检验的相加总，第二当联合检验另加设成立时，并不意味着对所有单资产的零假设成立。

把单资产写成矩阵模式

（2-11）

== =0 =0 i=1，2，3，…N （2-12）

2.2.1Wald检验

适用于线性和非线性约束条件检验，当零假设α =0， Wald只需要估计一个无约束模型，但需要估计约束条件的方差协方差矩阵，**并根据约束条件构造统计量，进行假设检验具体步骤如下：**

**1利用最小二乘法估计模型参数**

**2.计算Wald检验统计量**

文本, 信件, 示意图

描述已自动生成

（2-13）

**3.根据统计量计算 *p*-value（chi2cdf, fcdf）**

文本, 信件

描述已自动生成

（2-14）

当值小于给定显著性水平的临界值时，不拒绝零假设CAPM成立。反之，则拒绝零假设，认为CAPM不成立，样本容量很大时该统计量服从自由度为N（即10）的卡方分布

|  |  |  |  |
| --- | --- | --- | --- |
| Wald Test1 | Wald Test2 | LR Test | LM Test |
| 14.20851, | 0.32292, | 9.12215 | 6.20003 |
| 0.07649 | 0.89907 | 0.33210 | 0.62484 |

表2-9 CAPM多资产联合检验结果

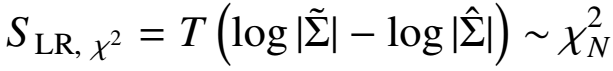
2.2.2似然比检验（LR Test）

如果约束条件成立则相应的约束模型和非约束模型的极大似然函数是相等的

具体步骤如下：

1. 估计限制模型的参数（*αi* = 0）

2.估计无限制模型的参数

3..构造统计量（|A|矩阵A的行列式）

（2-15）

4.根据统计量计算 *p*-value

文本, 信件

描述已自动生成文本, 信件

描述已自动生成

（2-16）

（2-17）

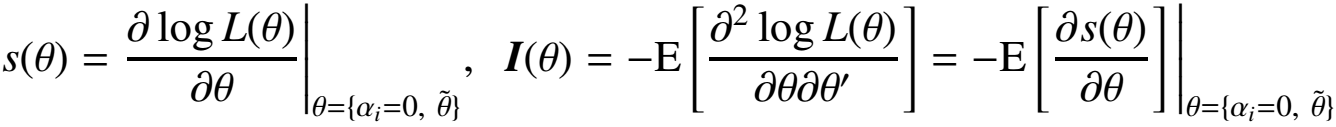
如果零假设α =0成立，限制和非限制得到的参数极大似然估计值没有差别，当的值小于给定显著水平的临界值时，不拒绝零假设，CAPM成立。反之，则拒绝零假设，CAPM不成立，得到的统计量为11.2358，p值等于0.1374不能拒绝原假设。

2.2.3LM检验（拉格朗日检验）

记L为非限制模型的对数似然函数，只需估计约束模型。约束条件简单时，适用于LM检验

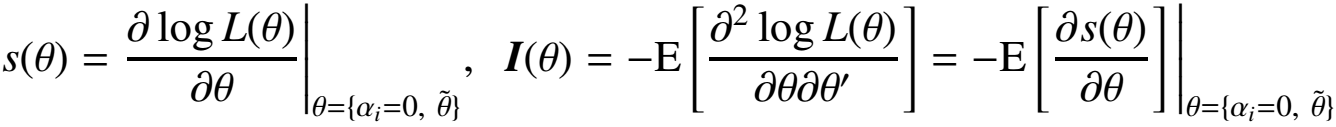
具体步骤为：

1. 估计限制模型的参数（*αi* = 0）



（2-17）

1. 计算无限制模型似然函数的得分向量和信息矩阵



（2-18）

1. 构造统计量



（2-19）

4.根据统计量计算 *p*-value

文本, 信件

描述已自动生成

文本

描述已自动生成

（2-20） （2-21）

如果零假设α =0成立L在，一阶条件得分向量在α =0的取值接近0，当值小于给定显著水平的临界值时，不拒绝零假设，CAPM成立。反之，则拒绝零假设，CAPM不成立，从表2-1中得到的统计量为11.30502，p值等于0.18501，不能拒绝原假设。

2-3单因子资产定价模型的横切面检验

惯性效应是指过去表现好的股票（赢者）会继续表现好，过去表现差的的股票（输者）会继续表现差。与惯性效应刚好相反，反转效应指过去表现好的股票接下来会继续表现差，过去表现差的股票接下来会继续表现好

实证分析以2011作为起始年，用十年的数据计算β值，具体步骤为利用*t*-1时刻之前的信息估计每个股票的*β，*将股票按照*β*大小排序、分组构造投资组合，计算每个投资组合在*t*时刻的收益率为，将序位最小和最大的两组再平均分成两组形成4个组合。计算4个组合从2011年到2021年的平均收益率，然后分别计算β值最高的组合和β值最低的组合的平均收益率，计算t值和pvalue。

零假设：

图片包含 游戏机, 桌子, 钟表

描述已自动生成

（2-21）

文本, 信件

描述已自动生成

（2-22）

文本, 信件

描述已自动生成

（2-23）

根据上述步骤，所得结果为

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序位 | 1 | 2 | 3 | 4 |
|  | 0.033948137 | 0.00688175 | 0.0053916 | 0.001441303 |

表2-10资产按b值排序

T检验结果

T值：1.53975788

P值：0.7825

3 结果与讨论

3.1单因子资产定价模型的时间序列检验

透过实证检验8只股票通过单资产实证分析，p值非常大不论在0.01或是0.05的显著性水平下都不拒绝=0的原假设，仅股票000831和股票000926的p值分别为为0和0.099，**这两只股票拒绝=0的原假设，不能被CAPM所支持，其余六只股票，constant的p值很大，不能拒绝原假设，单资产检验结果支持CAPM模型**

通过多资产实证分析，本文得出的结论为：联合检验不能拒绝原假设，因此，检验股票不存在超额回报率。所以多资产联合检验支持CAPM

在计算多资产联合检验的时候，翻阅文献大多采用的是月度数据而非老师所用的日度数据，如果想对CAPM在中国A股市场的适用性进行研究，就必须考虑到不同时期政府和经济状态对股市的影响，进一步会影响CAPM的*β*值，这时候就需要进行数据的分组，产生非同步交易的情况，结论可能会出现偏差，另外在数据库里面的股票数据常常出现连续几周以上不交易，在实证的时候必须将其剔除，导致可用的数据大为减少，综上，月度数据作为实证检验较为合适。

3.2单因子资产定价模型的横切面检验

透过表2-10可得知，以前β高的投资组合，再后面十年表现得还是相当稳健，存在惯性效应，不过在实证过程中换一组数据却有明显不同的现象，而存在反转效应，是值得关注的现象，

3.3总结

图表, 折线图, 直方图

描述已自动生成随着我国资本市场的不断发展，经历过金融危机后股票市场规范程度的提高，我国证券市场可能越来越接近CAPM模型的假设条件。透过CAPM模型的实证检验一方面可以使投资者认识到我国股票市场上风险和收益之间的关系，助于做出正确的投资决策，另一方面可以加强管理者对我国证券市场的认识，从而制定出更符合我国证券市场的政策，在阅读文献后发现，CAPN的适用性在不同行业板块也有不同的现象，本文选取的8只股票进行了单资产时间序列和多资产的联合检验，均支持CAPM模型。但鉴于所选数据的局限性，样本数不够足够，得出中国A股市场存在惯性效应，而非反转效应，此次研究无法准确代表是否所有股票皆存在显著惯性效应，往后需要大量考察更多数据才能得知。

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

图表, 瀑布图

描述已自动生成

4 参考文献

1. 靳云汇 刘 霖．中国股票市场 CAPM 的实证研究[J] 金 融 研 究，2001（07)：106-115．
2. 贾 权 陈章武 中国股市有效性的实证分析[J]．金 融 研 究，2003(07)：86-92
3. Huai-Long Shi Zhi-Qiang Jiang，Wei-Xing Zhou Profitavility of contrarian strategies in the Chinese Stock Market[J]2015,14:1-22
4. Levy H．Equilibrium in an Imperfect Market：A Constraint on the Number of Securities in the Portfolio[J]．ne American Eco· nomic Review，1978，68(4)：643-658
5. CHEN, YUE-MING, WENG, YU-TING, DONG, XIAOYU, et al. Wald tests for variance-adjusted equivalence assessment with normal endpoints[J]. Journal of biopharmaceutical statistics,2017,27(2):308-316.

5 附录

plt.figure(1)  
onedata=pd.read\_csv('onedata.csv',encoding='GB2312')  
onedata.columns= ['code','date','close','rf']  
onedata.dropna(inplace=True)  
stk\_onecodes = np.unique(onedata['code'].values)  
print(stk\_onecodes)  
stock\_data1 = onedata[onedata['code'] == stk\_onecodes[0]]  
stock\_data1['return'] = np.log(stock\_data1['close']) - np.log(stock\_data1['close'].shift(periods=1))#对数收益  
stock\_data1.dropna(inplace=True)  
ind = (stock\_data1['return'] >= -0.1) & (stock\_data1['return'] <= 0.1)  
stock\_data1 = stock\_data1.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data1['return'].plot()  
data300=pd.read\_csv('1EShowData\_data\_Index\_daily\_price\_2001-2021.csv',encoding='GB2312', usecols=[1, 5])  
data300.columns=['date','close']  
data300.dropna(inplace=True)  
data300['return'] = np.log(data300['close']) - np.log(data300['close'].shift(periods=1))  
merge\_data = pd.merge(left=stock\_data1[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(2)  
stock\_data2 = onedata[onedata['code'] == stk\_onecodes[1]]  
stock\_data2['return'] = np.log(stock\_data2['close']) - np.log(stock\_data2['close'].shift(periods=1))#对数收益  
stock\_data2.dropna(inplace=True)  
ind = (stock\_data2['return'] >= -0.1) & (stock\_data2['return'] <= 0.1)  
stock\_data2 = stock\_data2.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data2['return'].plot()  
merge\_data = pd.merge(left=stock\_data2[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(3)  
stock\_data3 = onedata[onedata['code'] == stk\_onecodes[2]]  
stock\_data3['return'] = np.log(stock\_data3['close']) - np.log(stock\_data3['close'].shift(periods=1))#对数收益  
stock\_data3.dropna(inplace=True)  
ind = (stock\_data3['return'] >= -0.1) & (stock\_data3['return'] <= 0.1)  
stock\_data3 = stock\_data3.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data3['return'].plot()  
merge\_data = pd.merge(left=stock\_data3[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(4)  
stock\_data4 = onedata[onedata['code'] == stk\_onecodes[3]]  
stock\_data4['return'] = np.log(stock\_data4['close']) - np.log(stock\_data4['close'].shift(periods=1))#对数收益  
stock\_data4.dropna(inplace=True)  
ind = (stock\_data4['return'] >= -0.1) & (stock\_data4['return'] <= 0.1)  
stock\_data4 = stock\_data4.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data4['return'].plot()  
merge\_data = pd.merge(left=stock\_data4[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(5)  
stock\_data5 = onedata[onedata['code'] == stk\_onecodes[4]]  
stock\_data5['return'] = np.log(stock\_data5['close']) - np.log(stock\_data5['close'].shift(periods=1))#对数收益  
stock\_data5.dropna(inplace=True)  
ind = (stock\_data5['return'] >= -0.1) & (stock\_data5['return'] <= 0.1)  
stock\_data5 = stock\_data5.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data5['return'].plot()  
merge\_data = pd.merge(left=stock\_data5[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(6)  
stock\_data6 = onedata[onedata['code'] == stk\_onecodes[5]]  
stock\_data6['return'] = np.log(stock\_data6['close']) - np.log(stock\_data6['close'].shift(periods=1))#对数收益  
stock\_data6.dropna(inplace=True)  
ind = (stock\_data6['return'] >= -0.1) & (stock\_data6['return'] <= 0.1)  
stock\_data6 = stock\_data6.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data6.dropna(inplace=True)  
stock\_data6['return'].plot()  
merge\_data = pd.merge(left=stock\_data6[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(7)  
stock\_data7 = onedata[onedata['code'] == stk\_onecodes[6]]  
stock\_data7['return'] = np.log(stock\_data7['close']) - np.log(stock\_data7['close'].shift(periods=1))#对数收益  
stock\_data7.dropna(inplace=True)  
ind = (stock\_data7['return'] >= -0.1) & (stock\_data7['return'] <= 0.1)  
stock\_data7= stock\_data7.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data7.dropna(inplace=True)  
stock\_data7['return'].plot()  
merge\_data = pd.merge(left=stock\_data7[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()  
plt.figure(8)  
stock\_data8 = onedata[onedata['code'] == stk\_onecodes[7]]  
stock\_data8['return'] = np.log(stock\_data8['close']) - np.log(stock\_data8['close'].shift(periods=1))#对数收益  
stock\_data8.dropna(inplace=True)  
ind = (stock\_data8['return'] >= -0.1) & (stock\_data8['return'] <= 0.1)  
stock\_data8= stock\_data8.loc[ind, :]  
plt.subplot(2,2,1)  
stock\_data8.dropna(inplace=True)  
stock\_data8['return'].plot()  
merge\_data = pd.merge(left=stock\_data8[['date', 'return', 'rf']],right=data300[['date','return']],  
 on='date',  
 how='inner')#第一个为股票第二为指数按照日期去拼  
merge\_data.columns = ['date', 'return\_stk', 'rfreturn', 'return\_ind']  
merge\_data.dropna(inplace=True)  
stk\_ret = merge\_data['return\_stk'].values  
rf\_ret = merge\_data['rfreturn'].values  
ind\_ret = merge\_data['return\_ind'].values  
plt.subplot(2,2,2)  
plt.plot(ind\_ret - rf\_ret, stk\_ret - rf\_ret, 'o', ms=5, mfc='w', lw=2)#股票收益减去无风险  
plt.xlabel(r'$r\_m - r\_f$', fontsize = 20)  
plt.ylabel(r'$r\_i - r\_f$', fontsize = 20)  
x = sm.add\_constant(ind\_ret-rf\_ret)#定义x  
y = stk\_ret - rf\_ret  
model = sm.OLS(y, x)  
results = model.fit()  
print(results.summary())  
plt.show()

stock\_data = pd.read\_csv('month data2.csv',encoding='GB2312')  
stock\_data.dropna(inplace=True)  
stock\_data.columns = ['code','date', 'close', 'rfreturn']  
# # 股票代码\_Stkcd 日期\_Date 收盘价\_Clpr 日无风险收益率\_DRfRet  
stk\_codes = np.unique(stock\_data['code'].values)  
print(stk\_codes)  
plt.figure(2)  
plt.subplot(2,2,1)  
stock\_data50 = stock\_data[stock\_data['code'] == stk\_codes[0]]  
stock\_data50['date'] = pd.to\_datetime(stock\_data50['date'])  
stock\_data50.sort\_values(by=['date'], inplace=True)  
stock\_data50['return'] = np.log(stock\_data50['close']) - np.log(stock\_data50['close'].shift(periods=1))  
stock\_data50.dropna(inplace=True)  
ind = (stock\_data50['return'] >= -0.1) & (stock\_data50['return'] <= 0.1)  
stock\_data50 = stock\_data50.loc[ind, :]  
print(stock\_data50)  
plt.plot(stock\_data50['return'].values)  
plt.subplot(2,2,2)  
stock\_data51 = stock\_data[stock\_data['code'] == stk\_codes[1]]  
stock\_data51['date'] = pd.to\_datetime(stock\_data51['date'])  
stock\_data51.sort\_values(by=['date'], inplace=True)  
stock\_data51['return'] = np.log(stock\_data51['close']) - np.log(stock\_data51['close'].shift(periods=1))  
stock\_data51.dropna(inplace=True)  
ind = (stock\_data51['return'] >= -0.1) & (stock\_data51['return'] <= 0.1)  
stock\_data51 = stock\_data51.loc[ind, :]  
plt.plot(stock\_data51['return'].values)  
plt.subplot(2,2,3)  
stock\_data52 = stock\_data[stock\_data['code'] == stk\_codes[2]]  
stock\_data52['date'] = pd.to\_datetime(stock\_data52['date'])  
stock\_data52.sort\_values(by=['date'], inplace=True)  
stock\_data52['return'] = np.log(stock\_data52['close']) - np.log(stock\_data52['close'].shift(periods=1))  
stock\_data52.dropna(inplace=True)  
ind = (stock\_data52['return'] >= -0.1) & (stock\_data52['return'] <= 0.1)  
stock\_data52 = stock\_data52.loc[ind, :]  
plt.plot(stock\_data52['return'].values)  
plt.subplot(2,2,4)  
stock\_data53 = stock\_data[stock\_data['code'] == stk\_codes[3]]  
stock\_data53['date'] = pd.to\_datetime(stock\_data53['date'])  
stock\_data53.sort\_values(by=['date'], inplace=True)  
stock\_data53['return'] = np.log(stock\_data53['close']) - np.log(stock\_data53['close'].shift(periods=1))  
stock\_data53.dropna(inplace=True)  
ind = (stock\_data53['return'] >= -0.1) & (stock\_data53['return'] <= 0.1)  
stock\_data53 = stock\_data53.loc[ind, :]  
plt.plot(stock\_data53['return'].values)  
# plt.show()  
  
plt.figure(3)  
plt.subplot(2,2,1)  
stock\_data54 = stock\_data[stock\_data['code'] == stk\_codes[4]]  
stock\_data54['date'] = pd.to\_datetime(stock\_data54['date'])  
stock\_data54.sort\_values(by=['date'], inplace=True)  
stock\_data54['return'] = np.log(stock\_data54['close']) - np.log(stock\_data54['close'].shift(periods=1))  
stock\_data54.dropna(inplace=True)  
ind = (stock\_data54['return'] >= -0.1) & (stock\_data54['return'] <= 0.1)  
stock\_data54 = stock\_data54.loc[ind, :]  
plt.plot(stock\_data54['return'].values)  
  
plt.subplot(2,2,2)  
stock\_data55 = stock\_data[stock\_data['code'] == stk\_codes[5]]  
stock\_data55['date'] = pd.to\_datetime(stock\_data55['date'])  
stock\_data55.sort\_values(by=['date'], inplace=True)  
stock\_data55['return'] = np.log(stock\_data55['close']) - np.log(stock\_data55['close'].shift(periods=1))  
stock\_data55.dropna(inplace=True)  
ind = (stock\_data55['return'] >= -0.1) & (stock\_data55['return'] <= 0.1)  
stock\_data55 = stock\_data55.loc[ind, :]  
plt.plot(stock\_data55['return'].values)  
  
plt.subplot(2,2,3)  
stock\_data56 = stock\_data[stock\_data['code'] == stk\_codes[6]]  
stock\_data56['date'] = pd.to\_datetime(stock\_data56['date'])  
stock\_data56.sort\_values(by=['date'], inplace=True)  
stock\_data56['return'] = np.log(stock\_data56['close']) - np.log(stock\_data56['close'].shift(periods=1))  
stock\_data56.dropna(inplace=True)  
ind = (stock\_data56['return'] >= -0.1) & (stock\_data56['return'] <= 0.1)  
stock\_data56 = stock\_data56.loc[ind, :]  
plt.plot(stock\_data56['return'].values)  
  
plt.subplot(2,2,4)  
stock\_data57 = stock\_data[stock\_data['code'] == stk\_codes[7]]  
stock\_data57['date'] = pd.to\_datetime(stock\_data57['date'])  
stock\_data57.sort\_values(by=['date'], inplace=True)  
stock\_data57['return'] = np.log(stock\_data57['close']) - np.log(stock\_data57['close'].shift(periods=1))  
stock\_data57.dropna(inplace=True)  
ind = (stock\_data57['return'] >= -0.1) & (stock\_data57['return'] <= 0.1)  
stock\_data57 = stock\_data57.loc[ind, :]  
plt.plot(stock\_data57['return'].values)  
# plt.show()  
  
stock1= stock\_data[stock\_data['code'] == stk\_codes[7]]  
stock1.dropna(inplace=True)  
stock1['date'] = pd.to\_datetime(stock1['date'])  
stock1['return'] = np.log(stock1['close']) - np.log(stock1['close'].shift(periods=1))  
stock1.dropna(inplace=True)  
ind = (stock1['return'] >= -0.1) & (stock1['return'] <= 0.1)  
stock1 = stock1.loc[ind, :]  
print(stock1)  
  
index=pd.read\_csv('300month.csv',encoding='GB2312')  
index.columns=['code','date','close']  
index.index=pd.to\_datetime(index['date'])  
index.dropna(inplace=True)  
indexm=index.resample('BM').last()#取营业日最后一日记得把index转为datetime  
indexm['date'] = pd.to\_datetime(indexm['date'])  
indexm['return'] = np.log(indexm['close']) - np.log(indexm['close'].shift(periods=1))  
indexm.dropna(inplace=True)  
ind = (indexm['return'] >= -0.1) & (indexm['return'] <= 0.1)  
indexm = indexm.loc[ind, :]  
indexm.columns=['code','date','close','return']  
#  
data\_matrix = pd.merge(left=indexm[[ 'return']],#index已经是date改成日期  
 right=stock1[['date','return']],  
 on='date',  
 how='inner',  
 sort=True)  
  
#  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data50[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data51[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data52[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data53[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data54[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data55[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data56[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
data\_matrix = pd.merge(left=data\_matrix,  
 right=stock\_data57[['date', 'return']],  
 on='date',  
 how='inner',  
 sort=True)  
ret\_rf = stock\_data[['date', 'rfreturn']]  
ret\_rf.dropna(inplace=True)  
ret\_rf.drop\_duplicates(inplace=True, subset=['date'])  
# subset : column label or sequence of labels, optional用来指定特定的列，默认所有列  
ret\_rf.sort\_values(by=['date'], inplace=True)  
ret\_rf['date'] = pd.to\_datetime(ret\_rf['date'])  
data\_matrix = pd.merge(left=data\_matrix,  
 right=ret\_rf,  
 on='date',  
 how='inner',  
 sort=True)  
data\_matrix.drop\_duplicates(inplace=True, subset=['date'])  
data\_matrix.reset\_index(inplace=True,drop=True)  
data\_matrix.columns = ['date', 'ind', 'stk1', 'stk2', 'stk3', 'stk4', 'stk5', 'stk6', 'stk7','stk8','stk9','rf']  
data\_matrix['ind'] = data\_matrix['ind'] - data\_matrix['rf']  
data\_matrix['stk1'] = data\_matrix['stk1'] - data\_matrix['rf']  
data\_matrix['stk2'] = data\_matrix['stk2'] - data\_matrix['rf']  
data\_matrix['stk3'] = data\_matrix['stk3'] - data\_matrix['rf']  
data\_matrix['stk4'] = data\_matrix['stk4'] - data\_matrix['rf']  
data\_matrix['stk5'] = data\_matrix['stk5'] - data\_matrix['rf']  
data\_matrix['stk6'] = data\_matrix['stk6'] - data\_matrix['rf']  
data\_matrix['stk7'] = data\_matrix['stk7'] - data\_matrix['rf']  
data\_matrix['stk8'] = data\_matrix['stk8'] - data\_matrix['rf']  
data\_matrix['stk9'] = data\_matrix['stk9'] - data\_matrix['rf']  
print(data\_matrix)  
ret\_ind = data\_matrix['ind'].values#指数超额收益  
T = len(ret\_ind)#数据长度  
N = 8  
mu\_market = np.mean(ret\_ind)#均值  
sigma\_market = np.sum((ret\_ind - mu\_market)\*\*2)/T#sigma公式  
ret\_stocks = data\_matrix[['stk1', 'stk2', 'stk3', 'stk4', 'stk5', 'stk6','stk7','stk8','stk9']].values  
x = sm.add\_constant(ret\_ind)  
y = ret\_stocks[:, 3]  
model = sm.OLS(y,x)  
results = model.fit()  
print(results.summary())  
#无限制模型参数估计  
x = np.ones((T, 2))#定义X一列是1一列是指数  
x[:, 1] = ret\_ind#收益率指数  
y = ret\_stocks#股票收益率  
xTx = np.dot(np.transpose(x), x)#x转制乘以x  
xTy = np.dot(np.transpose(x), y)#x转制乘以y  
AB\_hat = np.dot(np.linalg.inv(xTx), xTy)  
ALPHA = AB\_hat[0]#第一位  
print(ALPHA)  
BETA = AB\_hat[1]#第二位  
RESD = y - np.dot(x, AB\_hat)  
COV = np.dot(np.transpose(RESD), RESD)/T#斜方差  
invCOV = np.linalg.inv(COV)  
# 限制模型参数估计#a=0  
xr = np.ones((T, 1))  
xr[:, 0] = ret\_ind  
yr = ret\_stocks  
xrTxr = np.dot(np.transpose(xr), xr)  
xrTyr = np.dot(np.transpose(xr), yr)  
ABr\_hat = np.dot(np.linalg.inv(xrTxr), xrTyr)  
RESDr = yr - np.dot(xr, ABr\_hat)  
COVr = np.dot(np.transpose(RESDr), RESDr)/T  
invCOVr = np.linalg.inv(COVr)  
trans\_ALPHA = np.ones((len(ALPHA), 1))#a转置  
trans\_ALPHA[:, 0] = ALPHA  
#卡方分布#wald检验  
SWchi2 = T\*(1/(1+mu\_market\*\*2/sigma\_market))\*np.dot(np.dot(ALPHA, invCOV), trans\_ALPHA)  
SWF = (T-N-1)/N\*(1/(1+mu\_market\*\*2/sigma\_market))\*np.dot(np.dot(ALPHA, invCOV), trans\_ALPHA)  
pvalue\_Wchi2 = 1 - chi2.cdf(SWchi2[0], N)  
pvalue\_WF = 1 - f.cdf(SWF[0], N, T-N-1)#累计概率  
print(pvalue\_Wchi2)  
print(pvalue\_WF)  
SLRchi2 = T\*(np.log(np.linalg.det(COVr)) - np.log(np.linalg.det(COV)))  
pvalue\_SLRchi2 = 1 - chi2.cdf(SLRchi2, N)  
print(pvalue\_SLRchi2)  
  
#似然检  
a = np.zeros((9, 1))  
a[:, 0] = np.sum(RESDr, axis=0)#限制模型残差相  
salpha = np.dot(invCOVr, a)  
b = np.dot(ret\_ind, RESDr)  
sbeta = np.zeros((9, 1))  
sbeta[:, 0] = np.dot(invCOVr, b)  
score = np.concatenate((salpha, sbeta), axis=0)#得分函数  
  
a = np.concatenate((invCOVr\*T, invCOVr\*np.sum(ret\_ind)), axis=1)  
b = np.concatenate((invCOVr\*np.sum(ret\_ind), invCOVr\*np.sum(ret\_ind\*\*2)), axis=1)  
Minfo = np.concatenate((a, b), axis=0)#信息矩阵  
SLMchi2 = np.dot(np.dot(np.transpose(score), np.linalg.inv(Minfo)), score)  
pvalue\_SLMchi2 = 1 - chi2.cdf(SLMchi2[0][0], N)  
print(pvalue\_SLMchi2)  
  
print('{:>10s}, {:>10s}, {:>10s}, {:>10s}'.format('Wald Test1', 'Wald Test2', 'LR Test', 'LM Test'))  
print('{:10.5f}, {:10.5f}, {:10.5f}, {:10.5f}'.format(SWchi2[0], SWF[0], SLRchi2, SLMchi2[0][0]))  
print('{:10.5f}, {:10.5f}, {:10.5f}, {:10.5f}'.format(pvalue\_Wchi2, pvalue\_WF, pvalue\_SLRchi2, pvalue\_SLMchi2))  
  
# # 读取股票数据  
matrix = stock\_data50  
matrix2= stock\_data51  
matrix3= stock\_data52  
matrix4= stock\_data53  
matrix5= stock\_data54  
print(matrix)  
print(matrix2)  
print(matrix3)  
print(matrix4)  
print(matrix5)  
matrix=pd.merge(left=matrix[['date','return']],right=matrix2[['date','return']],on='date',how='inner',sort=True)  
matrix.columns=['date','stk1','stk2']  
matrix=pd.merge(left=matrix,right=matrix3[['date','return']],on='date',how='inner',sort=True)  
matrix.columns=['date','stk1','stk2','stk3']  
matrix=pd.merge(left=matrix,right=matrix4[['date','return']],on='date',how='inner',sort=True)  
matrix.columns=['date','stk1','stk2','stk3','stk4']  
matrix=pd.merge(left=matrix,right=matrix5[['date','return']],on='date',how='inner',sort=True)  
matrix.columns=['date','stk1','stk2','stk3','stk4','stk5']  
print(matrix)  
returns\_df = matrix  
  
# 定义参数  
N\_values = [1, 3, 6, 12]  
M\_values = [1, 3, 6, 12]  
num\_portfolios = 5  
  
# 进行实证检验  
results = {}  
for N in N\_values:  
 for M in M\_values:  
 momentum\_portfolio\_returns = []  
 for t in range(N + M, len(returns\_df) - M):  
 past\_returns = returns\_df.iloc[t - N:t].sum()  
 past\_returns\_sorted = past\_returns.sort\_values()  
 portfolio\_returns = []  
 group\_size = len(past\_returns\_sorted) // num\_portfolios  
 for p in range(num\_portfolios):  
 group = past\_returns\_sorted.iloc[p \* group\_size: (p + 1) \* group\_size]  
 group\_return = returns\_df.iloc[t + 1:t + M + 1][group.index].mean(axis=1).sum()  
 portfolio\_returns.append(group\_return)  
 momentum\_portfolio\_returns.append(portfolio\_returns)  
 results[(N, M)] = pd.DataFrame(momentum\_portfolio\_returns, columns=[f'P{i+1}' for i in range(num\_portfolios)])  
  
# 输出结果  
for key, value in results.items():  
 print(f"\nN = {key[0]}, M = {key[1]}")  
 sorted\_mean\_returns = value.mean().sort\_values(ascending=False)  
 print(sorted\_mean\_returns)  
  
 # 判断是惯性效应还是反转效应  
 if sorted\_mean\_returns.index[-1] == 'P1' and sorted\_mean\_returns.index[0] == 'P5':  
 print('Market exhibits a momentum effect.')  
 elif sorted\_mean\_returns.index[-1] == 'P5' and sorted\_mean\_returns.index[0] == 'P1':  
 print('Market exhibits a reversal effect.')  
 else:  
 print('Market does not exhibit a clear momentum or reversal effect.')  
  
def plot\_results(N, M, results):  
 data = results[(N, M)].mean()  
 data.plot(kind='bar')  
 plt.title(f"N = {N}, M = {M}")  
 plt.xlabel('Portfolio')  
 plt.ylabel('Average Return')  
 plt.show()  
  
for N in N\_values:  
 for M in M\_values:  
 plot\_results(N, M, results)

图表, 散点图

描述已自动生成图表, 散点图

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

参数八个

[[-0.01749214 -0.01464767 0.00134807 0.00621282 -0.00874138 0.01266745

0.00316747 -0.013634 ]

[ 0.79282827 1.12014137 0.2117188 0.16506766 0.61127784 -0.06787655

0.43113959 0.70651723]]

[-0.01749214 -0.01464767 0.00134807 0.00621282 -0.00874138 0.01266745

0.00316747 -0.013634 ]

卡方分布检验的 P 值= 0.07648991644906711

Wald 检验的 P 值= 0.8990734189851131

LR 检验的 P 值= 0.3320965421241103

LR 检验的 P 值= 0.6248361358833479

Wald Test1, Wald Test2, LR Test, LM Test

14.20851, 0.32292, 9.12215, 6.20003

0.07649, 0.89907, 0.33210, 0.62484

[178 rows x 5 columns]

date stk1 stk2 stk3 stk4 stk5

0 2000-02-29 -0.063087 0.043711 0.083332 -0.007463 -0.011398

1 2000-04-28 -0.013070 -0.024232 -0.043117 -0.029886 0.036348

2 2000-10-31 -0.011567 0.029510 -0.053714 0.069892 0.034196

3 2000-12-29 -0.066884 -0.010425 0.052644 0.043894 -0.064668

4 2001-02-28 0.009732 -0.053135 -0.032102 0.084481 -0.061251

5 2001-06-29 -0.028866 0.063605 0.038715 -0.058173 -0.056505

6 2001-08-31 0.034476 -0.055936 -0.081106 -0.036248 -0.084451

7 2001-11-30 0.062079 0.055620 0.018028 -0.013673 -0.028647

8 2002-07-31 -0.048518 -0.061196 -0.001832 -0.024130 0.000666

9 2002-08-30 -0.001631 -0.029559 0.057600 -0.027512 0.025651

10 2002-10-31 -0.066101 -0.036905 -0.079893 -0.023025 -0.061397

11 2003-02-28 0.006198 0.011696 0.019098 0.041090 -0.031444

12 2003-05-30 0.082203 0.046340 -0.024601 0.034924 0.009608

13 2003-07-31 -0.045120 -0.025318 -0.063044 -0.046520 -0.044247

14 2003-11-28 -0.054210 0.069819 -0.026492 0.076755 0.029058

15 2004-09-30 0.046968 0.092475 0.019944 -0.012552 0.000000

16 2006-02-28 0.070725 0.010695 -0.008265 0.015666 0.074333

17 2009-04-30 -0.054927 0.074324 0.018692 -0.004684 0.023560

18 2010-12-31 -0.035293 0.010471 -0.035121 0.044390 -0.037295

19 2011-10-31 0.041359 -0.052751 -0.014297 0.000000 0.051044

20 2012-09-28 -0.022473 0.010038 0.012922 0.000000 -0.077638

21 2013-11-29 0.055229 0.097321 0.010840 0.021471 -0.021819

22 2014-04-30 -0.023430 -0.081856 -0.022045 0.071500 0.033778

23 2015-02-27 0.015969 0.011583 -0.029296 -0.063946 0.004298

24 2016-08-31 -0.059566 0.012180 0.025009 0.010110 0.031035

25 2016-10-31 0.012308 0.025632 -0.051774 0.049425 0.008782

26 2016-11-30 0.025165 0.029757 0.022912 0.002535 0.042787

27 2016-12-30 -0.076405 0.009464 -0.088922 0.024178 -0.048267

28 2017-01-26 -0.067691 0.061640 -0.015212 -0.014102 0.024961

29 2017-02-28 0.005724 0.016106 -0.019190 0.052087 0.015949

30 2017-03-31 -0.002286 -0.057543 0.013040 0.042692 -0.033247

31 2017-06-30 0.026446 0.045863 0.050956 0.023942 0.020442

32 2017-12-29 0.031343 -0.071730 0.043890 0.008110 -0.005997

33 2018-01-31 -0.055211 0.014764 -0.073996 -0.081592 0.054858

34 2018-05-31 -0.074236 -0.014014 0.019446 -0.068188 -0.063740

35 2020-12-31 -0.068468 0.037882 0.093218 -0.066691 -0.020472

36 2021-02-26 -0.053843 0.008210 -0.002561 -0.023759 -0.076944

37 2021-04-30 -0.022076 -0.019894 -0.073755 -0.051469 0.056527

38 2021-05-31 0.008889 -0.087995 0.048202 0.071884 0.038329

date code return

0 2000-02-29 stk1 -0.063087

1 2000-04-28 stk1 -0.013070

2 2000-10-31 stk1 -0.011567

3 2000-12-29 stk1 -0.066884

4 2001-02-28 stk1 0.009732

.. ... ... ...

190 2018-05-31 stk5 -0.063740

191 2020-12-31 stk5 -0.020472

192 2021-02-26 stk5 -0.076944

193 2021-04-30 stk5 0.056527

194 2021-05-31 stk5 0.038329

[195 rows x 3 columns]

date code return ret\_1m ... ret\_6m ret\_12m rank group

0 2000-02-29 stk1 -0.063087 NaN ... NaN NaN 0.0 0

1 2000-04-28 stk1 -0.013070 -0.792822 ... NaN NaN 111.0 0

2 2000-10-31 stk1 -0.011567 -0.114992 ... NaN NaN 141.0 0

3 2000-12-29 stk1 -0.066884 4.782140 ... NaN NaN 321.0 2

4 2001-02-28 stk1 0.009732 -1.145513 ... NaN NaN 155.0 0

.. ... ... ... ... ... ... ... ... ...

190 2018-05-31 stk5 -0.063740 -2.161903 ... -3.553627 -2.887043 109.0 0

191 2020-12-31 stk5 -0.020472 -0.678828 ... -2.283539 -5.763039 325.0 2

192 2021-02-26 stk5 -0.076944 2.758569 ... 1.314303 -3.479246 371.0 3

193 2021-04-30 stk5 0.056527 -1.734656 ... 1.765273 5.436992 390.0 4

194 2021-05-31 stk5 0.038329 -0.321945 ... -7.391266 -0.104207 275.0 1

[195 rows x 9 columns]

group date return ... cum\_return\_6m cum\_return\_12m cum\_return\_all

0 0 2000-02-29 0.009019 ... NaN NaN 0.009019

1 0 2000-04-30 -0.014791 ... NaN NaN -0.005906

2 0 2000-10-31 0.013664 ... NaN NaN 0.007677

3 0 2000-12-31 0.028704 ... NaN NaN 0.036602

4 0 2001-02-28 -0.025759 ... NaN NaN 0.009900

.. ... ... ... ... ... ... ...

113 4 2017-12-31 0.027781 ... NaN NaN -0.470821

114 4 2018-05-31 -0.074236 ... NaN NaN -0.510105

115 4 2021-02-28 -0.022817 ... NaN NaN -0.521283

116 4 2021-04-30 0.056527 ... NaN NaN -0.494222

117 4 2021-05-31 0.048202 ... NaN NaN -0.469842

[118 rows x 8 columns]

市场存在惯性效应