1. 经济是一个动态的充满生机的网络，网络中各个节点会维持持续动态更新、删减和添加；一种新产品的上市，是不是能经受住市场的考验，是不是能被消费认可；一种游戏，上市能不能流行起来，都是要经过现实数据的训练筛选，没有历史数据，只能实时产生，并迅速反馈，每一次运行都即是train也是test；消费反馈修改生产权重，更新经济结构，这也是历史见证行业企业兴衰交替，更迭发展过程的原因；
2. 比如我们最近提到的一路一带，这会在全球经济网络产生总体的影响，加入很多新节点，形成很多新路径，并且对原有路径节点的权重形成影响并更新；然后这些节点也会接受全球经济环境的“经济数据”训练，进行资源配置的寻优；寻优的目标函数：人尽其才，无尽其用；从而使得世界更快更好的发展；全球经济网中，著名的波罗的海指数（BDI）
3. Bug也是很明显的，现实运行中调节和反馈系统相当复杂，我们所能接收到的反馈有限的，还有很多隐含的或者我们不可见的反馈在运行，我们仅仅对我们可见的反馈做出回应，这种回应的不全面性，将会产生错误，当这种错误增大到一定程度时，就会产生一定程度的崩溃，就是所谓的经济危机；经济危机将是对网络硬性强力修改；
4. 交易：一级市场和二级市场；
   1. 一级市场：我们最熟悉的“打新”应该属于参与一级市场行为，是一级市场的最后环节，和IPO对应（即上市公司通过券商发起IPO首次公开发行卖出股票，股民通过打新股买入这些股票）；
   2. 其它的一级市场行为：一般包括天使投资，VC/PE等风险股权投资。每次投资都会增加公司的资本且一般发生在ipo前。
   3. 个人也参与一级市场，但是需要的资金量大，有较好的风险承担能力和金融知识；
   4. 二级市场：由持股股东自由交易转让股票的交易属于二级市场。卖家是投资者，买家也是投资者。一般的股票交易就在此列。实际上，二级市场的交易无法对上市公司本身的资本造成任何影响，是一种权益转让，影响的只有持股人；一般发生在ipo后。
   5. 增发可算作一级半市场把，是公司发起IPO后增加资本的主要手段。

从生产到消费过程每一级的彼此连接，都是通过交易完成，并且由此建立反馈修正基础（类似正向传播中的a(1),a(2),...）；接下来，我们进入经济的微观结构，交易；什么是交易？

价差：商品质量和价值预期；商品质量是基于商品的实用性进行的评估，从而影响价差；当我们讨论股票等投资标的的时候，它们并不具有这种实在的实用性，所以影响价差的因素仅仅是价值预期；另外投资标的的买方和卖方对买卖具有更高的统一性，即一个市场上的交易者即是买方，也是卖方；所以买卖报价也都是对等的；

1. 1. 交易基本概念，摆单拍单；冰山一角
   1. 买和卖 是相当模糊的概念，也仅仅对我们大陆A股市场适应；实际“买”和“卖”代表的是方向，即做多标的和做空标的；
2. 2. 投资交易市场
3. 3. 量化基金概况
4. 4. 市场分析--交易优势，T+0，开盘时卖掉，收盘时买回，ai增加交易优势；联系赌场；交易优势，阈值，达到一定程度激活；

胜率：并不是寻找一种方法 以至于百发百中；而是寻求一个相对稳定的概率，在相当长的时间里，赢面可以持续稳定超越亏损；

1. 数据获取
   1. 打开交易软件解释一下；
   2. 上证指数获取作图
2. **import** tushare as ts
3. **import** matplotlib.pyplot as plt
4. **import** numpy as np
5. **from** pylab **import** \*
6. mpl.rcParams['font.sans-serif'] = ['SimHei']
7. mpl.rcParams['axes.unicode\_minus'] = False
9. data=ts.get\_k\_data('000001',ktype='D',autype=None,index=True,start='2016-01-01',end='2017-12-10')
10. fig=plt.figure()
11. ax=fig.add\_subplot(1,1,1)
12. ax.plot(data['close'])
13. #ax.plot(data['high'],'r\*')
14. dates=data['date']
15. n\_k=len(dates)
16. xindex=list(map(int,np.linspace(0,n\_k-1,12)))
17. ax.set\_xticks(xindex)
18. ax.set\_xticklabels(dates[xindex],rotation=60)
19. ax.grid()
20. ax.set\_title('000001')
21. ax.set\_xlabel('日期')
22. ax.set\_xlabel('价格')
    1. 平安银行数据获取作图
       1. Index=False
    2. 浦发银行
       1. 展示是否复权，分钟线；

data=ts.get\_k\_data('600000',ktype='D',autype=None,index=False,start='2016-01-01',end='2017-12-10')

ax.plot(data['close'],'k')

让学生自己下载数据，进行调试参数，五分钟，是否复权，指数等；

df = ts.get\_stock\_basics()

df[df[‘name’]==’贵州茅台’].index[0]

Note: 没有市净率，市盈率，总市值的历史值，如何获取？

市盈/净率：每股股价/每股收益（每股净资产） ts.get\_k\_data, ts.get\_report\_data(2014,3)中的esp和bvps

总市值：每股股价\*总股本;ts.get\_k\_data,ts.get\_stock\_basics()中的totals，单位是（亿股）

df = ts.get\_realtime\_quotes('000581')

实时监测某个/几个指标是否到达某个值；

1. **from** pylab **import** \*
2. **import** tushare as ts
3. **import** tkinter as tk
4. **import** time
6. mpl.rcParams['font.sans-serif'] = ['SimHei']
7. mpl.rcParams['axes.unicode\_minus'] = False
9. targetPrice=3.4
10. fig=0
11. **while** 1:
12. **try**:
13. time.sleep(1)
14. tem=float(ts.get\_realtime\_quotes('600519')['price'])
15. **if** tem>targetPrice:
16. fig=1
17. **break**
18. **else**:
19. **print**('还有 '+str(abs(tem-targetPrice))+ ' 个点，才能到达目标价位。')
20. **except**:
21. **print**('用户退出!')
22. **break**
23. **if** fig:
24. window = tk.Tk()
25. window.title('Warning')
26. window.geometry('300x200')
27. #        var = tk.StringVar()
28. l = tk.Label(window,bg='yellow',width=50,text='get target',font = 'SimHei -30 bold')
29. l.pack()
30. window.mainloop()
31. 数据可视化

Matplotlib.finance标准格式

1. **import** matplotlib.pyplot as plt
2. **import** matplotlib.finance as mpf
3. **from** matplotlib.pylab **import** date2num
4. **import** tushare as ts
5. #import numpy as np
6. **import** datetime
8. data=ts.get\_k\_data('600519',ktype='D',autype='qfq',start='2017-09-12',end='2017-12-11')
9. prices=data[['open','high','low','close']]
10. dates=data['date']
11. dates=[date2num(datetime.datetime.strptime(x, '%Y-%m-%d')) **for** x **in** dates]
12. candleData=np.column\_stack([dates,prices])
14. plt.figure(figsize=(12,8))
15. ax=plt.subplot()
16. mpf.candlestick\_ohlc(ax,candleData,width=0.5,colorup='r',colordown='g')
17. ax.xaxis\_date()
18. ax.grid()
19. plt.xticks(rotation=60)

修改一下，自定义x轴

1. **import** matplotlib.pyplot as plt
2. **import** matplotlib.finance as mpf
3. #from matplotlib.pylab import date2num
4. **import** tushare as ts
5. **import** numpy as np
6. #import datetime
8. data=ts.get\_k\_data('600000',ktype='D',autype='qfq',start='2017-10-12',end='2017-12-11')
9. prices=data[['open','high','low','close']]
10. dates=data['date']
11. candleData=np.column\_stack([list(range(len(dates))),prices])
13. plt.figure(figsize=(12,8))
14. ax=plt.subplot()
15. mpf.candlestick\_ohlc(ax,candleData,width=0.5,colorup='r',colordown='g')
16. xindex=list(map(int,np.linspace(0,len(dates)-1,12)))
17. ax.set\_xticks(xindex)
18. ax.set\_xticklabels(dates[dates.index[xindex]],rotation=60)
19. #ax.set\_xticklabels(dates[xticks])
20. ax.grid()

添加成交量及均线并标记点

1. **import** matplotlib.pyplot as plt
2. **import** matplotlib.finance as mpf
3. #from matplotlib.pylab import date2num
4. **import** tushare as ts
5. **import** numpy as np
6. #import datetime
8. data=ts.get\_k\_data('600519',ktype='D',autype='qfq',start='2017-09-12',end='2017-12-11')
9. prices=data[['open','high','low','close']]
10. dates=data['date']
11. candleData=np.column\_stack([list(range(len(dates))),prices])
13. fig=plt.figure(figsize=(12,8))
14. ax=fig.add\_axes([0.1,0.3,0.8,0.6])
15. #ax=plt.subplot(3,1,1)
16. mpf.candlestick\_ohlc(ax,candleData,width=0.5,colorup='r',colordown='g')
17. ma5=[np.mean(data['close'][i-4:i+1]) **for** i **in** range(4,len(dates))]
18. ax.plot(range(4,len(dates)),ma5)
19. ma10=[np.mean(data['close'][i-9:i+1]) **for** i **in** range(9,len(dates))]
20. ax.plot(range(9,len(dates)),ma10)
22. tmp=[True **if** (ma5[i+5]-ma10[i])\*(ma5[i+4]-ma10[i-1])<0 **else** False **for** i **in** range(1,len(ma10))]
23. tmp=np.array(range(10,len(dates)))[tmp]
24. ax.plot(tmp,np.array(data['high'])[tmp],'k\*',linewidth=12)
26. xindex=list(map(int,np.linspace(0,len(dates)-1,12)))
27. ax.set\_xticks(xindex)
28. ax.grid()
30. ax1=fig.add\_axes([0.1,0.1,0.8,0.2])
31. ax1.bar(range(len(dates)),data['volume'])
32. ax1.set\_xticks(xindex)
33. ax1.set\_xticklabels(dates[dates.index[xindex]],rotation=60)
34. #ax.set\_xticklabels(dates[xticks])
35. ax1.grid()
36. 量化策略
    1. 收盘卖掉，开盘买入；

T+1

1. **from** pylab **import** \*
2. **import** matplotlib.pyplot as plt
3. **import** matplotlib.finance as mpf
4. **import** tushare as ts
5. **import** numpy as np
7. mpl.rcParams['font.sans-serif']=['SimHei']
8. mpl.rcParams['axes.unicode\_minus']=False
10. stock='000005'
11. data=ts.get\_k\_data(stock,ktype='D',autype='qfq',index=False,start='2010-09-12',end='2017-12-11')
12. closes=np.array(data['close'])
13. opens=np.array(data['open'])
14. diff=1-opens[1:]/closes[:-1]#-0.0013
15. tmp=[ True **if**  abs(closes[i]/closes[i-1]-1)>=0.094 **else** False **for** i **in** range(1,len(closes))]
16. diff[tmp]=0
18. tmp=[ True **if**  closes[i]/closes[i-1]-1>=0.0 **else** False **for** i **in** range(1,len(closes))]
19. diff[tmp]=0
21. #diff=[]
22. #for i in range(1,len(closes)):
23. #    diff.append(1-opens[i]/closes[i-1])
24. #diff=np.array(diff)
26. plt.figure(figsize=(10,6))
27. ax=plt.subplot()
28. ax.plot(diff.cumsum(),label='价差套利')
29. diff1=closes[1:]/closes[:-1]-1
30. ax.plot(diff1.cumsum(),label='正常走势')
31. dates=data['date']
32. xindex=list(map(int,np.linspace(0,len(dates)-1,6)))
33. ax.set\_xticks(xindex)
34. ax.set\_xticklabels(dates[dates.index[xindex]],rotation=30)
35. plt.legend(prop={'size':10})
36. plt.title(stock)
37. plt.grid()

Question1: 真的这么好吗？

Answer1：需要去除手续费；

这只是一个交易优势，而且要应用是有条件的，这一个手续费可以由多个交易优势分摊，或者筛选对个交易优势大的个股；