根据基金发售力度基金买卖策略

2019023136 刘嗣馨

背景

本策略源自阅读论文《Do Measures of Investor Sentiment Predict Returns?》——Robert Neal and Simon M. Wheatley (1998)，作者对世俗经验“在人们投资情绪高涨时应该卖出股票，在人们投资情绪落入低谷时买入股票”进行验证，结合数据对封闭基金折价、基金赎回、散户交易情况指标进行检验，验证它们是否能作为预测股价涨跌的有效指标。 作者的研究中发现，能够提供信息来预测规模溢价的因子是小公司股票的均价，它们之间存在正相关性。同时发现净赎回对预测也有着经济上的合理解释和统计上的显著性，但是其标准误过大，无法用于预测。而将作者的结论放到中国市场中，在2015年时市场表现印证了作者的结论，中证1000的溢价远远超过了沪深300。而综合在市场最高点时，小公司的溢价远远超过大公司，预示着股市高点。但是在2020年至今的市场二者却出现了背离现象，中证1000指数在2020年8月就已见顶，随后一路下跌，而沪深300却仍旧一路高涨。究其原因，可能是由于2020年发行基金过多。结合现实解释，疫情也许对人们的投资情绪产生了影响，疫情期间正常的商业活动无法开展，人们在家中隔离，而在此期间，金融市场正常运转，并经历了暴跌暴涨，甚至创出超过2019年12月未受疫情影响前的新高，可能吸引了更多投资者进入金融市场。到6月，股票市场走高，为这部分进场的投资者带来了收益，由此产生了正向效应，更多的投资者开始购入更多的公募基金，也进一步对推高股价产生了正向作用，由此可以解释8月后机构集中持仓的股票表现优于持股更少的股票。 在将沪深300与中证1000背离的现实下，本次策略着眼于基金发首力度对沪深300的影响，并结合中金公司2021年初发布的研报《警惕抱团瓦解》进行探究。

In [73]:
```
#载入包和字体
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
%matplotlib inline
font = FontProperties(fname = r'C:\Windows\Fonts\simhei.ttf')
```

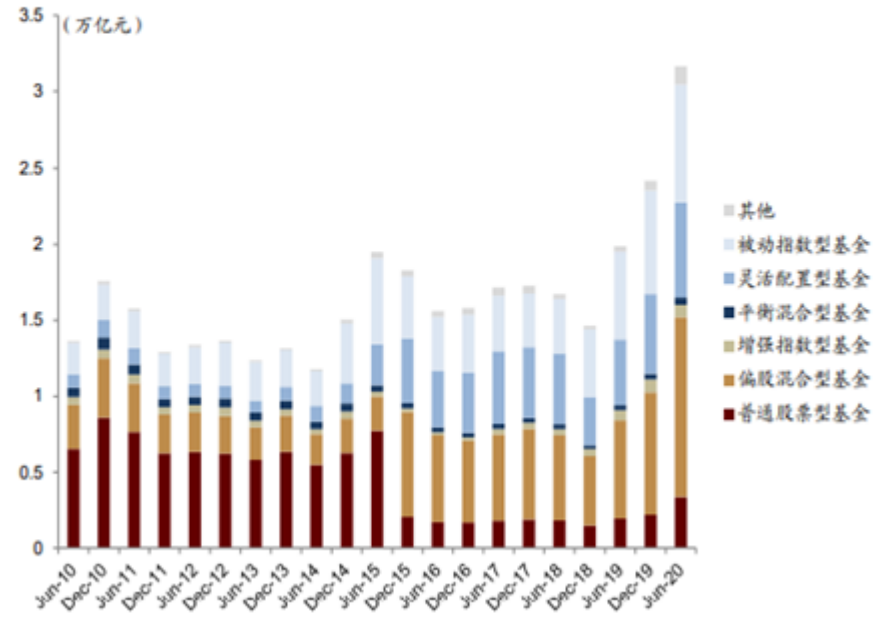
In [74]:
```
#绘图
fig = plt.figure(figsize = (12,6))
df300 = get_market_data(['open', 'close', 'high', 'low', 'volume'], stock_code = ['000300.SH'], start_time = '20150101',
                        end_time = '20210528', period = '1d')
zz1000 = get_market_data(['open', 'close', 'high', 'low', 'volume'], stock_code = ['000852.SH'], start_time = '20150101',
                        end_time = '20210528', period = '1d')
trade_df = pd.DataFrame(index = df300.index)
plt.xticks(list(range(0, len(df300.index), 50)), list(df300.index[np.arange(0, len(df300.index), 50)]), rotation = 90)
plt.grid = ()
plt.plot(np.arange(len(df300)), df300['close'], label = '沪深300')
plt.legend(loc = (.45,.89), frameon = False, prop = font)
plt.twinx()
plt.plot(np.arange(len(zz1000)), zz1000['close'], color = 'orange', label = '中证1000')
plt.legend( loc = (.45, .94), frameon = False, prop = font)
```

Out[74]: <matplotlib.legend.Legend at 0x206f4af1978>

策略思路

结合中金研报中公募基金规模变化进行对沪深300的买卖，在公募基金规模低于均值后买入，在公募基金规模超过均值后卖出

图表 3：公募基金持股规模近年增长迅速，主动规模创历史新高



如图所示，在2010年一月默认买入，2010年12月规模超过均值，在2011年一月第一个交易日卖出。又如在2014年6月，规模低于均值，在7月第一个交易日买入，在2015年7月1日规模增长超过均值时卖出。 默认在空仓期间购入理财产品。并在规模变化的后一个月的第一个交易日执行操作，假设数据的获取的滞后性。

代码部分

本策略中会加入定投与仅购买理财策略，进行对比

In [75]:
```
#数据载入
df300 = get_market_data(['open', 'close', 'high', 'low', 'volume'], stock_code = ['000300.SH'], start_time = '20090101',
                        end_time = '20210528', period = '1d')
#获取索引
trade_df = pd.DataFrame(index = df300.index)
#手动确认买卖点
sell_date = [486, 1598, 2676]
buy_date = [1354, 2432]
```

In [76]:
```
trade_df['money'] = 0
indexes = np.arange(0, len(trade_df.index), 20)              #index设置 20日定投
trade_df.iloc[indexes, 0] = 3000
log_ret = np.diff(np.log(df300['close']))                    #得到每日涨跌波动
log_ret = np.append([1], np.diff(np.log(df300['close'])))    #归一处理
sell_date = [486, 1598, 2676]
buy_date = [1354, 2432]
```

In [77]:
```
#计算定投与理财收益
trade_df['定投累计收益率']=log_ret.cumsum()
print(len(trade_df))
trade_df['基金份额'] = df300['close']/3000
rf = (4.0/100+1)** (1.0/250) -1
trade_df['基金份额'] = trade_df['money']/trade_df['基金份额']
trade_df['总基金份额'] = trade_df['基金份额'].cumsum()
trade_df['总投入资金'] = trade_df['money'].cumsum()
trade_df['指数定投资金'] = trade_df['定投累计收益率'] * trade_df['总基金份额']
trade_df['无风险收益率']= (4.0/100+1)** (1.0/250) -1
trade_df['无风险收益率净值']=(trade_df['无风险收益率']+1).cumprod()
trade_df['理财份额'] = trade_df['money']/trade_df['无风险收益率净值']
trade_df['总理财份额'] = trade_df['理财份额'].cumsum()
trade_df['理财定投资金'] = trade_df['总理财份额'] * trade_df['无风险收益率净值']
```

```
3014
```

In [78]:
```
initial_monry = trade_df.iloc[-1,-2]/(1 + 0.000157) ** 3014                    #对所有策略总投入资金进行贴现，供该策略在开始时使用
trade_df['策略交易总资金'] = initial_monry
trade_df['策略交易净值'] = 1.0
#手动交易
trade_df.iloc[:sell_date[0], -1] = trade_df.iloc[:sell_date[0], 1]
trade_df.iloc[sell_date[0]:buy_date[0], -1] = (np.diff(trade_df.iloc[sell_date[0] - 1:buy_date[0], 7]) + 1) * trade_df.iloc[
trade_df.iloc[buy_date[0]:sell_date[1], -1] = (1 + np.diff(trade_df.iloc[buy_date[0]:sell_date[1] + 1, 1]).cumsum()) * trade
trade_df.iloc[sell_date[1]:buy_date[1], -1] = trade_df.iloc[sell_date[1]:buy_date[1], 7] + trade_df.iloc[sell_date[1] - 1, -
trade_df.iloc[buy_date[1]:sell_date[2], -1] = (1 + np.diff(trade_df.iloc[buy_date[1] - 1:sell_date[2], 1]).cumsum()) * trade
trade_df.iloc[sell_date[2]:, -1] = trade_df.iloc[sell_date[2]:, 7] + trade_df.iloc[sell_date[2] - 1, -1] - trade_df.iloc[sel
newdf = trade_df[['定投累计收益率', '无风险收益率净值', '策略交易净值']]
newdf
```

Out[78]:

	定投累计收益率	无风险收益率净值	策略交易净值
20090105	1.000000	1.000157	1.000000
20090106	1.031285	1.000314	1.031285
20090107	1.025286	1.000471	1.025286
20090108	1.002668	1.000628	1.002668
20090109	1.018626	1.000785	1.018626
20090112	1.019839	1.000942	1.019839
20090113	0.996398	1.001099	0.996398
20090114	1.037668	1.001256	1.037668
20090115	1.037479	1.001413	1.037479
20090116	1.055395	1.001570	1.055395
20090119	1.066513	1.001727	1.066513
20090120	1.072819	1.001884	1.072819
20090121	1.071099	1.002042	1.071099
20090122	1.082333	1.002199	1.082333
20090123	1.076510	1.002356	1.076510
20090202	1.088433	1.002513	1.088433
20090203	1.113326	1.002671	1.113326
20090204	1.140226	1.002828	1.140226
20090205	1.133074	1.002985	1.133074
20090206	1.172416	1.003143	1.172416
20090209	1.198615	1.003300	1.198615
20090210	1.211627	1.003457	1.211627
20090211	1.213512	1.003615	1.213512
20090212	1.208006	1.003772	1.208006
20090213	1.242232	1.003930	1.242232
20090216	1.268231	1.004087	1.268231
20090217	1.236476	1.004245	1.236476
20090218	1.189504	1.004402	1.189504
20090219	1.199373	1.004560	1.199373
20090220	1.219150	1.004718	1.219150
...	...	...	...
20210414	1.972711	1.597269	3.289849
20210415	1.966335	1.597520	3.290100
20210416	1.969806	1.597770	3.290350
20210419	1.993846	1.598021	3.290601
20210420	1.993129	1.598272	3.290852
20210421	1.996149	1.598522	3.291102
20210422	1.994284	1.598773	3.291353
20210423	2.003323	1.599024	3.291604
20210426	1.991922	1.599275	3.291855
20210427	1.994535	1.599526	3.292106
20210428	2.000161	1.599777	3.292357
20210429	2.008899	1.600028	3.292608
20210430	2.000991	1.600279	3.292859
20210506	1.988744	1.600530	3.293110
20210507	1.975803	1.600781	3.293361
20210510	1.975076	1.601032	3.293612
20210511	1.981194	1.601283	3.293863

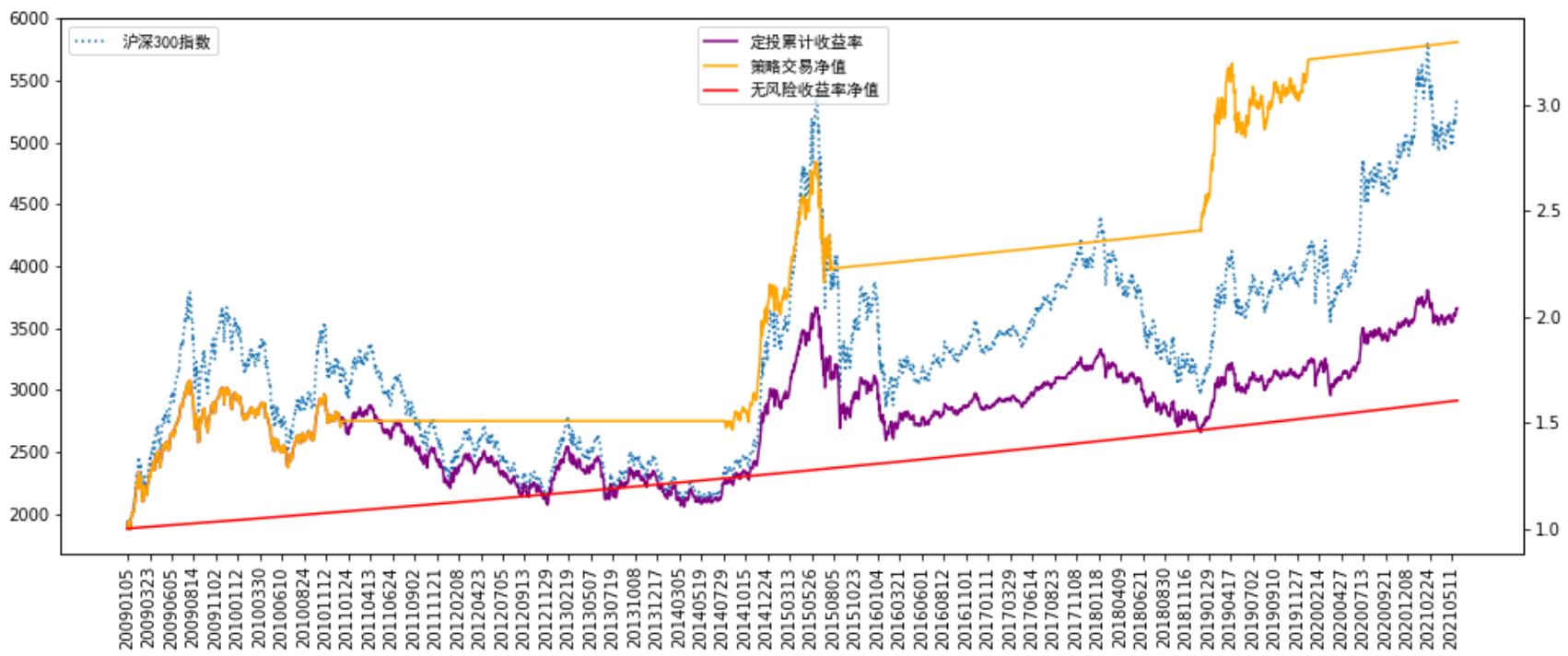
	定投累计收益率	无风险收益率净值	策略交易净值
20210512	1.985463	1.601535	3.294115
20210513	1.975187	1.601786	3.294366
20210514	1.998470	1.602037	3.294617
20210517	2.012922	1.602289	3.294868
20210518	2.013426	1.602540	3.295120
20210519	2.010467	1.602791	3.295371
20210520	2.013197	1.603043	3.295623
20210521	2.003069	1.603294	3.295874
20210524	2.007236	1.603546	3.296126
20210525	2.038342	1.603798	3.296377
20210526	2.038740	1.604049	3.296629
20210527	2.042050	1.604301	3.296881
20210528	2.038833	1.604553	3.297132

3014 rows × 3 columns

策略表现可视化

In [79]:
```
fig = plt.figure(figsize = (16,6))
cmap = plt.get_cmap('Spectral')
plt.xticks(list(np.arange(0, len(trade_df.index), 50)), list(trade_df.index[np.arange(0, len(trade_df.index), 50)]), rotation
plt.plot(np.arange(len(trade_df.index)), df300['close'], linestyle = 'dotted', label = '沪深300指数')
plt.legend(prop = font)
plt.twinx()
plt.plot(np.arange(len(trade_df.index)), trade_df['定投累计收益率'], color = 'purple')
plt.plot(np.arange(len(trade_df.index)), trade_df['策略交易净值'], color = 'orange')
plt.plot(np.arange(len(trade_df.index)), trade_df['无风险收益率净值'], color = 'red')
plt.legend(prop = font, loc = 'upper center')
```

Out[79]: <matplotlib.legend.Legend at 0x206f4c6ada0>



策略评价

In [80]:
```
def get_sharp(value, rf, n):
    df = pd.DataFrame()
    df['net_value'] = value
    m = len(value)
    for i in range(m):
        df.loc[i,'ri'] = round((df.iloc[i].net_value - df.iloc[i-1].net_value)/df.iloc[i-1].net_value, 3)
        df.loc[i,'rp'] = round((df.iloc[i].net_value - df.iloc[0].net_value)/df.iloc[0].net_value, 3)
    sharp = (df['rp'][m-1]/m*n-rf)/(df['ri'].std() * np.sqrt(n))
    return print('夏普比率是：{:.4}'.format(sharp))
print('年化收益率:{:.4}%'.format((pow(3.645158,1/12.5) - 1) * 100))
get_sharp(newdf['策略交易净值'],0.03,3000)
```

年化收益率:10.9%
夏普比率是：2.806

策略回顾与优化

可以看出根据基金规模进行买卖有一定效果，但是在2020年末，改策略提早卖出，可以看出这也是市场变化的影响，导致其错过一大波收益。 可以改进的方向：另取信息获取渠道，更快更新基金规模信息，再根据其信息建立回归模型或机器学习模型，自动判断当前是处于溢价还是折价状态进行自动化买卖。

补充

由于手动设置买卖点，在粘贴代码时，没注意更改了数据日期，导致策略执行有一定偏差，如2010至2014年期间，空仓却未计入理财利率。但策略大致效果相同。原策略效果如下