基础不简单之市盈率

最近，有同事人问市盈率，滚动市盈率的问题。因此，我在这篇文章里做了一个大概的回答，包括它们的意义，它们的获取方法（贴代码），它们的应用等。这篇文章会不断更新，因为这简单的概念的应用太广。

市盈率（Price earnings ratio，即 P/E ratio）也称“本益比”、“股价收益比率”或“市价盈利比率（简称市盈率）”。基本公式 = 股价 / 每股盈余* 100。每股盈余（EPS）又称每股税后利润，指税后利润与股本总数的比率。通俗地说，市盈率代表你以当前该证券的价格购买该证券后，那么以当前证券的每股盈余计算，购买成本返回的年数。例如，每股盈余 5，股价 50 元，那么市盈率 PE 为 10。这也就是说你购买一股，按照当前证券的收益需要经过 10 年才能收 回成本。

由于每股盈余（EPS）涉及财务报表，所以一般我们是按照一年中年初和年末的财务报表计算该值，此时计算出的市盈率也称为“静态市盈率”。但是公司的财务报表分位 4 个季度，每个季度的财务报表都能及时反映出该公司的经营状况。所以，我们有更好的市盈率指标——滚动市盈率 （PE(TTM)）。计算方式就是改变它的起始点，但却始终包括有四个不同的季度(1、2、3、4;2、3、4、1;3、4、1、2;4、1、2、3)。虽然这四个季度有可能属于两个不同的自然年度，但仍然弥补了上市公司季节性的客观差异所造成的影响。可能大家注意到，相隔两个季度之间的滚动市盈率，会出现 3 个季度的重合。这样的方式在一定程度上过滤掉小波动，进而更加客观地反映上市公司的真实情况。所以，很多证券公司选择的指标是滚动市盈率（PE(TTM)）。

现在我们已经知道了概念。可能，大家都猜到了：市盈率越高，代表投资者需要更长时间才能收回成 本，投资者的资金占用率越高。投资者应该不会选这种市盈率高的证券。但是实际的现象则是“啪啪”打脸。在中国证券市场，很多上涨速度快的证券的市盈率都很高，甚至高达 100 多，这意味着投资者需要 100 多年才能收回投资成本。为何？市盈率高代表股价高，风险大，但也意味着投资者看好该证券，认为该证券未来的发展值得投资。所以，市盈率也被看作为市场的情绪指标之一。

那到底是市盈率高好？还是低好？这个指标如何应用？一般地来说，在相同行业中，规模相当，上市时间都比较久的证券之间，比较市盈率是有意义的。市盈率低的证券更有投资价值。那么此时如何筛选最低市盈率的证券呢？接下来，我采用最近用的 **baostock 接口**实现在指定日期从全市场中选择最低市盈率的证券。

若有小伙伴不太了解 **baostock 接口，请参考：**www.baostock.com

具体代码如下：

```
#!/user/bin/env python
# -*- coding:utf-8 -*-
import pandas as pd
import baostock as bs
```

```
# 登陆系统
lg = bs.login()
# 显示登陆返回信息
print('login respond error_code:' + lg.error_code)
print('login respond  error_msg:' + lg.error_msg)

# 获取某一天的全市场的证券和指数代码
rs = bs.query_all_stock(day="2018-06-28")
print('query_all_stock respond error_code:' + rs.error_code)
print('query_all_stock respond  error_msg:' + rs.error_msg)
# 打印结果集
code_list = []
while (rs.error_code == '0') & rs.next():
    # 获取一条记录，将记录合并在一起
    code_list.append(rs.get_row_data()[0])
print(code_list)

df = pd.DataFrame()
# 获取证券的peTTM的历史数据
for code in code_list:
    # 详细指标参数，参见“历史行情指标参数”章节
    rs = bs.query_history_k_data(code,
                                 "date,code,peTTM,pbMRQ,pcfNcfTTM",
                                 start_date='2018-06-28',
end_date='2018-06-28',
                                 frequency="d", adjustflag="3")
    if rs.error_code == '0':
        result = rs.get_data()
        n = result.shape[0]
        if n <= 0:
            continue
        # 删除pe,pcf,pb为0的证券或指数
        if float(result.iloc[0, 2]) != 0:
            if df.empty:
                df = rs.get_data()
            else:
                df = pd.concat([df, rs.get_data()])

# 结果集输出到csv文件
df.to_csv("D:\\history_A_stock_k_data.csv", index=False)

df['peTTM'] = df['peTTM'].astype(float)

print(df)
```

```
# 以peTTM进行升序排序
df_sortby_peTTM = df.sort_values(by='peTTM')

df_sortby_peTTM.to_csv("D:\\history_A_stock_k_data2.csv",
index=False)
print("当天peTTM最小的证券: " + df_sortby_peTTM.iloc[0][1])

# 登出系统
bs.logout()
```

当然，根据我的代码，小伙伴可以修改 code_list 中的证券代码，从而选择同行业，规模相当，上市时间相当的证券代码集合，选择利于投资的最低 peTTM 的证券。这种比较操作的代码等到有小伙伴需要我，给我提出疑问后，我再具体更新代码上来吧。