☐ Get Data ▼ ☐ Analytics ▼ ☐ Classroom ▼ ☐ Research ▼ ☐ Support ▼

Search WRDS

Home / Support / Research Applications / Portfolio Construction and Market Anomalies / Portfolios by S / Size Portfolios for all CRSP Securities (Python)

Size Portfolios for all CRSP Securities (Python)

This sample program has been developed to calculate size portfolios for all CRSP securities. Graphs comparing these calculated monthly value-weighted returns and CRSP portfolio returns are also available for review.

Background

This program calculates size portfolios for all CRSP securities in Python. It computes monthly value-weighted returns per each decile. Returns are weighted by the market capitalization of securities at the end of the previous month. Portfolios are created on December of the previous year. Finally, it compares the portfolio monthly returns (by decile) against CRSP decile returns.

Top of Section

Q

Size Portfolios for all CRSP Securities

```
2
   # Size Portfolio for CRSP Securitie
3
   # July 2018
4
   # Qingyi (Freda) Song Drechsler
5
   6
7
   import pandas as pd
8
   import numpy as np
9
   import datetime as dt
   import wrds
10
11
   from dateutil.relativedelta import *
12
13
   #######################
14
   # Connect to WRDS #
15
   ######################
16
   conn = wrds.Connection()
17
18
   # Get CRSP Monthly Stocks for Decile Formation #
19
20
   msf = conn.raw_sal("""
21
22
                       select a.permno, a.date,
23
                      a.ret, a.shrout, a.prc
                      from crsp.msf as a
24
25
                       where a.date >= '12/01/1999'
                       """, date_cols=['date'])
26
27
28
   # keep only records with non missing ret prc and shrout value
29
   msf = msf[(msf['prc'].notna()) & (msf['ret'].notna()) & (msf['shrout'].notna())]
30
31
   msf['permno'] = msf['permno'].astype(int)
32
   msf['size'] = msf['shrout'] * msf['prc'].abs()
33
   msf['year'] = msf['date'].dt.year
34
   msf['month'] = msf['date'].dt.month
35
36
   # create msf_dec
37
   msf_dec = msf[msf['month']==12][['date','permno','year','size']]
38
39
   # create msf_ls
   msf_ls = msf.sort_values(['permno', 'date'])
40
41
   msf_ls['year_prev'] = msf_ls['year']-1
   msf_ls['size_lag'] = msf_ls.groupby('permno')['size'].shift(1)
42
   msf_ls['size_lag'] = np.where(msf_ls['size_lag'].isna(),\
43
44
    msf_ls['size']/(1+msf_ls['ret']), msf_ls['size_lag'])
45
46
   47
   # Compute Deciles for Each DEC #
48
   mst_dec = mst_dec.sort_values(['year'])
   msf_dec['decile']=1+msf_dec.groupby('year')['size']\
50
    .transform(lambda x: pd.qcut(x, 10, labels=False))
51
52
53
   54
   # Assign Size Group to All Months #
55
   56
   msf_groups = pd.merge(msf_ls[['permno', 'date', 'ret', 'size_lag', 'year_prev']], \
57
                      msf_dec[['permno','year','decile']], how='left', \
58
                      left_on=['permno','year_prev'], right_on=['permno','year'])
59
60
   msf_groups=msf_groups[msf_groups['decile'].notna()]
61
62
   63
   # Compute Size Weighted Returns #
```

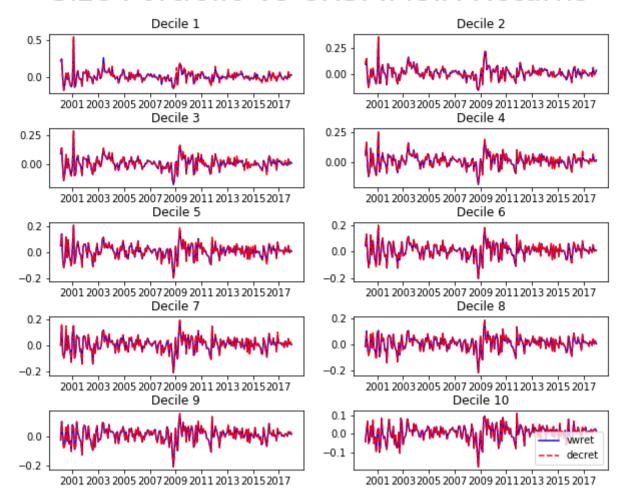
```
64
     msf_groups = msf_groups.sort_values(['decile', 'date'])
65
66
67
     # function to calculate value weighted return
68
     def wavg(group, avg_name, weight_name):
69
         d = group[avg_name]
70
        w = group[weight_name]
71
        try:
72
            return (d * w).sum() / w.sum()
73
        except ZeroDivisionError:
74
            return np.nan
75
76
     # value-weigthed return
77
     vwrets=msf_groups.groupby(['decile','date']).apply(wavg, 'ret','size_lag')\
     .to_frame().reset_index().rename(columns={0: 'vwret'})
78
79
80
     81
     # Compare Results with CRSP MSIX #
82
     msix = conn.raw_sql("""
83
84
                          select caldt, decret1, decret2, decret3, decret4, decret5,
85
                          decret6, decret7, decret8, decret9, decret10
86
                          from crsp.msix where caldt >= '12/01/1999'
                          """, date_cols=['caldt'])
87
88
89
     # transpose msix data
90
     msix1=pd.melt(msix, id_vars='caldt', \
                  value_vars=['decret1','decret2', 'decret3', 'decret4', 'decret5', 'decret6',
91
92
93
     'decret7', 'decret8', 'decret9', 'decret10'])
94
95
     # extract decile information from decret
     msix1['decile'] = msix1['variable'].str[6:].astype(int)
96
97
     # rename return column
     msix1 = msix1.rename(columns={'value':'decret', 'caldt':'date'})
98
     msix1 = msix1.drop(['variable'], axis=1)
99
100
101
     decile_returns = pd.merge(vwrets, msix1, how='left', on=['date','decile'])
102
103
     #########################
104
    # End of Program #
     #####################
```

Top of Section

Outcome Discussion

The figure below plots the comparison between vwrets calculated from the Python code against the decile returns obtained from CRSP.MSIX dataset.

Size Portfolio vs CRSP.MSIX Returns



Top of Section

Тор

Table of Contents

- <u>Background</u>
- » Size Portfolios for all CRSP Securities
- » Outcome Discussion



Wharton Research Data Services

About WRDS

WRDS FAQs

WRDS News

Account Preferences

Info / Support Request

Privacy Policy

3 Ways to use WRDS WRDS Account Types

Terms of Use

Sample Data

Conference Calendar Impactful Research

Unless otherwise noted, all material is © 1993-2020, The Wharton School, University of Pennsylvania. All rights reserved.