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

Size Portfolios for all CRSP Securities

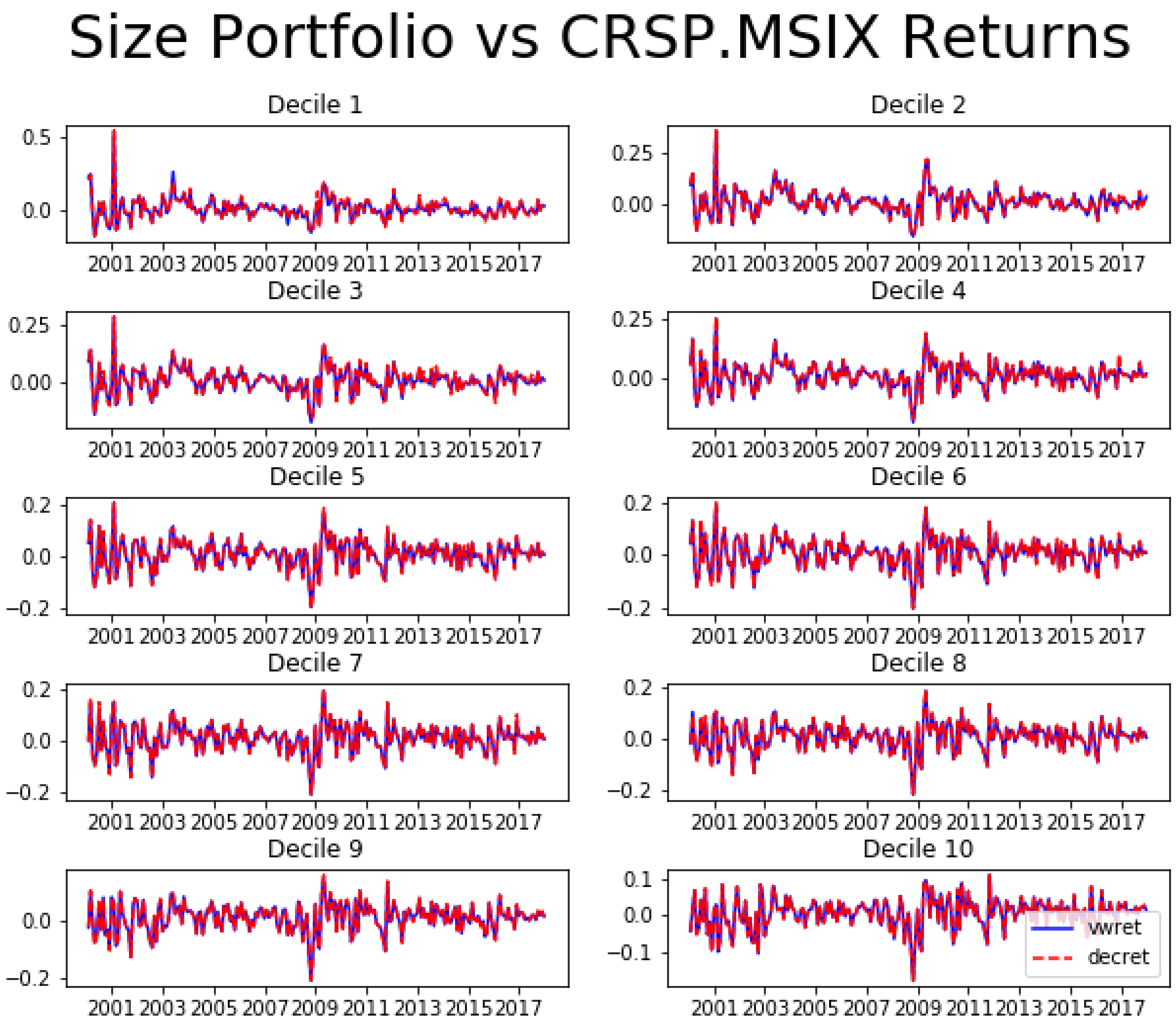
```python
##########################################
# Size Portfolio for CRSP Securitie      #
# July 2018                              #
# Qingyi (Freda) Song Drechsler          #
##########################################

import pandas as pd
import numpy as np
import datetime as dt
import wrds
from dateutil.relativedelta import *

###################
# Connect to WRDS #
###################
conn = wrds.Connection()

################################################
# Get CRSP Monthly Stocks for Decile Formation #
################################################
msf = conn.raw_sql("""
                      select a.permno, a.date,
                      a.ret, a.shrout, a.prc
                      from crsp.msf as a
                      where a.date >= '12/01/1999'
                      """, date_cols=['date'])

# keep only records with non missing ret prc and shrout value
msf = msf[(msf['prc'].notna()) & (msf['ret'].notna()) & (msf['shrout'].notna())]

msf['permno'] = msf['permno'].astype(int)
msf['size'] = msf['shrout'] * msf['prc'].abs()
msf['year'] = msf['date'].dt.year
msf['month'] = msf['date'].dt.month

# create msf_dec
msf_dec = msf[msf['month']==12][['date','permno','year','size']]

# create msf_ls
msf_ls = msf.sort_values(['permno', 'date'])
msf_ls['year_prev'] = msf_ls['year']-1
msf_ls['size_lag'] = msf_ls.groupby('permno')['size'].shift(1)
msf_ls['size_lag'] = np.where(msf_ls['size_lag'].isna(),\
 msf_ls['size']/(1+msf_ls['ret']), msf_ls['size_lag'])

#################################
# Compute Deciles for Each DEC  #
#################################
msf_dec = msf_dec.sort_values(['year'])
msf_dec['decile']=1+msf_dec.groupby('year')['size']\
.transform(lambda x: pd.qcut(x, 10, labels=False))

###################################
# Assign Size Group to All Months #
###################################
msf_groups = pd.merge(msf_ls[['permno','date','ret','size_lag','year_prev']], \
                      msf_dec[['permno','year','decile']], how='left', \
                      left_on=['permno','year_prev'], right_on=['permno','year'])

msf_groups=msf_groups[msf_groups['decile'].notna()]

#################################
# Compute Size Weighted Returns #
```

```
#################################
msf_groups = msf_groups.sort_values(['decile', 'date'])

# function to calculate value weighted return
def wavg(group, avg_name, weight_name):
    d = group[avg_name]
    w = group[weight_name]
    try:
        return (d * w).sum() / w.sum()
    except ZeroDivisionError:
        return np.nan

# value-weigthed return
vwrets=msf_groups.groupby(['decile','date']).apply(wavg, 'ret','size_lag')\
.to_frame().reset_index().rename(columns={0: 'vwret'})

##################################
# Compare Results with CRSP MSIX #
##################################
msix = conn.raw_sql("""
                      select caldt, decret1, decret2, decret3, decret4, decret5,
                      decret6, decret7, decret8, decret9, decret10
                      from crsp.msix where caldt >= '12/01/1999'
                      """, date_cols=['caldt'])

# transpose msix data
msix1=pd.melt(msix, id_vars='caldt', \
              value_vars=['decret1','decret2', 'decret3', 'decret4', 'decret5', 'decret6',
\
'decret7', 'decret8','decret9','decret10'])

# extract decile information from decret
msix1['decile'] = msix1['variable'].str[6:].astype(int)
# rename return column
msix1 = msix1.rename(columns={'value':'decret', 'caldt':'date'})
msix1 = msix1.drop(['variable'], axis=1)

decile_returns = pd.merge(vwrets, msix1, how='left', on=['date','decile'])

###################
# 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.



Top of Section

Top

Table of Contents