HW2

April 9, 2024

0.0.1 Luoyu Zhang, MIT ID: 914165258 , Email: luoyu@mit.edu

0.0.2 Yutong Lin, MIT ID:922403517, Email:brenda_l@mit.edu

[ ]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings("ignore")
```

1 Q1

1.1 Use of PERMNO vs. TICKERS:

PERMNO is a unique identifier assigned by the Center for Research in Security Prices (CRSP) to each security in the market. It is a consistent way to track securities over time, regardless of changes in ticker symbols, company names, or corporate actions such as mergers and acquisitions. This consistency is important for analysis over long periods, as it allows for the unambiguous identification of securities.

TICKERS are symbols assigned to securities traded on public exchanges. These can change over time due to various reasons, such as rebranding, mergers, or moving between exchanges, which can create confusion in long-term analysis.

2 Q2

2.1 Discrepancies between One-Month Price Change and PRC Column:

a. This discrepancy is not data error. The PRC column reflects the total return, and it includes the price appreciation/depreciation, dividends, and other distributions paid to investors. Total return is a comprehensive view of a stock's performance.

The percent changes of price simply reflects the movement in the security's market price. This calculation does not account for dividends or other returns of holding the security. It only focuses on price appreciation/depreciation.

b. For some companies, these values may always match. Because there are no dividends or distributions, or any other corporate actions that might affect the total return.

[ ]:
```python
# Price_Ret(T1)

file_path = 'sp500raw.xlsx'
sp500_data = pd.read_excel(file_path)
sp500_data.drop(columns=['Unnamed: 0'], inplace =True)
sp500_data['date'] = pd.to_datetime(sp500_data['date'])
sp500_data.sort_values(by=['permno', 'date'], inplace=True)

sp500_data['Price_Ret(T1)'] = sp500_data.groupby('permno')['price'].pct_change()


display(sp500_data.head())
```

```
      permno       date    price   shrout       prc          mcap  \
48     10104 2011-01-31  32.0300  5052420  0.024920  1.618290e+08
719    10104 2011-02-28  32.9000  5061000  0.027162  1.665069e+08
1100   10104 2011-03-31  33.4325  5060516  0.016185  1.691857e+08
1836   10104 2011-04-29  35.9600  5060516  0.077395  1.819762e+08
2410   10104 2011-05-31  34.2200  5068000 -0.048387  1.734270e+08

      Price_Ret(T1)
48              NaN
719        0.027162
1100       0.016185
1836       0.075600
2410      -0.048387
```

3 Q3

3.1 Variability in the Number of Companies:

The number of companies in the S&P 500 index does not always precisely equal 500 due to several factors, including mergers, acquisitions, bankruptcies, and the addition or removal of companies based on the market capitalization criteria set by the index. These events can cause the number of constituents to fluctuate over time. So it's not a mistake. It reflects the dynamic nature of the stock market and the index's composition adjustments to maintain its representation of the U.S. economy's leading companies.

4 Q4

4.1 303 companies are present over the entire sample

4.2 761 unique companies are in the sample

[ ]:
```python
date_counts = sp500_data.groupby('permno').size()
full_presence_companies = date_counts[date_counts == sp500_data['date'].
 ↪nunique()]
```

```
unique_companies = sp500_data['permno'].nunique()

(full_presence_companies.count(), unique_companies)
```

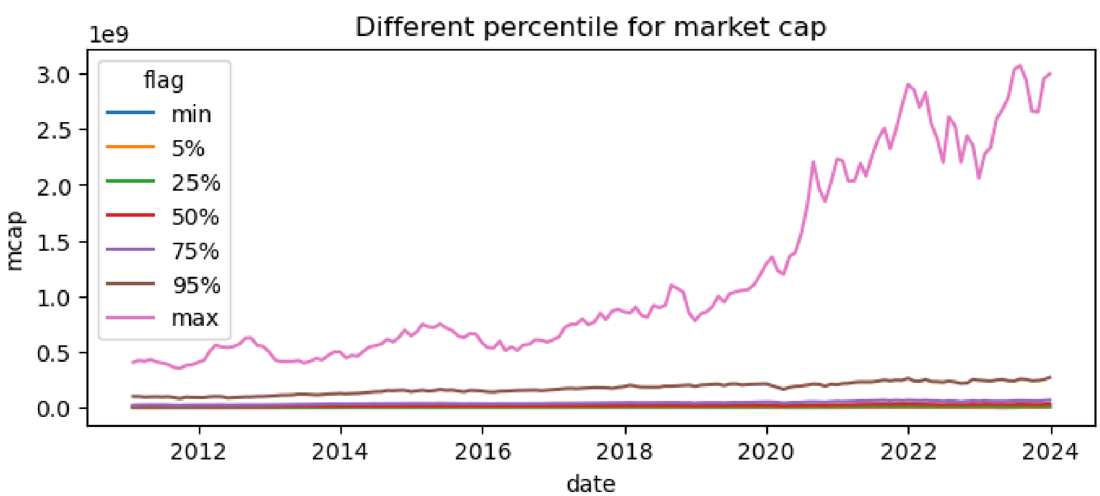
[ ]: (303, 761)

5 Q5

First, we plot the time series of market cap for all stocks each month

```
[ ]: mcap_plot = sp500_data.groupby('date')['mcap'].describe(percentiles=[0.05,0.
     ↪25,0.75,0.95]).reset_index()

col_plot = ['min', '5%', '25%', '50%', '75%', '95%','max']
df_list = []
for i in col_plot:
    tmp_df = mcap_plot[['date',i]]
    tmp_df['flag'] = i
    tmp_df.rename(columns={i:'mcap'},inplace=True)
    df_list.append(tmp_df)
mcap_final = pd.concat(df_list,axis=0)

plt.figure(figsize=(8,3))
plt.title('Different percentile for market cap')
sns.lineplot(data=mcap_final,x = 'date',y = 'mcap',hue='flag')
```

[ ]: <Axes: title={'center': 'Different percentile for market cap'}, xlabel='date', ylabel='mcap'>



[ ]:
```python
nrows = 4
ncols = 2
fig, axes = plt.subplots(nrows, ncols, figsize=(12, 8))

axes_flat = axes.flatten()

for i, col in enumerate(col_plot):
    # Filter the data for the current percentile
    data = mcap_plot[['date', col]].copy()
    data.rename(columns={col: 'mcap'}, inplace=True)

    # Plot on the i-th subplot in the flattened axes array
    sns.lineplot(ax=axes_flat[i], data=data, x='date', y='mcap')
    axes_flat[i].set_title(f'Market Cap {col}')
    axes_flat[i].set_ylabel('Market Cap')
    axes_flat[i].set_xlabel('Date')

if len(col_plot) % 2 != 0:
    axes_flat[-1].set_visible(False)

plt.tight_layout()
plt.show()
```

5.0.1 Q5 a

Calculate the percentile range for the month prior stock leaving

```
[ ]: percentiles=[0, 0.05,0.25,0.50, 0.75,0.95, 1]

last_dates = sp500_data.groupby('permno')['date'].max()
one_month_prior = last_dates - pd.DateOffset(months=1)

# Step 2: Convert 'one_month_prior' to a DataFrame for merging
one_month_prior_df = one_month_prior.reset_index()
one_month_prior_df.columns = ['permno', 'date']

prior_to_leaving = pd.merge(sp500_data, one_month_prior_df, how='inner',␣
 ↪on=['permno', 'date'])
prior_to_leaving['date'] = prior_to_leaving['date'].dt.to_period('M')

# Calculate the quantiles of mcap for all companies on each date
prior_percentiles = prior_to_leaving.groupby('date')['mcap'].quantile(np.
 ↪array(percentiles)).unstack()

# # Merge the permno information with the prior_percentiles DataFrame
# prior_percentiles = pd.merge(prior_percentiles, prior_to_leaving[['date',␣
 ↪'permno']], on='date')

prior_percentiles
```

```
[ ]:                0.00          0.05          0.25          0.50          0.75  \
date
2011-02   2075951.64  3.072294e+06  7.057662e+06  1.203937e+07  1.589526e+07
2011-09   1119252.00  1.452419e+06  2.785089e+06  4.450926e+06  6.116763e+06
2011-11   1803942.70  1.803943e+06  1.803943e+06  1.803943e+06  1.803943e+06
2012-02  26233301.16  2.623330e+07  2.623330e+07  2.623330e+07  2.623330e+07
2012-03   1212050.28  2.275960e+06  6.531599e+06  1.185115e+07  1.734458e+07
2012-04   3377462.40  3.860524e+06  5.792769e+06  8.208075e+06  1.062338e+07
2012-07   5268457.95  5.268458e+06  5.268458e+06  5.268458e+06  5.268458e+06
2012-11   2062613.50  2.062614e+06  2.062614e+06  2.062614e+06  2.062614e+06
2012-12   1647663.24  1.647663e+06  1.647663e+06  1.647663e+06  1.647663e+06
2013-04   2508740.00  3.546321e+06  7.696646e+06  1.288455e+07  1.807246e+07
2013-08  24212741.13  2.421274e+07  2.421274e+07  2.421274e+07  2.421274e+07
2013-09  10210123.72  1.021012e+07  1.021012e+07  1.021012e+07  1.021012e+07
2014-02   3066332.61  3.066333e+06  3.066333e+06  3.066333e+06  3.066333e+06
2014-04   3096085.00  3.096085e+06  3.096085e+06  3.096085e+06  3.096085e+06
2014-05   3333934.08  4.459060e+06  8.959565e+06  1.458520e+07  2.021083e+07
2014-06   3970910.59  3.970911e+06  3.970911e+06  3.970911e+06  3.970911e+06
2015-06  13297233.60  1.329723e+07  1.329723e+07  1.329723e+07  1.329723e+07
2015-07  15468946.30  1.546895e+07  1.546895e+07  1.546895e+07  1.546895e+07
2015-09   2298075.78  2.606799e+06  3.841693e+06  5.385310e+06  1.102737e+07
```

2015-10	15915135.35	1.620842e+07	1.738157e+07	1.884801e+07	2.031444e+07
2015-11	1851137.93	3.240462e+06	8.797758e+06	1.574438e+07	2.269100e+07
2016-01	1818688.76	1.891327e+06	2.181881e+06	4.534724e+06	8.410461e+06
2016-02	2445489.78	2.523534e+06	2.835710e+06	3.225929e+06	7.893580e+06
2016-11	7732550.52	8.477358e+06	1.145659e+07	1.518063e+07	1.890467e+07
2017-02	3739634.54	3.746957e+06	3.776248e+06	3.812861e+06	3.849475e+06
2017-06	4269163.76	4.729676e+06	6.571726e+06	8.874287e+06	1.117685e+07
2017-07	6665636.95	1.313331e+07	3.900398e+07	7.134233e+07	7.496822e+07
2018-02	6643920.78	6.643921e+06	6.643921e+06	6.643921e+06	6.643921e+06
2018-04	3451932.45	3.463228e+06	3.508410e+06	4.910450e+06	5.532378e+07
2018-07	14523478.01	1.452348e+07	1.452348e+07	1.452348e+07	1.452348e+07
2018-11	6654602.54	6.654603e+06	6.654603e+06	6.654603e+06	6.654603e+06
2018-12	2937233.62	3.027302e+06	3.387578e+06	3.837922e+06	4.288266e+06
2019-02	4517462.40	4.517462e+06	4.517462e+06	4.517462e+06	4.517462e+06
2019-04	4210730.75	4.277526e+06	4.544705e+06	4.878678e+06	5.212652e+06
2019-08	3079564.18	3.079564e+06	3.079564e+06	3.079564e+06	3.079564e+06
2019-09	70642020.00	7.064202e+07	7.064202e+07	7.064202e+07	7.064202e+07
2020-07	2790887.50	2.794907e+06	2.810985e+06	2.831082e+06	2.917197e+06
2020-09	7666398.16	7.666398e+06	7.666398e+06	7.666398e+06	7.666398e+06
2020-10	4748825.40	4.748825e+06	4.748825e+06	4.748825e+06	4.748825e+06
2020-11	11283611.40	1.151760e+07	1.245355e+07	1.362350e+07	1.479344e+07
2021-06	28295903.04	2.829590e+07	2.829590e+07	2.829590e+07	2.829590e+07
2021-12	6590562.95	1.119795e+07	2.962750e+07	5.266444e+07	5.283919e+07
2022-02	4756719.00	5.186794e+06	6.907092e+06	9.057465e+06	9.145936e+06
2022-11	4851051.93	4.851052e+06	4.851052e+06	4.851052e+06	4.851052e+06
2023-06	4178913.20	4.178913e+06	4.178913e+06	4.178913e+06	4.178913e+06
2023-07	4622472.00	4.629070e+06	4.655463e+06	4.688453e+06	4.721444e+06

	0.95	1.00
date		
2011-02	1.897997e+07	19751150.10
2011-09	7.449433e+06	7782600.00
2011-11	1.803943e+06	1803942.70
2012-02	2.623330e+07	26233301.16
2012-03	2.173933e+07	22838013.00
2012-04	1.255563e+07	13038687.68
2012-07	5.268458e+06	5268457.95
2012-11	2.062614e+06	2062613.50
2012-12	1.647663e+06	1647663.24
2013-04	2.222278e+07	23260362.54
2013-08	2.421274e+07	24212741.13
2013-09	1.021012e+07	10210123.72
2014-02	3.066333e+06	3066332.61
2014-04	3.096085e+06	3096085.00
2014-05	2.471133e+07	25836459.32
2014-06	3.970911e+06	3970910.59
2015-06	1.329723e+07	13297233.60

```
2015-07  1.546895e+07  15468946.30
2015-09  1.554102e+07  16669427.56
2015-10  2.148759e+07  21780876.17
2015-11  2.824830e+07  29637619.68
2016-01  1.235596e+07  13342339.50
2016-02  1.162770e+07  12561230.44
2016-11  2.188390e+07  22628707.20
2017-02  3.878766e+06   3886088.34
2017-06  1.301890e+07  13479411.00
2017-07  7.786893e+07  78594106.80
2018-02  6.643921e+06   6643920.78
2018-04  7.039583e+07  74163841.20
2018-07  1.452348e+07  14523478.01
2018-11  6.654603e+06   6654602.54
2018-12  4.648541e+06   4738610.11
2019-02  4.517462e+06   4517462.40
2019-04  5.479831e+06   5546625.84
2019-08  3.079564e+06   3079564.18
2019-09  7.064202e+07  70642020.00
2020-07  2.986089e+06   3003312.48
2020-09  7.666398e+06   7666398.16
2020-10  4.748825e+06   4748825.40
2020-11  1.572939e+07  15963381.24
2021-06  2.829590e+07  28295903.04
2021-12  5.297899e+07  53013945.72
2022-02  9.216713e+06   9234407.38
2022-11  4.851052e+06   4851051.93
2023-06  4.178913e+06   4178913.20
2023-07  4.747836e+06   4754434.36
```

Q5 b Calculate the percentile range of market caps one month before stocks entering

[ ]:
```
first_dates = sp500_data.groupby('permno')['date'].min()
first_dates_df = first_dates.reset_index()
first_dates_df.columns = ['permno', 'date']

upon_entering = pd.merge(sp500_data, first_dates_df, how = 'inner', on =
 ↪['permno', 'date'])
upon_entering['date'] = upon_entering['date'].dt.to_period('M')

entering_percentiles = upon_entering.groupby('date')['mcap'].quantile(np.
 ↪array(percentiles)).unstack()

entering_percentiles
```

[ ]:
```
                  0.00          0.05          0.25          0.50          0.75  \
date
```

```
2011-01  1.546778e+06  3.083643e+06  6.678938e+06  1.147552e+07  2.218710e+07
2011-02  1.020416e+07  1.020416e+07  1.020416e+07  1.020416e+07  1.020416e+07
2011-03  2.565047e+07  2.565047e+07  2.565047e+07  2.565047e+07  2.565047e+07
2011-04  8.302238e+06  8.463407e+06  9.108084e+06  9.913930e+06  1.791928e+07
2011-06  5.492196e+06  6.031846e+06  8.190446e+06  1.088870e+07  1.358694e+07
…                 …             …             …             …             …
2023-06  7.814901e+07  7.814901e+07  7.814901e+07  7.814901e+07  7.814901e+07
2023-08  4.413831e+07  4.413831e+07  4.413831e+07  4.413831e+07  4.413831e+07
2023-09  5.968635e+07  6.050417e+07  6.377542e+07  6.786448e+07  7.195355e+07
2023-10  1.448573e+07  1.473669e+07  1.574049e+07  1.699525e+07  3.232717e+07
2023-12  2.059155e+07  2.589709e+07  4.711924e+07  7.364693e+07  1.001746e+08

                 0.95          1.00
date
2011-01  1.007589e+08  4.068335e+08
2011-02  1.020416e+07  1.020416e+07
2011-03  2.565047e+07  2.565047e+07
2011-04  2.432356e+07  2.592463e+07
2011-06  1.574554e+07  1.628519e+07
…                 …             …
2023-06  7.814901e+07  7.814901e+07
2023-08  4.413831e+07  4.413831e+07
2023-09  7.522480e+07  7.604262e+07
2023-10  4.459270e+07  4.765909e+07
2023-12  1.213968e+08  1.267023e+08

[124 rows x 7 columns]
```

5.0.2 Q5 i-vi

[ ]:
```python
df = sp500_data.copy()
df.set_index('date', inplace=True)
df = df.sort_values(['permno', 'date'])

def geometric_return(returns):
    return np.prod(1+returns) - 1

# (i)         Trailing twelve month return based on prc (we will refer to this␣
↪PRC_Ret(T12))
df['PRC_Ret(T12)'] = df.groupby('permno')['prc'].transform(lambda x: x.
↪rolling(12).apply(geometric_return, raw = True))
# (ii)         Trailing twelve month return based on prices (we will refer to␣
↪this Prices_Ret(T12))
df['Prices_Ret(T12)'] = df.groupby('permno')['Price_Ret(T1)'].transform(lambda␣
↪x: x.rolling(12).apply(geometric_return, raw = True))
```

```python
# (iii)        Trailing twelve month return excluding the most recent trailing␣
↪month based on prc.
# In other words, this eleven months of return starting from 12 months ago␣
↪excluding the most recent month (we will to this as PRC_Ret(T12M1))
df['PRC_Ret(T12M1)'] = df.groupby('permno')['prc'].transform(lambda x: x.
↪shift(1).rolling(window=11).apply(geometric_return, raw = True))
# (iv)        Trailing twelve month return excluding the most recent trailing␣
↪month based on prices.
# In other words, this eleven months of return starting from 12 months ago␣
↪excluding the most recent month (we will to this as Prices_Ret(T12M1))
df['Prices_Ret(T12M1)'] = df.groupby('permno')['Price_Ret(T1)'].
↪transform(lambda x: x.shift(1).rolling(window=11).apply(geometric_return,␣
↪raw = True))

# one-month-return from 12 months ago
# (v)        The trailing one month return from exactly 12 months ago.
# In other words, if the EOM period is 2019-03 (March 2019), we want the return␣
↪for the stock for 2018-03 (March 2018).
# Do this based both on prc column and price based return. We will refer to␣
↪this as PRC_Ret(T12_1M) and Prices_Ret(T12_1M)
df['PRC_Ret(T12_1M)'] = df.groupby('permno')['prc'].shift(12)
df['Prices_Ret(T12_1M)'] =  df.groupby('permno')['Price_Ret(T1)'].shift(12)

# standard deviation of the monthly price based returns
# (vi)        Calculate the standard deviation of the monthly price_based␣
↪returns used in calculating Prices_Ret(T12M1); we call this␣
↪Vol_Prices_Ret(T12M1)
df['Vol_Prices_Ret(T12M1)'] = df.groupby('permno')['Price_Ret(T1)'].
↪transform(lambda x: x.shift(1).rolling(window=11).std())
# Divide Prices_Ret(T12M1) / Vol_Prices_Ret(T12M1). We will call this␣
↪SR_Prices_Ret(T12M1)
df['SR_Prices_Ret(T12M1)'] = df['Prices_Ret(T12M1)'] /␣
↪df['Vol_Prices_Ret(T12M1)']

df[df['permno'] == 13688].head(15)
```

[ ]:
```
            permno  price  shrout       prc         mcap  Price_Ret(T1)  \
date
2011-01-31   13688  46.28  392066 -0.032609  18144814.48            NaN
2011-02-28   13688  46.06  396258 -0.004754  18251643.48      -0.004754
2011-03-31   13688  44.18  396789 -0.030938  17530138.02      -0.040816
2011-04-29   13688  46.08  396789  0.043006  18284037.12       0.043006
2011-05-31   13688  43.38  397950 -0.058594  17263071.00      -0.058594
2011-06-30   13688  42.03  397950 -0.020632  16725838.50      -0.031120
2011-07-29   13688  41.43  397950 -0.014276  16487068.50      -0.014276
2011-08-31   13688  42.35  402245  0.022206  17035075.75       0.022206
```

```
2011-09-30   13688  42.30  402245  0.009563  17014963.50      -0.001181
2011-10-31   13688  42.90  402245  0.014185  17256310.50       0.014184
2011-11-30   13688  38.84  405883 -0.094639  15764495.72      -0.094639
2011-12-30   13688  41.22  405883  0.072992  16730497.26       0.061277
2012-01-31   13688  40.66  405883 -0.013586  16503202.78      -0.013586
2012-02-29   13688  41.68  412900  0.025086  17209672.00       0.025086
2012-03-30   13688  43.41  421211  0.052423  18284769.51       0.041507

            PRC_Ret(T12)  Prices_Ret(T12)  PRC_Ret(T12M1)  Prices_Ret(T12M1)  \
date
2011-01-31           NaN              NaN             NaN                NaN
2011-02-28           NaN              NaN             NaN                NaN
2011-03-31           NaN              NaN             NaN                NaN
2011-04-29           NaN              NaN             NaN                NaN
2011-05-31           NaN              NaN             NaN                NaN
2011-06-30           NaN              NaN             NaN                NaN
2011-07-29           NaN              NaN             NaN                NaN
2011-08-31           NaN              NaN             NaN                NaN
2011-09-30           NaN              NaN             NaN                NaN
2011-10-31           NaN              NaN             NaN                NaN
2011-11-30           NaN              NaN             NaN                NaN
2011-12-30     -0.100800              NaN       -0.161969                NaN
2012-01-31     -0.083118        -0.121435       -0.070489          -0.109334
2012-02-29     -0.055627        -0.095093       -0.078738          -0.117238
2012-03-30      0.025610        -0.017429       -0.025478          -0.056587

            PRC_Ret(T12_1M)  Prices_Ret(T12_1M)  Vol_Prices_Ret(T12M1)  \
date
2011-01-31              NaN                 NaN                    NaN
2011-02-28              NaN                 NaN                    NaN
2011-03-31              NaN                 NaN                    NaN
2011-04-29              NaN                 NaN                    NaN
2011-05-31              NaN                 NaN                    NaN
2011-06-30              NaN                 NaN                    NaN
2011-07-29              NaN                 NaN                    NaN
2011-08-31              NaN                 NaN                    NaN
2011-09-30              NaN                 NaN                    NaN
2011-10-31              NaN                 NaN                    NaN
2011-11-30              NaN                 NaN                    NaN
2011-12-30              NaN                 NaN                    NaN
2012-01-31        -0.032609                 NaN               0.045336
2012-02-29        -0.004754           -0.004754               0.045322
2012-03-30        -0.030938           -0.040816               0.045243

            SR_Prices_Ret(T12M1)
date
2011-01-31                   NaN
```

```
2011-02-28                  NaN
2011-03-31                  NaN
2011-04-29                  NaN
2011-05-31                  NaN
2011-06-30                  NaN
2011-07-29                  NaN
2011-08-31                  NaN
2011-09-30                  NaN
2011-10-31                  NaN
2011-11-30                  NaN
2011-12-30                  NaN
2012-01-31            -2.411622
2012-02-29            -2.586795
2012-03-30            -1.250717
```

[ ]:

```
# Calculate 1-month, 3-month, 6-month returns
df['PRC_Ret(F1M)'] = df.groupby('permno')['prc'].shift(-1)
df['PRC_Ret(F3M)'] = df.groupby('permno')['prc'].rolling(3).
 ↪apply(geometric_return).groupby(level=0).shift(-3).reset_index(level=0,␣
 ↪drop=True)
df['PRC_Ret(F6M)'] = df.groupby('permno')['prc'].rolling(6).
 ↪apply(geometric_return).groupby(level=0).shift(-6).reset_index(level=0,␣
 ↪drop=True)

df[df['permno'] == 13688].head(20)
```

[ ]:

date	permno	price	shrout	prc	mcap	Price_Ret(T1) \
2011-01-31	13688	46.28	392066	-0.032609	18144814.48	NaN
2011-02-28	13688	46.06	396258	-0.004754	18251643.48	-0.004754
2011-03-31	13688	44.18	396789	-0.030938	17530138.02	-0.040816
2011-04-29	13688	46.08	396789	0.043006	18284037.12	0.043006
2011-05-31	13688	43.38	397950	-0.058594	17263071.00	-0.058594
2011-06-30	13688	42.03	397950	-0.020632	16725838.50	-0.031120
2011-07-29	13688	41.43	397950	-0.014276	16487068.50	-0.014276
2011-08-31	13688	42.35	402245	0.022206	17035075.75	0.022206
2011-09-30	13688	42.30	402245	0.009563	17014963.50	-0.001181
2011-10-31	13688	42.90	402245	0.014185	17256310.50	0.014184
2011-11-30	13688	38.84	405883	-0.094639	15764495.72	-0.094639
2011-12-30	13688	41.22	405883	0.072992	16730497.26	0.061277
2012-01-31	13688	40.66	405883	-0.013586	16503202.78	-0.013586
2012-02-29	13688	41.68	412900	0.025086	17209672.00	0.025086
2012-03-30	13688	43.41	421211	0.052423	18284769.51	0.041507
2012-04-30	13688	44.18	415354	0.017738	18350339.72	0.017738
2012-05-31	13688	43.70	422320	-0.010865	18455384.00	-0.010865
2012-06-29	13688	45.27	422320	0.046339	19118426.40	0.035927
2012-07-31	13688	46.16	422320	0.019660	19494291.20	0.019660

```
2012-08-31   13688  43.41  426463 -0.059575  18512758.83      -0.059575

            PRC_Ret(T12)  Prices_Ret(T12)  PRC_Ret(T12M1)  Prices_Ret(T12M1)  \
date
2011-01-31           NaN              NaN             NaN                NaN
2011-02-28           NaN              NaN             NaN                NaN
2011-03-31           NaN              NaN             NaN                NaN
2011-04-29           NaN              NaN             NaN                NaN
2011-05-31           NaN              NaN             NaN                NaN
2011-06-30           NaN              NaN             NaN                NaN
2011-07-29           NaN              NaN             NaN                NaN
2011-08-31           NaN              NaN             NaN                NaN
2011-09-30           NaN              NaN             NaN                NaN
2011-10-31           NaN              NaN             NaN                NaN
2011-11-30           NaN              NaN             NaN                NaN
2011-12-30     -0.100800              NaN       -0.161969                NaN
2012-01-31     -0.083118        -0.121435       -0.070489          -0.109334
2012-02-29     -0.055627        -0.095093       -0.078738          -0.117238
2012-03-30      0.025610        -0.017429       -0.025478          -0.056587
2012-04-30      0.000763        -0.041233       -0.016679          -0.057943
2012-05-31      0.051502         0.007377        0.063052           0.018442
2012-06-29      0.123405         0.077088        0.073653           0.039734
2012-07-31      0.162081         0.114168        0.139675           0.092686
2012-08-31      0.069110         0.025030        0.136837           0.089965

            PRC_Ret(T12_1M)  Prices_Ret(T12_1M)  Vol_Prices_Ret(T12M1)  \
date
2011-01-31              NaN                 NaN                    NaN
2011-02-28              NaN                 NaN                    NaN
2011-03-31              NaN                 NaN                    NaN
2011-04-29              NaN                 NaN                    NaN
2011-05-31              NaN                 NaN                    NaN
2011-06-30              NaN                 NaN                    NaN
2011-07-29              NaN                 NaN                    NaN
2011-08-31              NaN                 NaN                    NaN
2011-09-30              NaN                 NaN                    NaN
2011-10-31              NaN                 NaN                    NaN
2011-11-30              NaN                 NaN                    NaN
2011-12-30              NaN                 NaN                    NaN
2012-01-31        -0.032609                 NaN               0.045336
2012-02-29        -0.004754           -0.004754               0.045322
2012-03-30        -0.030938           -0.040816               0.045243
2012-04-30         0.043006            0.043006               0.045089
2012-05-31        -0.058594           -0.058594               0.041669
2012-06-29        -0.020632           -0.031120               0.040467
2012-07-31        -0.014276           -0.014276               0.040988
2012-08-31         0.022206            0.022206               0.040912
```

```
            SR_Prices_Ret(T12M1)  PRC_Ret(F1M)  PRC_Ret(F3M)  PRC_Ret(F6M)
date
2011-01-31                   NaN     -0.004754      0.005932     -0.085788
2011-02-28                   NaN     -0.030938     -0.048486     -0.061023
2011-03-31                   NaN      0.043006     -0.038366     -0.021779
2011-04-29                   NaN     -0.058594     -0.091179     -0.048810
2011-05-31                   NaN     -0.020632     -0.013176     -0.085230
2011-06-30                   NaN     -0.014276      0.017249      0.002219
2011-07-29                   NaN      0.022206      0.046620      0.002921
2011-08-31                   NaN      0.009563     -0.073016      0.005746
2011-09-30                   NaN      0.014185     -0.014775      0.048444
2011-10-31                   NaN     -0.094639     -0.041753      0.052117
2011-11-30                   NaN      0.072992      0.084966      0.149471
2011-12-30                   NaN     -0.013586      0.064167      0.120918
2012-01-31             -2.411622      0.025086      0.097960      0.158697
2012-02-29             -2.586795      0.052423      0.059453      0.063001
2012-03-30             -1.250717      0.017738      0.053329      0.003421
2012-04-30             -1.285087     -0.010865      0.055318     -0.017533
2012-05-31              0.442571      0.046339      0.003349     -0.043417
2012-06-29              0.981883      0.019660     -0.047382     -0.092813
2012-07-31              2.261305     -0.059575     -0.069033     -0.055833
2012-08-31              2.198959     -0.006565     -0.046610      0.003979
```

[ ]:
```
df[df['permno'] == 13688].tail(20)
```

[ ]:
```
            permno  price   shrout       prc         mcap  Price_Ret(T1)  \
date
2018-08-31   13688  46.18   517151  0.071959  23882033.18       0.071959
2018-09-28   13688  46.01   517151 -0.003681  23794117.51      -0.003681
2018-10-31   13688  46.81   517151  0.017388  24207838.31       0.017388
2018-11-30   13688  26.38   518674 -0.436445  13682620.12      -0.436445
2018-12-31   13688  23.75   518674 -0.099697  12318507.50      -0.099697
2022-10-31   13688  14.93  1987700  0.194400  29676361.00      -0.371368
2022-11-30   13688  15.70  1987700  0.051574  31206890.00       0.051574
2022-12-30   13688  16.26  1987700  0.035669  32320002.00       0.035669
2023-01-31   13688  15.90  1987700 -0.022140  31604430.00      -0.022140
2023-02-28   13688  15.62  1987700 -0.017610  31047874.00      -0.017610
2023-03-31   13688  16.17  1987785  0.035211  32142483.45       0.035211
2023-04-28   13688  17.11  1987785  0.058132  34011001.35       0.058132
2023-05-31   13688  16.94  1995778 -0.009936  33808479.32      -0.009936
2023-06-30   13688  17.28  1995778  0.020071  34487043.84       0.020071
2023-07-31   13688  17.61  2568985  0.019097  45239825.85       0.019097
2023-08-31   13688  16.30  2091241 -0.074390  34087228.30      -0.074390
2023-09-29   13688  16.13  2091241 -0.010430  33731717.33      -0.010429
2023-10-31   13688  16.30  2133508  0.010539  34776180.40       0.010539
2023-11-30   13688  17.17  2133508  0.053374  36632332.36       0.053374
```

2023-12-29	13688	18.03	2133508	0.050670	38467149.24	0.050087

	PRC_Ret(T12)	Prices_Ret(T12)	PRC_Ret(T12M1)	Prices_Ret(T12M1) \
date				
2018-08-31	-0.338740	-0.343848	-0.383129	-0.387894
2018-09-28	-0.324276	-0.324277	-0.321780	-0.321780
2018-10-31	-0.189717	-0.189718	-0.203565	-0.203566
2018-11-30	-0.513643	-0.513643	-0.136983	-0.136984
2018-12-31	-0.470221	-0.470221	-0.411554	-0.411555
2022-10-31	-0.331439	-0.648126	-0.440254	-0.440255
2022-11-30	-0.274032	-0.617912	-0.309637	-0.636651
2022-12-30	-0.296745	-0.629866	-0.320965	-0.642613
2023-01-31	-0.344685	-0.655098	-0.329848	-0.647289
2023-02-28	-0.315070	-0.639511	-0.302792	-0.633049
2023-03-31	-0.278124	-0.620066	-0.302678	-0.632989
2023-04-28	-0.245380	-0.602832	-0.286838	-0.624652
2023-05-31	-0.303031	-0.633175	-0.296037	-0.629493
2023-06-30	-0.286416	-0.624429	-0.300456	-0.631819
2023-07-31	-0.285217	-0.623798	-0.298612	-0.630848
2023-08-31	0.173994	-0.382108	0.268346	-0.332449
2023-09-29	0.290398	-0.320842	0.303999	-0.313684
2023-10-31	0.091760	0.091762	0.080373	0.080375
2023-11-30	0.093628	0.093631	0.038215	0.038217
2023-12-29	0.109469	0.108856	0.055963	0.055966

	PRC_Ret(T12_1M)	Prices_Ret(T12_1M)	Vol_Prices_Ret(T12M1) \
date			
2018-08-31	0.039740	0.039740	0.073846
2018-09-28	-0.025007	-0.032538	0.081380
2018-10-31	-0.151564	-0.151564	0.071131
2018-11-30	-0.061104	-0.061104	0.070312
2018-12-31	-0.173488	-0.173488	0.140641
2022-10-31	-0.053536	-0.053536	0.141937
2022-11-30	-0.031581	-0.031581	0.173600
2022-12-30	0.069117	0.069117	0.172270
2023-01-31	0.049397	0.049397	0.171354
2023-02-28	-0.060087	-0.060087	0.172015
2023-03-31	-0.017771	-0.017771	0.172020
2023-04-28	0.012218	0.012218	0.173245
2023-05-31	0.071959	0.071959	0.172181
2023-06-30	-0.003681	-0.003681	0.171956
2023-07-31	0.017388	0.017388	0.172093
2023-08-31	-0.436445	-0.436445	0.122407
2023-09-29	-0.099697	-0.099697	0.121143
2023-10-31	0.194400	-0.371368	0.038987
2023-11-30	0.051574	0.051574	0.036241
2023-12-29	0.035669	0.035669	0.038131

```
            SR_Prices_Ret(T12M1)  PRC_Ret(F1M)  PRC_Ret(F3M)  PRC_Ret(F6M)
date
2018-08-31             -5.252737     -0.003681     -0.428756     -0.354049
2018-09-28             -3.954051      0.017388     -0.483808     -0.328537
2018-10-31             -2.861863     -0.436445     -0.393997     -0.354625
2018-11-30             -1.948229     -0.099697      0.130781      0.125019
2018-12-31             -2.926289      0.194400      0.300800      0.293600
2022-10-31             -3.101749      0.051574      0.064970      0.146015
2022-11-30             -3.667350      0.035669     -0.005095      0.078980
2022-12-30             -3.730258     -0.022140     -0.005535      0.062730
2023-01-31             -3.777486     -0.017610      0.076100      0.107546
2023-02-28             -3.680186      0.035211      0.084506      0.043532
2023-03-31             -3.679734      0.058132      0.068645     -0.002475
2023-04-28             -3.605598     -0.009936      0.029222     -0.047342
2023-05-31             -3.655991      0.020071     -0.037781      0.013576
2023-06-30             -3.674303      0.019097     -0.066552      0.043980
2023-07-31             -3.665742     -0.074390     -0.074391           NaN
2023-08-31             -2.715922     -0.010430      0.053373           NaN
2023-09-29             -2.589367      0.010539      0.118412           NaN
2023-10-31              2.061590      0.053374           NaN           NaN
2023-11-30              1.054517      0.050670           NaN           NaN
2023-12-29              1.467721           NaN           NaN           NaN
```

[ ]:
```
df.columns
```

[ ]:
```
Index(['permno', 'price', 'shrout', 'prc', 'mcap', 'Price_Ret(T1)',
       'PRC_Ret(T12)', 'Prices_Ret(T12)', 'PRC_Ret(T12M1)',
       'Prices_Ret(T12M1)', 'PRC_Ret(T12_1M)', 'Prices_Ret(T12_1M)',
       'Vol_Prices_Ret(T12M1)', 'SR_Prices_Ret(T12M1)', 'PRC_Ret(F1M)',
       'PRC_Ret(F3M)', 'PRC_Ret(F6M)'],
      dtype='object')
```

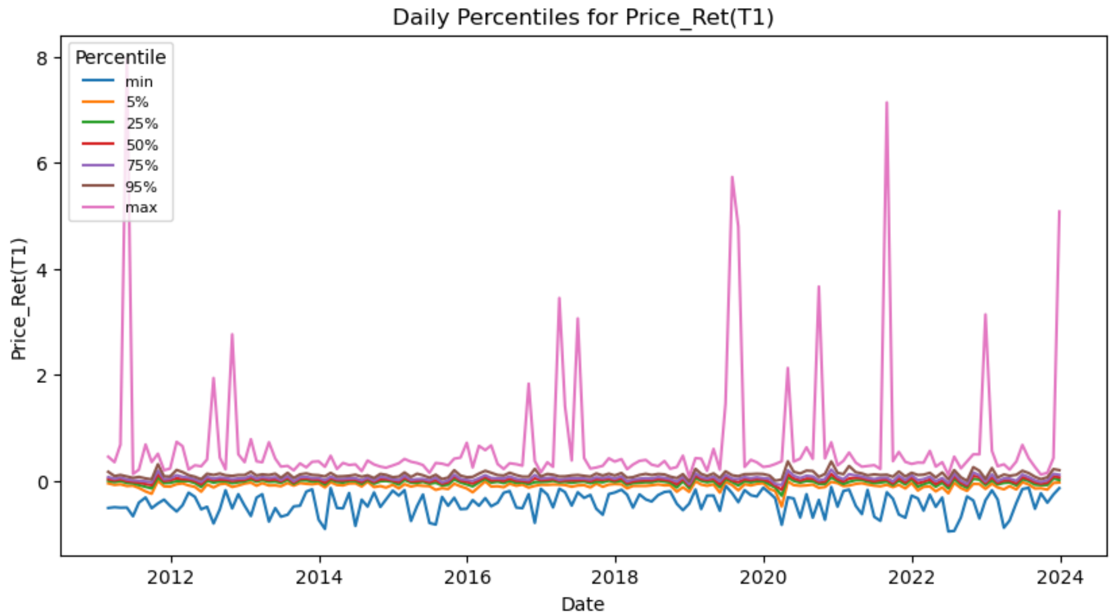
6 Q6

plot the time series of different variables (unnormalized)

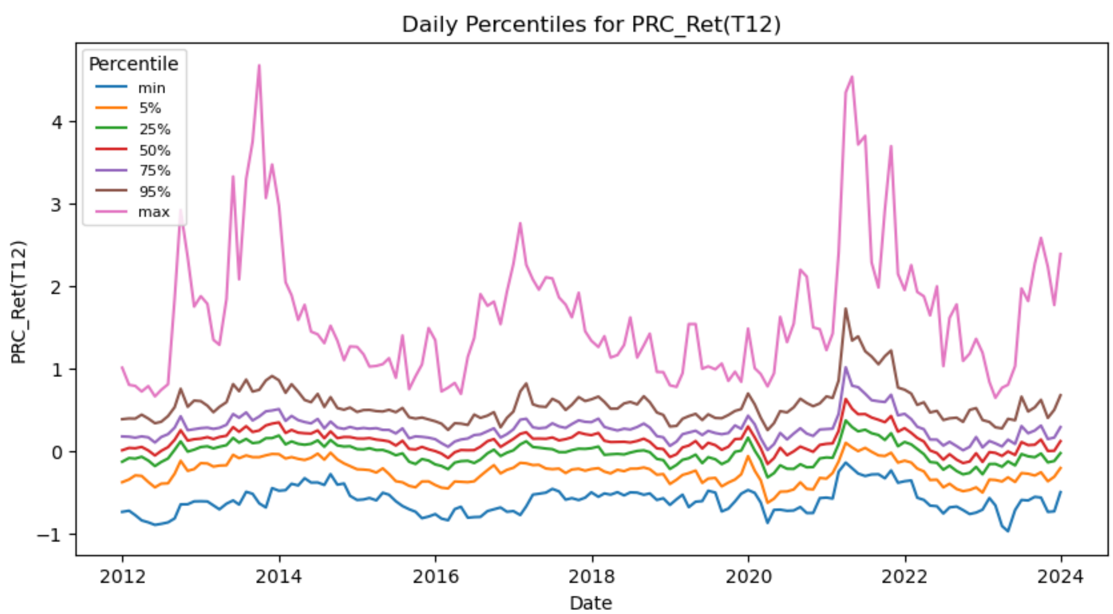
[ ]:
```
var_list = ['Price_Ret(T1)',
            'PRC_Ret(T12)', 'Prices_Ret(T12)',
            'PRC_Ret(T12M1)', 'Prices_Ret(T12M1)',
            'PRC_Ret(T12_1M)', 'Prices_Ret(T12_1M)',
            'Vol_Prices_Ret(T12M1)', 'SR_Prices_Ret(T12M1)',
            'PRC_Ret(F1M)', 'PRC_Ret(F3M)', 'PRC_Ret(F6M)']
```

[ ]:
```
medians_dict = {}

for var in var_list:
```

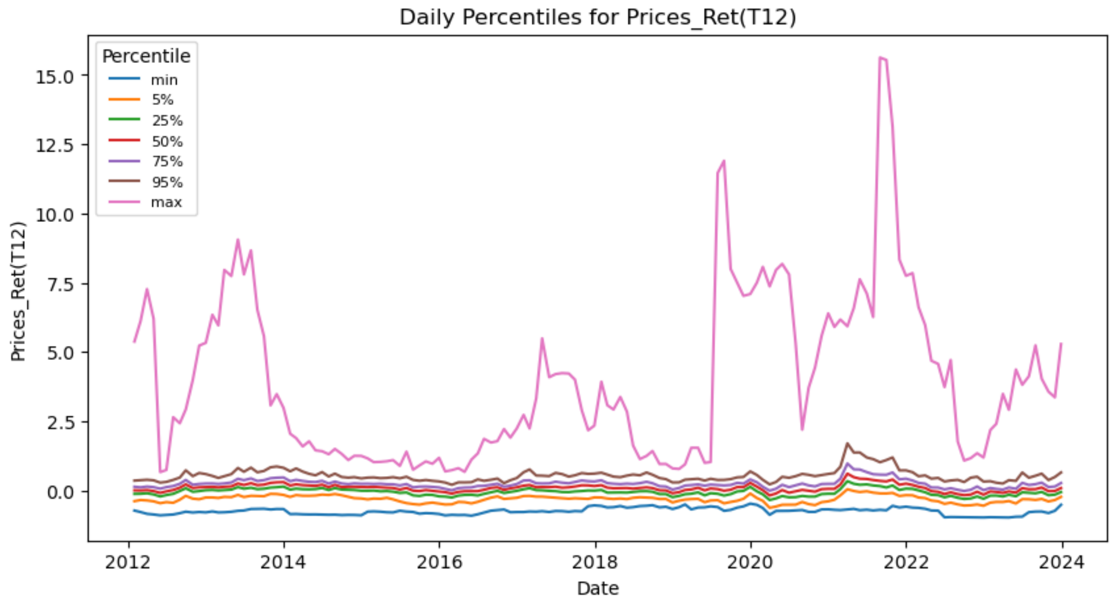
```
    # Calculate percentiles
    percentiles_df = df.groupby('date')[var].describe(percentiles=[0.0, 0.05, 0.
↪25, 0.5, 0.75, 0.95, 1.0]).reset_index()
    percentiles_df = percentiles_df[['date', 'min', '5%', '25%', '50%', '75%',␣
↪'95%', 'max']]

    # Melt the dataframe
    melted_df = percentiles_df.melt(id_vars=['date'], var_name='Percentile',␣
↪value_name=var)

    # Plotting
    plt.figure(figsize=(10, 5))
    sns.lineplot(data=melted_df, x='date', y=var, hue='Percentile')
    plt.title(f'Daily Percentiles for {var}')
    plt.xlabel('Date')
    plt.ylabel(var)
    plt.legend(title='Percentile',  fontsize="8", loc ="upper left")
    plt.show()

    # Calculating median values for each percentile
    medians_dict[var] = {}
    for percentile in ['min', '5%', '25%', '50%', '75%', '95%', 'max']:
        # Filter the dataframe for the current percentile
        filtered_df = melted_df[melted_df['Percentile'] == percentile]
        # Calculate the median of these values
        median_value = filtered_df[var].median()
        medians_dict[var][percentile] = median_value

# Print median values for each percentile of each variable
for var, percentiles in medians_dict.items():
    print(f"\n{var}:")
    for percentile, median_value in percentiles.items():
        print(f"  Median of the {percentile} percentile: {median_value}")
```

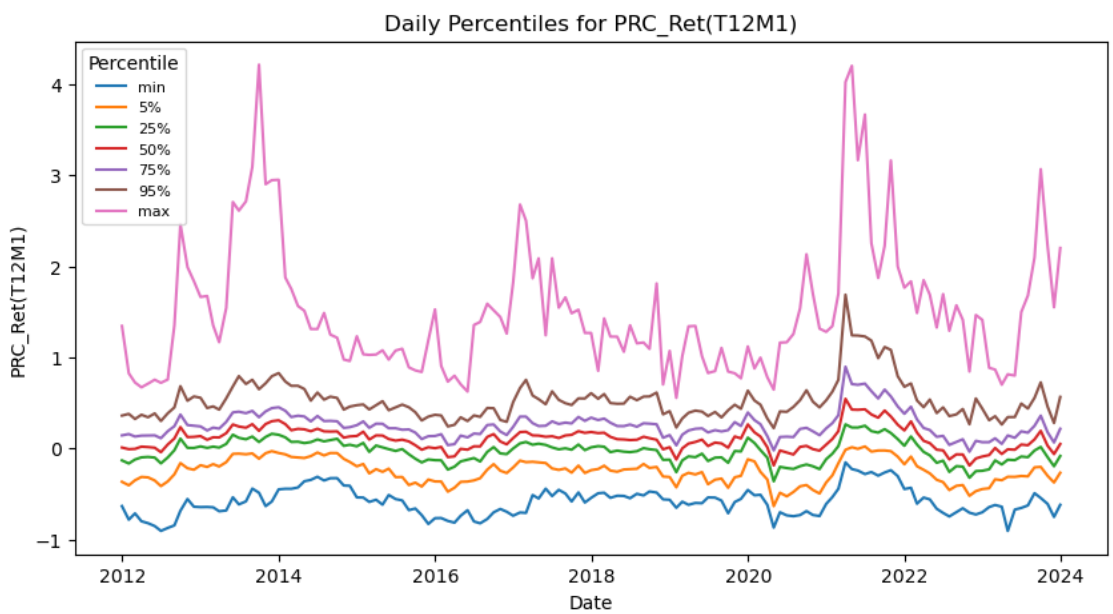


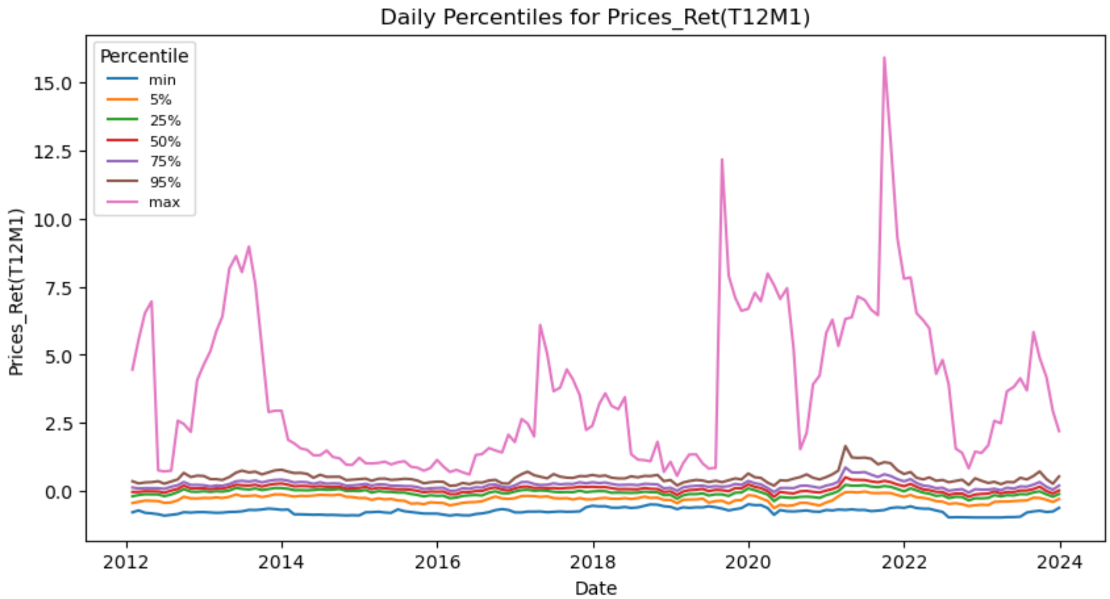
Daily Percentiles for Price_Ret(T1)
Percentile
min
5%
25%
50%
75%
95%
max
Price_Ret(T1)
Date



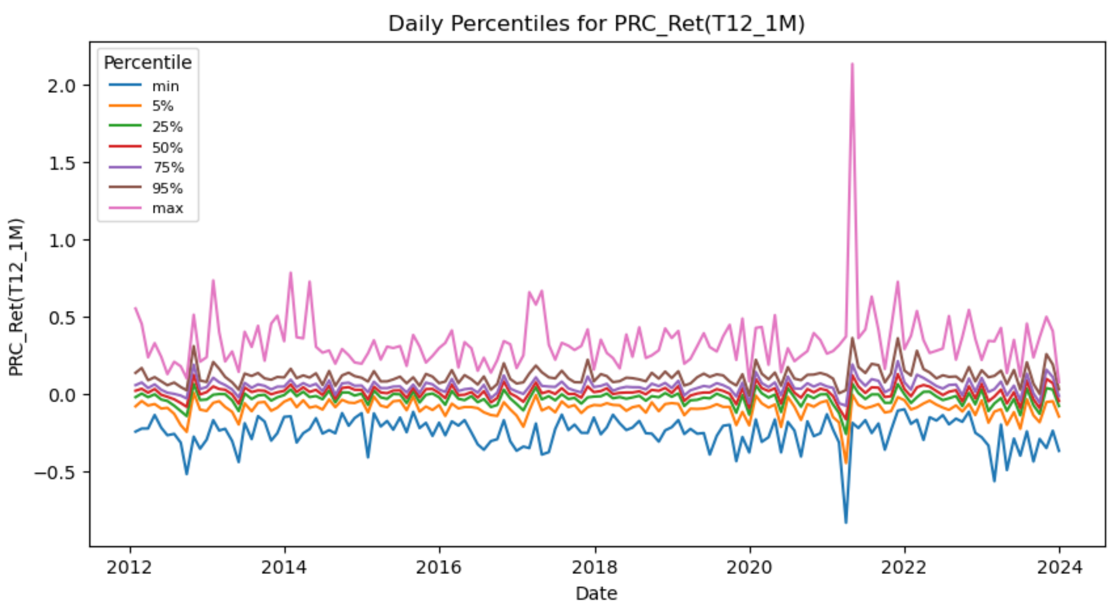
Daily Percentiles for PRC_Ret(T12)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(T12)
Date


Daily Percentiles for Prices_Ret(T12)
Percentile
min
5%
25%
50%
75%
95%
max
Prices_Ret(T12)
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2012
2014
2016
2018
2020
2022
2024
Date

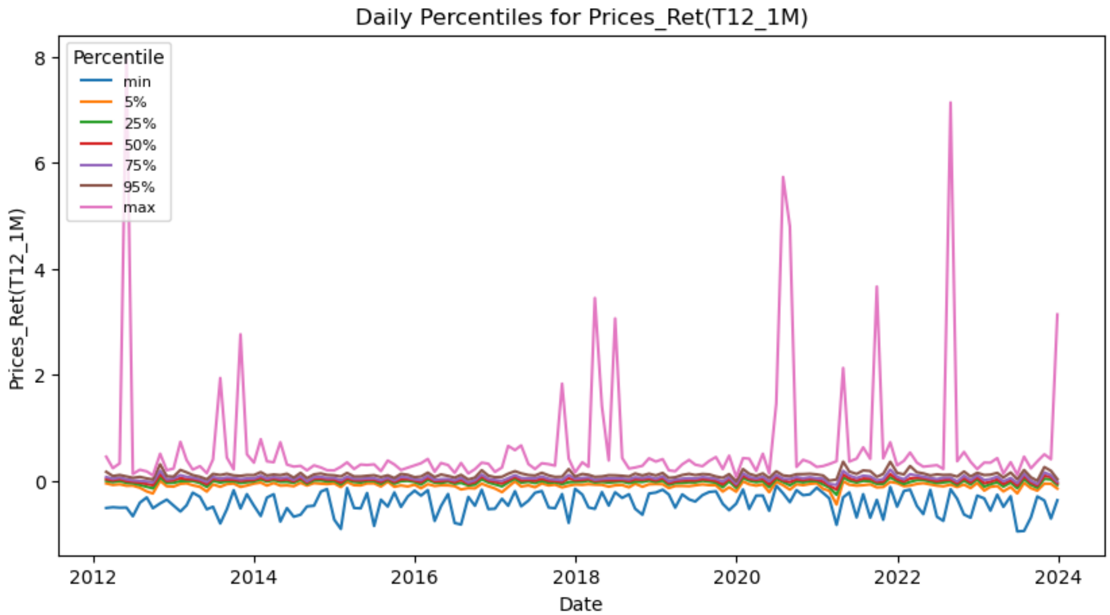

Daily Percentiles for PRC_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(T12M1)
4
3
2
1
0
−1
2012
2014
2016
2018
2020
2022
2024
Date

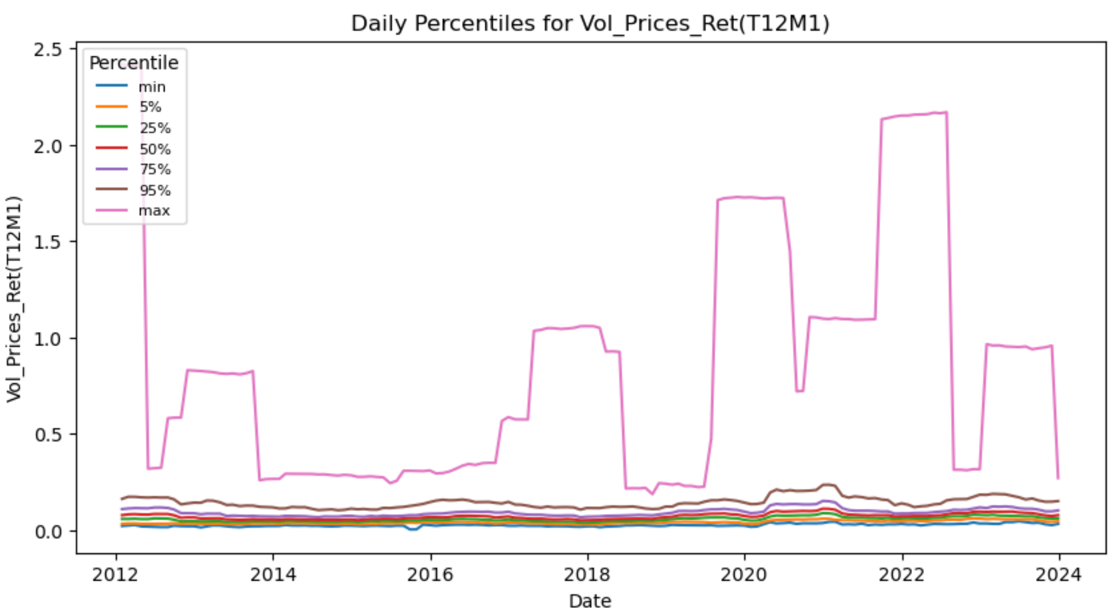


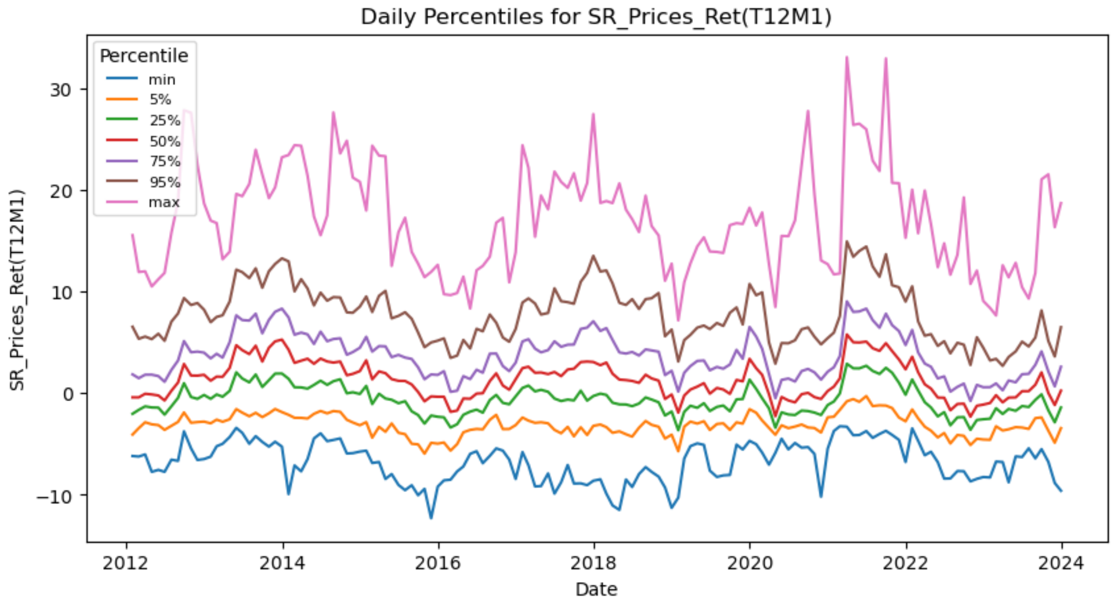
Daily Percentiles for Prices_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
Prices_Ret(T12M1)
15.0
12.5
10.0
7.5
5.0
2.5
0.0
2012
2014
2016
2018
2020
2022
2024
Date

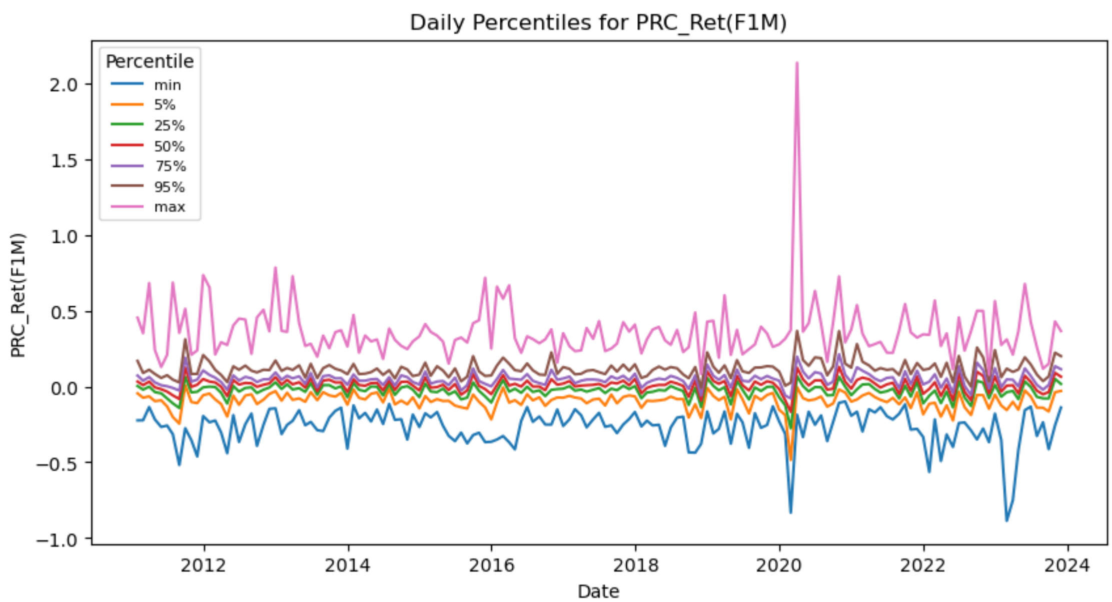


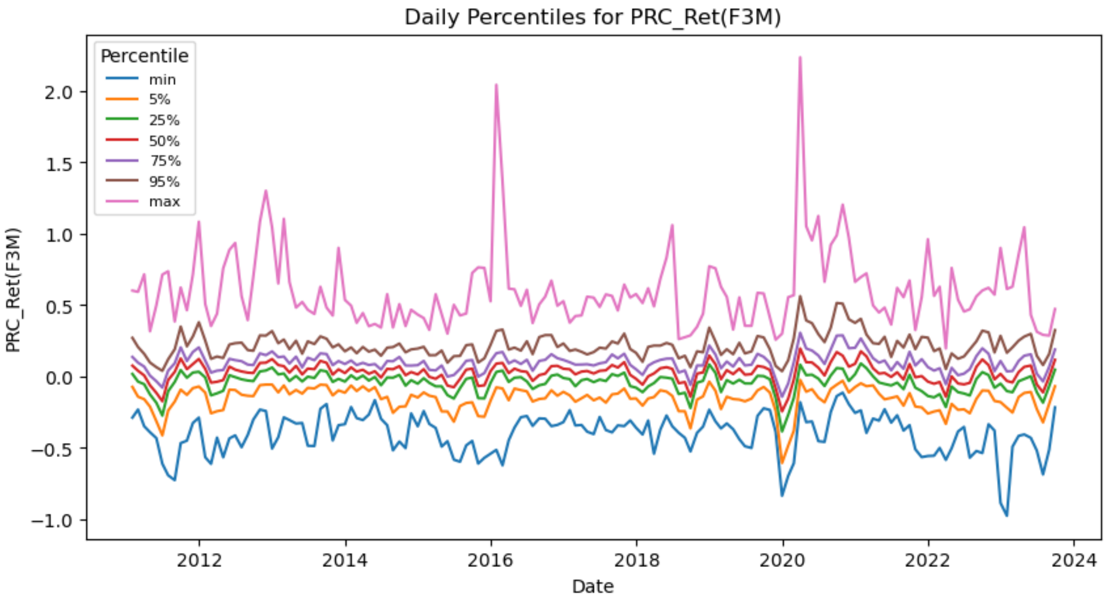
Daily Percentiles for PRC_Ret(T12_1M)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(T12_1M)
2.0
1.5
1.0
0.5
0.0
−0.5
2012
2014
2016
2018
2020
2022
2024
Date

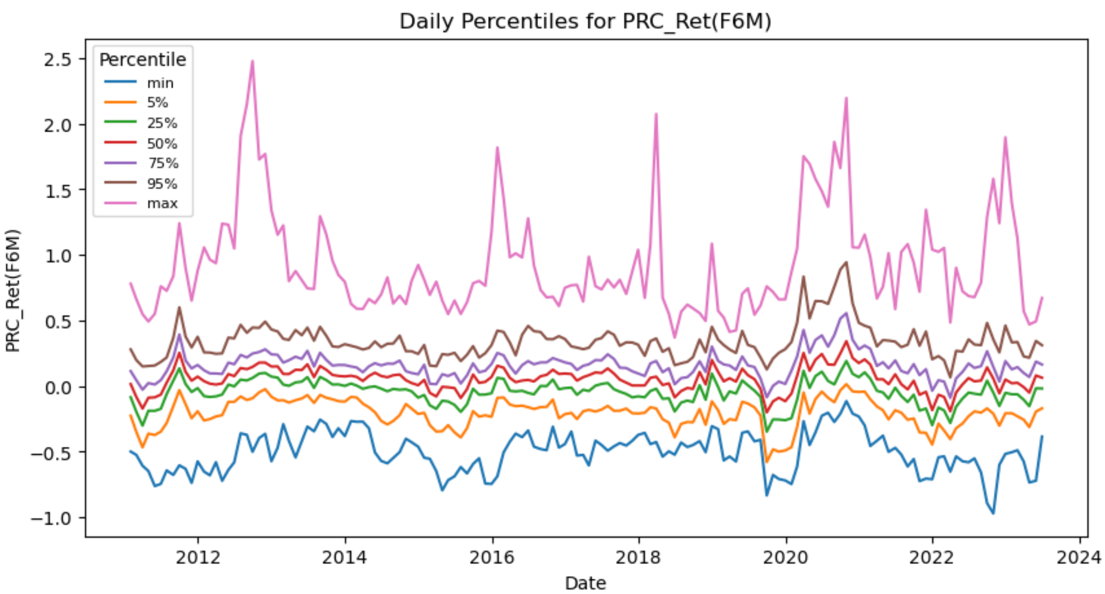

Daily Percentiles for Prices_Ret(T12_1M)
Percentile
min
5%
25%
50%
75%
95%
max
Prices_Ret(T12_1M)
Date


Daily Percentiles for Vol_Prices_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
Vol_Prices_Ret(T12M1)
Date


Daily Percentiles for SR_Prices_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
SR_Prices_Ret(T12M1)
Date
30
20
10
0
−10
2012
2014
2016
2018
2020
2022
2024


Daily Percentiles for PRC_Ret(F1M)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(F1M)
Date
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
2012
2014
2016
2018
2020
2022
2024






```
Price_Ret(T1):
  Median of the min percentile: -0.39167808645820357
  Median of the 5% percentile: -0.08980909941590991
  Median of the 25% percentile: -0.026080186843129516
  Median of the 50% percentile: 0.009840098400984099
  Median of the 75% percentile: 0.04500901155123471
  Median of the 95% percentile: 0.10878458921692565
```

```
  Median of the max percentile: 0.351092168353756

PRC_Ret(T12):
  Median of the min percentile: -0.6021358561790884
  Median of the 5% percentile: -0.2601085754336827
  Median of the 25% percentile: -0.029966241406540944
  Median of the 50% percentile: 0.11711134844528892
  Median of the 75% percentile: 0.2663292586916528
  Median of the 95% percentile: 0.5185331023066828
  Median of the max percentile: 1.4529489569958174

Prices_Ret(T12):
  Median of the min percentile: -0.7557145651081932
  Median of the 5% percentile: -0.30622521054976753
  Median of the 25% percentile: -0.05933295773842828
  Median of the 50% percentile: 0.08761902974020108
  Median of the 75% percentile: 0.23487339000570143
  Median of the 95% percentile: 0.4931673249181745
  Median of the max percentile: 3.0724807729108154

PRC_Ret(T12M1):
  Median of the min percentile: -0.5935068160952284
  Median of the 5% percentile: -0.25537950746545657
  Median of the 25% percentile: -0.027062382765922988
  Median of the 50% percentile: 0.10222840543367773
  Median of the 75% percentile: 0.23730038875006415
  Median of the 95% percentile: 0.48209692052144026
  Median of the max percentile: 1.34414294810801

Prices_Ret(T12M1):
  Median of the min percentile: -0.7517020803351337
  Median of the 5% percentile: -0.28932406616548423
  Median of the 25% percentile: -0.05661931829933811
  Median of the 50% percentile: 0.07559991404392996
  Median of the 75% percentile: 0.2093162870674058
  Median of the 95% percentile: 0.4624807893640177
  Median of the max percentile: 2.773236232603188

PRC_Ret(T12_1M):
  Median of the min percentile: -0.235482
  Median of the 5% percentile: -0.08451669999999999
  Median of the 25% percentile: -0.02132025
  Median of the 50% percentile: 0.01242625
  Median of the 75% percentile: 0.04806275
  Median of the 95% percentile: 0.1090796
  Median of the max percentile: 0.3072145

Prices_Ret(T12_1M):
```

```
  Median of the min percentile: -0.4043392504930967
  Median of the 5% percentile: -0.0869153216034124
  Median of the 25% percentile: -0.022640108870468856
  Median of the 50% percentile: 0.010797638145016442
  Median of the 75% percentile: 0.04478668054527535
  Median of the 95% percentile: 0.1072377184028884
  Median of the max percentile: 0.33552521058820717

Vol_Prices_Ret(T12M1):
  Median of the min percentile: 0.023235423243071512
  Median of the 5% percentile: 0.036986464081294496
  Median of the 25% percentile: 0.052832811519830866
  Median of the 50% percentile: 0.06965348806941386
  Median of the 75% percentile: 0.0883775945724741
  Median of the 95% percentile: 0.1380265945113937
  Median of the max percentile: 0.8108228070518557

SR_Prices_Ret(T12M1):
  Median of the min percentile: -6.536362197934961
  Median of the 5% percentile: -3.1516797428505585
  Median of the 25% percentile: -0.783304308343572
  Median of the 50% percentile: 1.2254821693188644
  Median of the 75% percentile: 3.6666254972570447
  Median of the 95% percentile: 7.626656448053435
  Median of the max percentile: 16.716384046487384

PRC_Ret(F1M):
  Median of the min percentile: -0.250578
  Median of the 5% percentile: -0.0877414
  Median of the 25% percentile: -0.021781000000000002
  Median of the 50% percentile: 0.011755
  Median of the 75% percentile: 0.0472615
  Median of the 95% percentile: 0.10952209999999996
  Median of the max percentile: 0.335631

PRC_Ret(F3M):
  Median of the min percentile: -0.3746256641295582
  Median of the 5% percentile: -0.14092031136842478
  Median of the 25% percentile: -0.030480130694339747
  Median of the 50% percentile: 0.03312106180321894
  Median of the 75% percentile: 0.10005245330751539
  Median of the 95% percentile: 0.20641178984396635
  Median of the max percentile: 0.5561335475798863

PRC_Ret(F6M):
  Median of the min percentile: -0.5019132744404435
  Median of the 5% percentile: -0.19659327999721904
  Median of the 25% percentile: -0.038681625917077356
```

```
Median of the 50% percentile: 0.058503106264278 57
Median of the 75% percentile: 0.1572815415464443
Median of the 95% percentile: 0.31806958610497854
Median of the max percentile: 0.7977277070728639
```

normalization

```
[ ]: norm_df = df.copy()
```

```
[ ]: def normalization(x):
         return (x - x.mean()) / x.std()

     # Apply the cross-sectional normalization
     for var in var_list:
         norm_df[var] = norm_df.groupby(norm_df.index)[var].transform(normalization)
```

```
[ ]: norm_df.describe()
```

```
[ ]:             permno          price       shrout          prc          mcap  \
     count  78481.000000   78481.000000  7.848100e+04  78440.000000  7.848100e+04
     mean   55562.798716     111.790334  6.116220e+05      0.010668  4.677985e+07
     std    29005.295471     206.740954  1.116013e+06      0.085216  1.115109e+08
     min    10104.000000       2.050000  3.179000e+03     -0.886269  6.203644e+05
     25%    24010.000000      39.885000  1.591360e+05     -0.035815  1.084019e+07
     50%    60943.000000      67.800000  3.004150e+05      0.011156  1.966517e+07
     75%    83143.000000     117.410000  5.844920e+05      0.055954  4.184465e+07
     max    93436.000000    7000.450200  2.920640e+07      2.135168  3.071345e+09

            Price_Ret(T1)  PRC_Ret(T12)  Prices_Ret(T12)  PRC_Ret(T12M1)  \
     count   7.772000e+04  7.024300e+04     6.955400e+04    7.024300e+04
     mean   -4.022630e-18  2.427719e-18    -3.269024e-18   -2.427719e-18
     std     9.990088e-01  9.989744e-01     9.989715e-01    9.989744e-01
     min    -1.233978e+01 -4.415152e+00    -4.617517e+00   -4.248586e+00
     25%    -4.943571e-01 -6.118526e-01    -4.984432e-01   -6.093929e-01
     50%    -3.180546e-03 -5.660440e-02    -5.032311e-02   -5.377272e-02
     75%     4.960473e-01  5.392211e-01     4.226046e-01    5.435153e-01
     max     2.202466e+01  1.301485e+01     2.006791e+01    1.317903e+01

            Prices_Ret(T12M1)  PRC_Ret(T12_1M)  Prices_Ret(T12_1M)  \
     count       6.955400e+04     6.951900e+04        6.883100e+04
     mean       -3.269024e-18    -2.044168e-18       -3.096901e-18
     std         9.989715e-01     9.989710e-01        9.989679e-01
     min        -4.924631e+00    -6.740546e+00       -1.267789e+01
     25%        -4.978291e-01    -5.807216e-01       -4.985315e-01
     50%        -4.794963e-02    -1.939954e-03       -8.548682e-04
     75%         4.281541e-01     5.707045e-01        5.043007e-01
     max         2.049279e+01     1.332261e+01        2.167890e+01
```

```
       Vol_Prices_Ret(T12M1)  SR_Prices_Ret(T12M1)  PRC_Ret(F1M)  \
count           6.955400e+04          69554.000000  7.772000e+04
mean           -2.451768e-17              0.000000 -2.742702e-18
std             9.989715e-01              0.998971  9.990088e-01
min            -2.096707e+00             -3.859849 -1.063666e+01
25%            -4.732970e-01             -0.692103 -5.712042e-01
50%            -1.752337e-01             -0.115771 -2.809995e-03
75%             2.319764e-01              0.577867  5.594344e-01
max             2.109337e+01              7.696053  1.321799e+01

       PRC_Ret(F3M)  PRC_Ret(F6M)
count  7.620700e+04  7.396800e+04
mean   6.340219e-18 -3.458190e-18
std    9.990022e-01  9.989923e-01
min   -7.420586e+00 -5.197883e+00
25%   -5.989960e-01 -6.117191e-01
50%   -1.338180e-02 -2.993152e-02
75%    5.790098e-01  5.667353e-01
max    1.186595e+01  1.171013e+01
```

[ ]:
```
norm_df[norm_df['permno'] == 13688].tail(20)
```

[ ]:
```
            permno  price   shrout       prc         mcap  Price_Ret(T1)  \
date
2018-08-31   13688  46.18   517151  0.071959  23882033.18       0.838340
2018-09-28   13688  46.01   517151 -0.003681  23794117.51      -0.062808
2018-10-31   13688  46.81   517151  0.017388  24207838.31       1.058523
2018-11-30   13688  26.38   518674 -0.436445  13682620.12      -5.567856
2018-12-31   13688  23.75   518674 -0.099697  12318507.50       0.049782
2022-10-31   13688  14.93  1987700  0.194400  29676361.00      -4.578703
2022-11-30   13688  15.70  1987700  0.051574  31206890.00      -0.136373
2022-12-30   13688  16.26  1987700  0.035669  32320002.00       0.522297
2023-01-31   13688  15.90  1987700 -0.022140  31604430.00      -1.038114
2023-02-28   13688  15.62  1987700 -0.017610  31047874.00       0.285922
2023-03-31   13688  16.17  1987785  0.035211  32142483.45       0.434698
2023-04-28   13688  17.11  1987785  0.058132  34011001.35       0.807505
2023-05-31   13688  16.94  1995778 -0.009936  33808479.32       0.344636
2023-06-30   13688  17.28  1995778  0.020071  34487043.84      -0.743447
2023-07-31   13688  17.61  2568985  0.019097  45239825.85      -0.215000
2023-08-31   13688  16.30  2091241 -0.074390  34087228.30      -0.526028
2023-09-29   13688  16.13  2091241 -0.010430  33731717.33       0.819754
2023-10-31   13688  16.30  2133508  0.010539  34776180.40       0.701242
2023-11-30   13688  17.17  2133508  0.053374  36632332.36      -0.455822
2023-12-29   13688  18.03  2133508  0.050670  38467149.24      -0.118962

            PRC_Ret(T12)  Prices_Ret(T12)  PRC_Ret(T12M1)  Prices_Ret(T12M1)  \
date
```

2018-08-31	-1.940569	-1.809446	-2.278698	-2.125198
2018-09-28	-1.844704	-1.703063	-1.923185	-1.781724
2018-10-31	-1.017202	-0.908586	-1.336724	-1.218756
2018-11-30	-2.476929	-2.326814	-0.712910	-0.602887
2018-12-31	-1.837462	-1.718570	-2.006677	-1.869544
2022-10-31	-0.845862	-1.792281	-1.141477	-0.999701
2022-11-30	-0.900287	-1.878631	-0.819356	-1.775541
2022-12-30	-0.690041	-1.722367	-0.923627	-1.906484
2023-01-31	-1.521615	-2.383467	-1.096665	-2.103911
2023-02-28	-1.359490	-2.292594	-1.472089	-2.370850
2023-03-31	-1.058395	-1.934625	-1.315498	-2.325002
2023-04-28	-1.128450	-2.092471	-1.353444	-2.076127
2023-05-31	-1.146777	-1.776124	-1.386558	-2.058255
2023-06-30	-1.435482	-2.077505	-1.427737	-2.057421
2023-07-31	-1.462780	-2.149704	-1.406606	-2.165069
2023-08-31	0.278687	-1.268839	0.523082	-1.209264
2023-09-29	0.467702	-1.311946	0.283370	-1.337195
2023-10-31	0.288897	0.312691	0.095761	0.136870
2023-11-30	0.234010	0.265630	0.412329	0.411065
2023-12-29	-0.175711	-0.092587	-0.122094	-0.032014

	PRC_Ret(T12_1M)	Prices_Ret(T12_1M)	Vol_Prices_Ret(T12M1) \
date			
2018-08-31	0.781315	0.816316	0.316145
2018-09-28	-0.860955	-0.857825	0.644607
2018-10-31	-2.502201	-2.240800	0.336968
2018-11-30	-1.499018	-1.463835	0.087739
2018-12-31	-3.411106	-3.314734	2.592583
2022-10-31	-1.411569	-1.202048	1.340508
2022-11-30	-0.090407	-0.035507	1.984494
2022-12-30	0.085009	0.109053	1.722606
2023-01-31	0.960206	0.971058	1.225713
2023-02-28	-0.602936	-0.570525	1.152623
2023-03-31	-0.573437	-0.533598	1.170932
2023-04-28	0.947464	0.964085	1.199831
2023-05-31	0.715837	0.745695	1.233189
2023-06-30	1.153905	0.987908	1.376828
2023-07-31	-0.878636	-0.627945	1.454083
2023-08-31	-6.014655	-5.086610	0.523787
2023-09-29	-0.118207	-0.081338	0.601206
2023-10-31	0.983379	-4.650653	-0.888084
2023-11-30	-0.260586	-0.169683	-0.882816
2023-12-29	1.521040	0.512341	-1.364996

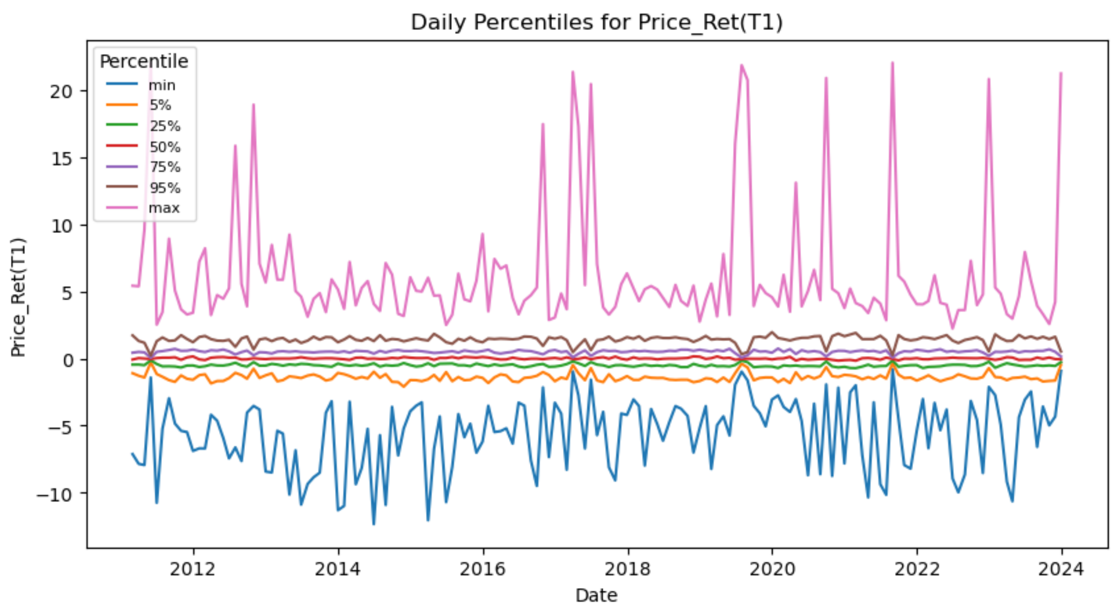
	SR_Prices_Ret(T12M1)	PRC_Ret(F1M)	PRC_Ret(F3M)	PRC_Ret(F6M)
date				
2018-08-31	-2.047335	-0.097830	-3.221805	-2.328881

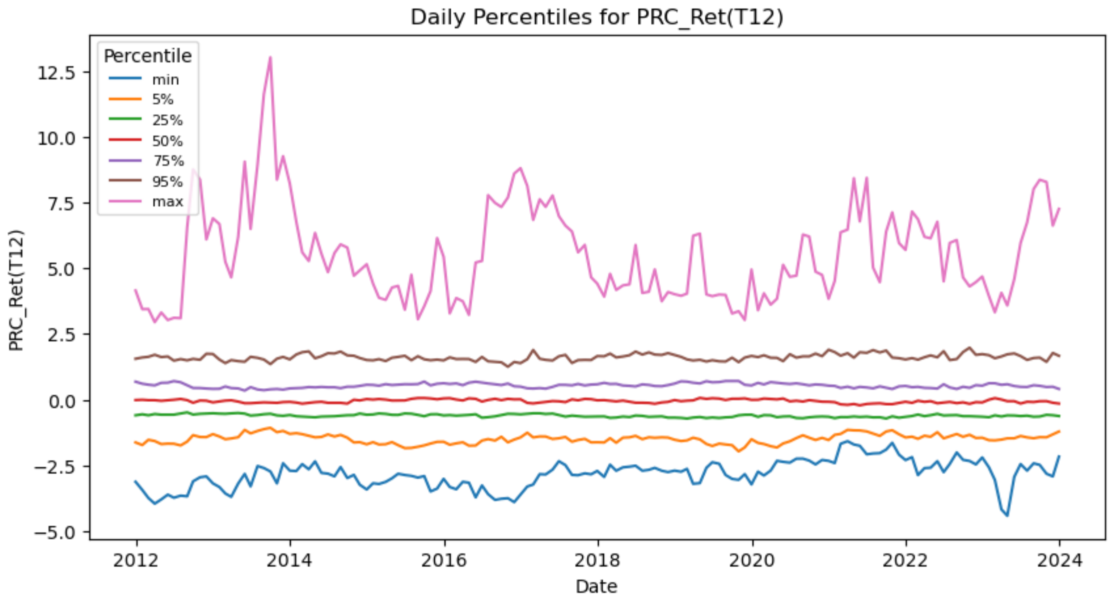
2018-09-28	-1.594592	1.044488	-2.700666	-1.979743
2018-10-31	-1.218128	-5.887187	-3.946722	-2.881442
2018-11-30	-0.605428	0.009959	1.073157	0.763731
2018-12-31	-0.997873	1.252255	1.237598	0.562316
2022-10-31	-0.450277	-0.214798	-0.200492	0.448685
2022-11-30	-0.992079	1.501962	0.075840	0.687993
2022-12-30	-1.101851	-1.055048	-0.272488	-0.101246
2023-01-31	-1.190937	0.253201	0.851233	0.364914
2023-02-28	-1.704291	0.411378	0.832687	-0.003445
2023-03-31	-1.439759	0.793134	0.211025	0.027878
2023-04-28	-1.502387	0.322579	-0.325374	0.006359
2023-05-31	-1.520544	-0.770847	-0.984749	-0.375167
2023-06-30	-1.584903	-0.240620	-0.176069	-0.157439
2023-07-31	-1.583424	-0.634540	0.320555	NaN
2023-08-31	-1.483973	0.784113	0.532947	NaN
2023-09-29	-1.456317	0.686640	-0.012088	NaN
2023-10-31	0.585127	-0.491400	NaN	NaN
2023-11-30	0.734855	-0.264917	NaN	NaN
2023-12-29	0.202221	NaN	NaN	NaN

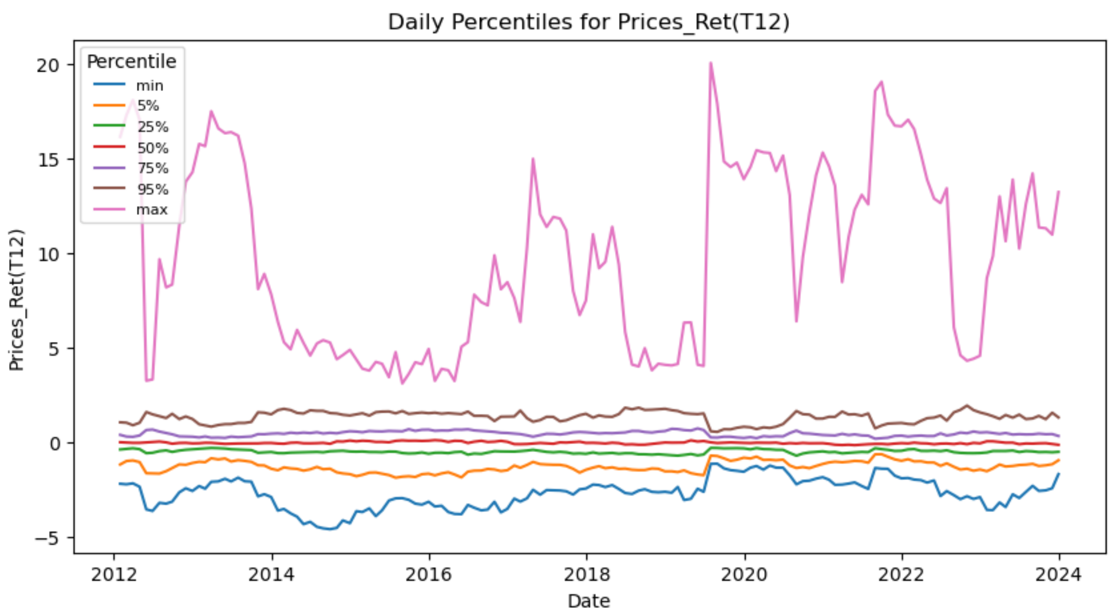
plot the time series of different variables (normalized)

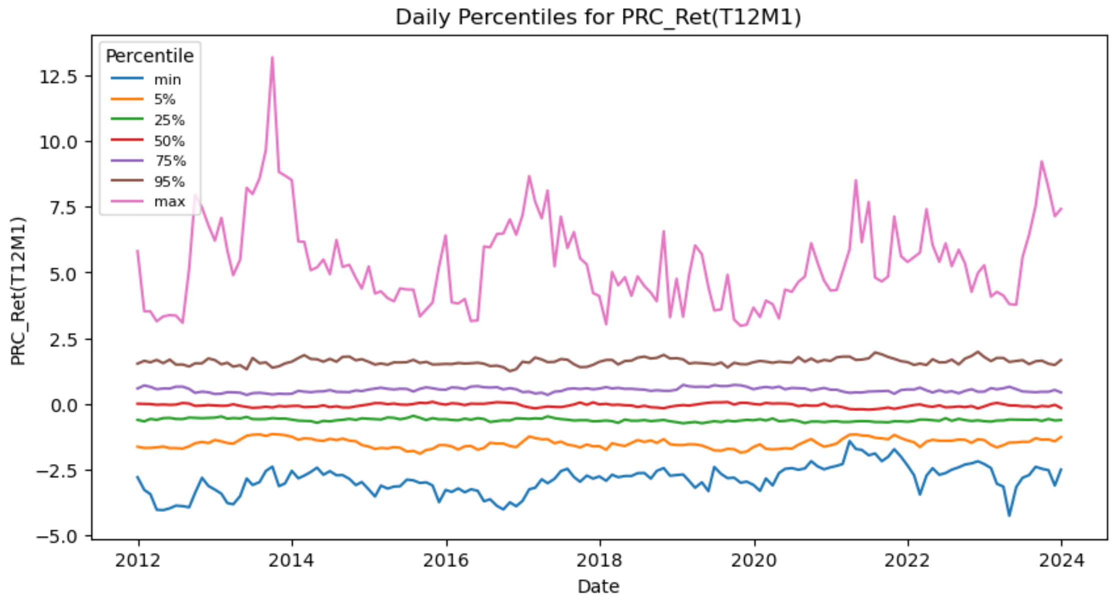
[ ]:
```
medians_dict = {}

for var in var_list:
    # Calculate percentiles
    percentiles_df = norm_df.groupby('date')[var].describe(percentiles=[0.0, 0.
↪05, 0.25, 0.5, 0.75, 0.95, 1.0]).reset_index()
    percentiles_df = percentiles_df[['date', 'min', '5%', '25%', '50%', '75%',␣
↪'95%', 'max']]

    # Melt the dataframe
    melted_df = percentiles_df.melt(id_vars=['date'], var_name='Percentile',␣
↪value_name=var)

    # Plotting
    plt.figure(figsize=(10, 5))
    sns.lineplot(data=melted_df, x='date', y=var, hue='Percentile')
    plt.title(f'Daily Percentiles for {var}')
    plt.xlabel('Date')
    plt.ylabel(var)
    plt.legend(title='Percentile',  fontsize="8", loc ="upper left")
    plt.show()

    # Calculating median values for each percentile
    medians_dict[var] = {}
    for percentile in ['min', '5%', '25%', '50%', '75%', '95%', 'max']:
```

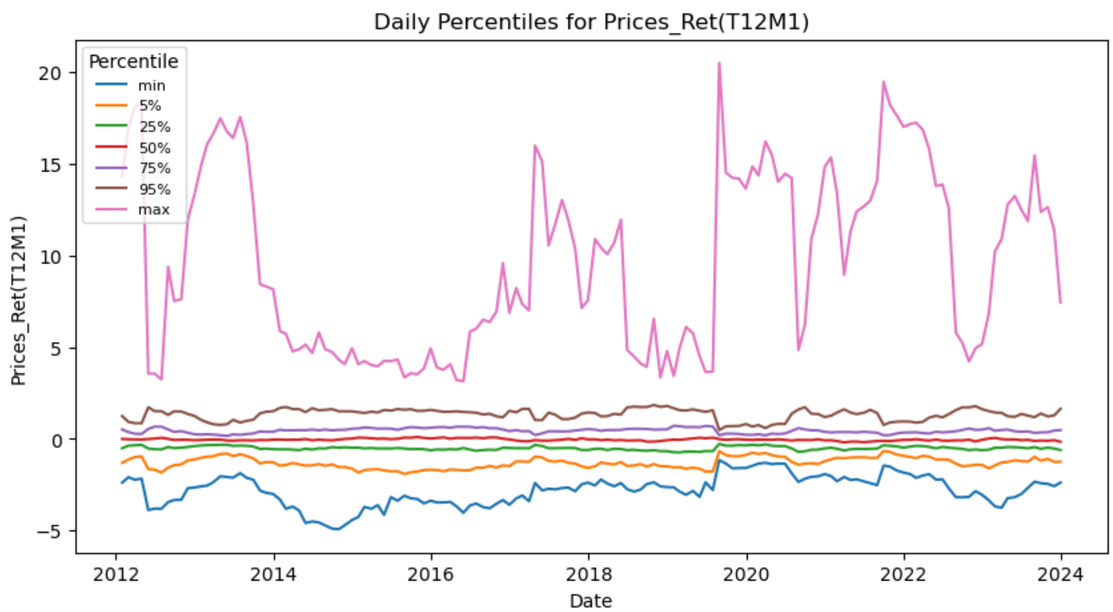
```
        # Filter the dataframe for the current percentile
        filtered_df = melted_df[melted_df['Percentile'] == percentile]
        # Calculate the median of these values
        median_value = filtered_df[var].median()
        medians_dict[var][percentile] = median_value

# Print median values for each percentile of each variable
for var, percentiles in medians_dict.items():
    print(f"\n{var}:")
    for percentile, median_value in percentiles.items():
        print(f"  Median of the {percentile} percentile: {median_value}")
```

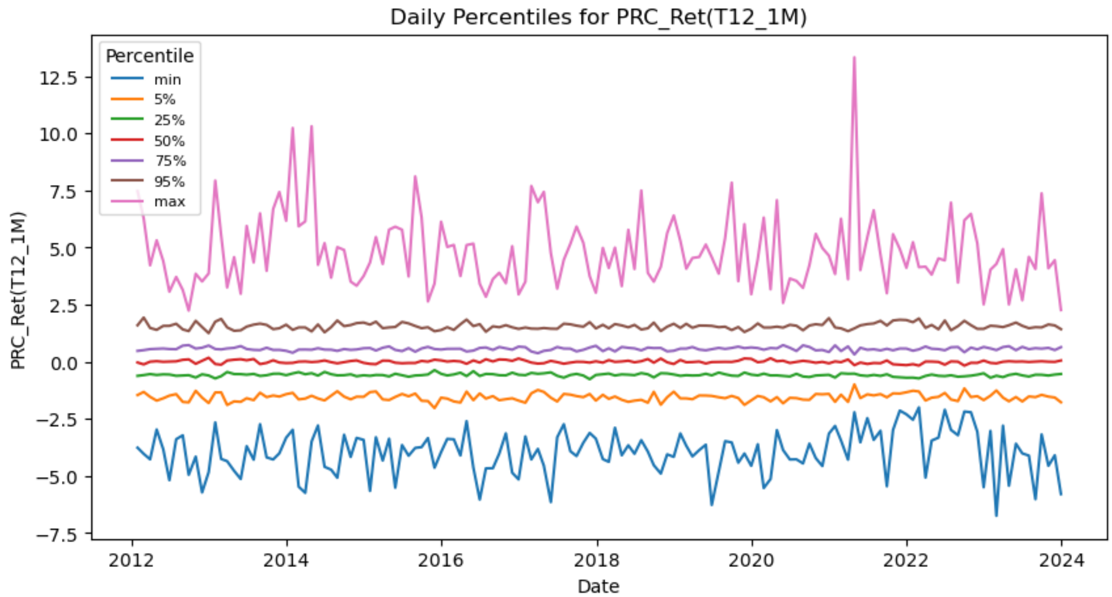


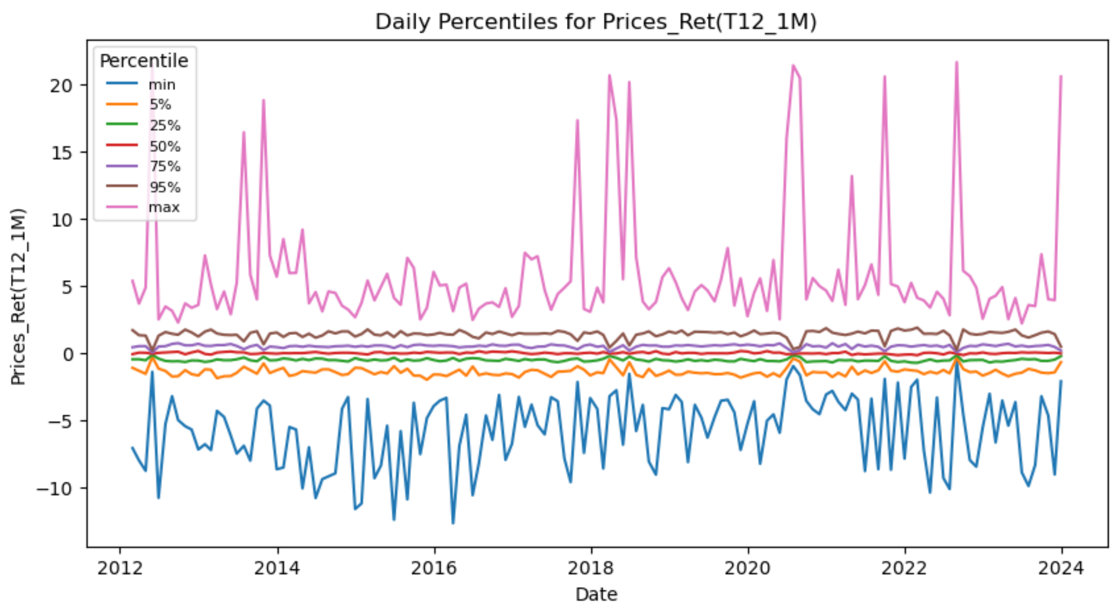

Daily Percentiles for PRC_Ret(T12)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(T12)
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
2012
2014
2016
2018
2020
2022
2024
Date

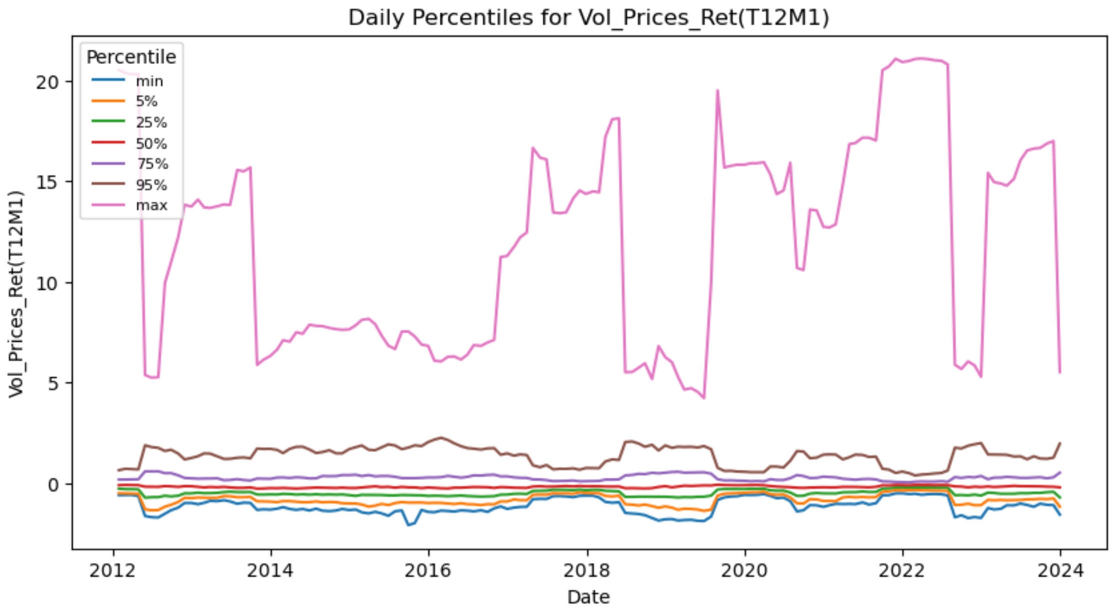

Daily Percentiles for Prices_Ret(T12)
Percentile
min
5%
25%
50%
75%
95%
max
Prices_Ret(T12)
20
15
10
5
0
−5
2012
2014
2016
2018
2020
2022
2024
Date

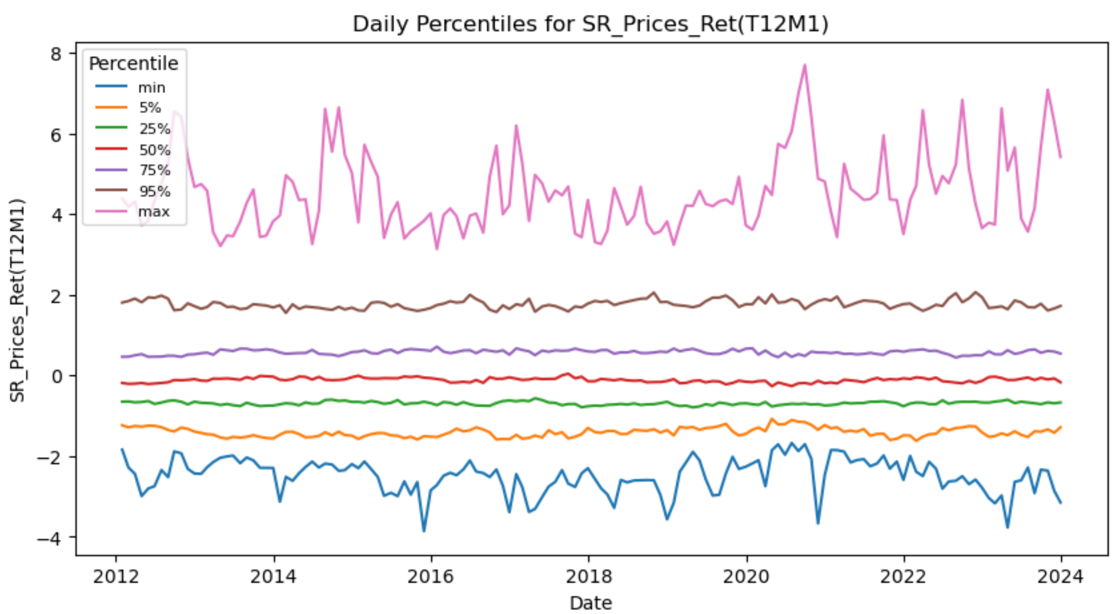

Daily Percentiles for PRC_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(T12M1)
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
2012
2014
2016
2018
2020
2022
2024
Date

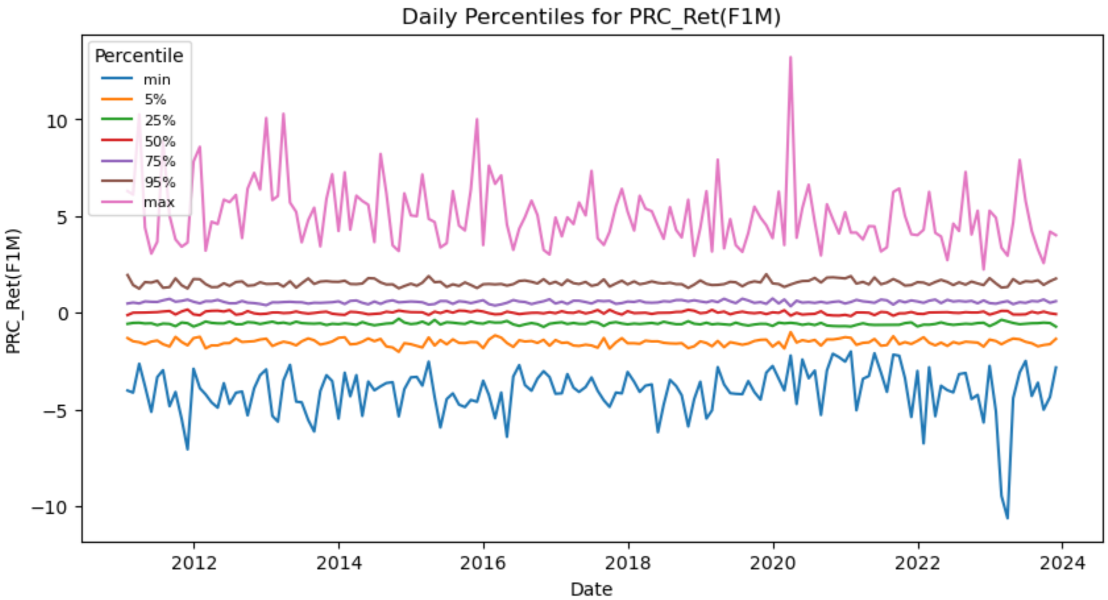

Daily Percentiles for Prices_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
Prices_Ret(T12M1)
20
15
10
5
0
−5
2012
2014
2016
2018
2020
2022
2024
Date

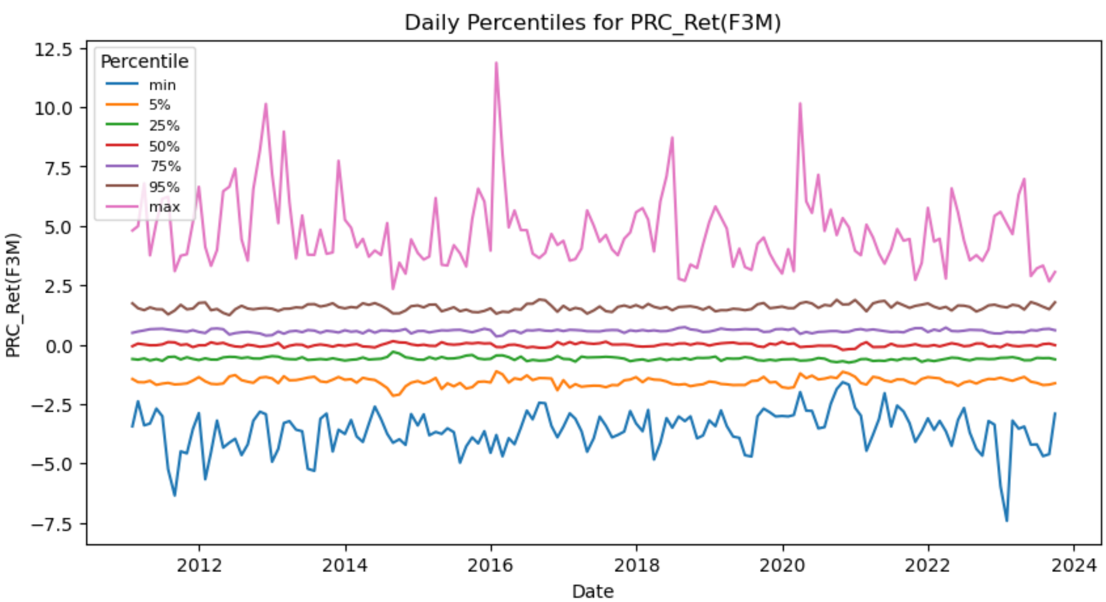

Daily Percentiles for PRC_Ret(T12_1M)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(T12_1M)
Date

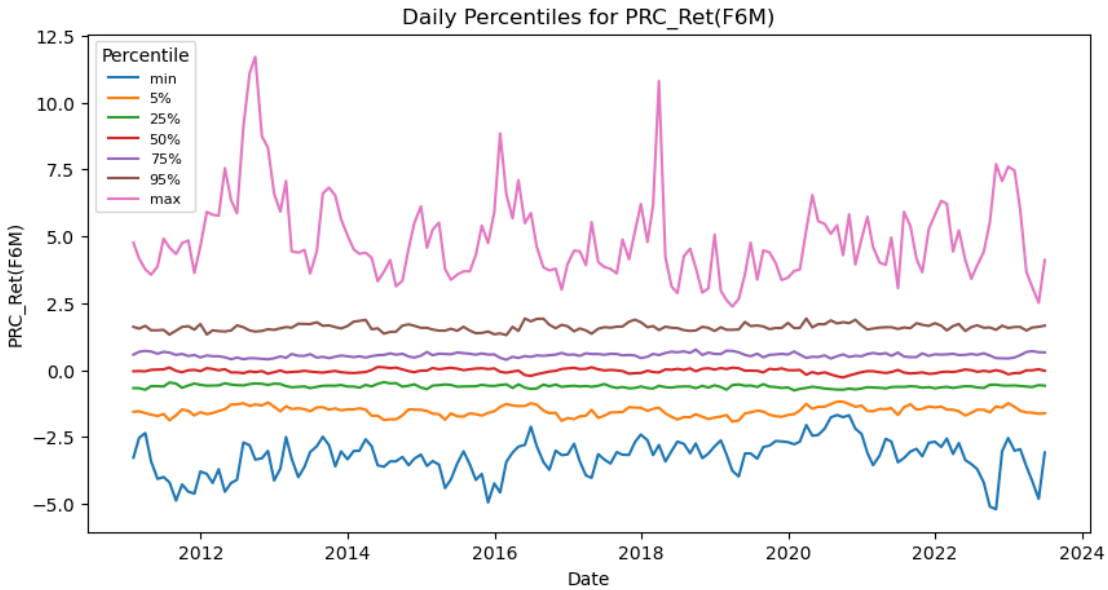

Daily Percentiles for Prices_Ret(T12_1M)
Percentile
min
5%
25%
50%
75%
95%
max
Prices_Ret(T12_1M)
Date


Daily Percentiles for Vol_Prices_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
20
15
10
5
0
Vol_Prices_Ret(T12M1)
2012
2014
2016
2018
2020
2022
2024
Date


Daily Percentiles for SR_Prices_Ret(T12M1)
Percentile
min
5%
25%
50%
75%
95%
max
8
6
4
2
0
−2
−4
SR_Prices_Ret(T12M1)
2012
2014
2016
2018
2020
2022
2024
Date


Daily Percentiles for PRC_Ret(F1M)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(F1M)
10
5
0
−5
−10
2012
2014
2016
2018
2020
2022
2024
Date


Daily Percentiles for PRC_Ret(F3M)
Percentile
min
5%
25%
50%
75%
95%
max
PRC_Ret(F3M)
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
2012
2014
2016
2018
2020
2022
2024
Date




```
Price_Ret(T1):
  Median of the min percentile: -5.198133709689026
  Median of the 5% percentile: -1.4437030258095545
  Median of the 25% percentile: -0.524688685724891
  Median of the 50% percentile: -0.0014935953394055615
  Median of the 75% percentile: 0.5342270934485629
  Median of the 95% percentile: 1.447557540184519
  Median of the max percentile: 4.838470475531766

PRC_Ret(T12):
  Median of the min percentile: -2.8122137017625346
  Median of the 5% percentile: -1.467993990581814
  Median of the 25% percentile: -0.6091559104894488
  Median of the 50% percentile: -0.05685047870553174
  Median of the 75% percentile: 0.5425013903778502
  Median of the 95% percentile: 1.6005720228106035
  Median of the max percentile: 5.149057318241457

Prices_Ret(T12):
  Median of the min percentile: -2.623484856196533
  Median of the 5% percentile: -1.315648860201541
  Median of the 25% percentile: -0.5263023796097035
  Median of the 50% percentile: -0.052164780835206134
  Median of the 75% percentile: 0.44391778732736
  Median of the 95% percentile: 1.3880584390999817
  Median of the max percentile: 9.737366764428469
```

```
PRC_Ret(T12M1):
  Median of the min percentile: -2.8301301754108787
  Median of the 5% percentile: -1.4926444633821028
  Median of the 25% percentile: -0.6098870236847719
  Median of the 50% percentile: -0.05389505337245977
  Median of the 75% percentile: 0.5514059392472089
  Median of the 95% percentile: 1.6040604838220167
  Median of the max percentile: 5.155394880887796

Prices_Ret(T12M1):
  Median of the min percentile: -2.693534184992621
  Median of the 5% percentile: -1.3341952696892703
  Median of the 25% percentile: -0.5193179568822941
  Median of the 50% percentile: -0.050156090692311334
  Median of the 75% percentile: 0.46044961079220365
  Median of the 95% percentile: 1.4188482241519078
  Median of the max percentile: 9.158296155924498

PRC_Ret(T12_1M):
  Median of the min percentile: -3.9413278721984
  Median of the 5% percentile: -1.55079159719361
  Median of the 25% percentile: -0.586017149784201
  Median of the 50% percentile: 0.0007278370506201418
  Median of the 75% percentile: 0.5698792839467659
  Median of the 95% percentile: 1.5437454768847636
  Median of the max percentile: 4.512428595710889

Prices_Ret(T12_1M):
  Median of the min percentile: -5.368377826968604
  Median of the 5% percentile: -1.462847497269528
  Median of the 25% percentile: -0.5299846673532286
  Median of the 50% percentile: 0.0008829260397438713
  Median of the 75% percentile: 0.5374256285206424
  Median of the 95% percentile: 1.4617555628557535
  Median of the max percentile: 4.592530937531173

Vol_Prices_Ret(T12M1):
  Median of the min percentile: -1.1862347612492783
  Median of the 5% percentile: -0.8603998246451678
  Median of the 25% percentile: -0.5352320465946132
  Median of the 50% percentile: -0.19496642714764345
  Median of the 75% percentile: 0.24956778785218497
  Median of the 95% percentile: 1.4122652895112808
  Median of the max percentile: 12.788111557892844

SR_Prices_Ret(T12M1):
  Median of the min percentile: -2.425651831754161
  Median of the 5% percentile: -1.4103678600567084
```

```
  Median of the 25% percentile: -0.6900796465285024
  Median of the 50% percentile: -0.11123105018101062
  Median of the 75% percentile: 0.5744054197609173
  Median of the 95% percentile: 1.758574158946772
  Median of the max percentile: 4.344820072932867

PRC_Ret(F1M):
  Median of the min percentile: -4.021855099907964
  Median of the 5% percentile: -1.5413294821610142
  Median of the 25% percentile: -0.5694740287258351
  Median of the 50% percentile: -0.0030454869953410003
  Median of the 75% percentile: 0.5572185492712635
  Median of the 95% percentile: 1.519735239806513
  Median of the max percentile: 4.788424663800764

PRC_Ret(F3M):
  Median of the min percentile: -3.4982550225861266
  Median of the 5% percentile: -1.5560498359342942
  Median of the 25% percentile: -0.6043054757486624
  Median of the 50% percentile: -0.011701510283801333
  Median of the 75% percentile: 0.5813402540711379
  Median of the 95% percentile: 1.5671776796146417
  Median of the max percentile: 4.430211130750474

PRC_Ret(F6M):
  Median of the min percentile: -3.1576765948687218
  Median of the 5% percentile: -1.525464800769815
  Median of the 25% percentile: -0.6098176122325272
  Median of the 50% percentile: -0.028150177943277784
  Median of the 75% percentile: 0.5693101818271424
  Median of the 95% percentile: 1.5930240459374843
  Median of the max percentile: 4.524673937869889
```

6.0.1 Q7 Fama-McBeth Cross-sectional Regression

a)

[ ]:
```
# Period: October 31st 2019 to November 30th 2019
# dependent var: PRC_Ret(F1M)
# independent variables: PRC_Ret(T12M1)

import statsmodels.api as sm

date = norm_df.index[norm_df.index <= '2019-11-30'].max()
print(date)
nov_2019_df = norm_df.loc[date].copy()
nov_2019_df.dropna(inplace=True)
```

```python
X = sm.add_constant(nov_2019_df['PRC_Ret(T12M1)'])
Y = nov_2019_df['PRC_Ret(F1M)']

print(nov_2019_df.shape)

model = sm.OLS(Y, X).fit()
print(model.summary())
```

```
2019-11-29 00:00:00
(472, 17)
                            OLS Regression Results
==============================================================================
Dep. Variable:            PRC_Ret(F1M)   R-squared:                       0.035
Model:                             OLS   Adj. R-squared:                  0.033
Method:                  Least Squares   F-statistic:                     17.26
Date:                 Tue, 09 Apr 2024   Prob (F-statistic):           3.87e-05
Time:                         09:03:36   Log-Likelihood:                -651.82
No. Observations:                  472   AIC:                             1308.
Df Residuals:                      470   BIC:                             1316.
Df Model:                            1
Covariance Type:             nonrobust
================================================================================
==
                     coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--
const             -0.0167      0.044     -0.376      0.707      -0.104
0.071
PRC_Ret(T12M1)    -0.1907      0.046     -4.155      0.000      -0.281
-0.101
==============================================================================
Omnibus:                       69.236   Durbin-Watson:                   1.967
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              179.860
Skew:                           0.729   Prob(JB):                     8.79e-40
Kurtosis:                       5.649   Cond. No.                         1.05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

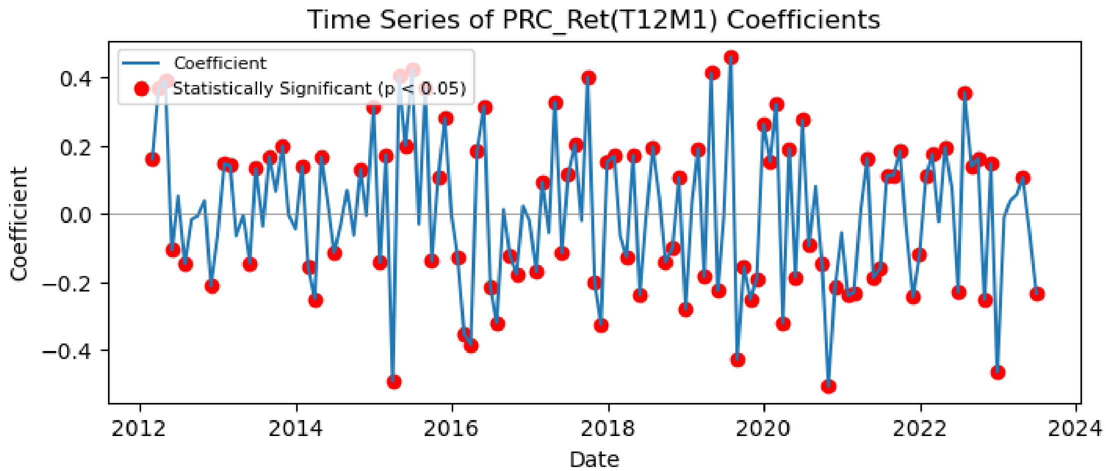
- **Economic Interpretation:** the negative coefficient of -0.1907 for PRC_Ret(T12M1) suggests a reversal effect, where stocks that had higher returns in the past eleven months (excluding the most recent month) tend to have lower returns in the following month. This could indicate that stocks which experienced an increase over the past eleven months could be overbought and may be subject to price corrections in the short-term future.
- **Statistical Interpretation:**

- The coefficient for PRC_Ret(T12M1) is -0.1907, with a standard error of 0.046. The negative coefficient suggests that there is an inverse relationship between the past returns (excluding the most recent month) and the forward one month returns.
- The t-statistic for PRC_Ret(T12M1) is -4.155 and the p-value is 0.000, indicating that the relationship is statistically significant at conventional levels ($p < 0.05$).
- The R-squared of the model is 0.035, which means that approximately 3.5% of the variability in forward one month returns can be explained by the past eleven months of returns. This suggests that while the model has found a significant relationship, it explains a relatively small portion of the variance in future returns.

b) Cross-sectinal regression

[ ]:
```
all_df = norm_df.copy()
all_df.dropna(inplace=True)

all_df_bfnorm = df.copy()
all_df_bfnorm.dropna(inplace=True)

dates = all_df.index.unique()
coefficients = []
p_values = []


for period in dates:
    period_data = all_df.loc[period].copy()

    X = sm.add_constant(period_data['PRC_Ret(T12M1)'])
    Y = period_data['PRC_Ret(F1M)']

    model = sm.OLS(Y, X).fit()
    coefficients.append(model.params['PRC_Ret(T12M1)'])
    p_values.append(model.pvalues['PRC_Ret(T12M1)'])

results_df = pd.DataFrame({'Date': dates, 'Coefficient': coefficients,␣
↪'P-Value': p_values})

plt.figure(figsize=(8, 3))
plt.plot(results_df['Date'], results_df['Coefficient'], label='Coefficient')
plt.axhline(0, color='grey', lw=0.5)
plt.title('Time Series of PRC_Ret(T12M1) Coefficients')
plt.xlabel('Date')
plt.ylabel('Coefficient')

significant_periods = results_df[results_df['P-Value'] < 0.05]
plt.scatter(significant_periods['Date'], significant_periods['Coefficient'],␣
↪color='red', label='Statistically Significant (p < 0.05)')

plt.legend(fontsize="8", loc ="upper left")
```

```python
plt.show()
```



[ ]:
```python
print(results_df[-50:-30])
```

```
           Date  Coefficient       P-Value
87   2019-05-31    -0.223891  1.114739e-06
88   2019-06-28    -0.009623  8.357974e-01
89   2019-07-31     0.458781  3.191481e-26
90   2019-08-30    -0.426440  3.013890e-22
91   2019-09-30    -0.156068  7.642035e-04
92   2019-10-31    -0.250612  3.123300e-08
93   2019-11-29    -0.190704  3.870269e-05
94   2019-12-31     0.261179  1.214144e-08
95   2020-01-31     0.152727  1.047135e-03
96   2020-02-28     0.320003  8.366097e-13
97   2020-03-31    -0.319613  3.585846e-12
98   2020-04-30     0.190599  2.808616e-05
99   2020-05-29    -0.185043  6.144411e-05
100  2020-06-30     0.274666  1.962104e-09
101  2020-07-31    -0.092968  4.729591e-02
102  2020-08-31     0.079700  8.480634e-02
103  2020-09-30    -0.145121  1.437561e-03
104  2020-10-30    -0.505004  7.784050e-32
105  2020-11-30    -0.213576  4.612879e-06
106  2020-12-31    -0.056299  2.237001e-01
```

- **Result:**
  - The time series plot of coefficients over the entire sample period reveals fluctuations in the predictive power of PRC_Ret(T12M1) over time.
  - There are periods where the coefficient is significantly positive, indicating periods where momentum strategies (buying past winners and selling past losers) would have performed

well. Conversely, there are also periods with significantly negative coefficients, indicating short-term reversals where momentum strategies would underperform.
  - The presence of both significantly positive and negative coefficients over the sample indicates that the effectiveness of a momentum strategy can vary greatly over time. This variation could be influenced by market conditions, investor sentiment, economic factors, or other variables not captured by the model.

- **Economic Interpretation**:
  - A positive coefficient implies that past positive returns are associated with higher future returns, which supports the momentum strategy. Conversely, a negative coefficient suggests that higher past returns are associated with lower future returns, which would contradict the momentum strategy. On specific dates, like in early 2012, we see positive, statistically significant coefficients. This implies that during these periods, a momentum strategy would have been effective—buying stocks with high past returns could have led to high future returns. There are periods, such as May 2012, where the coefficient is negative and significant, suggesting that a traditional momentum strategy would have been counterproductive. In such times, it could indicate a market reversal where past losers outperform past winners.
  - Economically, this analysis suggests that momentum investing is not universally effective across all periods. The strategy's effectiveness depends on the market environment, with some periods showing that past winners continue to outperform (momentum) and other periods indicating that past winners underperform (reversal).

- **Statistical Interpretation:**
  - the variation in the significance and direction of the coefficient across different periods suggests that momentum's effectiveness is not stable over time. This instability makes it challenging to apply a one-size-fits-all momentum strategy across all market conditions.

- **Explanation to layperson:**
  - Momentum investing, which involves buying stocks that have performed well in the past and selling those that have performed poorly, doesn't always work. Our analysis shows that sometimes, stocks that did well in the past year don't continue to do so in the following month. This pattern changes over time; in some months, momentum investing might work, while in others, it might not.

- **Stands out period:**
  - Observations Across Time | time | value | | — | — | | 2020-02-28 | 0.230302 | | 2020-03-31 | -0.236381 |
  - There is a abrupt shift of the coefficient from 0.23 to -0.23 for 2020-02 and 2020-03, which corresponds to the melt down at the beginning of the COVID. This abrupt shift of coefficient may be due to that all stocks are suffering from a huge downturn in that period.
  - Significant periods, such as those with particularly high positive or negative coefficients, highlight times when momentum strategies either performed exceptionally well or poorly. These standout periods could be associated with specific market events or economic conditions that influenced investor behavior and market dynamics.

In conclusion, while momentum strategies can be part of an investor's toolkit, their application requires careful consideration of current and historical market conditions. There's no guarantee that past winners will continue their streak, and as seen, the strategy's effectiveness can change significantly over time.

d)

- **Alpha**: It is the measure of an investment strategy's ability to beat the market, or its excess return independent of the market's movements. It is essential because it is supposed to represent the value added by a fund manager's skill. The interpretation of a single regression coefficient as alpha means looking at how much return the strategy can generate over and above the market return. Consistent positive alpha over time would indicate a strategy that consistently adds value beyond market performance.
- **Maximum Drawdown**: This is a crucial risk measure because it captures the largest single drop from peak to trough in the value of a portfolio, providing a real sense of the worst-case scenario an investor might experience. Investors are often more concerned with the potential losses they might incur than with the volatility of returns, making MDD a critical factor in the evaluation process.
- **Sortino Ratio**: As it focuses only on downside risk, the Sortino Ratio is particularly valuable for investors who are more concerned with the negative volatility of returns rather than overall volatility. This measure is significant in evaluating strategies that aim to minimize potential losses while still providing reasonable returns.

When it comes to the interpretation of the time-series of regression coefficients, such as alpha, the stability or variability can be quite telling. A stable, consistently positive alpha suggests that a manager has skill that is persistently contributing to outperformance. In contrast, large fluctuations in alpha might indicate changes in market conditions that the manager is either responding to effectively or being adversely affected by, or they could signal that what appeared to be skill was, in fact, luck. Viewing alpha across time allows investors to differentiate between a manager's skill and variance due to chance, helping to assess the long-term viability of an investment strategy.

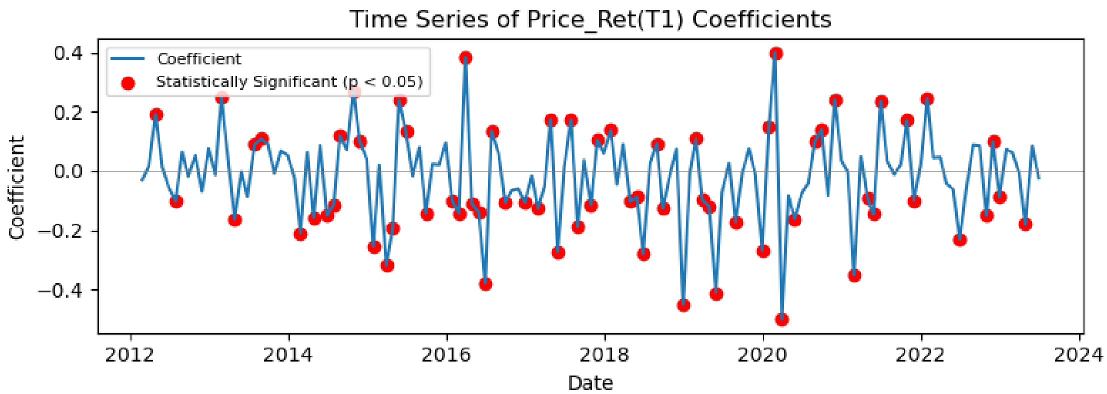
6.0.2 8 a) univariate regressions

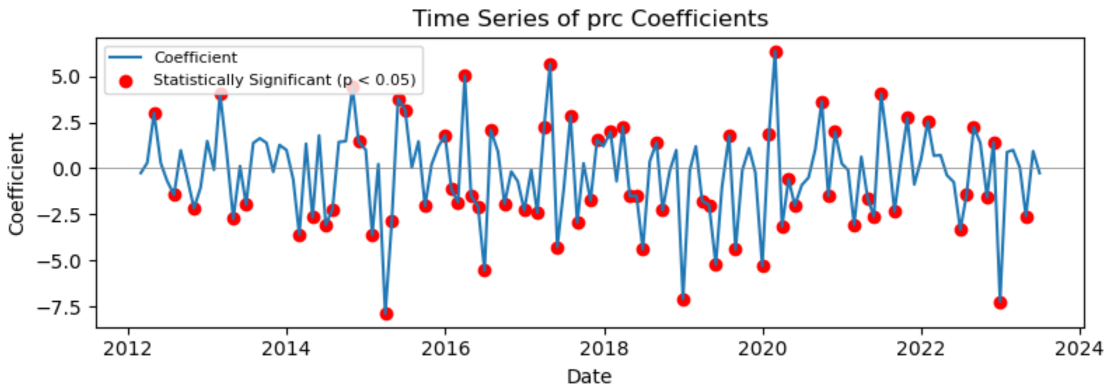
Price_Ret(T1); PRC; PRC_Ret(T12); Prices_Ret(T12); PRC_Ret(T12M1); Prices_Ret(T12M1); PRC_Ret(T12_1M); Prices_Ret(T12_1M); and SR_Prices_Ret(T12M1)

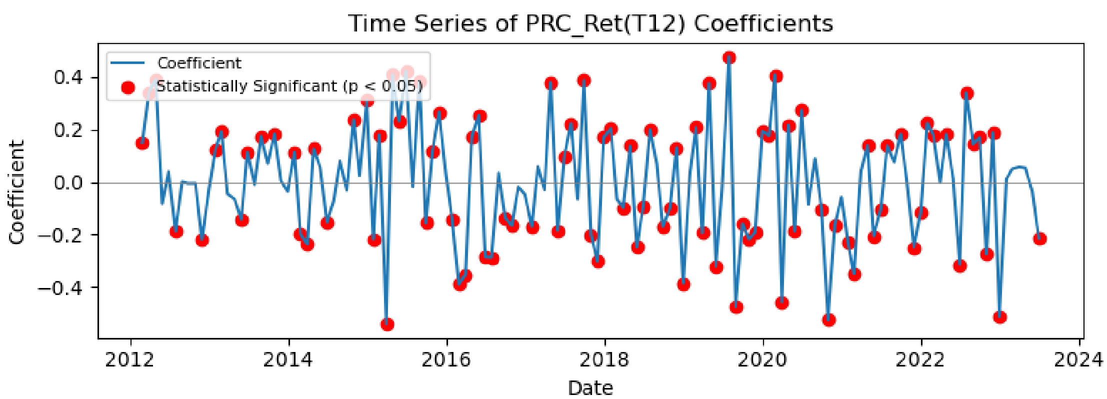
[ ]:
```
def cross_sectional_regression(df, independent_var, dependent_var):
    dates = df.index.unique()
    coefficients = []
    p_values = []

    for period in dates:
        period_data = df.loc[period]

        X = sm.add_constant(period_data[independent_var])
        Y = period_data[dependent_var]
```

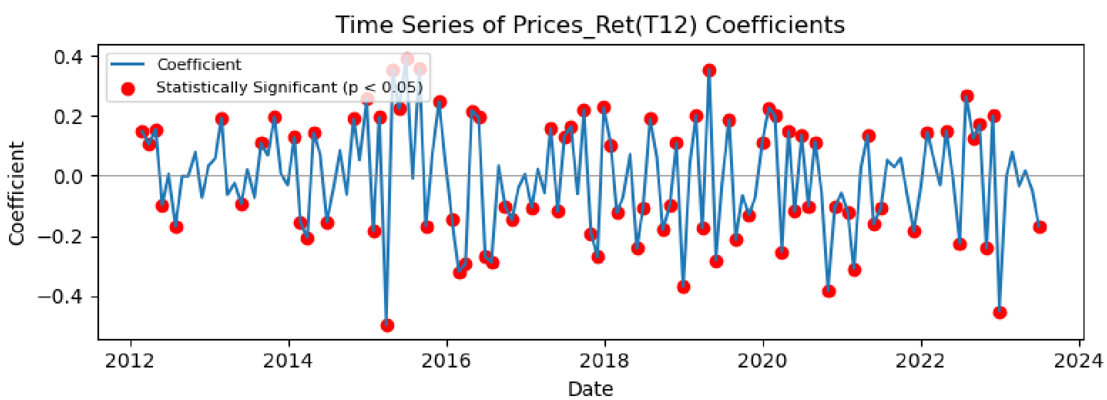
```
        model = sm.OLS(Y, X).fit()
        coefficients.append(model.params[independent_var])
        p_values.append(model.pvalues[independent_var])

    results_df = pd.DataFrame({'Date': dates, 'Coefficient': coefficients,␣
↪'P-Value': p_values})

    plt.figure(figsize=(8, 3))
    plt.plot(results_df['Date'], results_df['Coefficient'], label='Coefficient')
    plt.axhline(0, color='grey', lw=0.5)
    plt.title(f'Time Series of {independent_var} Coefficients')
    plt.xlabel('Date')
    plt.ylabel('Coefficient')

    significant_periods = results_df[results_df['P-Value'] < 0.05]
    plt.scatter(significant_periods['Date'],␣
↪significant_periods['Coefficient'], color='red', label='Statistically␣
↪Significant (p < 0.05)')

    plt.legend(fontsize="8", loc ="upper left")
    plt.tight_layout()
    plt.show()
```

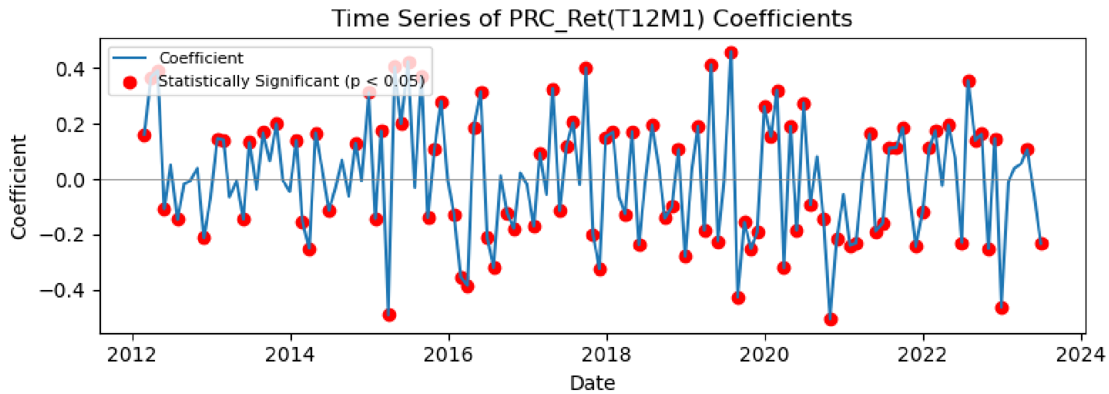
[ ]:
```
cross_sectional_regression(all_df, 'Price_Ret(T1)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'prc', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'PRC_Ret(T12)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'Prices_Ret(T12)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'PRC_Ret(T12M1)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'Prices_Ret(T12M1)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'PRC_Ret(T12_1M)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'Prices_Ret(T12_1M)', 'PRC_Ret(F1M)')
cross_sectional_regression(all_df, 'SR_Prices_Ret(T12M1)', 'PRC_Ret(F1M)')
```

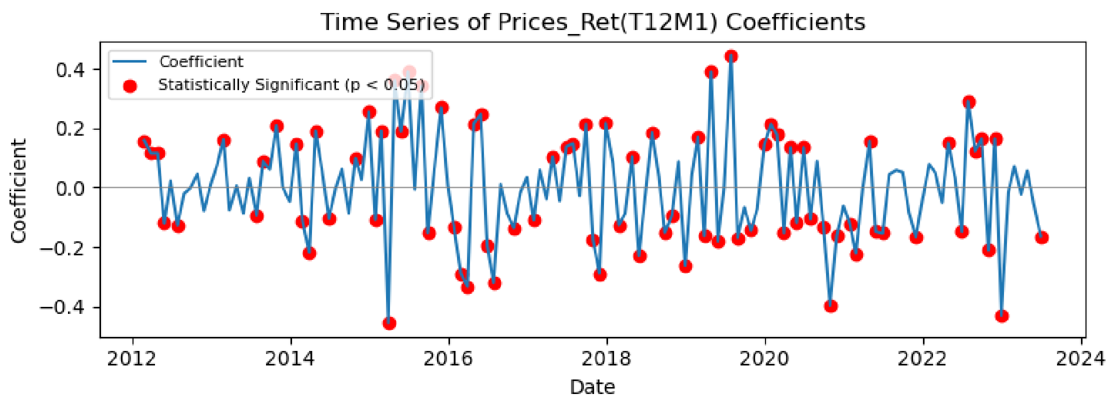


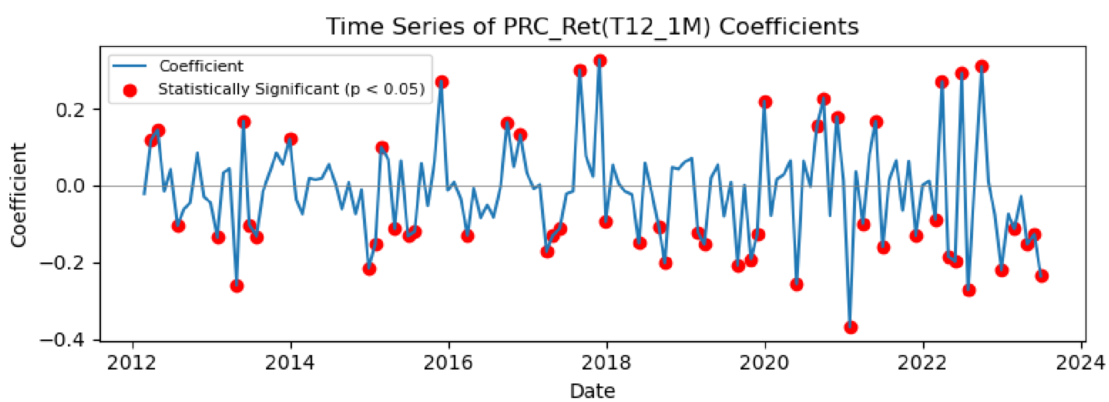

Time Series of prc Coefficients
Coefficient
Statistically Significant (p < 0.05)
5.0
2.5
0.0
−2.5
−5.0
−7.5
Coefficient
2012
2014
2016
2018
2020
2022
2024
Date

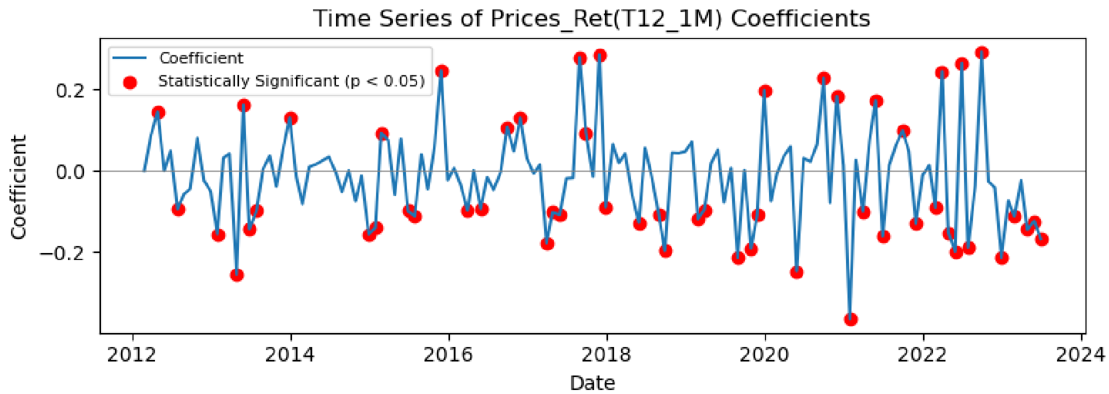

Time Series of PRC_Ret(T12) Coefficients
Coefficient
Statistically Significant (p < 0.05)
0.4
0.2
0.0
−0.2
−0.4
Coefficient
2012
2014
2016
2018
2020
2022
2024
Date

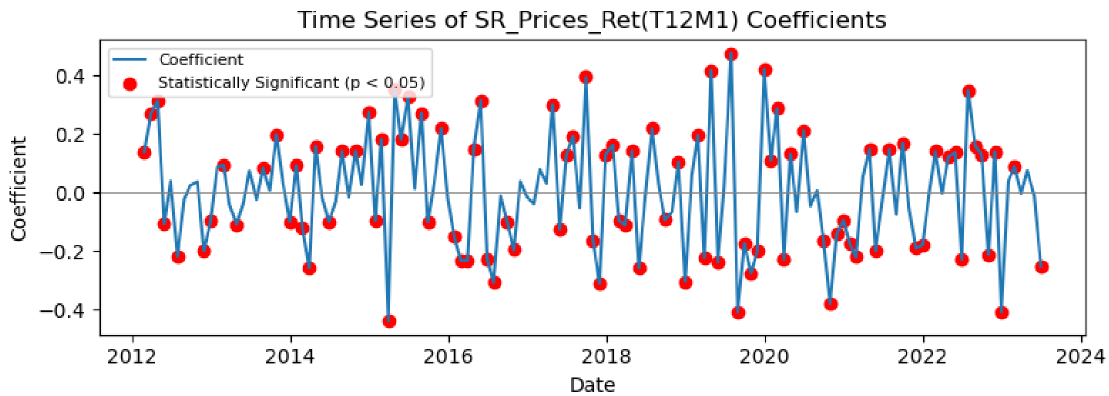

Time Series of Prices_Ret(T12) Coefficients
Coefficient
Statistically Significant (p < 0.05)
0.4
0.2
0.0
−0.2
−0.4
Coefficient
2012
2014
2016
2018
2020
2022
2024
Date


Time Series of PRC_Ret(T12M1) Coefficients
Coefficient
Statistically Significant (p < 0.05)
Coefficient
0.4
0.2
0.0
−0.2
−0.4
2012
2014
2016
2018
2020
2022
2024
Date


Time Series of Prices_Ret(T12M1) Coefficients
Coefficient
Statistically Significant (p < 0.05)
Coefficient
0.4
0.2
0.0
−0.2
−0.4
2012
2014
2016
2018
2020
2022
2024
Date


Time Series of PRC_Ret(T12_1M) Coefficients
Coefficient
Statistically Significant (p < 0.05)
Coefficient
0.2
0.0
−0.2
−0.4
2012
2014
2016
2018
2020
2022
2024
Date





Economically, I would not expect the sign of all the variables to be the same.

Survivorship Bias: Only including stocks that have survived through the entire period may bias the results.

Serial Correlation: Financial time series data often exhibit serial correlation, potentially inflating the significance of predictors.

8 b) Multivariate Regression:

i. Price_Ret(T1) and Prices_Ret(T12M1)

ii. PRC and PRC_Ret(T12M1)

iii. PRC and Prices_Ret(T12)

iv. PRC and SR_Prices_Ret(T12M1)

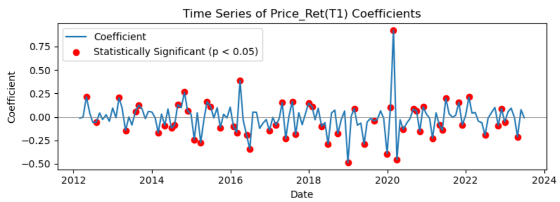
[ ]:
```
def multivariate_cross_sectional_regression(df, independent_var_list,␣
↪dependent_var):
    dates = df.index.unique()
```

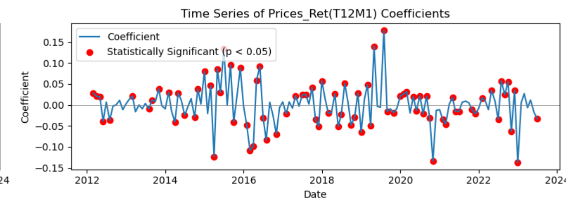
```
    results = []

    for period in dates:
        period_data = df.loc[period]

        X = sm.add_constant(period_data[independent_var_list])
        Y = period_data[dependent_var]

        model = sm.OLS(Y, X).fit()
        for var in independent_var_list:
            results.append({
                'Date' : period,
                'Variable' : var,
                'Coefficient' : model.params.get(var, 0),
                'p-value' : model.pvalues.get(var, 1)
            })

    results_df = pd.DataFrame(results)

    fig, axs = plt.subplots(1, len(independent_var_list), figsize=(16, 3))

    for i, var in enumerate(independent_var_list):
        var_data = results_df[results_df['Variable'] == var]
        axs[i].plot(var_data['Date'], var_data['Coefficient'],␣
 ↪label='Coefficient')
        axs[i].axhline(0, color='grey', lw=0.5)
        axs[i].set_title(f'Time Series of {var} Coefficients')
        axs[i].set_xlabel('Date')
        axs[i].set_ylabel('Coefficient')

        significant_periods = var_data[var_data['p-value'] < 0.05]
        axs[i].scatter(significant_periods['Date'],␣
 ↪significant_periods['Coefficient'], color='red', label='Statistically␣
 ↪Significant (p < 0.05)')

        axs[i].legend()

    plt.tight_layout()
    plt.show()
```

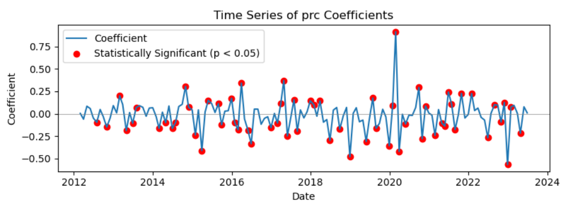
[ ]:

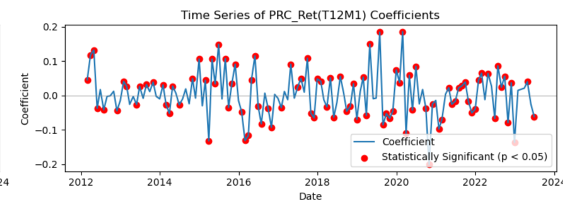
```
multivariate_cross_sectional_regression(all_df_bfnorm, ['Price_Ret(T1)',␣
 ↪'Prices_Ret(T12M1)'], 'PRC_Ret(F1M)')

multivariate_cross_sectional_regression(all_df_bfnorm, ['prc',␣
 ↪'PRC_Ret(T12M1)'], 'PRC_Ret(F1M)')
```

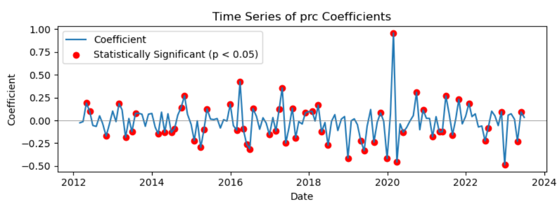
```
multivariate_cross_sectional_regression(all_df_bfnorm, ['prc',␣
↪'Prices_Ret(T12)'], 'PRC_Ret(F1M)')

multivariate_cross_sectional_regression(all_df_bfnorm, ['prc',␣
↪'SR_Prices_Ret(T12M1)'], 'PRC_Ret(F1M)')
```

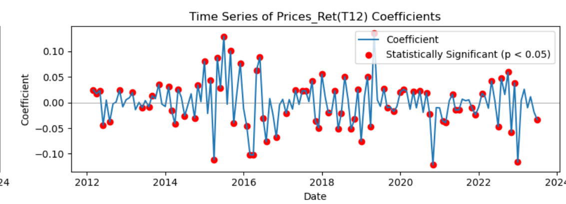


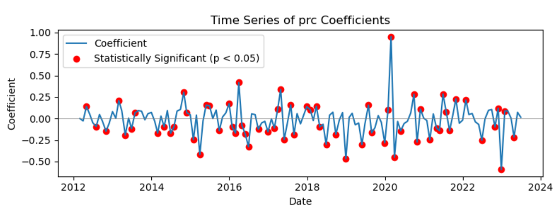


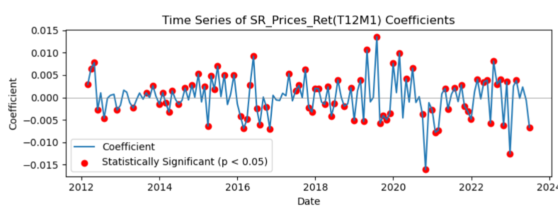






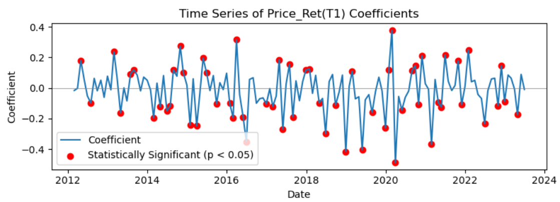


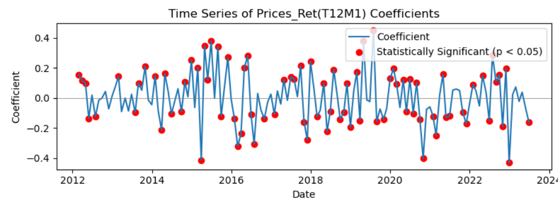


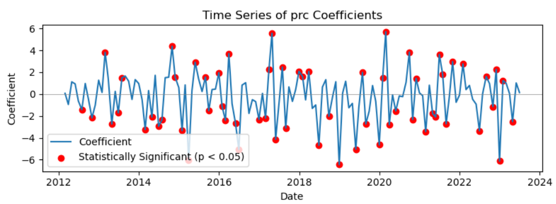


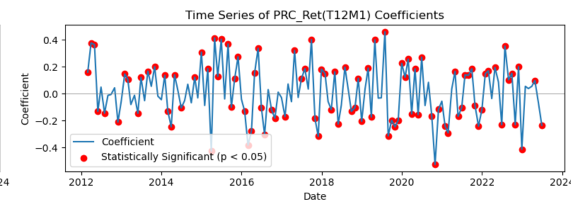
[ ]:
```
multivariate_cross_sectional_regression(all_df, ['Price_Ret(T1)',␣
↪'Prices_Ret(T12M1)'], 'PRC_Ret(F1M)')

multivariate_cross_sectional_regression(all_df, ['prc', 'PRC_Ret(T12M1)'],␣
↪'PRC_Ret(F1M)')
```

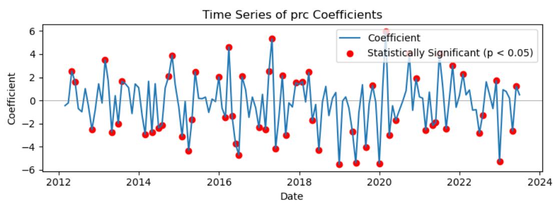
```
multivariate_cross_sectional_regression(all_df, ['prc', 'Prices_Ret(T12)'],␣
↪'PRC_Ret(F1M)')

multivariate_cross_sectional_regression(all_df, ['prc',␣
↪'SR_Prices_Ret(T12M1)'], 'PRC_Ret(F1M)')
```

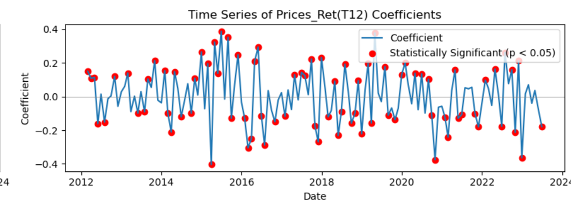


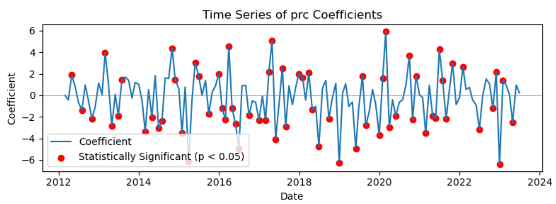


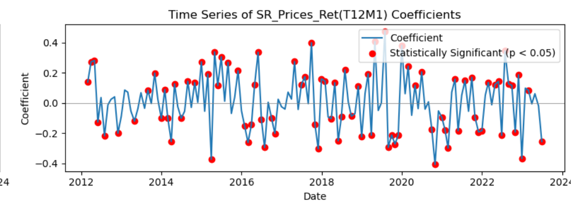












- Economically, I would not expect the sign of all the variables to be the same for different factors in the same multivariate regression because each factor represents a different aspect of price behavior and investor sentiment.
  - Price_Ret(T1) and Prices_Ret(T12M1): Price_Ret(T1) reflects immediate past return, potentially capturing short-term movements which may exhibit a reversal effect in the

subsequent period, whereas Prices_Ret(T12M1) indicates longer-term momentum, excluding the immediate past month. The economic expectation is that the short-term return might show a negative coefficient if prices revert to the mean, while the longer-term momentum should typically show a positive coefficient if past winners continue to perform well.
  - PRC and PRC_Ret(T12M1): PRC_Ret(T12M1) captures the momentum effect, which should have a positive coefficient, reflecting the tendency for past winners to continue yielding higher returns.
  - PRC and SR_Prices_Ret(T12M1): SR_Prices_Ret(T12M1) could be capturing a short-term reversal that may happen after a period of momentum. The expected sign might be negative, suggesting that after a period of sustained returns, prices might correct themselves.

- **Particularities with the Regressions**
  - when running the regression on unnormalized data and the normalized data, we found that if we use unnormalized data, the scale of coefficients vary in different regressions. For example, the regression on prc and PRC_Ret(T12M1) has the scale of coefficient of 'PRC_Ret(T12M1)' between [-0.2,0.2], but the regression on prc and SR_Prices_Ret(T12M1) has the scale of coefficient of 'PRC_Ret(T12M1)' between [-0.015,0.015]. So we use the cross-sectional normalized data to run the rest of the regressions.

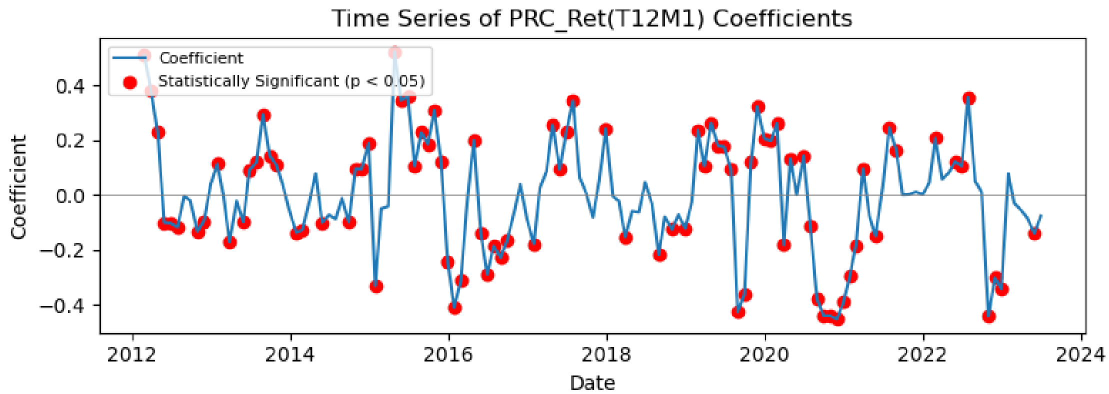
- **Particularities/Issues with the Regressions**
  - **Multicollinearity**: Since some of the independent variables are different forms of price returns, they might be highly correlated with each other, leading to multicollinearity. This could inflate the variance of the coefficient estimates and make the results less reliable.
  - **Time Variation**: The coefficients fluctuate over time, which suggests that the relationship between the predictors and future returns is not stable. This could be due to changes in market efficiency, investor behavior, or macroeconomic conditions.
  - **Autocorrelation**: In time-series data, there could be autocorrelation that violates the regression assumptions and could lead to misleading inference if not properly addressed.
  - **Risk Adjustments**: The regressions might not account for risk factors that could explain returns. For instance, stocks with high momentum might also have higher risk, which is not captured in a simple regression.

6.0.3 9 Cross-sectional regression:

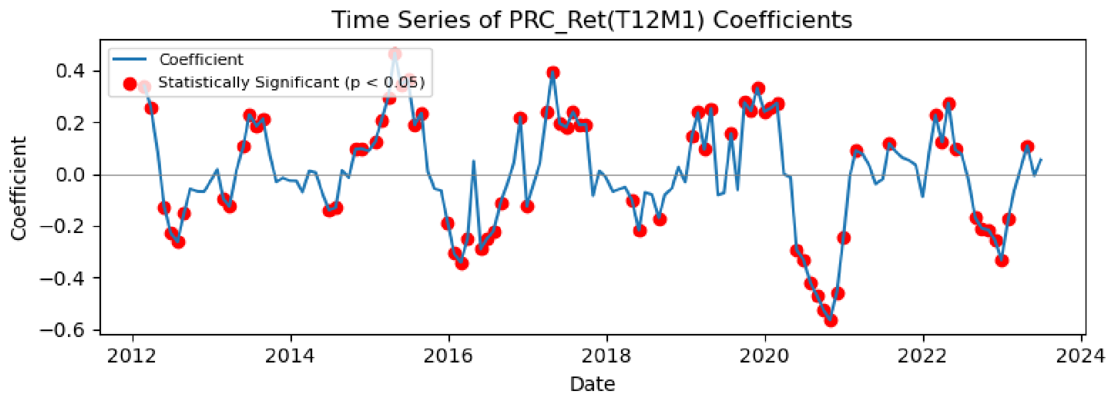
independent variables is PRC_Ret(T12M1)

dependent variables: PRC_Ret(F3M) and PRC_Ret(F6M).

[ ]:
```
cross_sectional_regression(all_df, 'PRC_Ret(T12M1)', 'PRC_Ret(F3M)')
```




```
[ ]: cross_sectional_regression(all_df, 'PRC_Ret(T12M1)', 'PRC_Ret(F6M)')
```



- **observation on result**:
  - **Coefficient Fluctuations:** The coefficients over time fluctuate around zero, which suggests that the predictive power of PRC_Ret(T12M1) on future returns varies over time and may not be consistent.
  - **Statistical Significance:** In both plots, red dots signify statistically significant coefficients at the 5% level. There appear to be many more statistically significant points for the F1M forward returns compared to F3M and F6M. This indicates that as the forward-looking period increases, the predictive power of PRC_Ret(T12M1) diminishes.
  - **Trend in Significance:** The trend of the coefficients does not seem to follow a clear pattern, which could suggest that the relationship between past returns and future returns is not stable over time and may be influenced by market conditions, behavioral factors, or other external variables.
- **interpretation:** Given the observed reduction in statistically significant coefficients as the forward return period increases, we can deduce that momentum signals, specifically PRC_Ret(T12M1), are less predictive for longer-term returns. This might be explained by:

- **Market Efficiency:** Markets may incorporate information more fully over longer periods, diluting the momentum effect.
- **Mean Reversion:** Over longer periods, there's a higher chance for mean reversion, where stocks that performed well in the past may underperform as they revert to their long-term average.

- **Potential Econometric/Statistical Issues:**
  - **Non-Stationarity:** Financial time series data may be non-stationary, particularly over different time horizons, which can affect the reliability of regression coefficients.
  - **Serial Correlation:** The presence of serial correlation in financial returns can lead to misestimation of the significance levels of the coefficients.
  - **Changing Risk Premia:** The risk-return tradeoff may change over time, which means the factors driving returns in one period might not be the same in another.
- **Correcting Issues:**
  - **Addressing Non-Stationarity:** Use techniques like differencing, detrending, or transforming the series into stationary through logarithmic or percentage change transformations.
  - **Accounting for Serial Correlation:** Employ models that specifically adjust for time-series correlation, like the Generalized Method of Moments (GMM) or Newey-West standard errors.
  - **Modeling Changing Risk Premia:** Introduce additional variables to the regression that can account for changes in risk premia over time, or use rolling regression windows to capture the evolving relationships.
- **Conclusion:**
  - The Fama-McBeth regression results, particularly the decrease in statistically significant coefficients for longer forward returns, suggest that the momentum effect identified by PRC_Ret(T12M1) diminishes over longer periods. This is in line with the economic theory of market efficiency and mean reversion over time. The reliability of these regression results can be improved by addressing the potential econometric and statistical issues mentioned, thus yielding more robust insights into the predictive power of momentum signals across different investment horizons.

10

- **Understanding of Momentum Signals:**
  - The homework solidifies the concept of momentum in finance—that past returns can influence future performance. It's a deep dive into how quantitative researchers can identify and leverage historical data patterns to predict future market movements.
  - Learning how to create various momentum signals from raw data teaches the practical aspects of data handling, signal processing, and the nuances of different financial metrics. It's crucial for developing any systematic trading strategy.
- **Econometric Skills**
  - Cross-sectional Regression Analysis: Performing Fama-McBeth regressions teaches us how to explore the relationships between return predictors and actual returns across different assets at a given time.
- **Understanding Statistical Significance:**
  - Through this exercise, we learn to critically evaluate the statistical significance of your findings, understanding not just when a signal appears to predict returns, but also gauging the reliability of these predictions.

- **Investment Strategy Insights**
  - Efficacy of Momentum Strategies: By analyzing momentum [-12, -1] or any other specified periods, we gain insights into the effectiveness of momentum-based investment strategies. It helps quantify how past performance can be indicative of future returns.
- **Diverse Time Horizons:**
  - Comparing short-term (F1M) with medium (F3M) and long-term (F6M) forward returns highlights how the predictive power of signals can vary over different investment horizons. This can inform the development of strategies tailored to specific time frames.
- **Innovation in Strategy Development:**
  - This exercise also underlines the importance of continuous learning in developing investment strategies. The financial markets are ever-changing, and strategies that worked in the past may not work in the future. Hence, a quantitative researcher must always be testing, learning, and adapting.