1 Brief summary of Cross-Asset Skew strategy

Strategy (Nick Baltas et al 2019): Calculate rolling (256 trading days = 1 trading year) skew of assets (here ETFs) in different asset classes (here Commodities, Equities and Fixed Income), rank them based on their skewness within each asset class and assign weights based on the rank (from low to high skewness). Then construct self financed portfolio (1 dollar short in lower ranked assets 1 dollar long in higher ranked assets) and rebalance each month based on the skewness ranking at the end of the previous month.

Backtest portfolio for different start and end dates by comparing the performance of the skew portfolios with the market portfolios of the individual assets and perform OLS regression to see if strategy yields significant alpha or just market beta.

The skew portfolio from different assets are barely correlated (check) thus advantageous to diversivy and combine the individual portfolios into one diverse portfolio (Global Skewness Factor Portfolio).

Global Skewness Factor Portfolio: Scale each asset class skew portfolio to have a full sample volatility of 10% and combine them all on an equal-weight basis.

Definition of Skew:

$$S = \frac{1}{N}\sum_{i=1}^{N}\frac{(r_i - \mu)^3}{\sigma^3} \tag{1}$$

σ rolling std

μ rolling mean

r_i daily returns of individual ETFs

Definition of weight:

$$w = z(RANK - \frac{M+1}{2}) \tag{2}$$

z normalization factor

$RANK$ ranking of ETF within asset class

M nummber of ETFs in asset class

[ ]:
```
import numpy as np
import pandas as pd
import time
import matplotlib.pyplot as plt
from pandas_datareader import data as pdr
```

```
import yfinance as yf
import statsmodels.api as sm
from scipy.stats import linregress
```

Pull data from yahoo finance

[ ]:
```
# import data
def get_data(stocks, start, end):
    stockData = yf.download(stocks, start, end)
    stockData['Ticker'] = stocks
    #stockData = stockData['Close']
    #returns = stockData.pct_change()
    #meanReturns = returns.mean()
    #covMatrix = returns.cov()
    return stockData  #, covMatrix
```

Set start and end date for the Backtest

[ ]:
```
startdate = "2015-01-01"
enddate = "2024-04-01"
```

2 Commodities

Select the ETFs in the commodity asset class and calculate the rolling skew for the last day of the month (EOM) for each ETF.

Make sure that data is clean

[ ]:
```
commodities = ["GLD", "SLV", "GSG", "USO", "PPLT", "UNG", "DBA"]

alldatacommodities  = []
for j in commodities:
  individualdf =  get_data(j,startdate,enddate)
  individualdf
  individualdf = individualdf.drop(columns=['Open','High','Low','Adj␣
 ↪Close','Volume'])
  individualdf['pct_change'] = individualdf.Close.pct_change()
  individualdf['ret'] = np.log(individualdf.Close) - np.log(individualdf.Close.
 ↪shift(1))
  individualdf['rolling_mean'] = individualdf.ret.rolling(256).mean()
  individualdf['rolling_std'] = individualdf.ret.rolling(256).std()
  individualdf['skew_day'] = ((individualdf.ret-individualdf.rolling_mean)/
 ↪individualdf.rolling_std)**3
  individualdf['rolling_skew'] = individualdf.skew_day.rolling(256).mean()
  individualdf = individualdf.reset_index()
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['Date']
```

```
 individualdf.groupby([individualdf.Date.dt.year, individualdf.Date.dt.
↪month],group_keys=False).max()
 individualdf['EOM'] = groupings.transform(lambda x: x.max())
 individualdf['EOM_rolling_skew'] = groupings.transform(lambda x:␣
↪individualdf[individualdf["Date"] == x.max()].rolling_skew)
 individualdf['EOM_rolling_skew_lookback'] = individualdf.EOM_rolling_skew.
↪shift(1)
 groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
↪Date.dt.month],group_keys=False)['EOM_rolling_skew']
 individualdf['EOM_rolling_skew'] = groupings.transform(lambda x: x.max())
 groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
↪Date.dt.month],group_keys=False)['EOM_rolling_skew_lookback']
 individualdf['EOM_rolling_skew_lookback'] = groupings.transform(lambda x: x.
↪max())

 alldatacommodities.append(individualdf)

alldatacommodities
```

```
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
```

[ ]:
```
[           Date       Close Ticker  pct_change       ret  rolling_mean  \
 0    2015-01-02  114.080002    GLD         NaN       NaN           NaN
 1    2015-01-05  115.800003    GLD    0.015077  0.014965           NaN
 2    2015-01-06  117.120003    GLD    0.011399  0.011334           NaN
 3    2015-01-07  116.430000    GLD   -0.005891 -0.005909           NaN
 4    2015-01-08  115.940002    GLD   -0.004209 -0.004217           NaN
 …           …          …    …          …       …           …
 2320 2024-03-22  200.350006    GLD   -0.008021 -0.008053      0.000450
 2321 2024-03-25  200.990005    GLD    0.003194  0.003189      0.000350
 2322 2024-03-26  201.639999    GLD    0.003234  0.003229      0.000361
 2323 2024-03-27  203.100006    GLD    0.007241  0.007215      0.000464
 2324 2024-03-28  205.720001    GLD    0.012900  0.012818      0.000448

       rolling_std  skew_day  rolling_skew        EOM  EOM_rolling_skew  \
 0             NaN       NaN           NaN 2015-01-30               NaN
 1             NaN       NaN           NaN 2015-01-30               NaN
 2             NaN       NaN           NaN 2015-01-30               NaN
 3             NaN       NaN           NaN 2015-01-30               NaN
 4             NaN       NaN           NaN 2015-01-30               NaN
 …           …       …           …         …                 …
```

```
2320     0.007818 -1.286600      0.315197 2024-03-28          0.240299
2321     0.007616  0.051818      0.210425 2024-03-28          0.240299
2322     0.007618  0.053338      0.210633 2024-03-28          0.240299
2323     0.007532  0.719991      0.243620 2024-03-28          0.240299
2324     0.007502  4.483327      0.240299 2024-03-28          0.240299

      EOM_rolling_skew_lookback
0                           NaN
1                           NaN
2                           NaN
3                           NaN
4                           NaN
…                           …
2320                   0.317191
2321                   0.317191
2322                   0.317191
2323                   0.317191
2324                   0.317191

[2325 rows x 12 columns],
           Date      Close Ticker  pct_change       ret  rolling_mean  \
0    2015-01-02  15.110000    SLV         NaN       NaN           NaN
1    2015-01-05  15.500000    SLV    0.025811  0.025483           NaN
2    2015-01-06  15.830000    SLV    0.021290  0.021067           NaN
3    2015-01-07  15.850000    SLV    0.001263  0.001263           NaN
4    2015-01-08  15.640000    SLV   -0.013249 -0.013338           NaN
…         …        …    …          …       …           …
2320 2024-03-22  22.559999    SLV   -0.003974 -0.003981      0.000486
2321 2024-03-25  22.580000    SLV    0.000887  0.000886      0.000353
2322 2024-03-26  22.340000    SLV   -0.010629 -0.010686      0.000300
2323 2024-03-27  22.510000    SLV    0.007610  0.007581      0.000358
2324 2024-03-28  22.750000    SLV    0.010662  0.010605      0.000294

      rolling_std  skew_day  rolling_skew        EOM  EOM_rolling_skew  \
0             NaN       NaN           NaN 2015-01-30               NaN
1             NaN       NaN           NaN 2015-01-30               NaN
2             NaN       NaN           NaN 2015-01-30               NaN
3             NaN       NaN           NaN 2015-01-30               NaN
4             NaN       NaN           NaN 2015-01-30               NaN
…           …       …           …        …                …
2320     0.015040 -0.026208      0.098882 2024-03-28          0.062796
2321     0.014884  0.000046      0.073724 2024-03-28          0.062796
2322     0.014899 -0.400878      0.072135 2024-03-28          0.062796
2323     0.014898  0.113970      0.072758 2024-03-28          0.062796
2324     0.014819  0.336910      0.062796 2024-03-28          0.062796

      EOM_rolling_skew_lookback
```

```
0                          NaN
1                          NaN
2                          NaN
3                          NaN
4                          NaN
…                          …
2320                  0.097062
2321                  0.097062
2322                  0.097062
2323                  0.097062
2324                  0.097062

[2325 rows x 12 columns],
           Date      Close Ticker  pct_change       ret  rolling_mean  \
0    2015-01-02  21.219999    GSG         NaN       NaN           NaN
1    2015-01-05  20.620001    GSG   -0.028275 -0.028683           NaN
2    2015-01-06  20.280001    GSG   -0.016489 -0.016626           NaN
3    2015-01-07  20.219999    GSG   -0.002959 -0.002963           NaN
4    2015-01-08  20.290001    GSG    0.003462  0.003456           NaN
…         …        …    …          …        …           …
2320 2024-03-22  21.799999    GSG   -0.003656 -0.003663      0.000525
2321 2024-03-25  21.980000    GSG    0.008257  0.008223      0.000592
2322 2024-03-26  21.809999    GSG   -0.007734 -0.007764      0.000535
2323 2024-03-27  21.830000    GSG    0.000917  0.000917      0.000499
2324 2024-03-28  22.059999    GSG    0.010536  0.010481      0.000534

      rolling_std  skew_day  rolling_skew        EOM  EOM_rolling_skew  \
0             NaN       NaN           NaN 2015-01-30               NaN
1             NaN       NaN           NaN 2015-01-30               NaN
2             NaN       NaN           NaN 2015-01-30               NaN
3             NaN       NaN           NaN 2015-01-30               NaN
4             NaN       NaN           NaN 2015-01-30               NaN
…           …       …           …        …                …
2320     0.011304 -0.050849     -0.073154 2024-03-28         -0.071546
2321     0.011298  0.308158     -0.071577 2024-03-28         -0.071546
2322     0.011303 -0.395809     -0.073415 2024-03-28         -0.071546
2323     0.011288  0.000050     -0.074202 2024-03-28         -0.071546
2324     0.011305  0.681124     -0.071546 2024-03-28         -0.071546

      EOM_rolling_skew_lookback
0                          NaN
1                          NaN
2                          NaN
3                          NaN
4                          NaN
…                          …
2320                  -0.107685
```

```
2321                  -0.107685
2322                  -0.107685
2323                  -0.107685
2324                  -0.107685

[2325 rows x 12 columns],
           Date       Close Ticker  pct_change       ret  rolling_mean  \
0    2015-01-02  159.119995    USO         NaN       NaN           NaN
1    2015-01-05  150.320007    USO   -0.055304 -0.056892           NaN
2    2015-01-06  144.399994    USO   -0.039383 -0.040179           NaN
3    2015-01-07  146.960007    USO    0.017729  0.017573           NaN
4    2015-01-08  148.399994    USO    0.009798  0.009751           NaN
…         …          …    …          …        …           …
2320 2024-03-22   76.680000    USO   -0.001822 -0.001824      0.000955
2321 2024-03-25   77.760002    USO    0.014085  0.013986      0.001101
2322 2024-03-26   77.290001    USO   -0.006044 -0.006063      0.001003
2323 2024-03-27   77.510002    USO    0.002846  0.002842      0.000921
2324 2024-03-28   78.730003    USO    0.015740  0.015617      0.000958

      rolling_std  skew_day  rolling_skew        EOM  EOM_rolling_skew  \
0             NaN       NaN           NaN 2015-01-30               NaN
1             NaN       NaN           NaN 2015-01-30               NaN
2             NaN       NaN           NaN 2015-01-30               NaN
3             NaN       NaN           NaN 2015-01-30               NaN
4             NaN       NaN           NaN 2015-01-30               NaN
…           …       …           …        …                …
2320     0.018613 -0.003328     -0.118061 2024-03-28         -0.117802
2321     0.018568  0.334186     -0.114106 2024-03-28         -0.117802
2322     0.018539 -0.055356     -0.116202 2024-03-28         -0.117802
2323     0.018484  0.001123     -0.119674 2024-03-28         -0.117802
2324     0.018504  0.497242     -0.117802 2024-03-28         -0.117802

      EOM_rolling_skew_lookback
0                           NaN
1                           NaN
2                           NaN
3                           NaN
4                           NaN
…                           …
2320                  -0.172024
2321                  -0.172024
2322                  -0.172024
2323                  -0.172024
2324                  -0.172024

[2325 rows x 12 columns],
           Date       Close Ticker  pct_change       ret  rolling_mean  \
```

```
0    2015-01-02  116.570000   PPLT         NaN       NaN           NaN
1    2015-01-05  117.599998   PPLT    0.008836  0.008797           NaN
2    2015-01-06  118.540001   PPLT    0.007993  0.007961           NaN
3    2015-01-07  118.320000   PPLT   -0.001856 -0.001858           NaN
4    2015-01-08  118.110001   PPLT   -0.001775 -0.001776           NaN
…         …         …    …          …         …           …
2320 2024-03-22   82.360001   PPLT   -0.013535 -0.013627     -0.000359
2321 2024-03-25   83.110001   PPLT    0.009106  0.009065     -0.000315
2322 2024-03-26   83.099998   PPLT   -0.000120 -0.000120     -0.000371
2323 2024-03-27   82.639999   PPLT   -0.005535 -0.005551     -0.000331
2324 2024-03-28   83.550003   PPLT    0.011012  0.010951     -0.000346

      rolling_std  skew_day  rolling_skew        EOM  EOM_rolling_skew  \
0             NaN       NaN           NaN 2015-01-30               NaN
1             NaN       NaN           NaN 2015-01-30               NaN
2             NaN       NaN           NaN 2015-01-30               NaN
3             NaN       NaN           NaN 2015-01-30               NaN
4             NaN       NaN           NaN 2015-01-30               NaN
…           …       …           …        …                …
2320     0.014377 -0.786090      0.008433 2024-03-28          0.009627
2321     0.014388  0.277080      0.009518 2024-03-28          0.009627
2322     0.014359  0.000005      0.007688 2024-03-28          0.009627
2323     0.014331 -0.048324      0.009728 2024-03-28          0.009627
2324     0.014317  0.491391      0.009627 2024-03-28          0.009627

      EOM_rolling_skew_lookback
0                           NaN
1                           NaN
2                           NaN
3                           NaN
4                           NaN
…                           …
2320                    -0.04676
2321                    -0.04676
2322                    -0.04676
2323                    -0.04676
2324                    -0.04676

[2325 rows x 12 columns],
           Date       Close Ticker  pct_change       ret  rolling_mean  \
0    2015-01-02  239.360001    UNG         NaN       NaN           NaN
1    2015-01-05  237.919998    UNG   -0.006016 -0.006034           NaN
2    2015-01-06  238.880005    UNG    0.004035  0.004027           NaN
3    2015-01-07  233.279999    UNG   -0.023443 -0.023722           NaN
4    2015-01-08  239.039993    UNG    0.024691  0.024391           NaN
…         …         …    …          …         …           …
2320 2024-03-22   15.110000    UNG   -0.013708 -0.013802     -0.003094
```

```
2321 2024-03-25   15.040000    UNG   -0.004633 -0.004643     -0.002850
2322 2024-03-26   14.960000    UNG   -0.005319 -0.005333     -0.002697
2323 2024-03-27   14.350000    UNG   -0.040775 -0.041630     -0.002999
2324 2024-03-28   14.570000    UNG    0.015331  0.015215     -0.002758

      rolling_std  skew_day  rolling_skew        EOM  EOM_rolling_skew  \
0             NaN       NaN           NaN 2015-01-30               NaN
1             NaN       NaN           NaN 2015-01-30               NaN
2             NaN       NaN           NaN 2015-01-30               NaN
3             NaN       NaN           NaN 2015-01-30               NaN
4             NaN       NaN           NaN 2015-01-30               NaN
…             …         …             …          …                 …
2320     0.035324 -0.027860     -0.016252 2024-03-28         -0.009509
2321     0.035097 -0.000133     -0.007814 2024-03-28         -0.009509
2322     0.034999 -0.000427     -0.005542 2024-03-28         -0.009509
2323     0.035001 -1.344572     -0.012619 2024-03-28         -0.009509
2324     0.034913  0.136420     -0.009509 2024-03-28         -0.009509

      EOM_rolling_skew_lookback
0                           NaN
1                           NaN
2                           NaN
3                           NaN
4                           NaN
…                           …
2320                  -0.032926
2321                  -0.032926
2322                  -0.032926
2323                  -0.032926
2324                  -0.032926

[2325 rows x 12 columns],
           Date      Close Ticker  pct_change       ret  rolling_mean  \
0    2015-01-02  24.639999    DBA         NaN       NaN           NaN
1    2015-01-05  24.940001    DBA    0.012175  0.012102           NaN
2    2015-01-06  25.110001    DBA    0.006816  0.006793           NaN
3    2015-01-07  24.990000    DBA   -0.004779 -0.004790           NaN
4    2015-01-08  24.900000    DBA   -0.003601 -0.003608           NaN
…         …         …    …          …        …             …
2320 2024-03-22  24.080000    DBA    0.006689  0.006667      0.000755
2321 2024-03-25  24.639999    DBA    0.023256  0.022989      0.000848
2322 2024-03-26  24.520000    DBA   -0.004870 -0.004882      0.000831
2323 2024-03-27  24.670000    DBA    0.006117  0.006099      0.000851
2324 2024-03-28  24.760000    DBA    0.003648  0.003642      0.000879

      rolling_std   skew_day  rolling_skew        EOM  EOM_rolling_skew  \
0             NaN        NaN           NaN 2015-01-30               NaN
```

```
1              NaN        NaN           NaN 2015-01-30               NaN
2              NaN        NaN           NaN 2015-01-30               NaN
3              NaN        NaN           NaN 2015-01-30               NaN
4              NaN        NaN           NaN 2015-01-30               NaN
…              …          …             …          …                 …
2320      0.007805   0.434550     -0.622806 2024-03-28         -0.537701
2321      0.007927  21.788557     -0.537693 2024-03-28         -0.537701
2322      0.007935  -0.373263     -0.539151 2024-03-28         -0.537701
2323      0.007942   0.288505     -0.538039 2024-03-28         -0.537701
2324      0.007939   0.042128     -0.537701 2024-03-28         -0.537701

      EOM_rolling_skew_lookback
0                           NaN
1                           NaN
2                           NaN
3                           NaN
4                           NaN
…                           …
2320                  -0.754285
2321                  -0.754285
2322                  -0.754285
2323                  -0.754285
2324                  -0.754285

[2325 rows x 12 columns]]
```

Add all ETF dataframes into one single dataframe

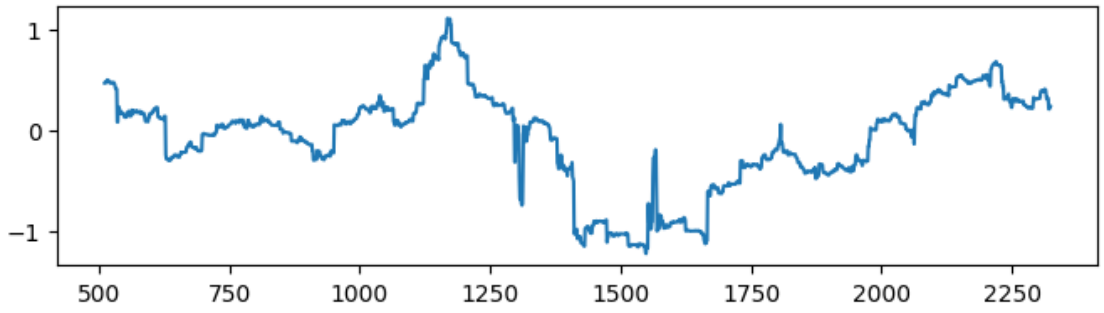
[ ]:
```
alldatacommodities_df = pd.concat(alldatacommodities)
alldatacommodities_df.tail(5)
```

[ ]:
```
           Date      Close Ticker  pct_change       ret  rolling_mean  \
2320 2024-03-22  24.080000    DBA    0.006689  0.006667      0.000755
2321 2024-03-25  24.639999    DBA    0.023256  0.022989      0.000848
2322 2024-03-26  24.520000    DBA   -0.004870 -0.004882      0.000831
2323 2024-03-27  24.670000    DBA    0.006117  0.006099      0.000851
2324 2024-03-28  24.760000    DBA    0.003648  0.003642      0.000879

      rolling_std   skew_day  rolling_skew        EOM  EOM_rolling_skew  \
2320     0.007805   0.434550     -0.622806 2024-03-28         -0.537701
2321     0.007927  21.788557     -0.537693 2024-03-28         -0.537701
2322     0.007935  -0.373263     -0.539151 2024-03-28         -0.537701
2323     0.007942   0.288505     -0.538039 2024-03-28         -0.537701
2324     0.007939   0.042128     -0.537701 2024-03-28         -0.537701

      EOM_rolling_skew_lookback
2320                  -0.754285
```

```
2321                -0.754285
2322                -0.754285
2323                -0.754285
2324                -0.754285
```

Plot rolling skew of GLD

[ ]:
```
plt.figure(figsize=(8,2))
alldatacommodities_df[alldatacommodities_df["Ticker"] == "GLD"].rolling_skew.
  ↪plot()
plt.show()
```



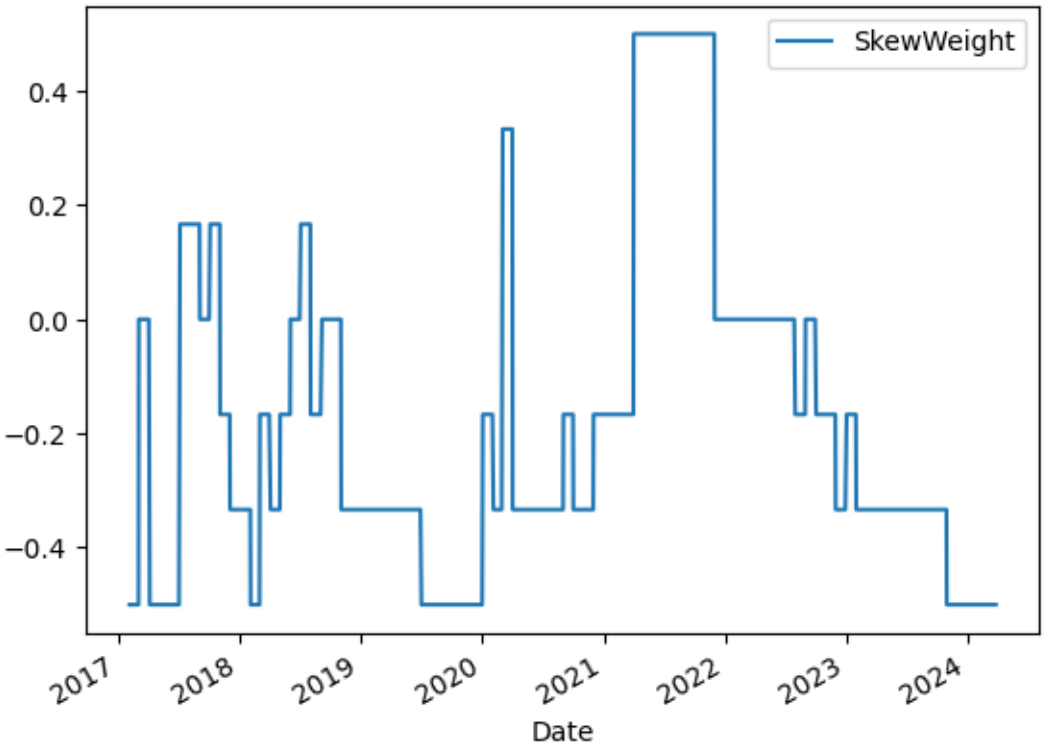
Assign a weight to each ETF in the asset class based on their skew

[ ]:
```
commodities = ["GLD", "SLV", "GSG", "USO", "PPLT", "UNG", "DBA"]
alldatacommodities_df['SkewWeightRaw']=alldatacommodities_df.
  ↪groupby('Date')['EOM_rolling_skew_lookback'].
  ↪rank(ascending=False)-(len(commodities)+1)/2
alldatacommodities_df['SkewWeight']=alldatacommodities_df['SkewWeightRaw']/np.
  ↪sum(np.arange(1, alldatacommodities_df['SkewWeightRaw'].max()+0.1, 1))
```

Plot skew weight of GLD

[ ]:
```
plt.figure(figsize=(8,2))
alldatacommodities_df[alldatacommodities_df["Ticker"] == "GLD"].plot(x='Date',␣
  ↪y='SkewWeight')
plt.show()
```

```
<Figure size 800x200 with 0 Axes>
```



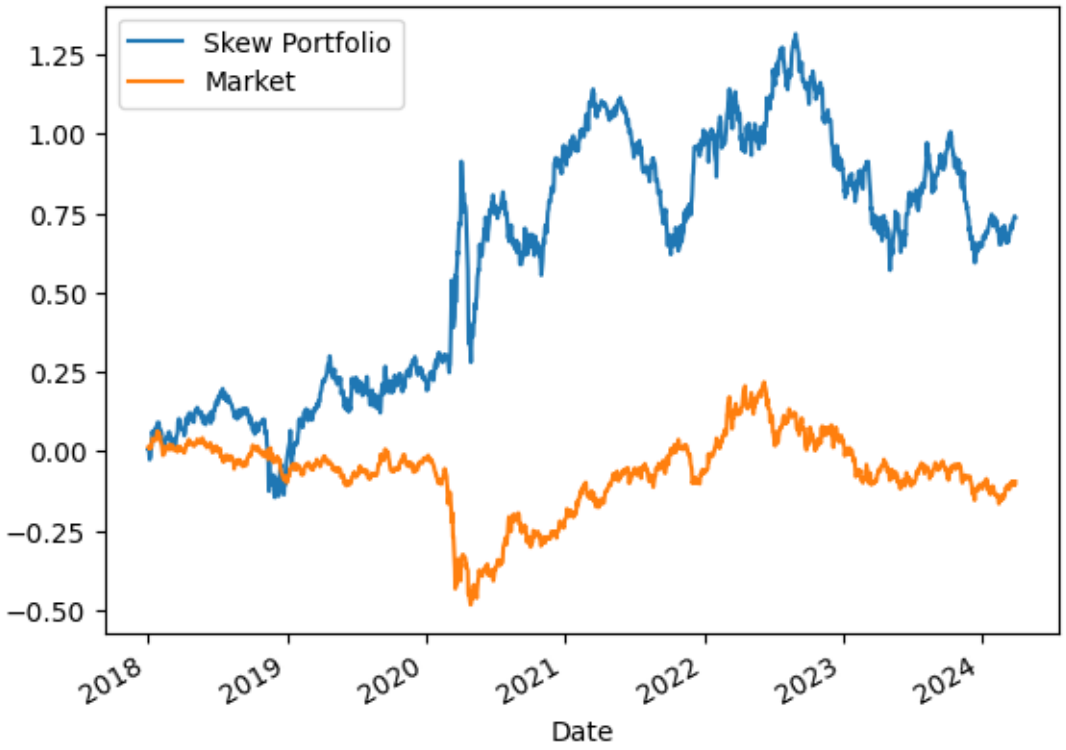
Calculate the returns of the skew portfolio and of the market consisting out of the commodity ETFs

[ ]:
```
alldatacommodities_df['WeightxLogret']=(alldatacommodities_df['SkewWeight']*alldatacommodities
alldatacommodities_df = alldatacommodities_df[alldatacommodities_df["Date"] >=␣
 ↪"2018-01-01"]
groupings = alldatacommodities_df.
 ↪groupby(['Ticker'],group_keys=False)['WeightxLogret'].cumsum()
alldatacommodities_df['ReturnIndividual'] = groupings.transform(lambda x: x)
groupings = alldatacommodities_df.
 ↪groupby(['Date'],group_keys=False)['ReturnIndividual'].sum()
alldatacommodities_df['PortfolioReturn'] = groupings.transform(lambda x: x)
alldatacommodities_df.tail(50)
groupings = alldatacommodities_df.
 ↪groupby(['Date'],group_keys=False)['ReturnIndividual'].sum()
PortfolioReturnsCommodities = groupings.transform(lambda x: x)
PortfolioReturnsCommodities.plot()

groupings = alldatacommodities_df.groupby(['Ticker'],group_keys=False)['ret'].
 ↪cumsum()
alldatacommodities_df['MarketReturnIndividual'] = groupings.transform(lambda x:␣
 ↪x)
```

```
groupings = alldatacommodities_df.
  ↪groupby(['Date'],group_keys=False)['MarketReturnIndividual'].sum()
MarketReturnsCommodities = groupings.transform(lambda x: x)
MarketReturnsCommodities = MarketReturnsCommodities/len(commodities)
MarketReturnsCommodities.plot()

plt.gca().legend(('Skew Portfolio','Market'))
plt.show
```

```
<ipython-input-58-d0155b3a514d>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  alldatacommodities_df['ReturnIndividual'] = groupings.transform(lambda x: x)
<ipython-input-58-d0155b3a514d>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  alldatacommodities_df['PortfolioReturn'] = groupings.transform(lambda x: x)
```

[ ]: <function matplotlib.pyplot.show(close=None, block=None)>



Can see that skew portfolio outperforms the market during the testing period

Perform OLS regression to determine alpha and market beta (and their respective p values).

[ ]:
```
MarketReturnsCommodities1 = sm.add_constant(MarketReturnsCommodities)

result = sm.OLS(PortfolioReturnsCommodities, MarketReturnsCommodities1).fit()

# printing the summary table
print(result.summary())
result.params
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       ReturnIndividual   R-squared:                       0.001
Model:                            OLS   Adj. R-squared:                 -0.000
Method:                 Least Squares   F-statistic:                    0.8679
Date:                Thu, 23 May 2024   Prob (F-statistic):              0.352
Time:                        21:37:59   Log-Likelihood:                -726.59
No. Observations:                1570   AIC:                             1457.
Df Residuals:                    1568   BIC:                             1468.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
==========
                             coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
----------
const                      0.6194      0.011     55.586      0.000       0.598
0.641
MarketReturnIndividual     0.0749      0.080      0.932      0.352      -0.083
0.233
==============================================================================
Omnibus:                     9222.891   Durbin-Watson:                   0.003
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              136.206
Skew:                          -0.278   Prob(JB):                     2.65e-30
Kurtosis:                       1.668   Cond. No.                         8.32
==============================================================================

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

[ ]:
```
const                     0.619366
MarketReturnIndividual    0.074890
dtype: float64
```

[ ]:
```
linregress(MarketReturnsCommodities,PortfolioReturnsCommodities)
```

[ ]:
```
LinregressResult(slope=0.07488976611078871, intercept=0.6193664390096213,
rvalue=0.023519989023177576, pvalue=0.3516857861382099,
stderr=0.0803881757735674, intercept_stderr=0.011142394715860414)
```

3 Repeat process for Equity ETFs

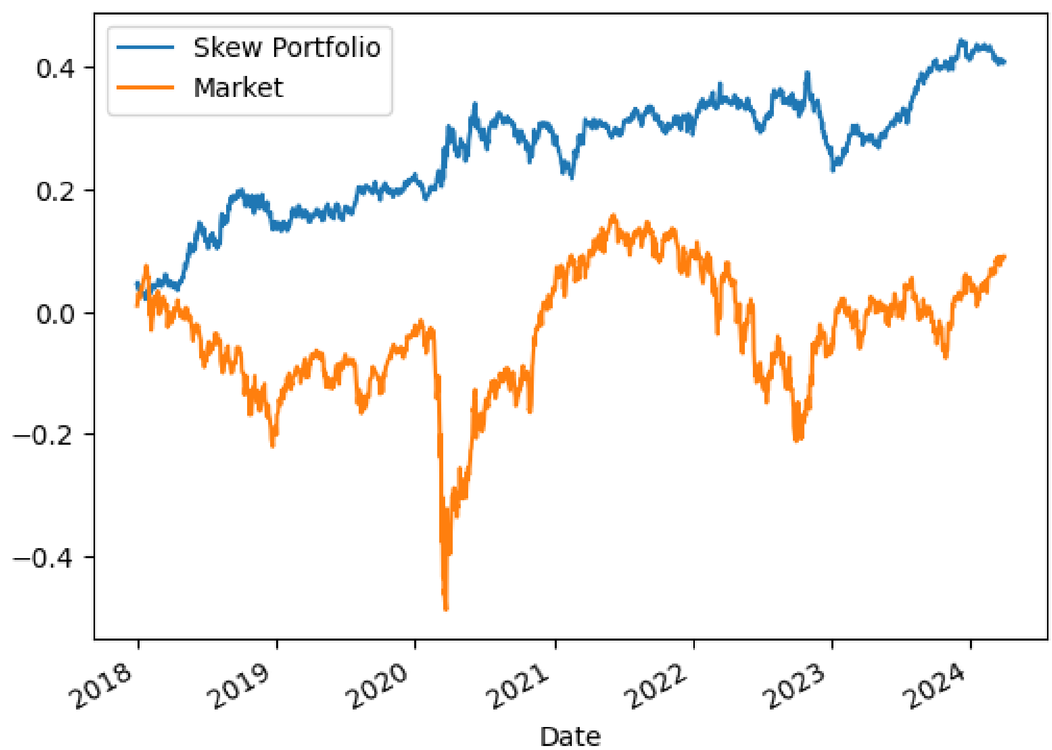
[ ]:
```
equity = ["SPY", "EWU", "EWJ", "INDA", "EWG", "EWL", "EWP", "EWQ",
                        "VTI", "FXI", "EWZ", "EWY", "EWA", "EWC", "EWG",
                        "EWH", "EWI", "EWN", "EWD", "EWT", "EZA", "EWW",␣
 ↪"ENOR", "EDEN", "TUR"]

alldataequity  = []
for j in equity:
  individualdf =  get_data(j,startdate,enddate)
  individualdf
  individualdf = individualdf.drop(columns=['Open','High','Low','Adj␣
 ↪Close','Volume'])
  individualdf['pct_change'] = individualdf.Close.pct_change()
  individualdf['ret'] = np.log(individualdf.Close) - np.log(individualdf.Close.
 ↪shift(1))
  individualdf['rolling_mean'] = individualdf.ret.rolling(256).mean()
  individualdf['rolling_std'] = individualdf.ret.rolling(256).std()
  individualdf['skew_day'] = ((individualdf.ret-individualdf.rolling_mean)/
 ↪individualdf.rolling_std)**3
  individualdf['rolling_skew'] = individualdf.skew_day.rolling(256).mean()
  individualdf = individualdf.reset_index()
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['Date']
  individualdf.groupby([individualdf.Date.dt.year, individualdf.Date.dt.
 ↪month],group_keys=False).max()
  individualdf['EOM'] = groupings.transform(lambda x: x.max())
  individualdf['EOM_rolling_skew'] = groupings.transform(lambda x:␣
 ↪individualdf[individualdf["Date"] == x.max()].rolling_skew)
  individualdf['EOM_rolling_skew_lookback'] = individualdf.EOM_rolling_skew.
 ↪shift(1)
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['EOM_rolling_skew']
  individualdf['EOM_rolling_skew'] = groupings.transform(lambda x: x.max())
```

```
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['EOM_rolling_skew_lookback']
  individualdf['EOM_rolling_skew_lookback'] = groupings.transform(lambda x: x.
 ↪max())

  alldataequity.append(individualdf)

#alldataequity

alldataequity_df = pd.concat(alldataequity)

alldataequity_df['SkewWeightRaw']=alldataequity_df.
 ↪groupby('Date')['EOM_rolling_skew_lookback'].
 ↪rank(ascending=False)-(len(equity)+1)/2
alldataequity_df['SkewWeight']=alldataequity_df['SkewWeightRaw']/np.sum(np.
 ↪arange(1, alldataequity_df['SkewWeightRaw'].max()+0.1, 1))

alldataequity_df['WeightxLogret']=(alldataequity_df['SkewWeight']*alldataequity_df['ret'])
alldataequity_df = alldataequity_df[alldataequity_df["Date"] >= "2018-01-01"]
groupings = alldataequity_df.
 ↪groupby(['Ticker'],group_keys=False)['WeightxLogret'].cumsum()
alldataequity_df['ReturnIndividual'] = groupings.transform(lambda x: x)
groupings = alldataequity_df.
 ↪groupby(['Date'],group_keys=False)['ReturnIndividual'].sum()
alldataequity_df['PortfolioReturn'] = groupings.transform(lambda x: x)
alldataequity_df.tail(50)
groupings = alldataequity_df.
 ↪groupby(['Date'],group_keys=False)['ReturnIndividual'].sum()
PortfolioReturnsEquity = groupings.transform(lambda x: x)
PortfolioReturnsEquity.plot()

groupings = alldataequity_df.groupby(['Ticker'],group_keys=False)['ret'].
 ↪cumsum()
alldataequity_df['MarketReturnIndividual'] = groupings.transform(lambda x: x)

groupings = alldataequity_df.
 ↪groupby(['Date'],group_keys=False)['MarketReturnIndividual'].sum()
MarketReturnsEquity = groupings.transform(lambda x: x)
MarketReturnsEquity = MarketReturnsEquity/len(equity)
MarketReturnsEquity.plot()

plt.gca().legend(('Skew Portfolio','Market'))
plt.show
```

```
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
```

```
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
```

```
[ ]: <function matplotlib.pyplot.show(close=None, block=None)>
```



[ ]:
```
MarketReturnsEquity1 = sm.add_constant(MarketReturnsEquity)

result = sm.OLS(PortfolioReturnsEquity, MarketReturnsEquity1).fit()

# printing the summary table
print(result.summary())
result.params
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       ReturnIndividual   R-squared:                       0.057
Model:                            OLS   Adj. R-squared:                  0.057
Method:                 Least Squares   F-statistic:                     95.00
Date:                Thu, 23 May 2024   Prob (F-statistic):           7.82e-22
Time:                        21:38:23   Log-Likelihood:                 1485.6
No. Observations:                1570   AIC:                            -2967.
Df Residuals:                    1568   BIC:                            -2956.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
==========
                             coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
----------
const                      0.2691      0.003    107.189      0.000       0.264
0.274
MarketReturnIndividual     0.2262      0.023      9.747      0.000       0.181
0.272
==============================================================================
Omnibus:                       85.562   Durbin-Watson:                   0.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               98.898
Skew:                          -0.613   Prob(JB):                     3.35e-22
Kurtosis:                       3.084   Cond. No.                         9.79
==============================================================================

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

[ ]:
```
const                    0.269098
MarketReturnIndividual   0.226160
dtype: float64
```

[ ]:
```
linregress(MarketReturnsEquity,PortfolioReturnsEquity)
```

[ ]:
```
LinregressResult(slope=0.22616040262392453, intercept=0.26909838683521675,
rvalue=0.23901080538747735, pvalue=7.81517219279385e-22,
stderr=0.023203462476002936, intercept_stderr=0.0025105150412339113)
```

4 Repeat process for fixed income ETFs

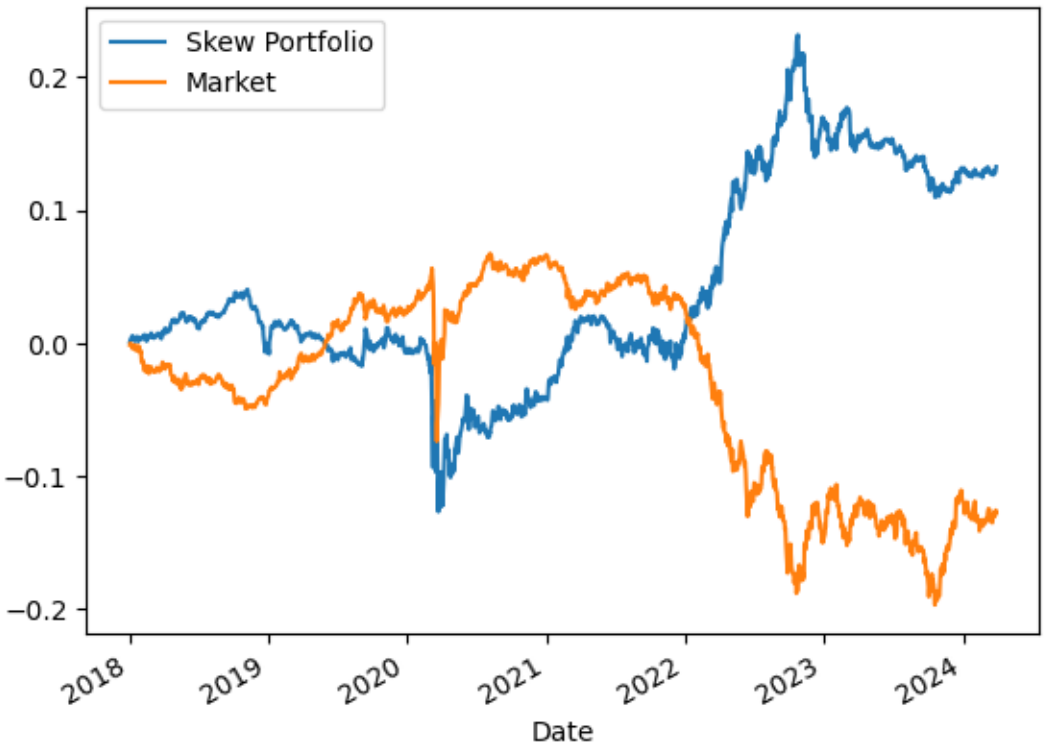
[ ]:
```
FI = ["AGG", "TLT", "LQD", "JNK", "MUB", "MBB", "IGOV", "EMB", "BND", "BNDX",␣
 ↪"VCIT", "VCSH", "BSV", "SRLN"]

alldataFI  = []
for j in FI:
  individualdf =  get_data(j,startdate,enddate)
  individualdf
  individualdf = individualdf.drop(columns=['Open','High','Low','Adj␣
 ↪Close','Volume'])
  individualdf['pct_change'] = individualdf.Close.pct_change()
  individualdf['ret'] = np.log(individualdf.Close) - np.log(individualdf.Close.
 ↪shift(1))
  individualdf['rolling_mean'] = individualdf.ret.rolling(256).mean()
  individualdf['rolling_std'] = individualdf.ret.rolling(256).std()
  individualdf['skew_day'] = ((individualdf.ret-individualdf.rolling_mean)/
 ↪individualdf.rolling_std)**3
  individualdf['rolling_skew'] = individualdf.skew_day.rolling(256).mean()
  individualdf = individualdf.reset_index()
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['Date']
  individualdf.groupby([individualdf.Date.dt.year, individualdf.Date.dt.
 ↪month],group_keys=False).max()
  individualdf['EOM'] = groupings.transform(lambda x: x.max())
  individualdf['EOM_rolling_skew'] = groupings.transform(lambda x:␣
 ↪individualdf[individualdf["Date"] == x.max()].rolling_skew)
  individualdf['EOM_rolling_skew_lookback'] = individualdf.EOM_rolling_skew.
 ↪shift(1)
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['EOM_rolling_skew']
  individualdf['EOM_rolling_skew'] = groupings.transform(lambda x: x.max())
  groupings = individualdf.groupby([individualdf.Date.dt.year, individualdf.
 ↪Date.dt.month],group_keys=False)['EOM_rolling_skew_lookback']
  individualdf['EOM_rolling_skew_lookback'] = groupings.transform(lambda x: x.
 ↪max())

  alldataFI.append(individualdf)

#alldataFI
```

```
alldataFI_df = pd.concat(alldataFI)

alldataFI_df['SkewWeightRaw']=alldataFI_df.
 ↪groupby('Date')['EOM_rolling_skew_lookback'].
 ↪rank(ascending=False)-(len(FI)+1)/2
alldataFI_df['SkewWeight']=alldataFI_df['SkewWeightRaw']/np.sum(np.arange(1,␣
 ↪alldataFI_df['SkewWeightRaw'].max()+0.1, 1))

alldataFI_df['WeightxLogret']=(alldataFI_df['SkewWeight']*alldataFI_df['ret'])
alldataFI_df = alldataFI_df[alldataFI_df["Date"] >= "2018-01-01"]
groupings = alldataFI_df.groupby(['Ticker'],group_keys=False)['WeightxLogret'].
 ↪cumsum()
alldataFI_df['ReturnIndividual'] = groupings.transform(lambda x: x)
groupings = alldataFI_df.groupby(['Date'],group_keys=False)['ReturnIndividual'].
 ↪sum()
alldataFI_df['PortfolioReturn'] = groupings.transform(lambda x: x)
alldataFI_df.tail(50)
groupings = alldataFI_df.groupby(['Date'],group_keys=False)['ReturnIndividual'].
 ↪sum()
PortfolioReturnsFI = groupings.transform(lambda x: x)
PortfolioReturnsFI.plot()

groupings = alldataFI_df.groupby(['Ticker'],group_keys=False)['ret'].cumsum()
alldataFI_df['MarketReturnIndividual'] = groupings.transform(lambda x: x)

groupings = alldataFI_df.
 ↪groupby(['Date'],group_keys=False)['MarketReturnIndividual'].sum()
MarketReturnsFI = groupings.transform(lambda x: x)
MarketReturnsFI = MarketReturnsFI/len(FI)
MarketReturnsFI.plot()

plt.gca().legend(('Skew Portfolio','Market'))
plt.show
```

```
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
[**********************100%%**********************]  1 of 1 completed
```

```
[**********100%%**********]  1 of 1 completed
```

[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



[ ]:
```
MarketReturnsFI1 = sm.add_constant(MarketReturnsFI)

result = sm.OLS(PortfolioReturnsFI, MarketReturnsFI1).fit()

# printing the summary table
print(result.summary())
result.params
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       ReturnIndividual   R-squared:                       0.846
Model:                            OLS   Adj. R-squared:                  0.846
Method:                 Least Squares   F-statistic:                     8596.
Date:                Thu, 23 May 2024   Prob (F-statistic):               0.00
Time:                        21:38:28   Log-Likelihood:                 3295.5
No. Observations:                1570   AIC:                            -6587.
Df Residuals:                    1568   BIC:                            -6576.
Df Model:                           1
Covariance Type:            nonrobust
```

```
==============================================================================
==========
                             coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
----------
const                       0.0105      0.001     12.867      0.000       0.009
0.012
MarketReturnIndividual     -0.9312      0.010    -92.714      0.000      -0.951
-0.912
==============================================================================
Omnibus:                      407.671   Durbin-Watson:                   0.025
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1231.767
Skew:                          -1.304   Prob(JB):                    3.35e-268
Kurtosis:                       6.468   Cond. No.                         13.4
==============================================================================

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

[ ]:
```
const                    0.010537
MarketReturnIndividual  -0.931231
dtype: float64
```

[ ]:
```
linregress(MarketReturnsFI,PortfolioReturnsFI)
```

[ ]:
```
LinregressResult(slope=-0.9312311393755575, intercept=0.010536975976605373,
rvalue=-0.9196346741533906, pvalue=0.0, stderr=0.010044140312317765,
intercept_stderr=0.0008189060526586914)
```

5 Global Skewness Factor Portfolio

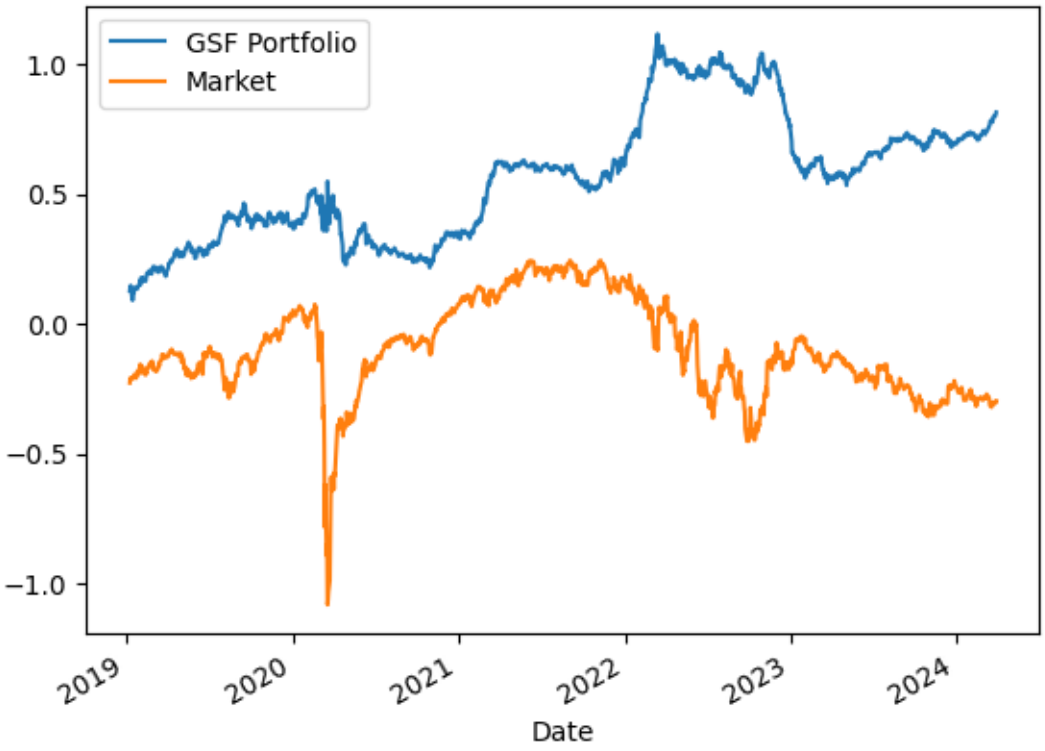
Skew portfolios from different assets have low correlation (check) => combine them into one diverse portfolio.

Scale each asset class skew portfolio to have a full sample volatility of 10% and combine them all on an equal-weight basis.

[ ]:
```
ReturnsFI_df = PortfolioReturnsFI.to_frame()
ReturnsFI_df["MarketReturnIndividual"] = MarketReturnsFI
ReturnsFI_df["Asset"] = "FI"

ReturnsEquity_df = PortfolioReturnsEquity.to_frame()
ReturnsEquity_df["MarketReturnIndividual"] = MarketReturnsEquity
ReturnsEquity_df["Asset"] = "Equity"
```

```
ReturnsCommodities_df = PortfolioReturnsCommodities.to_frame()
ReturnsCommodities_df["MarketReturnIndividual"] = MarketReturnsCommodities
ReturnsCommodities_df["Asset"] = "Commodities"

ReturnsCombined_df = ReturnsCommodities_df._append(ReturnsEquity_df,␣
 ↪ignore_index=False)
ReturnsCombined_df = ReturnsCombined_df._append(ReturnsFI_df,␣
 ↪ignore_index=False)

ReturnsCombined_df = ReturnsCombined_df.reset_index()
ReturnsCombined_df


groupings = ReturnsCombined_df.
 ↪groupby(['Asset'],group_keys=False)['ReturnIndividual']
ReturnsCombined_df['Std'] = groupings.transform(lambda x: x.rolling(256).std())
ReturnsCombined_df['NormReturnIndividual'] = 0.1*ReturnsCombined_df.
 ↪ReturnIndividual/ReturnsCombined_df.Std
ReturnsCombined_df['NormMarketReturnIndividual'] = 0.1*ReturnsCombined_df.
 ↪MarketReturnIndividual/ReturnsCombined_df.Std
ReturnsCombined_df

groupings = ReturnsCombined_df.
 ↪groupby(['Date'],group_keys=False)['NormReturnIndividual'].mean()
GSF_Portfolio = groupings.transform(lambda x: x)
groupings = ReturnsCombined_df.
 ↪groupby(['Date'],group_keys=False)['NormMarketReturnIndividual'].mean()
GSF_Market = groupings.transform(lambda x: x)
GSF_Portfolio.plot()
GSF_Market.plot()

plt.gca().legend(('GSF Portfolio','Market'))
plt.show
```

[ ]: <function matplotlib.pyplot.show(close=None, block=None)>



[ ]:
```
GSF_Market.dropna()
GSF_Market.dropna()

GSF_Market1 = sm.add_constant(GSF_Market)

result = sm.OLS(GSF_Portfolio.dropna(), GSF_Market1.dropna()).fit()

# printing the summary table
print(result.summary())
result.params
```

OLS Regression Results

Dep. Variable:	NormReturnIndividual	R-squared:	0.006
Model:	OLS	Adj. R-squared:	0.005
Method:	Least Squares	F-statistic:	7.958
Date:	Thu, 23 May 2024	Prob (F-statistic):	0.00486
Time:	21:38:29	Log-Likelihood:	4.0825
No. Observations:	1315	AIC:	-4.165
Df Residuals:	1313	BIC:	6.198
Df Model:	1		
Covariance Type:	nonrobust		

```
==============
                                 coef    std err          t      P>|t|
[0.025      0.975]
------------------------------------------------------------------------------
--------------
const                          0.5556      0.008     73.923      0.000
0.541       0.570
NormMarketReturnIndividual    -0.1000      0.035     -2.821      0.005
-0.170      -0.030
==============================================================================
Omnibus:                      136.510   Durbin-Watson:                   0.003
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               50.544
Skew:                           0.242   Prob(JB):                     1.06e-11
Kurtosis:                       2.171   Cond. No.                         5.38
==============================================================================

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

[ ]:
```
const                          0.555588
NormMarketReturnIndividual    -0.100035
dtype: float64
```

TODO: Check if skew is providing new information that is not available from Value, Momentum, and Carry Factors

For equity ETFs simple to check via value/momentum/carry ETFs