# Clustering Analysis

## US Stock Return Correlation

Group 3 - 2021 April 3

## Contents

## Steps

- Downloaded 1000+ Nasdaq stock + 2000+ NYSE stock prices from Yahoo! Finance
- Calculated daily return for the stocks and calculated correlation among the returns (SP500 component stocks etc.)
- Transfer the correlation to distance -> $d = \sqrt{\frac{1}{2}(1-\rho)}$
- Clustering based on the distance (for SP500 companies, sectors, industries)

## Data Downloading

- Prepared symbol list file (tickers of small, mid, large, mega cap companies) (Nasdaq 1358, NYSE 2061)
- Use downloading script to get historical price data from Yahoo! Finance (time interval from 2011-03-16 to 2021-03-16)
- Use concurrency: futures.ThreadPoolExecutor with max 4 workers -> slightly faster
- Took around 30 minutes to download the data set
- Alternatively, you can use the shared data set here: https://drive.google.com/file/d/1Uy0VmrkbKUAskGKAAQo45F8unrphAF14/view?usp=sharing

## Data Downloading

```
def download_one(symbol):
    if os.path.exists(FILEDIR + '{}.csv'.format(symbol)):
        # logging.info('already downloaded, skip')
        return


    logging.info('downloading for symbol: 
{}'.format(symbol))
    try:
        abc = downloader.downloader(symbol,start,end)
        abc = abc.decode('utf-8')


        with open(FILEDIR + '{}.csv'.format(symbol), "w") 
as f:
            f.writelines(str(abc))
        logging.info('downloaded successfully')
    except:
        logging.error('got error when trying to download: 
{}'.format(symbol))
```

```
def download_many(symbols: list[str]) -> None:
    with futures.ThreadPoolExecutor(max_workers=4) as 
executor:
        res = executor.map(download_one, symbols)


def main():
    logging.info('start downloading process...')


    nasdaq_symbols = read_symbols('nasdaq_symbols.csv')
    nyse_symbols = read_symbols('nyse_symbols.csv')
    symbols = nasdaq_symbols + nyse_symbols


    download_many(symbols)


    logging.info('finished downloading process!')
```

## Data Downloading

Code: downloader.py （https://stackoverflow.com/questions/44030983/yahoo-finance-url-not-working）

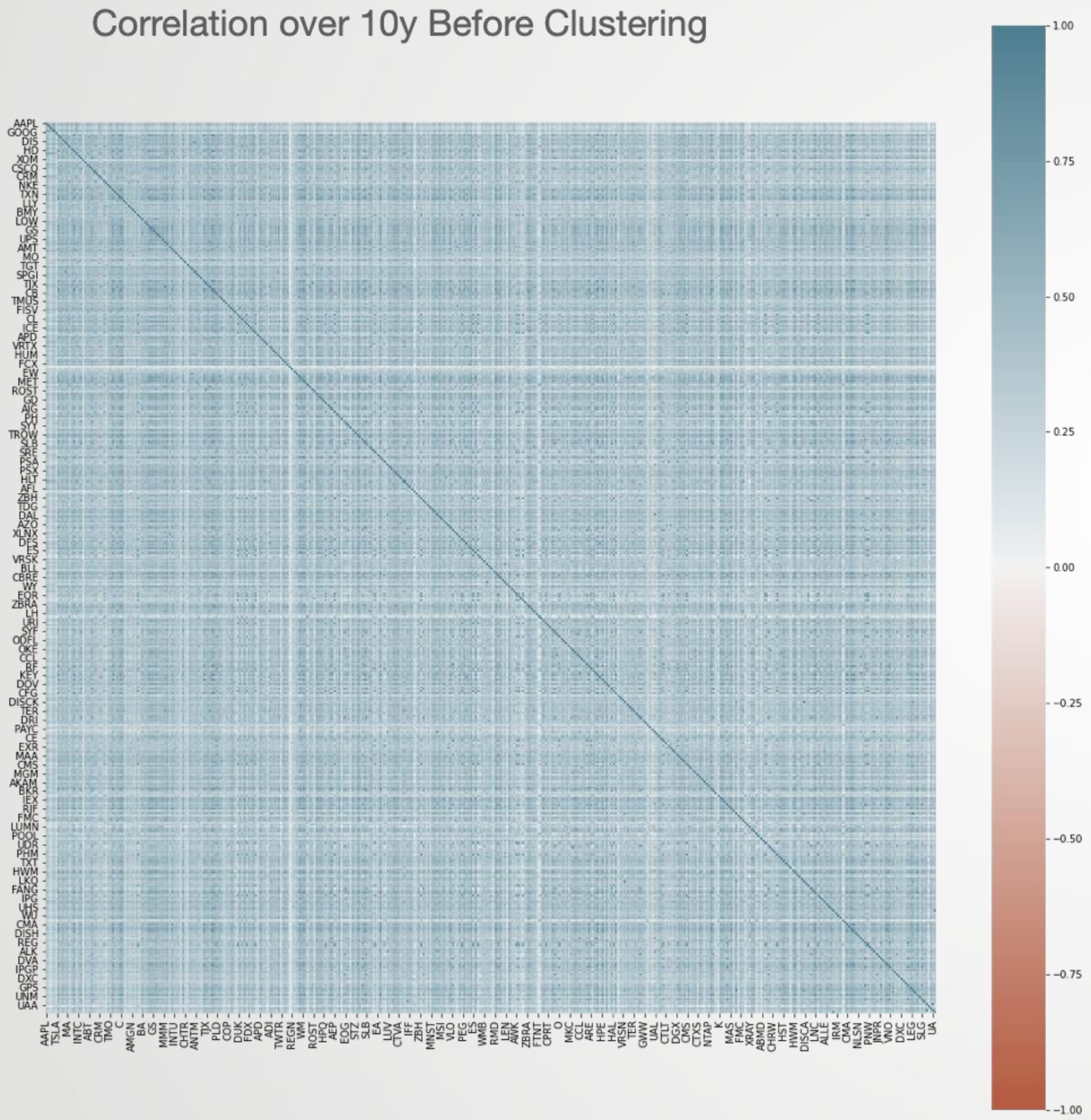
```python
import requests
import re
import json
import time as time

def run(symbol, initial, final):
    start = int(time.mktime(time.strptime(str(initial), 
'%Y-%m-%d')))
    end = int(time.mktime(time.strptime(str(final), '%Y-
%m-%d')))

    abc = downloader(symbol,start,end)
    abc = abc.decode('utf-8')

    with open('{}.csv'.format(symbol), "w") as f:
        f.writelines(str(abc))


def downloader(symbol, start, end):
    ses = requests.session()
    resp = ses.get(

f'https://finance.yahoo.com/quote/{symbol}/history?period
1={start}&period2={end}&interval=1d&filter=history&freque
ncy=1d'
    )
    resp.raise_for_status()
    data = resp.content.decode('utf-8')
    crumb_m = 
re.search(r'"CrumbStore":\{"crumb":("[^"]+")\}', data)
    assert crumb_m
    crumb = json.loads(crumb_m.group(1))
    csvresp = ses.get(

f'https://query1.finance.yahoo.com/v7/finance/download/{s
ymbol}?period1={start}&period2={end}&interval=1d&events=h
istory&crumb={crumb}'
    )
    csvresp.raise_for_status()
   return csvresp.content
```

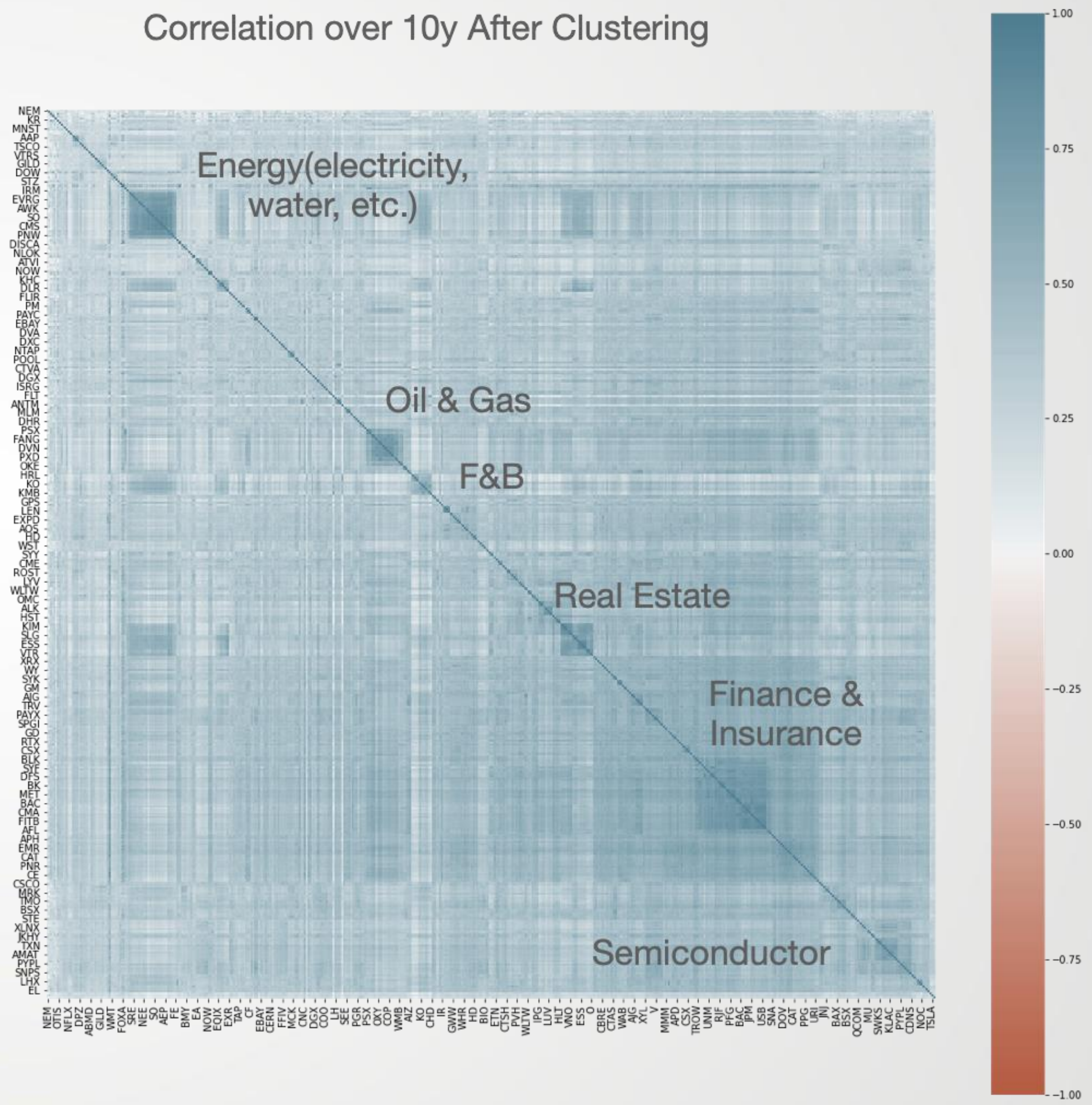
## Data Downloading

Code: auto_download.py

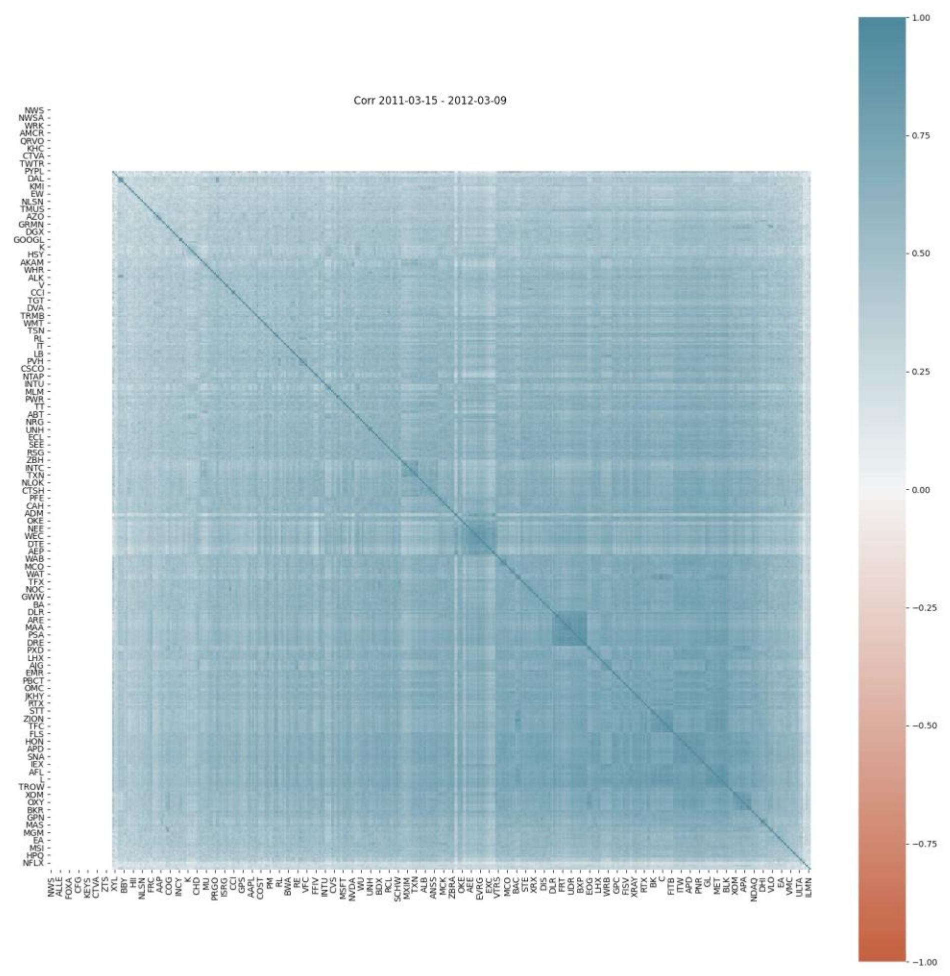
```python
import time
import logging
import downloader
import os
import argparse
from concurrent import futures

FILEDIR = os.path.join('downloaded_data', 'data')
initial = '2011-3-16'
final = '2021-3-16'

formatter = '%(asctime)s - %(name)s - %(levelname)s - %(message)s'
logging.basicConfig(filename='auto_download.log', filemode='w',
level=logging.INFO, format=formatter)

parser = argparse.ArgumentParser()
parser.add_argument("-i", "--initial", help="initial date in %Y-%m-%d format")
parser.add_argument("-f", "--final", help="final date in %Y-%m-%d format")
args = parser.parse_args()
if args.initial:
    initial = args.initial

if args.final:
    final = args.final

logging.info('date range: from {} to {}'.format(initial, final))

start = int(time.mktime(time.strptime(str(initial), '%Y-%m-%d')))
end = int(time.mktime(time.strptime(str(final), '%Y-%m-%d')))

if not os.path.exists(FILEDIR):
    os.makedirs(FILEDIR)

def read_symbols(symfile):
    syms = []
    with open(symfile, 'r') as f:
        content = f.readlines()
        for s in content:
            syms.append(s.strip())
    return syms

def download_one(symbol):
    if os.path.exists(FILEDIR + '{}.csv'.format(symbol)):
        # logging.info('already downloaded, skip')
        return

    logging.info('downloading for symbol: {}'.format(symbol))
    try:
        abc = downloader.downloader(symbol,start,end)
        abc = abc.decode('utf-8')

        with open(FILEDIR + '{}.csv'.format(symbol), "w") as f:
            f.writelines(str(abc))
        logging.info('downloaded successfully')
    except:
        logging.error('got error when trying to download: {}'.format(symbol))


def download_many(symbols: list[str]) -> None:
    with futures.ThreadPoolExecutor(max_workers=4) as executor:
        res = executor.map(download_one, symbols)

def main():
    logging.info('start downloading process...')

    nasdaq_symbols = read_symbols('nasdaq_symbols.csv')
    nyse_symbols = read_symbols('nyse_symbols.csv')
    symbols = nasdaq_symbols + nyse_symbols

    download_many(symbols)

    logging.info('finished downloading process!')


if __name__ == '__main__':
    main()
```

## S&P500 Correlation





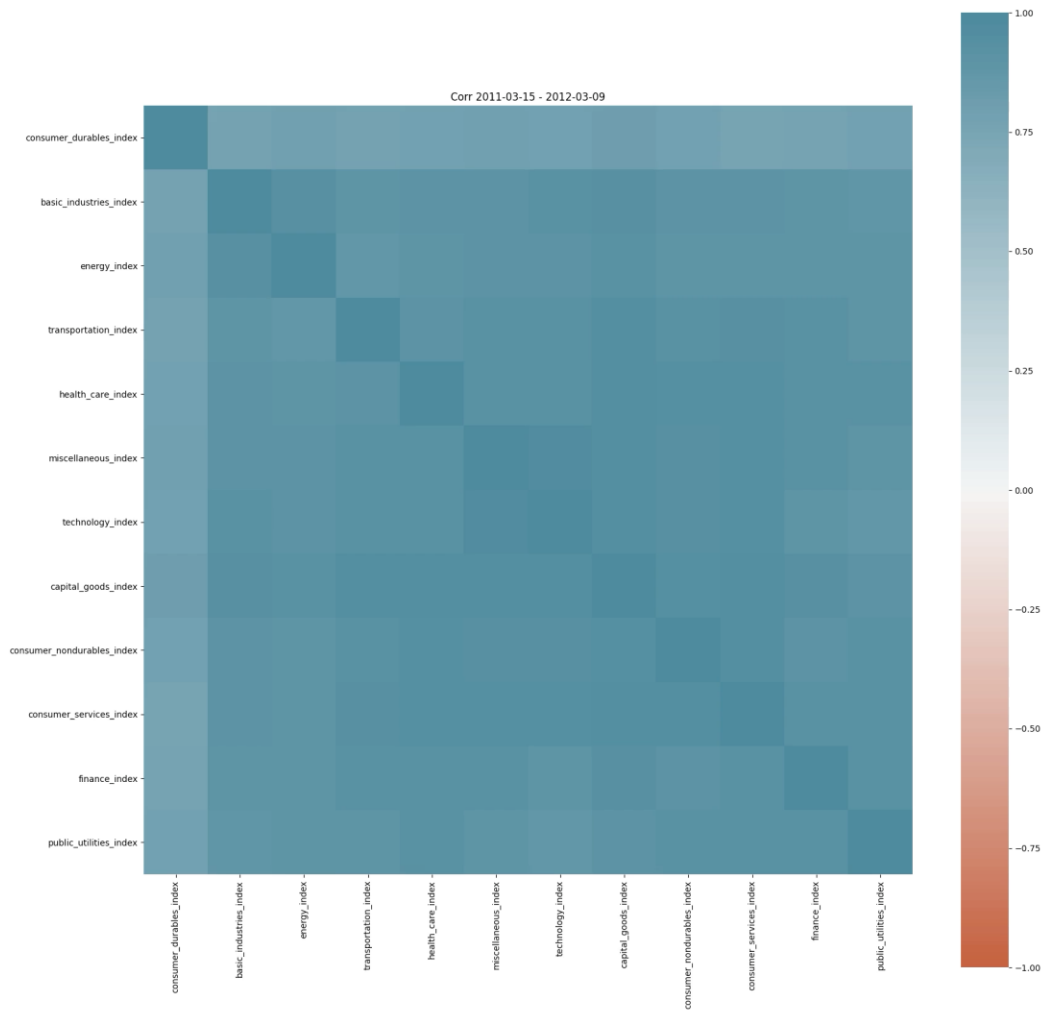
## S&P500 Correlation Clustering



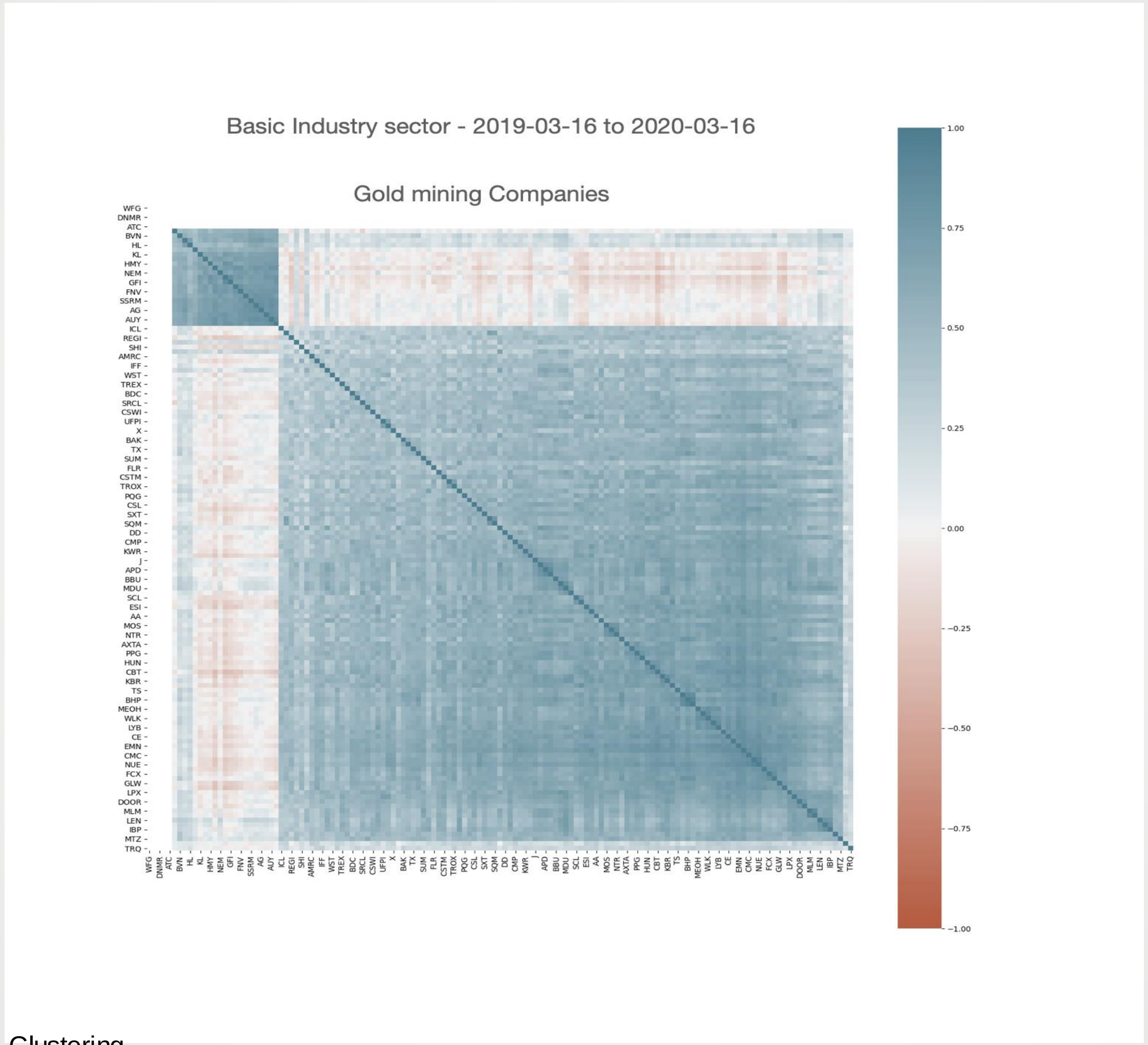
## Sectors

- basic industries
- capital goods
- consumer durables
- consumer nondurables
- consumer services
- energy
- finance
- health care
- public utilities
- technology
- transportation
- miscellaneous

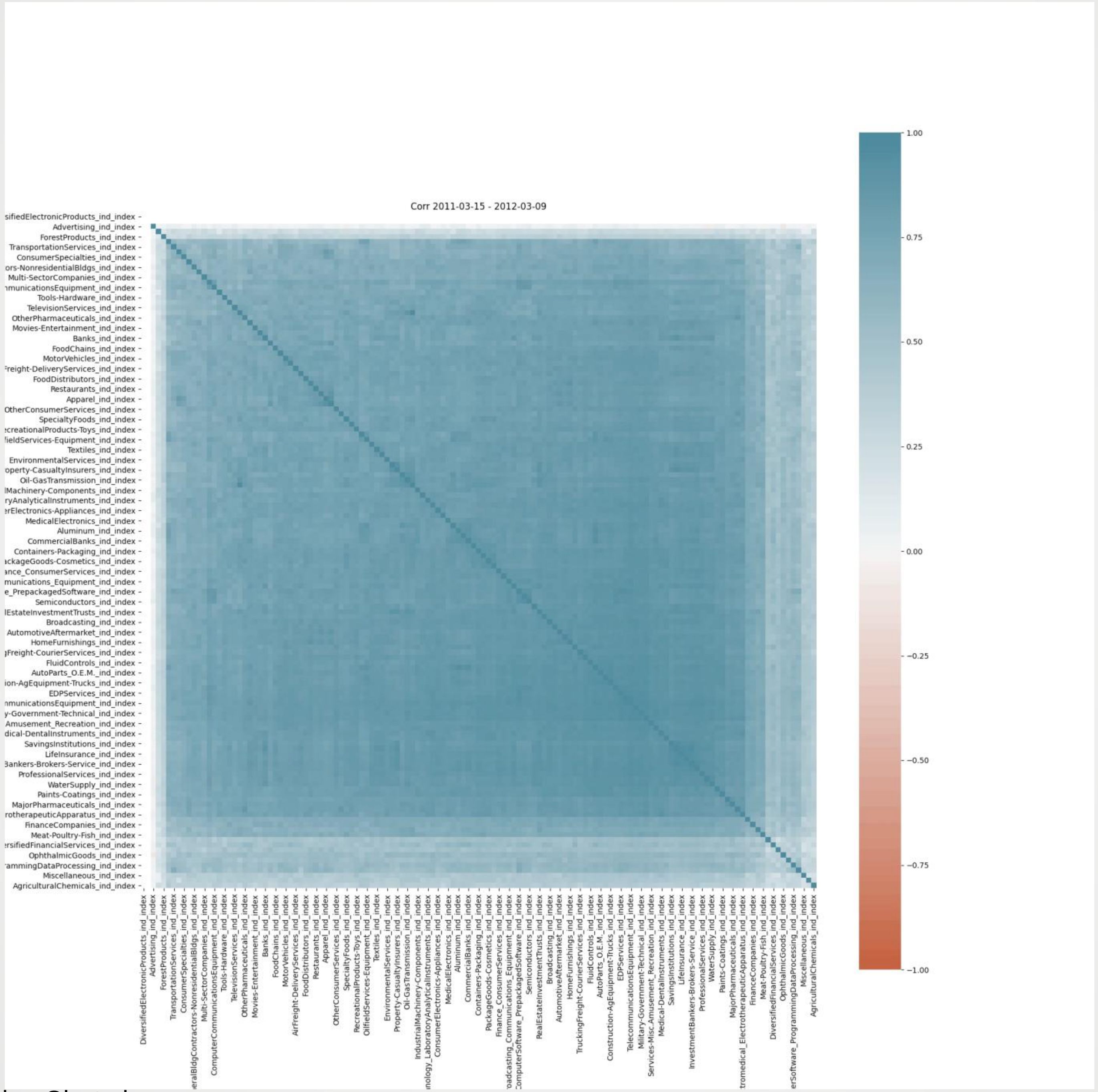
## Correlation among Sectors



## Correlation within Sector



## Industries


Corr 2011-03-15 - 2012-03-09
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
DiversifiedElectronicProducts_ind_index
Advertising_ind_index
ForestProducts_ind_index
TransportationServices_ind_index
ConsumerSpecialties_ind_index
Multi-SectorCompanies_ind_index
Tools-Hardware_ind_index
TelevisionServices_ind_index
OtherPharmaceuticals_ind_index
Movies-Entertainment_ind_index
Banks_ind_index
FoodChains_ind_index
MotorVehicles_ind_index
FoodDistributors_ind_index
Restaurants_ind_index
Apparel_ind_index
OtherConsumerServices_ind_index
SpecialtyFoods_ind_index
Textiles_ind_index
EnvironmentalServices_ind_index
Oil-GasTransmission_ind_index
MedicalElectronics_ind_index
Aluminum_ind_index
CommercialBanks_ind_index
Containers-Packaging_ind_index
Semiconductors_ind_index
Broadcasting_ind_index
AutomotiveAftermarket_ind_index
HomeFurnishings_ind_index
FluidControls_ind_index
AutoParts_O.E.M._ind_index
EDPServices_ind_index
SavingsInstitutions_ind_index
LifeInsurance_ind_index
ProfessionalServices_ind_index
WaterSupply_ind_index
Paints-Coatings_ind_index
MajorPharmaceuticals_ind_index
FinanceCompanies_ind_index
Meat-Poultry-Fish_ind_index
OphthalmicGoods_ind_index
Miscellaneous_ind_index
AgriculturalChemicals_ind_index

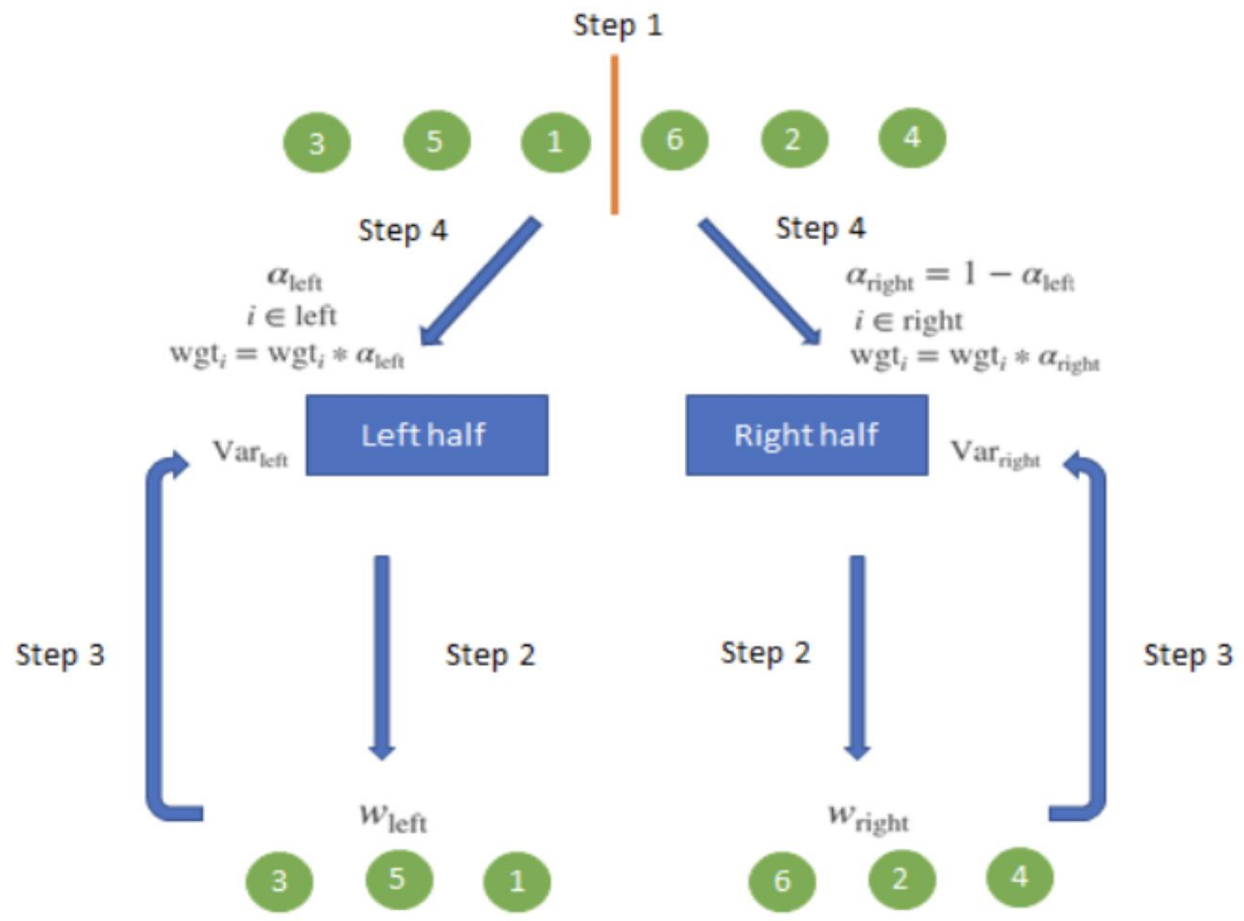
## HRP (hierarchical risk parity)

$$w = \frac{(diag(V))^{-1}}{trace(diag(V)^{-1})}$$

$$\tilde{V} = w'Vw$$

$$\alpha_{left} = 1 - \frac{\tilde{V}_{left}}{\tilde{V}_{left} + \tilde{V}_{right}}$$

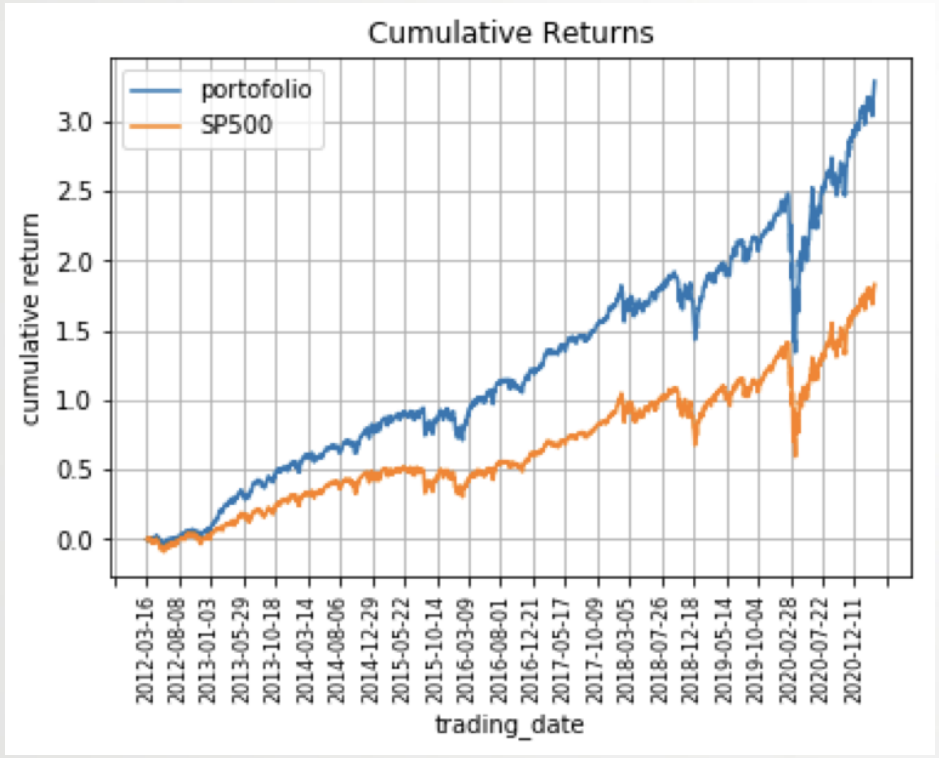
$$\alpha_{right} = 1 - \alpha_{left}$$



(C) Bisect bottom up and top down algorithm

## Backtest Result

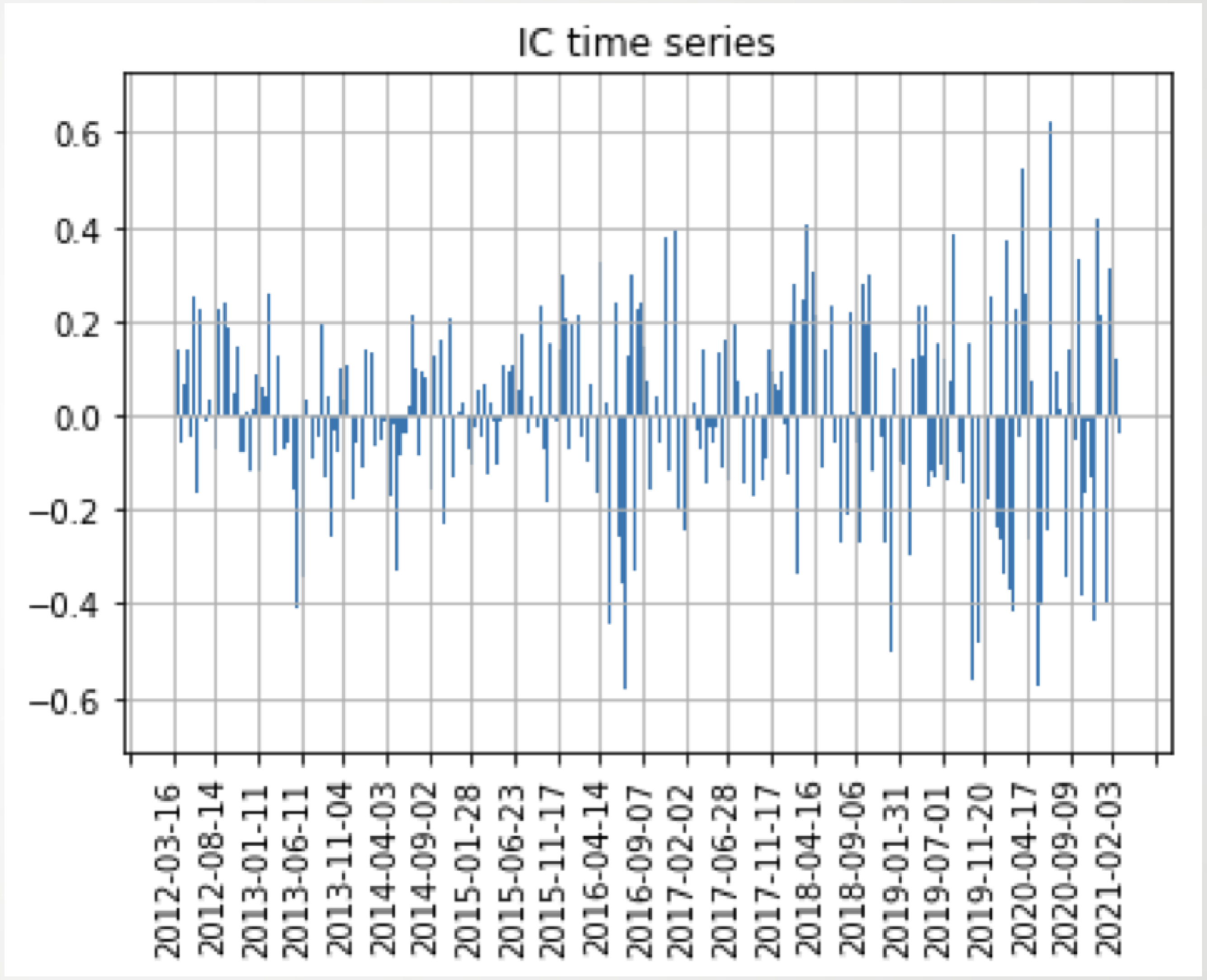
- We use HRP method to compute daily weight of stock in SP500 with the past 252 days stock return.
- It is a long only strategy since the weight is all positive.



| | **Performance** |
|---|---|
| **IC** | 0.0048 |
| **IR** | 0.0231 |
| **annual_return** | 0.1902 |
| **Sharpe** | 1.2309 |
| **Maxdrawdown** | 1.1372 |

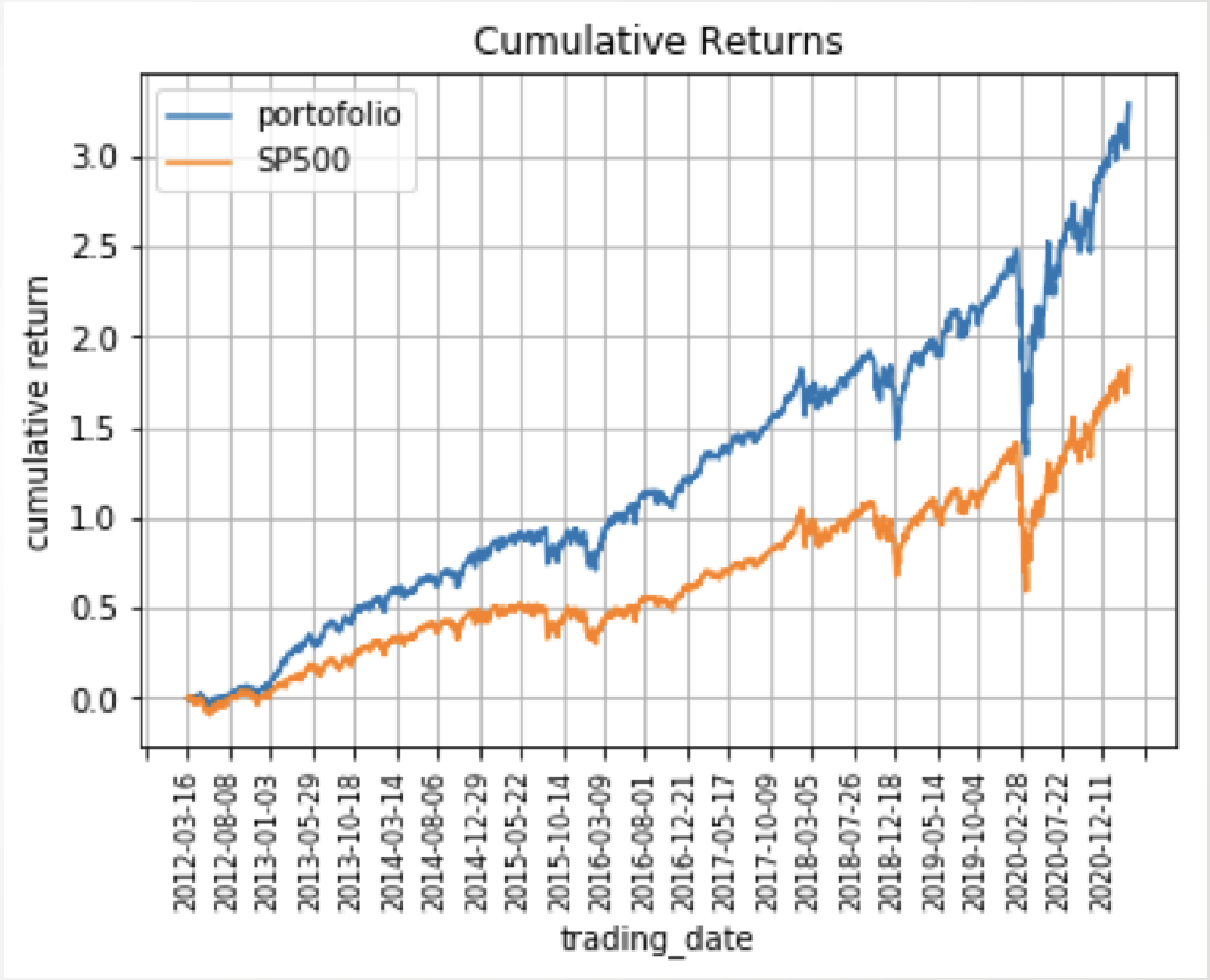
## Result Analysis

- Average IC = 0.005
- IR = 0.023
- It has relative strong signal but not stable



## Result Analysis

- It can beat SP500 index in the past 8 years especially in the bull market.
- It has larger drawdown than SP500 in bear market like the beginning of 2020.



## Further Study

- HRP doesn’t use the hierarchical clustering information sufficiently. It only use the sorted order of clustering data.
- It means this method is not exactly risk parity, e.g. our backtest of SP500 stocks.
- HCRP model use the covariance of assets, which can better weight assets with high correlation.

## References

Marcos Lopez de Prado (2018): *Advances in Financial Machine Learning*. Wiley.

Michaud, R. (1998): *Efficient Asset Allocation: A Practical Guide to Stock Portfolio Optimization and Asset Allocation*, 1st ed. Harvard Business School Press.

Ledoit, O. and M. Wolf (2003): “Improved estimation of the covariance matrix of stock returns with an application to portfolio selection.” *Journal of Empirical Finance*, Vol. 10, No. 5, pp. 603–621.

Raffinot, T. (2017): “Hierarchical clustering based asset allocation.” *Journal of Portfolio Management*

Rokach, L. and O. Maimon (2005): “Clustering methods,” in Rokach, L. and O. Maimon, eds., *Data Mining and Knowledge Discovery Handbook*. Springer, pp. 321–352.

https://stackoverflow.com/questions/44030983/yahoo-finance-url-not-working

**Thanks for listening!**