Investment Competition

KAIST 20214418 Jihyeon Seong

My portfolio is constructed by…

Portfolio Optimization

Markowitz Portfolio Theory(MAIN)

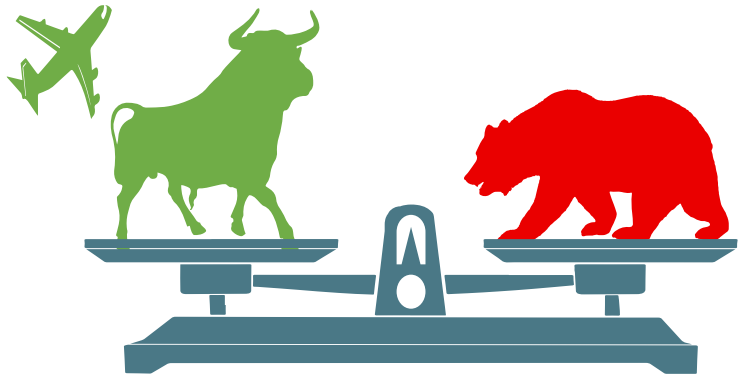
Diversified investment makes good and low risk portfolio.

HRP (Hierarchical Risk Parity)

Use hierarchical tree clustering method for portfolio allocation.

RL Portfolio Allocation (Reinforcement Learning)

Choose ticker(what to trade) and weight based on RL.



Hedging

Risk management

Deep Hedging – call option

Use neural network for hedging with position short

Deep Hedging – put option

Use neural network for hedging with position long

Trading Process

Alpha Square

Make investment

Model

Portfolio Optimization
or Hedging

Try several model

For portfolio allocation, try different models to get better return and Sharpe ratio

Data Preprocessing

Get N Ticker
and Close value

Get N ticker from 369 ticker

Calculate “annualized return and volatility” and filter “return>0.1 and vol<0.3”

Get Data

Get daily Close data
from FinanceDataReader

FinanceDataReader

Can get daily stock data
Stock list from Dacon competition

Before investment - (1) Get Data

① From Dacon “Stock Close Price Prediction Competition”

```
stock_list = pd.read_csv('./stock_list.csv')
stock_list['종목코드'] = stock_list['종목코드'].apply(lambda x : str(x).zfill(6))
stock_list.set_index('종목코드', inplace=True)
stock_list
```

	종목명	종목코드	상장시장
0	삼성전자	005930	KOSPI
1	SK하이닉스	000660	KOSPI
2	NAVER	035420	KOSPI
3	카카오	035720	KOSPI
4	삼성바이오로직스	207940	KOSPI
...	...	...	...
365	맘스터치	220630	KOSDAQ
366	다날	064260	KOSDAQ
367	제이시스메디칼	287410	KOSDAQ
368	크리스에프앤씨	110790	KOSDAQ
369	쎄트렉아이	099320	KOSDAQ

370 rows × 3 columns

② Get Data from FinanceDataReader

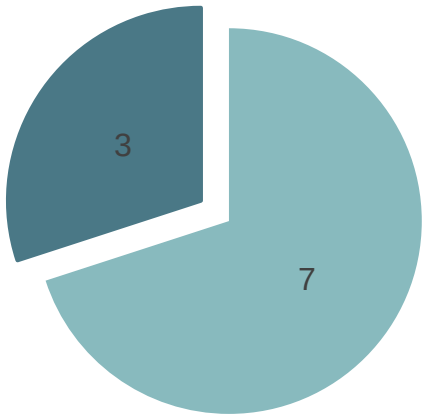
```
ticker=  stock_list['종목코드']
datetime = ['2022-01-01', '2022-05-22']
data = pd.DataFrame()
#data = pd.read_csv('./data.csv')
for code in tqdm(list(ticker)):
    df = pd.DataFrame()
    stock = fdr.DataReader(code, start = datetime[0], end = datetime[1])
    if not stock.empty:
      df[code] = stock['Close']

      df.reset_index(inplace=True, drop=True)

      data = pd.concat([data, df], axis=1)
data = data.interpolate(method='linear', limit_direction='both')
data
```

```
100% 370/370 [05:53<00:00, 1.11it/s]
```

Pre-Investment



시장정보 종목정보 발굴분석 커뮤니티 트레이딩

투자랭킹 투자현황 주식주문

FINAI_평균 계좌보기

평균 4일째 운용 중

총 평가자산	10,000,000원		
보유현금	10,000,000	주식평가금	0
평가수익금	0	누적수익률	0.00%

Let's Start Trading!

I had 10million won for initial value.
Then, what should I buy for the first?

Pre-Investment of bond and ETF

1. Buy stock of US government bond
2. Buy stock of US bond
3. Buy ETF of S&P and Nasdaq

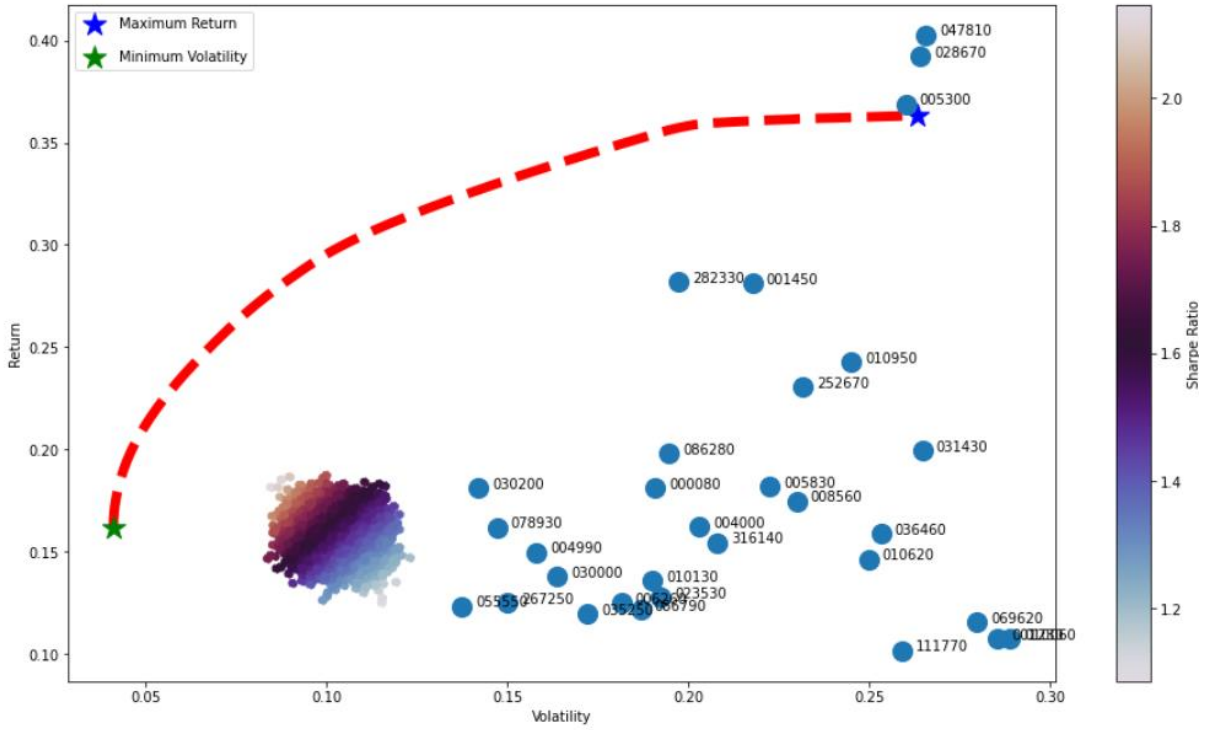
➢ This is for low risk!

KODEX 국채선물10년	현재가 63,935	64,000원 · 5주 -2,020 (-0.63%)
KODEX 미국채10년선물	현재가 11,545	11,565원 · 50주 -4,061 (-0.70%)
TIGER 미국달러단기채권액티브	현재가 10,795	10,830원 · 50주 -4,613 (-0.85%)
KBSTAR KIS단기종합채권(AA-이상)액티브	현재가 100,875	100,875원 · 5주 -2,674 (-0.53%)
KINDEX 미국고배당S&P	현재가 10,470	10,450원 · 50주 -1,773 (-0.34%)
KOSEF 미국방어배당성장나스닥	현재가 13,350	13,335원 · 25주 -1,394 (-0.42%)
ARIRANG 스마트베타Quality채권혼합	현재가 10,640	10,629원 · 10주 -454 (-0.43%)

Portfolio Optimization

Trade 1 & 2 & 3

Markowitz Portfolio-trade 1

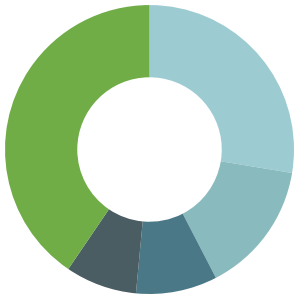


$\mu - \sigma$ diagram

Minimum Volatility Optimization

- Expected return: 0.1621.
- Standard Deviation: 0.0411
- Maximum Return Optimization
  get only 1 company

Portfolio Allocation



Kodex 200선물인버스2X

롯데

신한

현대글로비스

기타 (KT, 현대중공업, 메리츠증권, 한국항공우주, OCI, 제일기획, BFG리테일, LS, 대웅제약, 롯데정밀화학, 신세계 인터네셔날)

Before investment - (2) Data Pre-Processing

① Get annualized return and volatility

```
num_assets = 200
table = data.copy().iloc[:, :num_assets]
returns = table.pct_change()
mean_returns = returns.mean()
cov_matrix = returns.cov()
risk_free_rate = 0.0178
```

```
an_vol = pd.DataFrame(np.std(returns) * np.sqrt(94), columns=['vol'])
an_rt = pd.DataFrame(mean_returns * 94, columns=['rt'])
```

```
an_rt.describe()
```

	rt
count	200.000000
mean	-0.058588
std	0.189259
min	-0.717570
25%	-0.162273
50%	-0.054317
75%	0.057951
max	0.607462

```
an_vol.describe()
```

	vol
count	200.000000
mean	0.221283
std	0.083073
min	0.001110
25%	0.167664
50%	0.220559
75%	0.264303
max	0.569313

② Get ticker with return above 0.1 and volatility below 0.3 *(optional)*

```
table = table.loc[:, an_rt[an_rt['rt']>0.1].index & an_vol[an_vol['vol']<0.3].index]
table
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Index.__and__ operating as a set operation is deprecated, in the future this will be a logical operation r
  """Entry point for launching an IPython kernel.
```

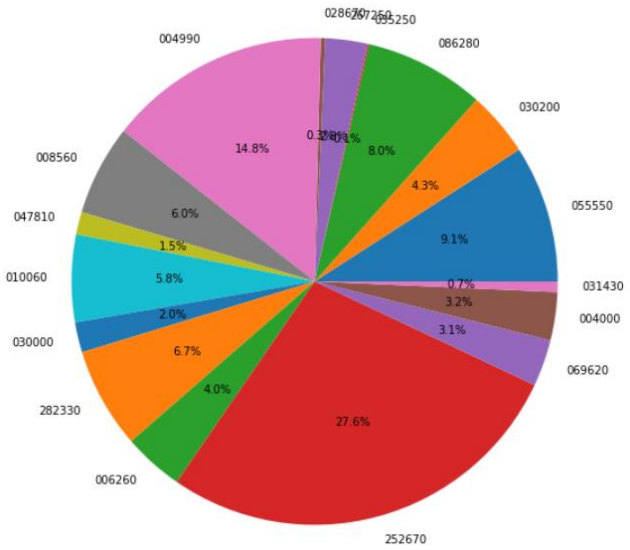
	055550	086790	010950	010130	030200	316140	086280	035250	267250	078930	...	000080	006260	001450	252670	001230	069620	004000	111770	031430	005300
0	37250	42350	85900	510000	30350	12800	172500	24200	53600	39400	...	30250	54500	23450	2150	16100	151500	73000	44150	29100	131000
1	37600	42700	90200	511000	30100	13000	173000	23950	54900	39550	...	30300	55000	24050	2155	17150	147500	74800	44600	29100	132500
2	37500	43000	90200	503000	30800	13050	173000	24000	55500	39550	...	29900	54000	24950	2215	17400	144500	74500	46050	28400	131000
3	37300	43200	92400	509000	30250	12850	184000	24100	54800	40050	...	30450	54000	25300	2260	18100	141000	72600	47750	28000	131000
4	37650	43450	97300	514000	30250	13100	184000	24550	55500	39700	...	30750	54500	25450	2195	18250	143000	75500	47300	28100	136000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
89	42350	46450	109500	552000	36650	15700	206000	27550	59800	46800	...	35850	59400	31250	2720	17650	164000	81000	48250	33650	186500
90	42400	46250	113000	552000	36550	15700	208000	27100	60900	47000	...	36950	60600	30950	2665	17900	164500	83300	48550	34400	189500
91	42050	46200	110000	557000	36250	14900	209500	27200	61400	46000	...	36800	59400	31050	2655	17650	167500	82900	50400	34350	182500
92	40700	45300	107000	561000	36350	14200	201500	26650	60400	45450	...	36100	60500	30850	2735	16800	162000	82000	46250	34200	182500
93	41700	46950	106000	573000	35950	14600	206000	26850	60000	45750	...	35550	60700	30250	2630	17200	163500	84000	47250	34250	182500

94 rows × 31 columns

Markowitz Portfolio Result-trade 1

- Minimum Volatility

	weight	종목명
055550	9.1	신한지주
030200	4.3	KT
086280	8.0	현대글로비스
035250	0.1	강원랜드
267250	2.8	현대중공업지주
028670	0.3	팬오션
004990	14.8	롯데지주
008560	6.0	메리츠증권
047810	1.5	한국항공우주
010060	5.8	OCI
030000	2.0	제일기획
282330	6.7	BGF리테일
006260	4.0	LS
252670	27.6	KODEX 200선물인버스2X
069620	3.1	대웅제약
004000	3.2	롯데정밀화학
031430	0.7	신세계인터내셔날



VS

- Maximum Return

	weight	종목명
047810	100.0	한국항공우주

100% all-in?
It's too risky!
But…
Can we ignore this result?

Actual Trading Result-trade 1

Max-Ret

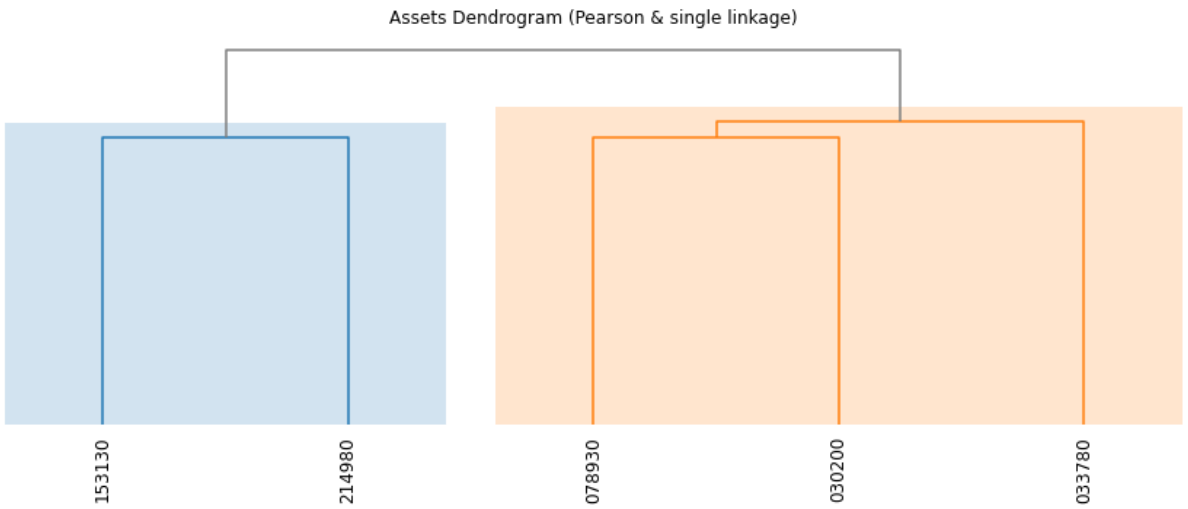
I buy stock based on Min-Volatility portfolio allocation result weight with multiplying 2,000,000 won.

However, I can't ignore Max-Return portfolio allocation result.
It means "한국항공우주" stock growing stock and will bring high return.
So I buy "한국항공우주" 10% more.

Min-Vol

Other portfolio optimization model I tried…

HRP-trade 2

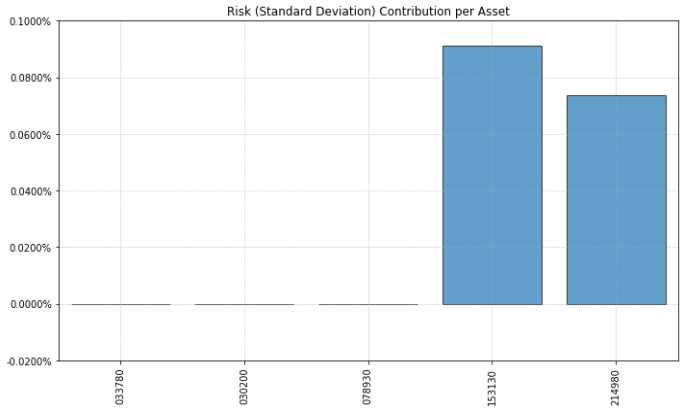


Asset Dendogram

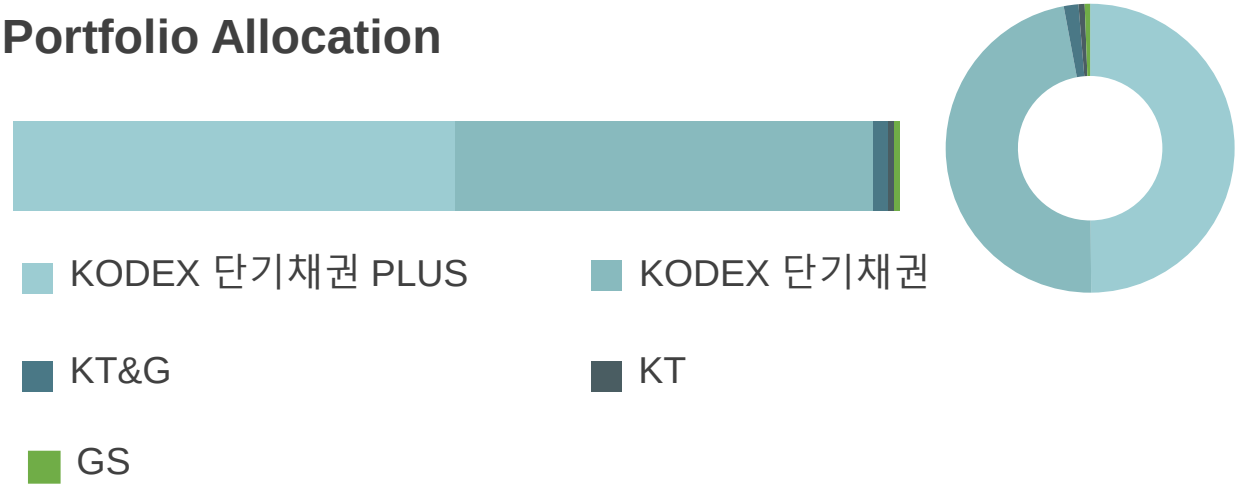
(Pearson & Single Linkage)

Naïve Risk Parity

- Expected return: 0.0
- Standard Deviation: 0.2
- Risk contribution per Asset “bar chart” ➔



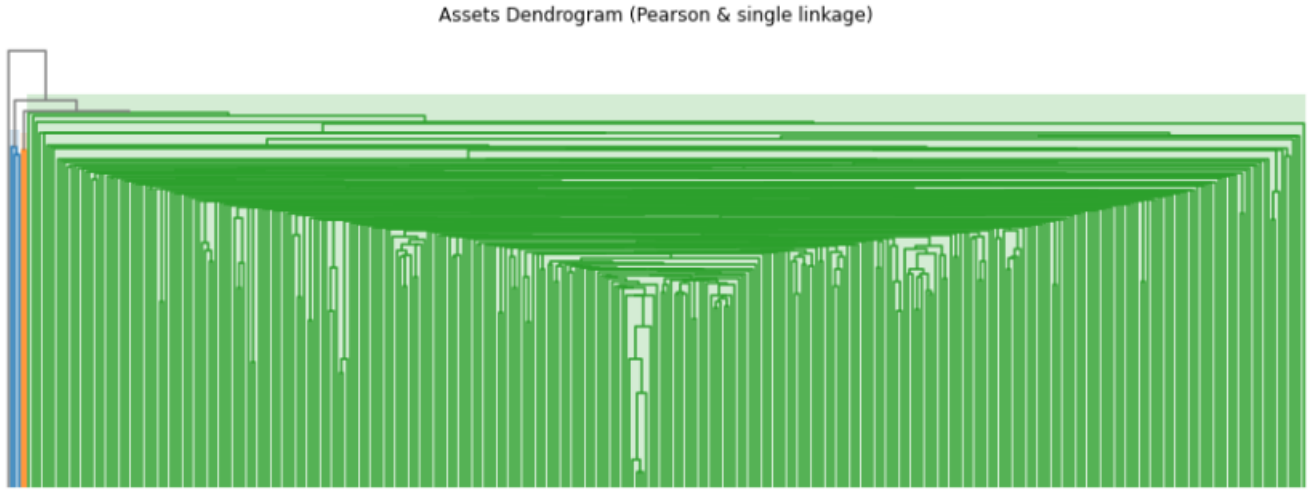
Portfolio Allocation



Before investment - (2) Data Pre-Processing

① Get all assets' dendogram

```
ax = rp.plot_dendrogram(returns=Y,
                        codependence='pearson',
                        linkage='single',
                        k=None,
                        max_k=10,
                        leaf_order=True,
                        ax=None)
```



② Get ticker with weights above 0.05% above *(optional)*

```
w['filter'] = w['weights']>0.0004
w_ = w.loc[w['filter']==True]
w_
```

	weights	filter
033780	1.6457%	True
030200	0.6055%	True
078930	0.6849%	True
153130	49.8890%	True
214980	47.1748%	True

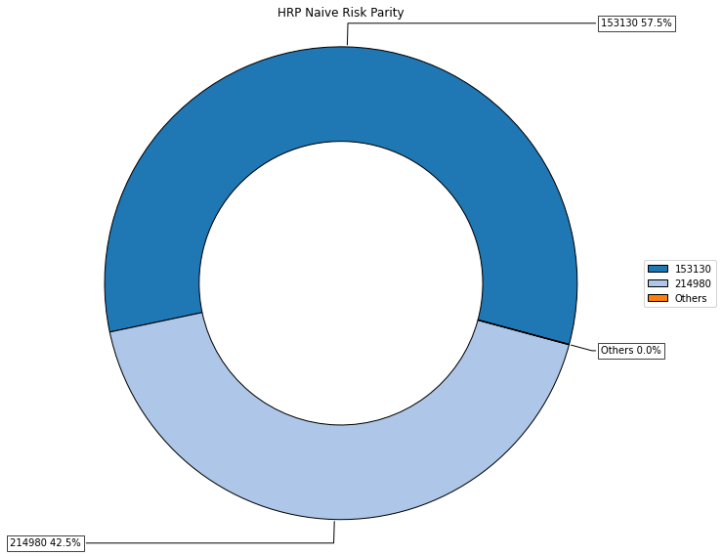
HRP Portfolio Result-trade 2

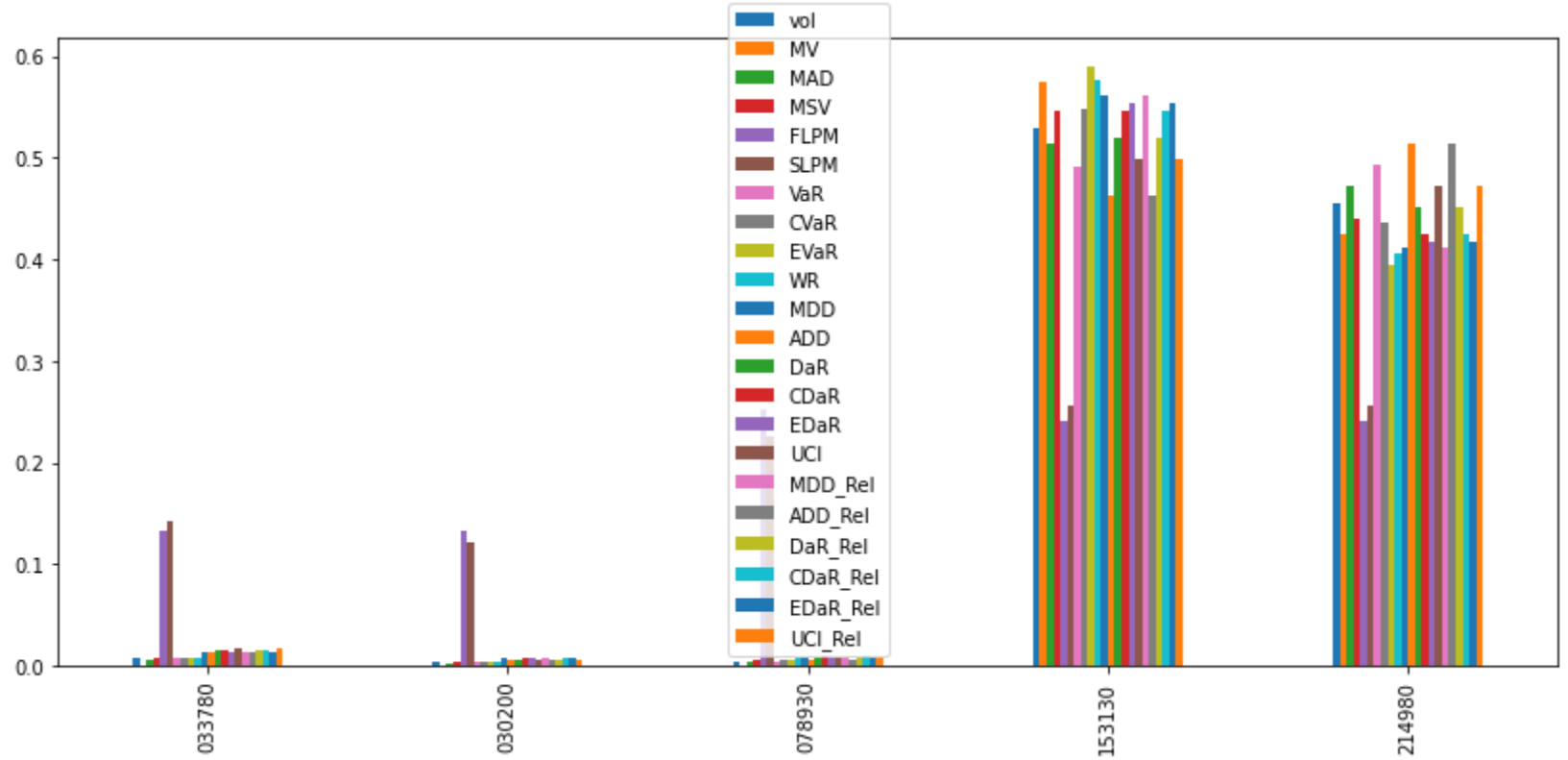
	weights	종목명
033780	1.6457%	KT&G
030200	0.6055%	KT
078930	0.6849%	GS
153130	49.8890%	KODEX 단기채권
214980	47.1748%	KODEX 단기채권PLUS

VS

- Several different method for portfolio optimization with HRP

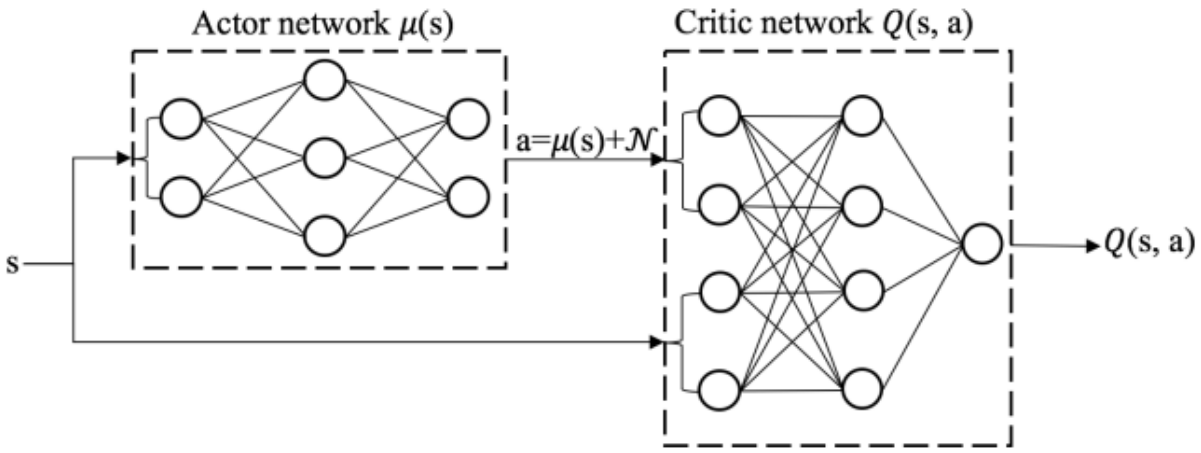
	vol	MV	MAD	MSV	FLPM	SLPM	VaR	CVaR	EVaR	WR	MDD	ADD	DaR	CDaR	EDaR	UCI	MDD_Rel	ADD_Rel	DaR_Rel	CDaR_Rel	EDaR_Rel	UCI_Rel
033780	0.70%	0.02%	0.62%	0.68%	13.25%	14.16%	0.69%	0.66%	0.68%	0.68%	1.23%	1.35%	1.59%	1.41%	1.34%	1.64%	1.24%	1.35%	1.56%	1.41%	1.35%	1.65%
030200	0.27%	0.00%	0.24%	0.29%	13.25%	12.07%	0.31%	0.34%	0.37%	0.40%	0.71%	0.49%	0.61%	0.67%	0.70%	0.61%	0.70%	0.48%	0.60%	0.66%	0.69%	0.61%
078930	0.43%	0.00%	0.40%	0.48%	25.31%	22.57%	0.45%	0.51%	0.60%	0.66%	0.79%	0.48%	0.71%	0.85%	0.81%	0.68%	0.80%	0.48%	0.71%	0.86%	0.83%	0.68%
153130	53.02%	57.48%	51.50%	54.57%	24.09%	25.59%	49.17%	54.85%	58.95%	57.71%	56.10%	46.32%	51.92%	54.66%	55.47%	49.89%	56.10%	46.32%	51.93%	54.66%	55.47%	49.89%
214980	45.58%	42.49%	47.23%	43.97%	24.11%	25.61%	49.39%	43.64%	39.39%	40.54%	41.16%	51.36%	45.17%	42.42%	41.67%	47.17%	41.17%	51.37%	45.19%	42.41%	41.66%	47.17%





Other portfolio optimization model I tried…

RL Portfolio Allocation-trade 2



DDPG algorithm

Paper: Practical Deep Reinforcement Learning Approach for Stock Trading

- cumulative return: -0.12105
- annul volatility: 0.29518
- max drawdown: -0.12963

Portfolio Allocation

NeurIPS 2018 Workshop on Challenges and Opportunities for AI in financial Services

Before investment - (2) Data Pre-Processing

① Download Git

```
## install finrl library
!pip install git+https://github.com/AI4Finance-Foundation/FinRL.git
```

```
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
# matplotlib.use('Agg')
import datetime

%matplotlib inline
from finrl.finrl_meta.preprocessor.yahoodownloader import YahooDownloader
from finrl.finrl_meta.preprocessor.preprocessors import FeatureEngineer, data_split
from finrl.finrl_meta.env_stock_trading.env_stocktrading import StockTradingEnv
from finrl.agents.stablebaselines3.models import DRLAgent
from finrl.finrl_meta.data_processor import DataProcessor

from finrl.plot import backtest_stats, backtest_plot, get_daily_return, get_baseline
from pprint import pprint

import sys
sys.path.append("../FinRL-Library")

import itertools
```

② Get data with “technical indicator” *(optional)*

Macd, bollingerband, rsi, cci, dx, sma, vix, turbulence

	date	tic	open	high	Low	close	volume	day	macd	boll_ub	boll_lb	rsi_30	cci_30	dx_30	close_30_sma	close_60_sma	vix	turbulence
0	2021-01-04	60	14650.0	14650.0	14000.0	14250.0	876899.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	14250.0	14250.0	26.969999	0.0
1	2021-01-04	80	31950.0	32000.0	31500.0	32000.0	462233.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	32000.0	32000.0	26.969999	0.0
2	2021-01-04	100	72447.0	76275.0	72256.0	75797.0	1086898.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	75797.0	75797.0	26.969999	0.0
3	2021-01-04	120	167500.0	169000.0	163500.0	168500.0	118757.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	168500.0	168500.0	26.969999	0.0
4	2021-01-04	150	52500.0	52700.0	51000.0	52000.0	126798.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	52000.0	52000.0	26.969999	0.0
5	2021-01-04	240	14300.0	14450.0	14150.0	14400.0	271945.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	14400.0	14400.0	26.969999	0.0
6	2021-01-04	250	83200.0	83200.0	79800.0	81800.0	883070.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	81800.0	81800.0	26.969999	0.0
7	2021-01-04	270	62400.0	64800.0	61700.0	64000.0	5602314.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	64000.0	64000.0	26.969999	0.0
8	2021-01-04	660	124500.0	128000.0	120500.0	126000.0	7995016.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	126000.0	126000.0	26.969999	0.0
9	2021-01-04	670	532000.0	540000.0	516000.0	540000.0	2547.0	4.0	0.0	14432.842712	13867.157288	0.0	-66.666667	100.0	540000.0	540000.0	26.969999	0.0

③ DDPG algorithm (model)

```
agent = DRLAgent(env = env_train)
model_ddpg = agent.get_model("ddpg")
```

```
{'batch_size': 128, 'buffer_size': 50000, 'learning_rate': 0.001}
Using cuda device
```

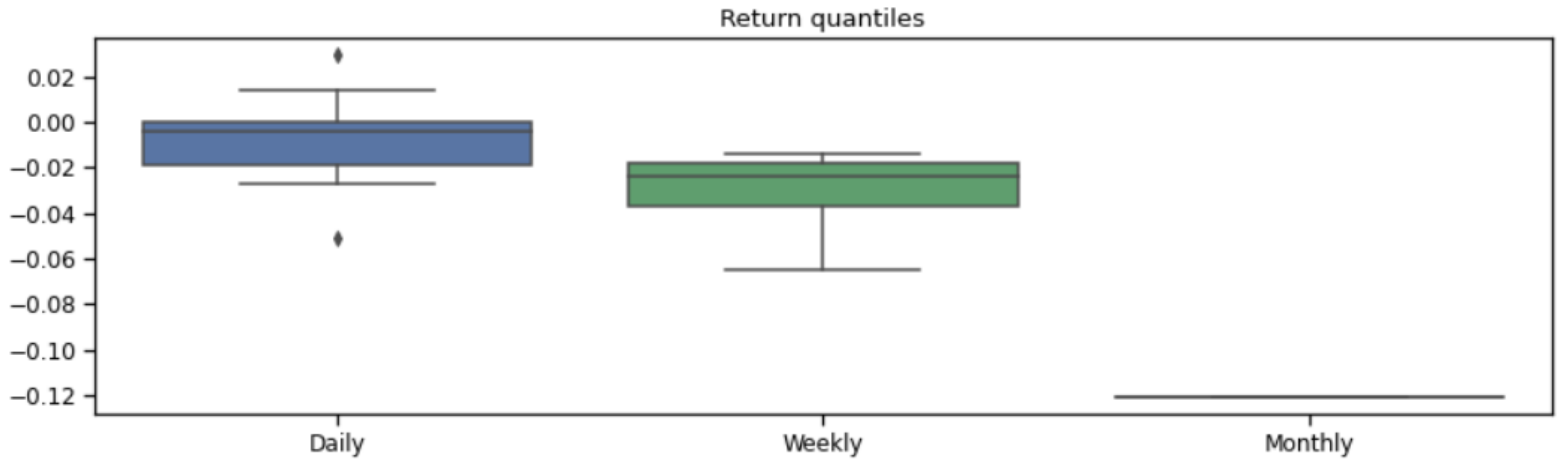
DDPG Portfolio Result - backtesting

- Several method and result of backtesting

```
==========Get Backtest Results==========
Annual return          -0.852312
Cumulative returns     -0.121051
Annual volatility       0.295178
Sharpe ratio           -6.716849
Calmar ratio           -6.575172
Stability               0.876530
Max drawdown           -0.129626
Omega ratio             0.278091
Sortino ratio          -7.009352
Skew                         NaN
Kurtosis                     NaN
Tail ratio              0.537502
Daily value at risk    -0.045057
dtype: float64
```

Worst drawdown periods	Net drawdown in %	Peak date	Valley date
0	12.96	2022-05-02	2022-05-19

Stress Events	mean	min	max
New Normal	-0.79%	-5.14%	2.96%



Actual Trading Result-trade 2

GS 42,850원 · 10주	428,500원
KT&G 84,900원 · 5주	424,500원

Sell which is not making good portfolio

마감내역

KBSTAR KIS단기종합채권(AA-이상)액티브 22.05.23. ~ 22.05.30.	-2,597 -0.51%
KBSTAR 미국장기국채선물레버리지(합성 H) 22.05.23. ~ 22.05.30.	+388 +0.40%
KODEX 국채선물10년 22.05.20. ~ 22.05.30.	-226 -0.07%
TIGER 미국달러단기채권액티브 22.05.23. ~ 22.05.30.	-11,586 -2.14%
KODEX 200선물인버스2X 22.05.23. ~ 22.05.30.	-12,512 -2.24%
KT&G 22.05.24. ~ 22.05.30.	-6,733 -1.59%
롯데정밀화학 22.05.23. ~ 22.05.30.	-843 -1.00%

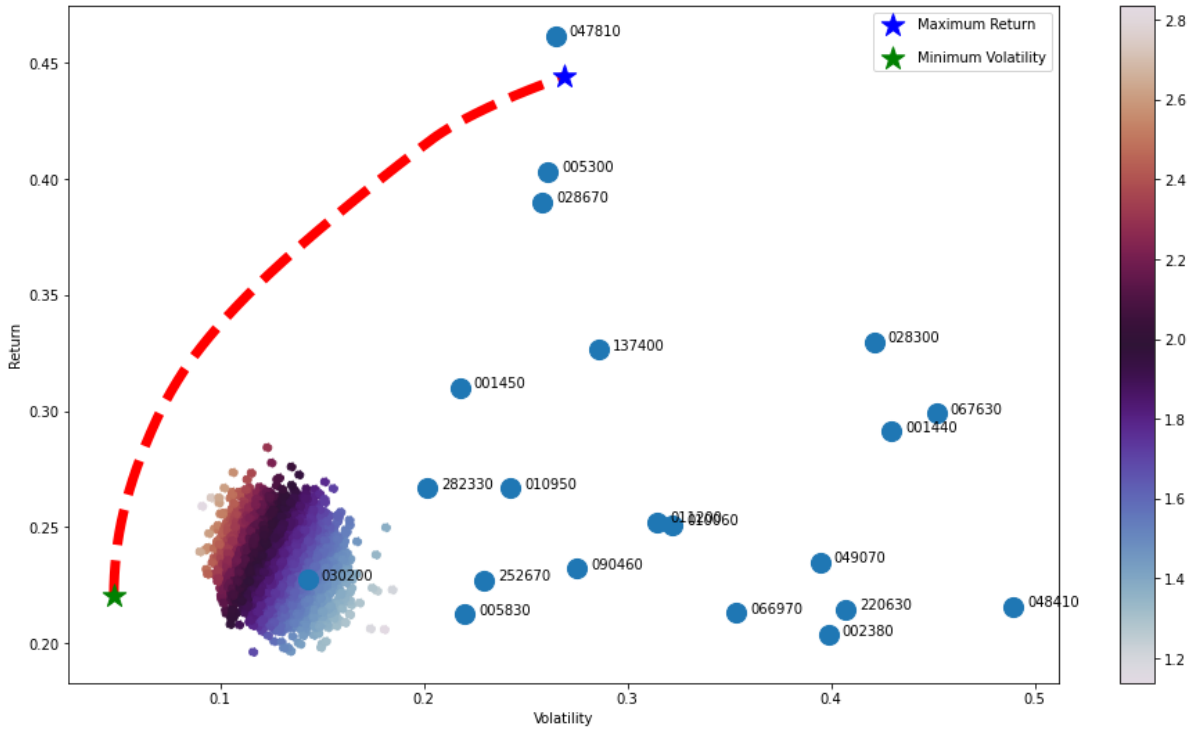
HRP

I buy stock based on HRP portfolio allocation result weight with multiplying 1,000,000 won.

Also with Reinforcement Learning based on NIPS paper, FinRL, deep portfolio managing is quite useful for real-world trading. So I buy “LG전자” based on model’s backtest result.

RL

Markowitz Portfolio-trade 3

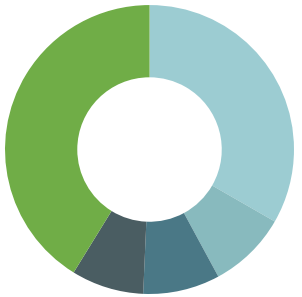


$\mu - \sigma$ diagram

Minimum Volatility Optimization

- Expected return: 0.4442
- Standard Deviation: 0.2691
- Maximum Return Optimization
  get only 1 company

Portfolio Allocation



Kodex 200선물인버스2X

BGF리테일

KT

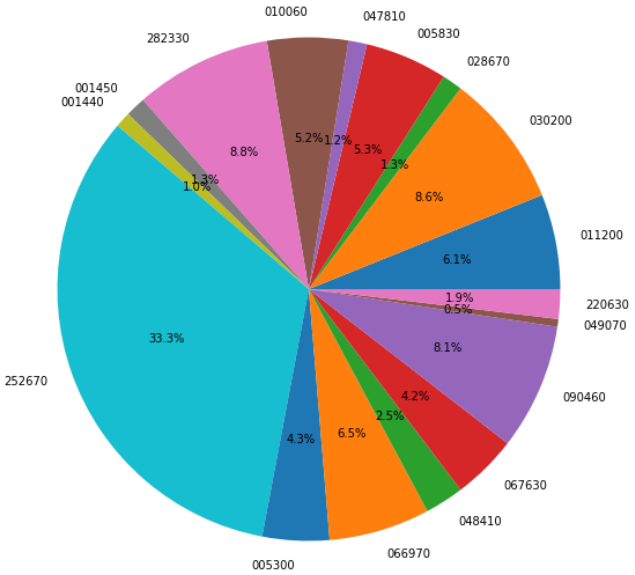
비에이치

기타 (HMM, 팬오션, DB손해보험, 한국항공우주, OCI, 현대해상, 대한전선, 롯데칠성, 현대바이오 에이치엘비, 인탑스, 엘앤에프)

Markowitz Portfolio Result-trade 3

	weight	종목명
011200	6.1	HMM
030200	8.6	KT
028670	1.3	팬오션
005830	5.3	DB손해보험
047810	1.2	한국항공우주
010060	5.2	OCI
282330	8.8	BGF리테일
001450	1.3	현대해상
001440	1.0	대한전선
252670	33.3	KODEX 200선물인버스2X
005300	4.3	롯데칠성
066970	6.5	엘앤에프
048410	2.5	현대바이오
067630	4.2	에이치엘비생명과학
090460	8.1	비에이치
049070	0.5	인탑스
220630	1.9	맘스터치

- Minimum Volatility



Nowadays,
Kospi & Kosdaq
is growing!
Should we buy
KODEX 200 inverse?

Actual Trading Result-trade 3

체결내역

매수 05.31.	인탑스 38,000원 · 3주	114,000원	매수 05.31.	롯데칠성 196,500원 · 5주	982,500원
매수 05.31.	비에이치 28,000원 · 4주	112,000원	매수 05.31.	대한전선 2,230원 · 10주	22,295원
매수 05.31.	HLB 44,600원 · 3주	133,800원	매수 05.31.	현대해상 31,350원 · 10주	313,500원
매수 05.31.	현대바이오 29,400원 · 1주	29,400원	매수 05.31.	BGF리테일 182,500원 · 1주	182,500원
매수 05.31.	엘앤에프 258,500원 · 1주	258,500원	매수 05.31.	OCI 127,438원 · 8주	1,019,500원

Inverse ETF

Markowitz model is based on history data. From half first 2022 year, Korea stock market was not good. However nowadays, it starts to grow. So I did not buy market inverse ETF even if model recommended to buy.

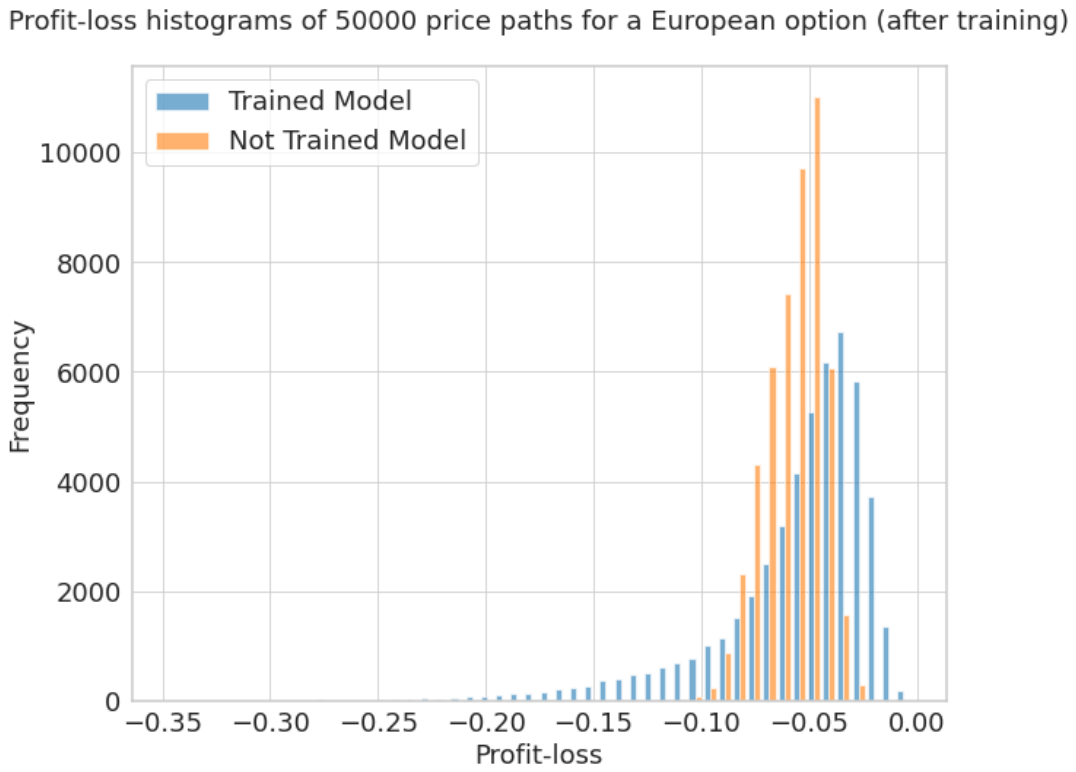
Difference with trade1 is, I picked stock indices with return more than 0.2 and vol lower than 0.5 which means get more return but more risky. However

More risky Trading

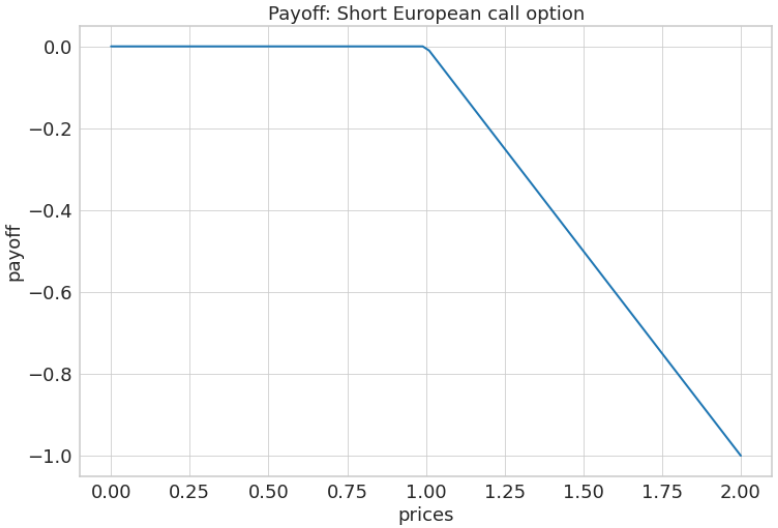
Hedging

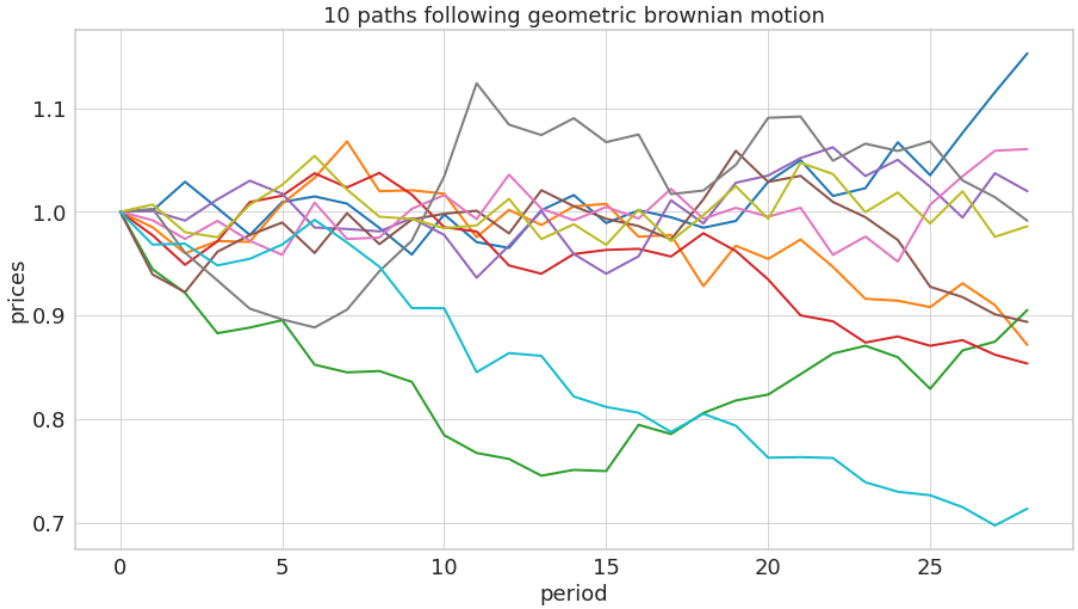
Just trying…

Deep Hedging(call)-no trade

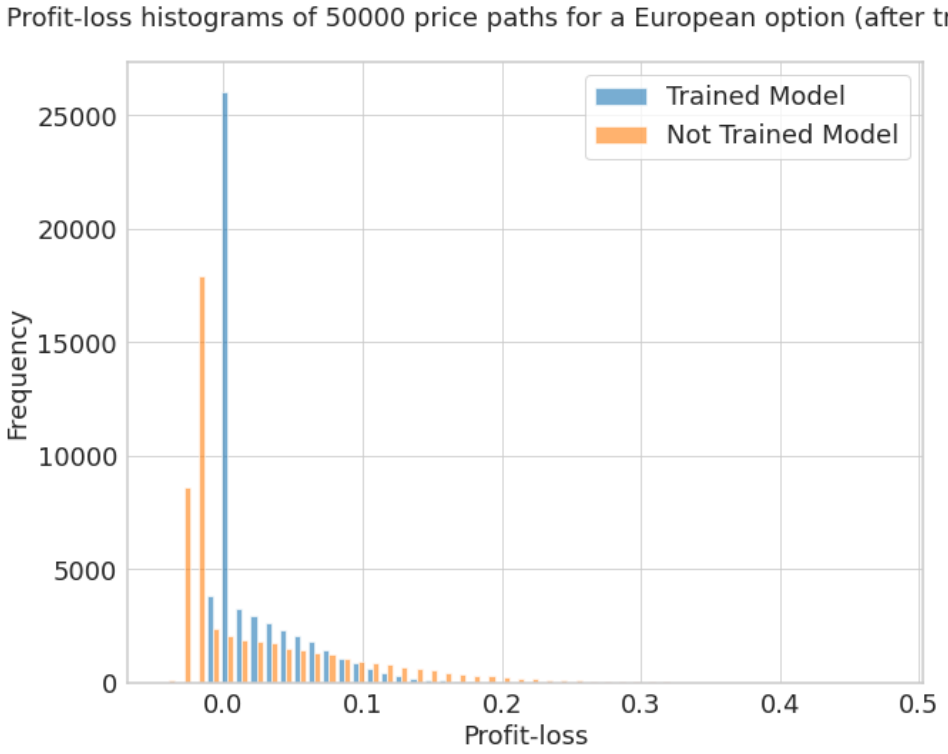


PnL – Frequency histogram

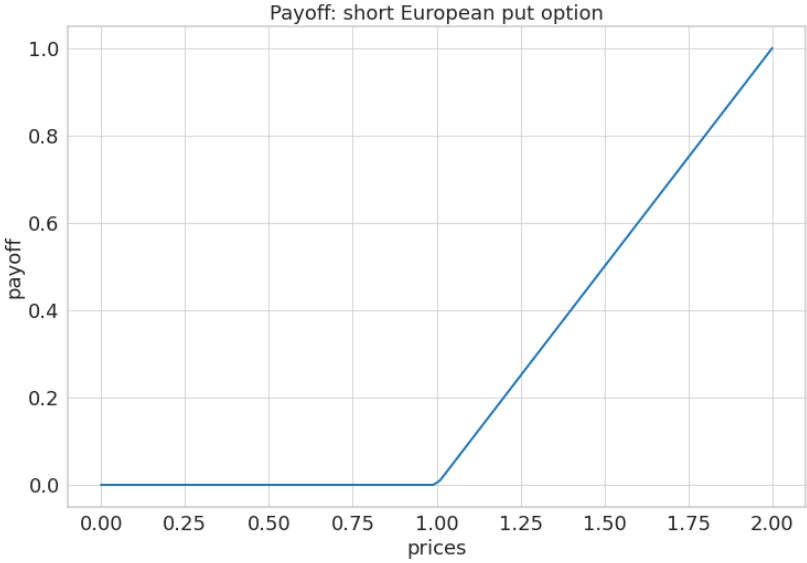


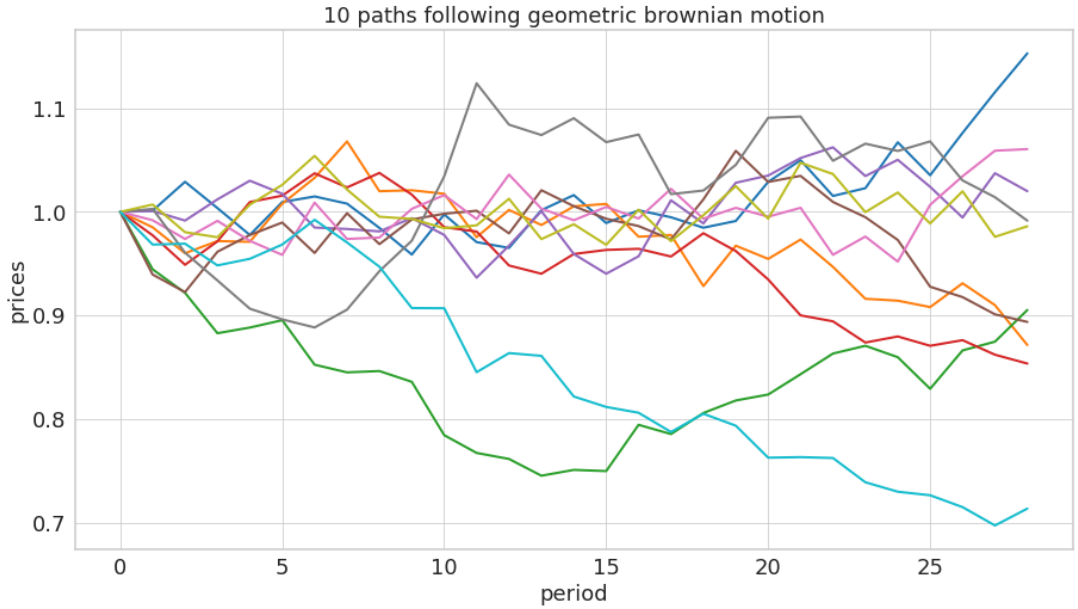


Deep Hedging(put)-no trade



PnL – Frequency histogram

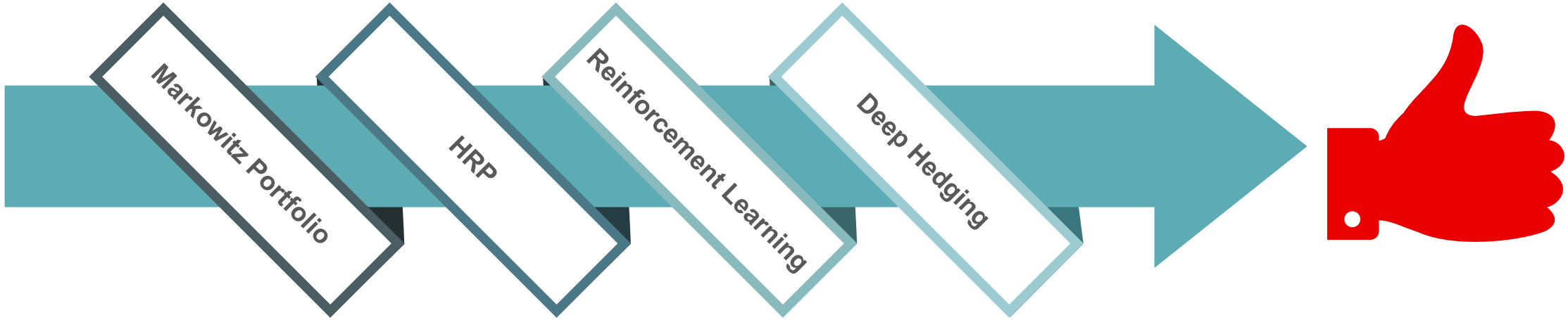




Conclusion

After Trading…

After Trading…



01

What I learned?

I tried several portfolio managing method and learned how to manage my portfolio in bull and bear market. Especially Markowitz Model was the best approach for me to get risk free investment portfolio.

02

Evaluation of my portfolio - Pros

I tried to make profit through analyzing historical data and pick tickers which have high risk but low volatility.

03

Evaluation of my portfolio - Cons

I can’t expect when the market is going to bear or bull. So yesterday’s best stock was going worst tomorrow and this makes my portfolio fluctuating

04

Summary

For four week, I managed 10,000,000 won through investing several stocks in Korea. Portfolio managing from model driven approach to data drive approach. It was really useful experience for trying what I learned in the class.

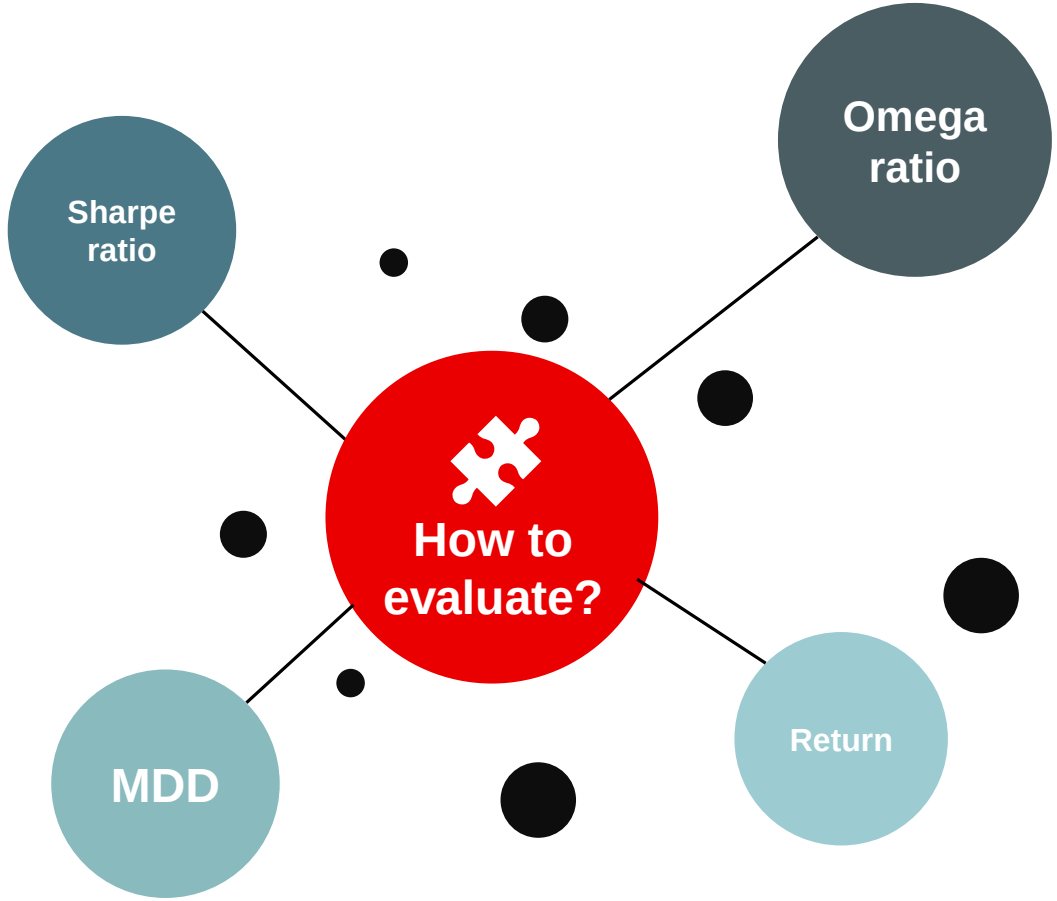
Portfolio Evaluation Method

Other method for evaluating portfolio except Sharpe ratio

Omega Ratio

The Omega ratio is a risk-return performance measure of an investment asset, portfolio, or strategy. It was devised by Con Keating and William F. Shadwick in 2002 and is defined as the probability weighted ratio of gains versus losses for some threshold return target. The ratio is an alternative for the widely used Sharpe ratio and is based on information the Sharpe ratio discards.

$$\Omega(\theta) = \frac{\int_{\theta}^{\infty} [1 - F(r)]\, dr}{\int_{-\infty}^{\theta} F(r)\, dr}$$



My final Portfolio

Final return, 1.14%
Cumulative return, 0.51%

It was great honor to join investment competition!

GitHub https://github.com/monouns/Portfolio-Allocation-tutorial

Thank You

KAIST 20214418 Jihyeon Seong