야, 너두 주식 할 수 있어!

AIStock Company
(ASC)

이샛별, 정지윤, 홍석훈, (오희주)

포트폴리오란?

- 두 개 이상 여러 자산의 조합

- 일반적으로,

  주식, 채권, 부동산등의 금융자산이나 실물 자산의 결합

- 재무(finance)에서는,

  투자 위험을 줄이기 위한 분산투자 조합안으로써 사용

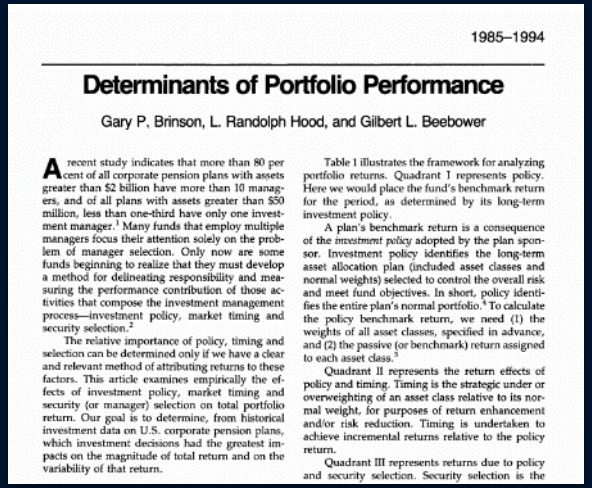
포트폴리오, 얼마나 중요할까?

" 1974년에서 1983년까지의 91개 미국 대형연기금 데이터를 분석한 결과,

자산배분이 투자전략(마켓 타이밍과 종목 선택)보다 중요하며,

총 수익률 변동성의 95.6%를 설명한다. "

연구논문 '포트폴리오 성과의 결정 요인들'



1985–1994

Determinants of Portfolio Performance

Gary P. Brinson, L. Randolph Hood, and Gilbert L. Beebower

A recent study indicates that more than 80 per cent of all corporate pension plans with assets greater than $2 billion have more than 10 managers, and of all plans with assets greater than $50 million, less than one-third have only one investment manager.[1] Many funds that employ multiple managers focus their attention solely on the problem of manager selection. Only now are some funds beginning to realize that they must develop a method for delineating responsibility and measuring the performance contribution of those activities that compose the investment management process—investment policy, market timing and security selection.[2]

The relative importance of policy, timing and selection can be determined only if we have a clear and relevant method of attributing returns to these factors. This article examines empirically the effects of investment policy, market timing and security (or manager) selection on total portfolio return. Our goal is to determine, from historical investment data on U.S. corporate pension plans, which investment decisions had the greatest impacts on the magnitude of total return and on the variability of that return.

Table 1 illustrates the framework for analyzing portfolio returns. Quadrant I represents policy. Here we would place the fund's benchmark return for the period, as determined by its long-term investment policy.

A plan's benchmark return is a consequence of the *investment policy* adopted by the plan sponsor. Investment policy identifies the long-term asset allocation plan (included asset classes and normal weights) selected to control the overall risk and meet fund objectives. In short, policy identifies the entire plan's normal portfolio.[4] To calculate the policy benchmark return, we need (1) the weights of all asset classes, specified in advance, and (2) the passive (or benchmark) return assigned to each asset class.[5]

Quadrant II represents the return effects of policy and timing. Timing is the strategic under or overweighting of an asset class relative to its normal weight, for purposes of return enhancement and/or risk reduction. Timing is undertaken to achieve incremental returns relative to the policy return.

Quadrant III represents returns due to policy and security selection. Security selection is the

자산관리, 대체 어떻게 해야하지?

- **주식에 대한 관심도 증가**

  → 성인 1명당 1계좌

- **자산관리**

  = 종목 관리 = 포트폴리오 구성

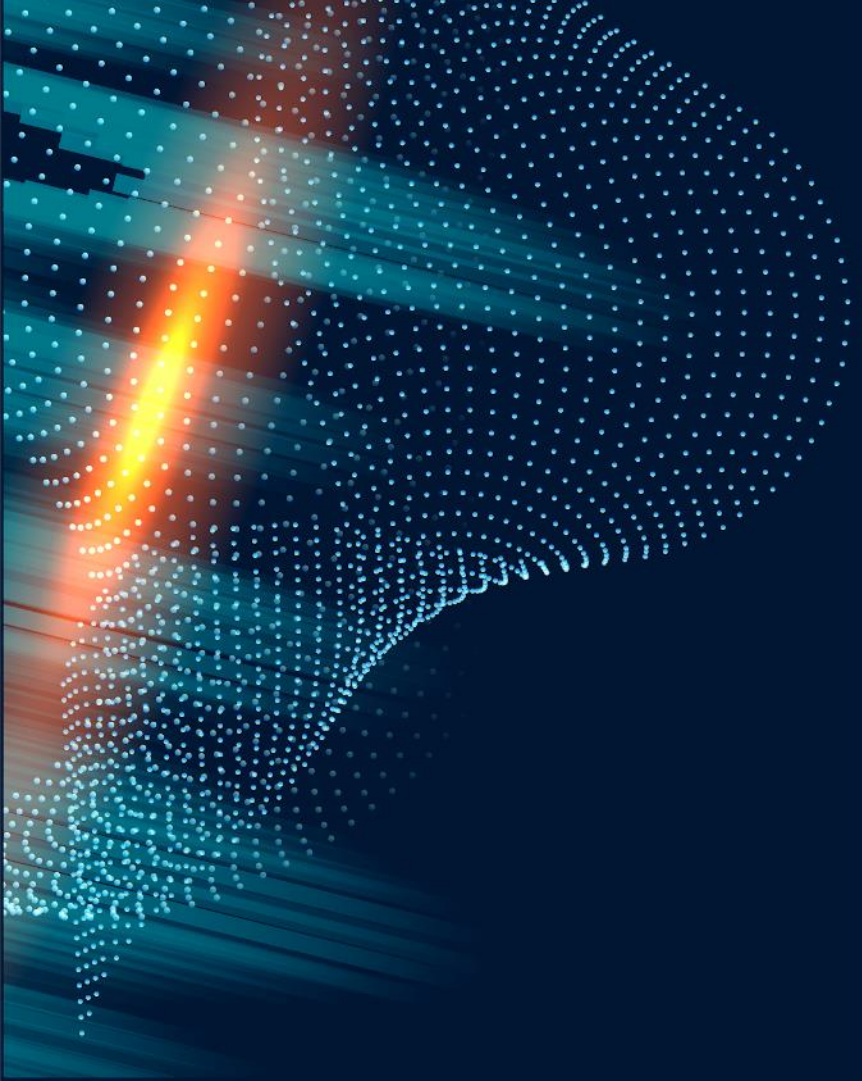
  자산운용사 : 코덱스, 타이거, 각종 은행사 및 증권사

  AI 로보어드바이저 : 에임, 파운트, 핀트 등

- **투자 필수 시대,**

  아직도 놀고만 계실겁니까? 저희랑 함께 가시죠. 렛츠 게릿 😎

목차



프로젝트 배경

1. 주제
2. 목적
3. 개요
4. 기대효과

주제

- 맞춤형 주식 포트폴리오 추천

목적

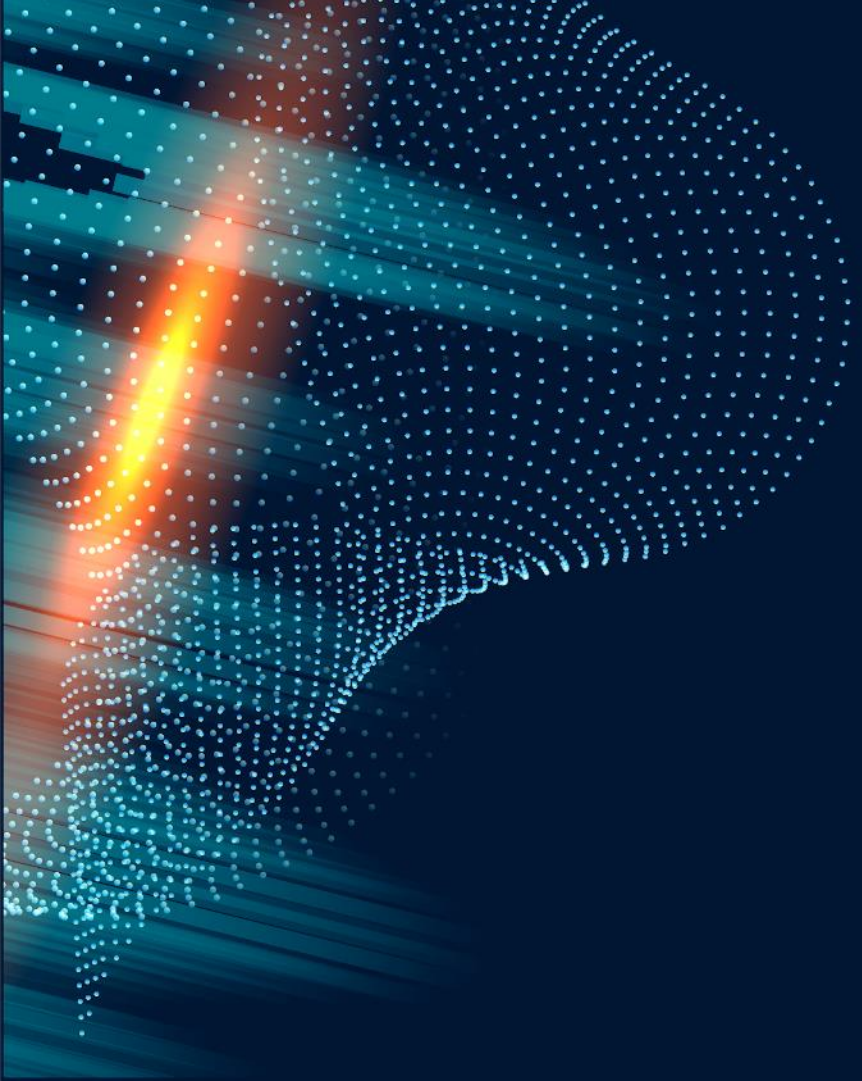
- 투자 금액, 기간, 종목, 리스크에 따른 맞춤형 포트폴리오 추천
- 소액 투자로도, 객관적인 추천을 받을 수 있는 서비스 제공

개요

- 투자 성향에 맞추어 원하는 종목으로 구성된 포트폴리오 자동 최적화 (Max 100)
- 관심 종목이 없을시, 5가지 전략별 서비스를 제시후 자동 최적화

기대 효과

- 언제어디서든 데이터 기반으로 객관적인 추천 받기 가능
- 고액으로 시작해야 한다는 부담감 Zero
- 개인의 자산 관리 보조 역할 (의사결정 도움)



팀원소개

오희주 (PM)

- 전략별 종목 코드 작성 (jupyter)
- 주가 데이터 수집 / 정제
- LSTM 코드 작성

이샛별

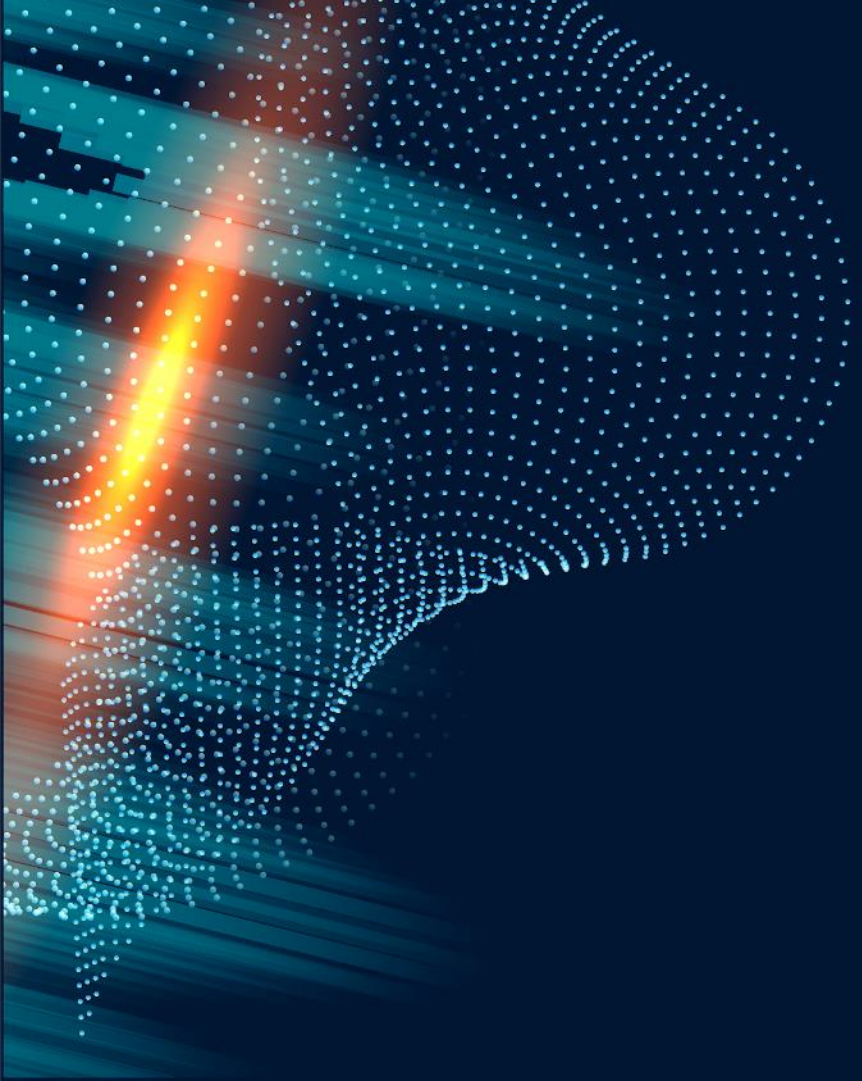
- 분석기획 및 웹 디자인 설계
- 주가 데이터 수집 / 정제
- 포트폴리오 추천 알고리즘 수립
- 알고리즘 테스트 및 검증
- PPT 제작

정지윤

- 수집/정제된 데이터 검증
- 모든 포트폴리오 코드 수집 후 .py 파일 생성
- Class화 및 함수화로 한 줄 코드 완성
- LSTM 코드 개선
- PPT 제작

홍석훈

- 전반적인 웹 서버 담당
- Docker 환경 구축
- 함수화된 코드를 Django 와 연결
- PPT 제작



수행 절차

1. 사전 기획
2. 개발
3. 수정 / 보완

구분	기간	활동	비고
사전 기획	•6/7(월) ~ 7/7(수)	•프로젝트 기획 및 자료 조사 •교재 스터디 •기획안 작성	• 자료조사 및 기능 추리기
	•7/26(월)	•프로젝트 중간 발표	•프로젝트 주제 선정 배경 및 개요 발표
개발	•6/21(월) ~ 8/6(금)	•주식 데이터 수집 및 전처리 •포트폴리오 최적화 알고리즘 구현	•Research •Develop
	•6/7(월) ~ 8/27(금)	•포트폴리오 코드와 웹의 연결	•Docker, Django 이용한 웹 서버 구현
수정/보완	•8/7(월) ~ 8/31(금)	•포트폴리오 알고리즘 개선	•최적화, 오류 수정
총 개발기간	•6/7(월) ~ 8/31(금)(12주)		

타임라인

~ Day 10

프로젝트 기획

~ Day 40

교재 스터디,

자료조사

~ Day 70

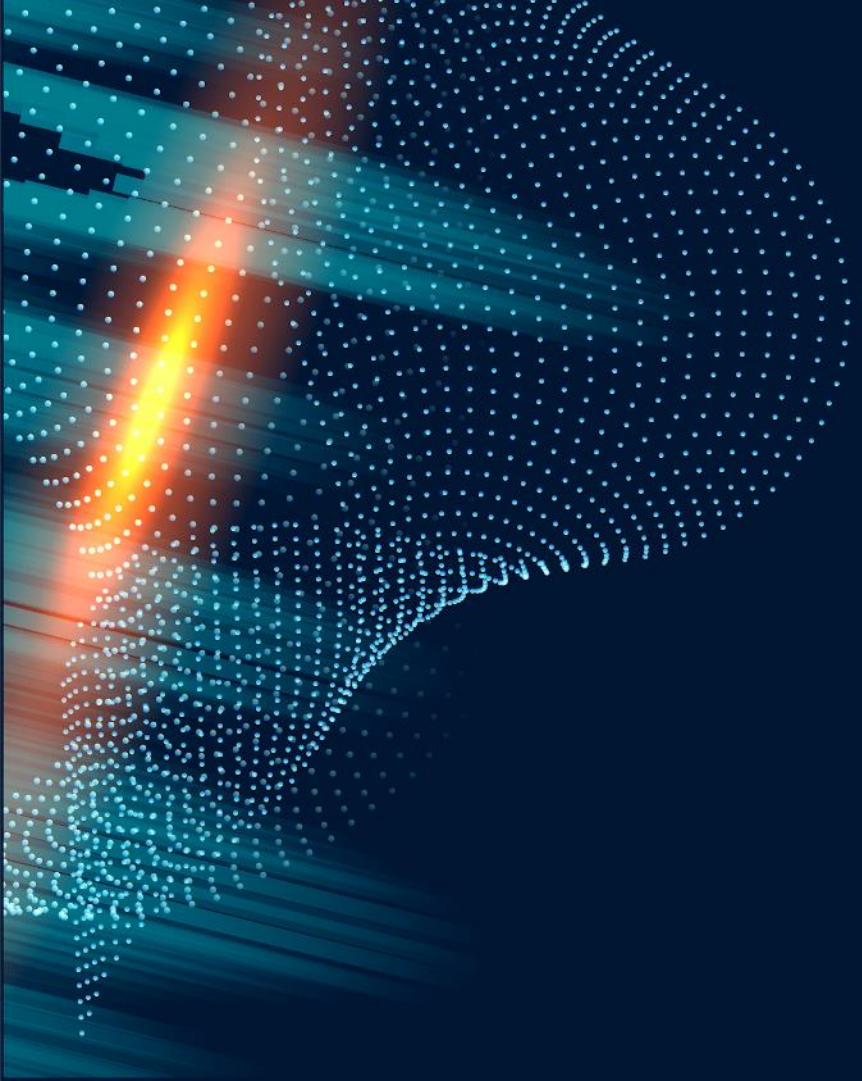
알고리즘 개발,

웹 서버 개발

~ Day 82

코드 개선 및

연결



수행 결과

1. 포트폴리오 최적화
2. 주가 예측
3. 웹 서비스

01

포트폴리오 최적화

1. 알고리즘 Flow
2. 데이터 수집 및 정제
3. 데이터 분석 및 활용
    a. 투자종목 선택 전략
    b. 포트폴리오 최적화
1. 데이터 분석 결과

02

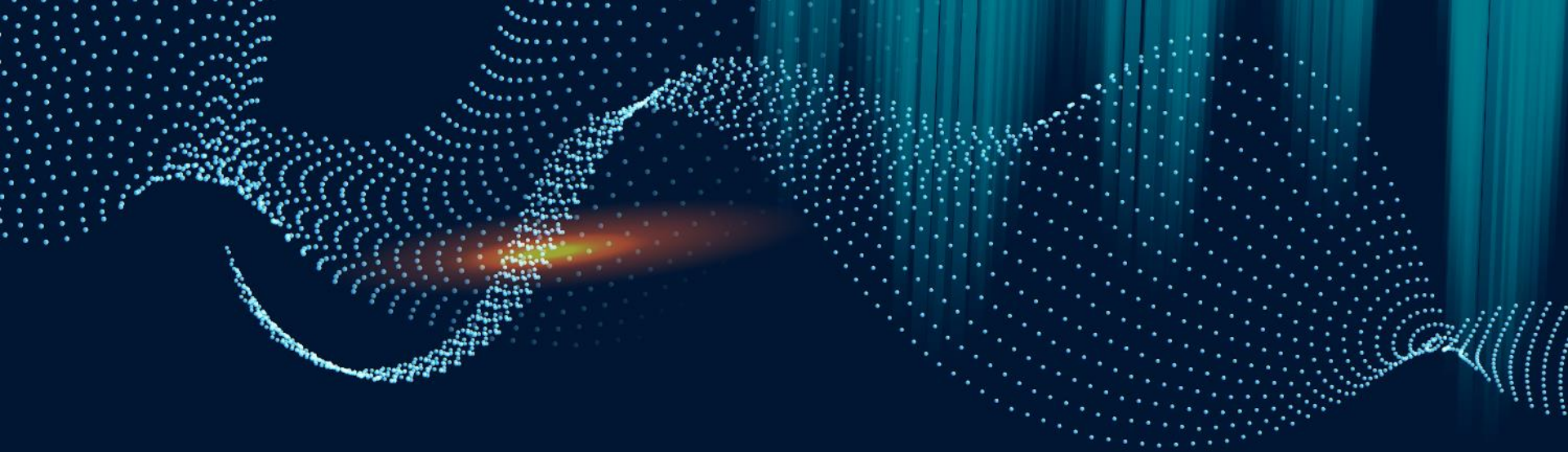
주가예측

1. 내일의 종가를 예측
2. LSTM 의 한계점

03

웹 서비스

사용자 편의를 위해 제공



01 포트폴리오 최적화

1. 알고리즘 Flow
2. 데이터 수집 및 정제
3. 데이터 분석 및 활용
    a. 투자종목 선택 전략
    b. 포트폴리오 최적화
4. 데이터 분석 결과

1. 알고리즘 Flow

투자종목 선택

✓ 직접 선택 (최대 100개)
✓ **5가지 전략** 중 선택
- 급등주
- 상승일 빈도
- 1개월 모멘텀
- 3개월 모멘텀
- 듀얼 모멘텀

주가 데이터

✓ 분석 데이터
: 과거 n년간 주가데이터
(n = 1 ~ 5)

✓ 데이터 전처리
- KRX 관리종목 삭제
- 대상 기간 중 종가에 결측값이 있는 종목 삭제

기대수익률

Risk model
(공분산 행렬)

포트폴리오 최적화

Efficient Frontier
(효율적 투자선)

✓ **목적함수**
- **Max. 샤프비율**
- **Max. 수익률**
(Risk 범위 이내)

✓ **제약변수**
- **Risk**

투자비용에 따라 매수수량 할당

최적의 포트폴리오

2. 데이터 수집 및 정제

✓ 데이터 수집

FinanceDataReader 라이브러리를 활용하여, KOSPI와 KOSDAQ 상장종목의 5년치 수정종가 데이터를 수집

✓ 데이터 정제

1. **KRX 관리종목 삭제**
   : 현재 거래 정지 중일 가능성이 높으며, 상장폐지 위험이 있는 종목이기때문에 사전에 제외 함.

1. **대상 기간 중 종가에 결측값이 있는 종목 삭제**
   : 동일한 기간동안의 충분한 양의 과거 데이터를 기반으로 신뢰도 높은 분석을 하기 위함.

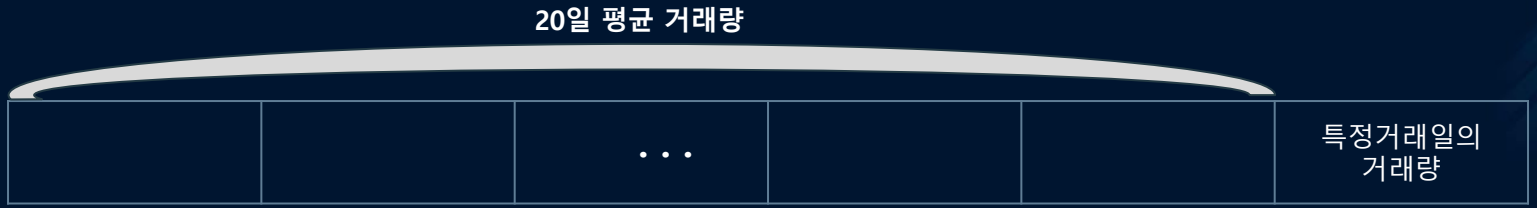
1. **포트폴리오 최적화 결과 종목별 투자비중(weights)이 0.01% 미만일 경우, 0으로 변환**

3. 데이터 분석 및 활용

> 투자종목 선택 전략

01. 급등주

특정 거래일의 거래량이 이전 20일의 평균 거래량보다 1,000% 이상 급증하는 종목을 선택



02. 상승일 빈도

최근 1년의 데이터를 기준으로, 전일 기준 상승 빈도가 높은 상위 30개 종목을 선택

* 투자종목 선택 전략의 대상 종목은 KOSPI와 KOSDAQ 상장종목으로 한정

3. 데이터 분석 및 활용

> 투자종목 선택 전략

03. 모멘텀 전략

✓ **모멘텀이란?**

- 물체가 한 방향으로 지속적으로 변동하려는 경향 (관성의 법칙)
- 금융시장에서의 모멘텀

: 괜찮았던 투자는 계속해서 괜찮고, 형편없는 투자는 계속해서 형편없게 되는 현상

a. **1개월 모멘텀** : 최근 1개월 수익률이 가장 높은 상위 30개 종목을 선택

b. **3개월 모멘텀** : 최근 3개월 수익률이 가장 높은 상위 30개 종목을 선택

c. **듀얼 모멘텀** : 상대 모멘텀 x 절대 모멘텀

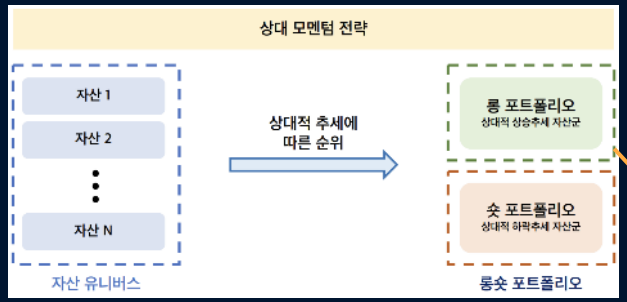
* 투자종목 선택 전략의 대상 종목은 KOSPI와 KOSDAQ 상장종목으로 한정

3. 데이터 분석 및 활용

> 투자종목 선택 전략

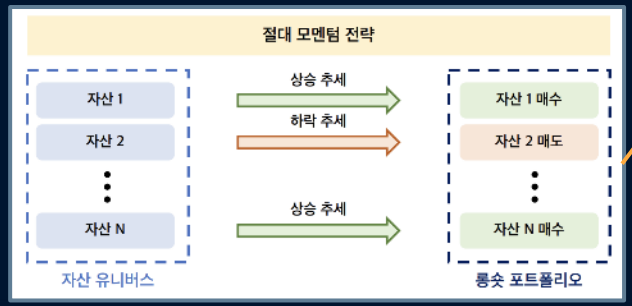
✓ 상대모멘텀

- 전체 자산 유니버스 중에서 **상대적**인 상승추세와 상대적인 하락추세를 판단하여 롱숏 포트폴리오를 구성



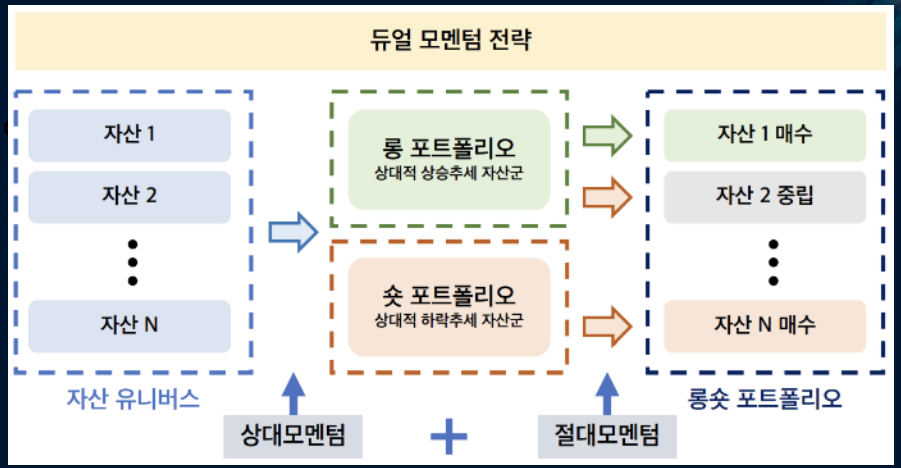
✓ 절대모멘텀

- 개별적으로 각 자산들의 과거 움직임으로 미래를 예측하는 것



✓ 듀얼모멘텀

- 절대 모멘텀 X 상대 모멘텀
- 장점: MDD(최대낙폭지수)를 줄이고,
  장기적으로 더 높은 수익률을 달성할 수 있다.



* 투자종목 선택 전략의 대상 종목은 KOSPI와 KOSDAQ 상장종목으로 한정

3. 데이터 분석 및 활용

> 포트폴리오 최적화

Step 1. 기대수익률 & Risk의 상관관계 구하기

- **기대수익률**

  : 주식 시장에서는 매년 수익이 발생하면서 복리로 불어나기 때문에,

  CAGR(Compound Annual Growth Rate; 연평균 성장률)로 각 종목별 연평균 수익률을 계산

$$\text{CAGR} = \left(\left(\frac{\text{최종가격}}{\text{최초가격}}\right)^{\frac{1}{\text{년수}}}\right) - 1$$

- **수익률의 공분산 = Risk의 상관관계**

  : Risk는 수익률의 표준편차로 계산하며, 주가의 변동성을 의미함

  Risk의 공분산

  각 종목의 연간 수익률간의 상관관계

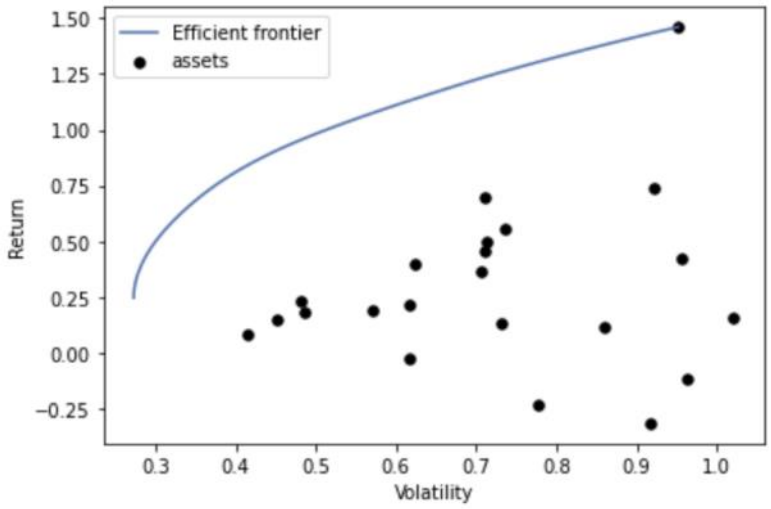
3. 데이터 분석 및 활용

> 포트폴리오 최적화

Step 2. 효율적 투자선(Efficient Frontier) 구하기

- 투자기회집합 전체에서 지배원리를 만족시키는 포트폴리오의 집합
- 지배원리 : 동일한 기대수익률 -> 낮은 위험
  동일한 위험 -> 높은 기대수익률
- 합리적인 투자자는 효율적 투자선 위의 한 포트폴리오를 선택
- 파란색 실선으로 표시된 부분이 효율적 투자선
- 실선 아래의 점들은 개별 종목의 수익률과 변동성을 나타내며, 효율적투자선 위에 위치한 포트폴리오보다 열세인 포트폴리오.

```
# 효율적 투자선와 각 종목의 수익률 & 변동성
import matplotlib.pyplot as plt

ef = EfficientFrontier(mu, S)

fig, ax = plt.subplots()
plotting.plot_efficient_frontier(ef, ax=ax, show_assets=True)
plt.legend(loc='best')
plt.show()
```



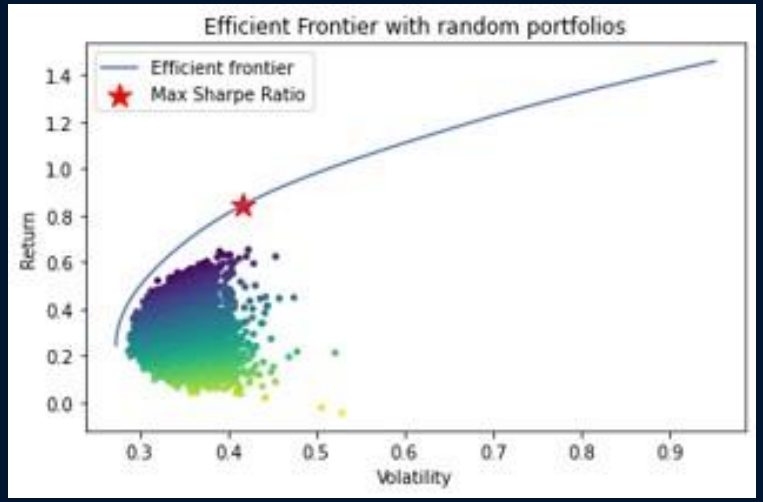
3. 데이터 분석 및 활용

> 포트폴리오 최적화

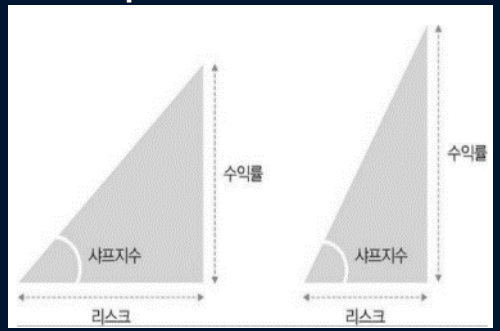
Step 3. 포트폴리오 최적화

Maximize Sharpe Ratio

★ 샤프지수를 최대로 하는 포트폴리오



✓ **Sharpe Ratio (샤프지수)**



- 한 단위의 위험을 부담하는 대신 얻을 수 있는 수익률
- 실제로 투자 포트폴리오에서 사용되는 개념으로,
  보유한 투자 주식 종목에서 최상의 구성비율을 찾는 데 이용
- 샤프비율 = (수익률 - 무위험수익률) / 수익률의 표준편차(Risk)
- 무위험수익률은 투자기간에 따른 국채수익률로 설정함

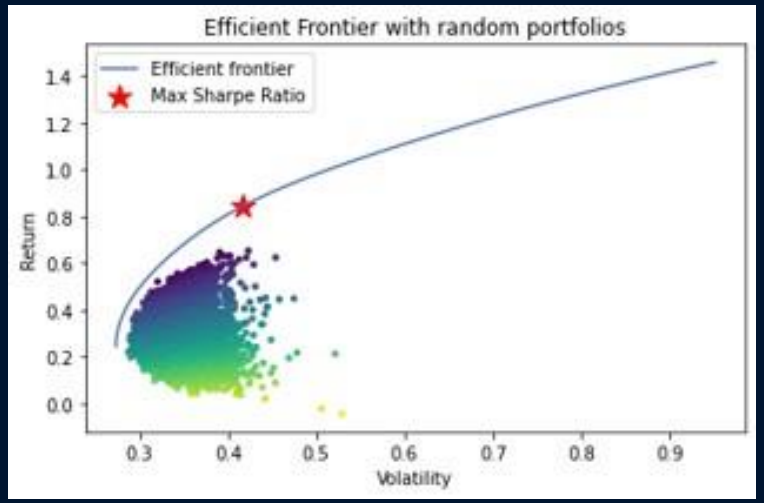
3. 데이터 분석 및 활용

> 포트폴리오 최적화

Step 3. 포트폴리오 최적화

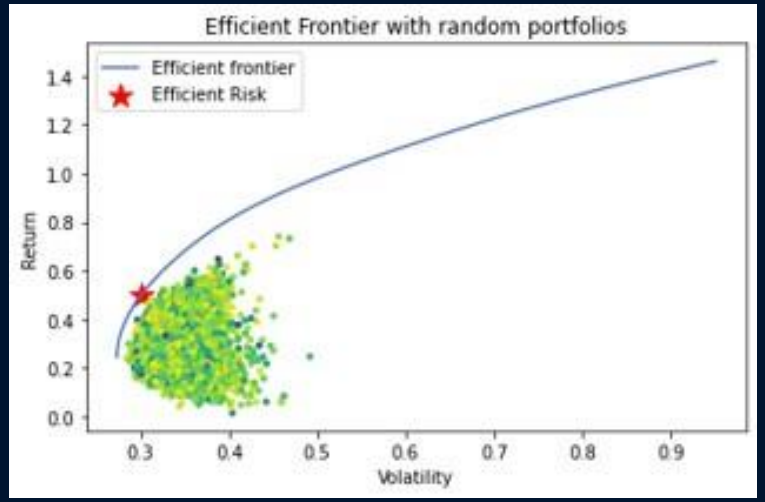
Maximize Sharpe Ratio

★ 샤프지수를 최대로 하는 포트폴리오



Maximize Return

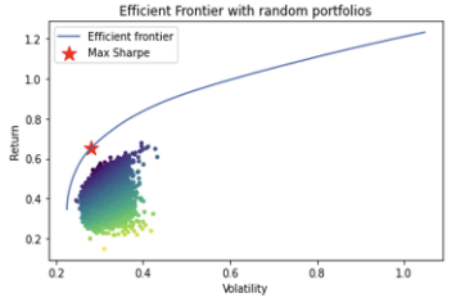
★ "주어진 Risk 보다 낮은 범위 내에서 "
최대의 수익률을 가지는 포트폴리오



4. 데이터 분석 결과

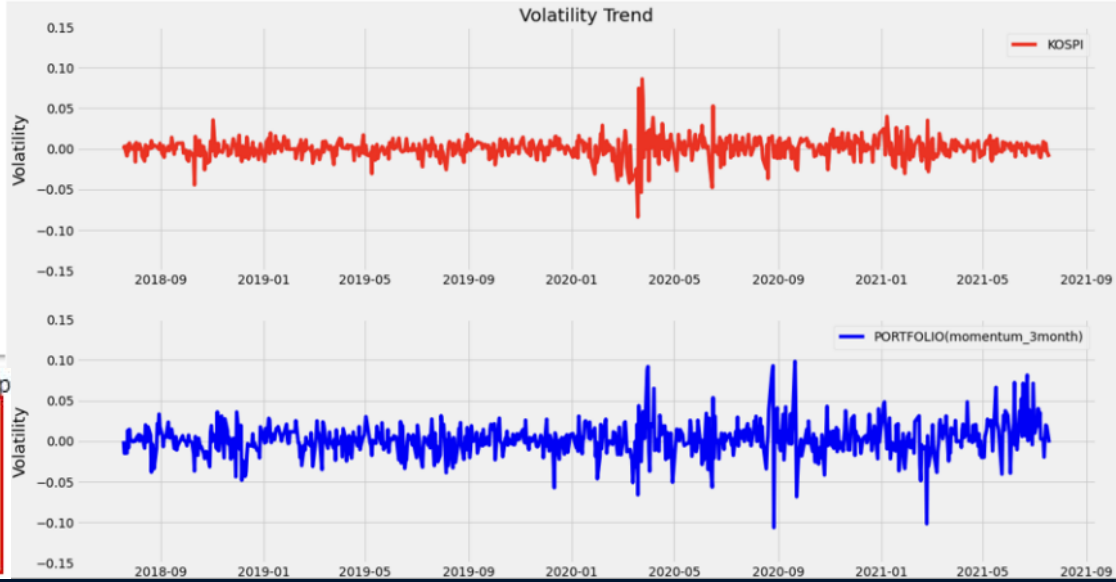
- ✓ **종목선택 전략 :** 모멘텀 3개월
- ✓ **최적화 함수 :** Max Sharpe Ratio
- ✓ **투자금액 :** 1,500 만원

- ✓ **12개 종목으로 포트폴리오 구성**
- ✓ **기대수익률 :** 65.2 %
- ✓ **변동률 :** 28 %
- ✓ **샤프비율 :** 2.25



	종목명	종목코드	수량(주)	투자금액(원)	투자비중
0	한전산업	130660	246.0	3923700.0	0.261593
1	삼성공조	006660	104.0	2797600.0	0.186516
2	KG ETS	151860	94.0	1950500.0	0.130040
3	진원생명과학	011000	30.0	1548000.0	0.103205
4	한일단조	024740	242.0	1231780.0	0.082123
5	씨이랩	189330	9.0	772200.0	0.051483
6	범양건영	002410	94.0	769860.0	0.051327
7	카스	016920	108.0	599400.0	0.039962
8	광진윈텍	090150	66.0	492360.0	0.032826
9	네이처셀	007390	13.0	409500.0	0.027301
10	씨미시스코	136510	7.0	266700.0	0.017781
11	이루온	065440	35.0	237650.0	0.015844
합계	NaN	NaN	1048.0	14999250.0	1.000000

```
-----  Momentum 3month portfolio p
Funds: 15000000 KRW
Funds Remaining:  750.0  KRW
Expected annual return: 65.2%
Annual volatility: 28.0%
Sharpe Ratio: 2.25
Allocation has RMSE: 0.001
```

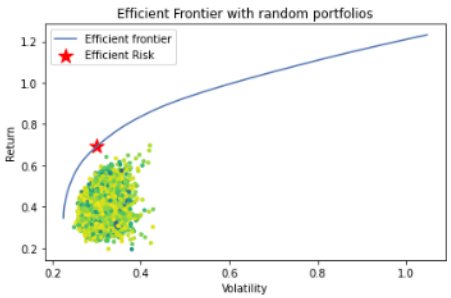


4. 데이터 분석 결과

- ✓ **종목선택 전략 :** 모멘텀 3개월
- ✓ **최적화함수 :** Max Return
- ✓ **투자금액 :** 1,500 만원
- ✓ **Risk :** 30%

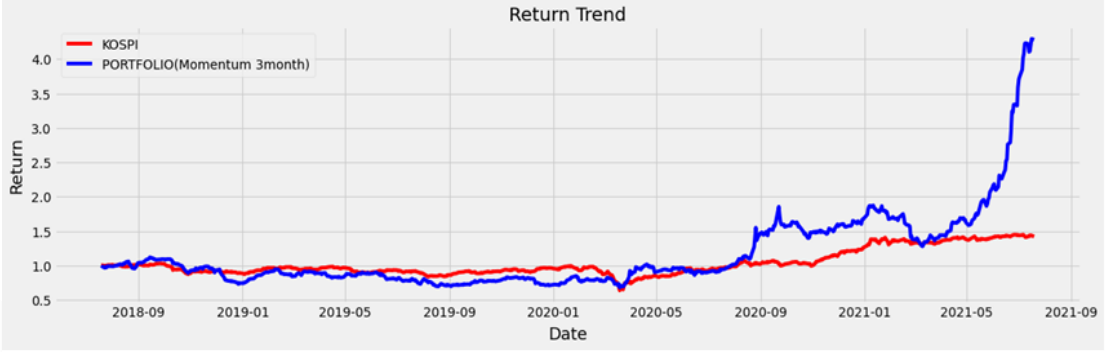
⬇

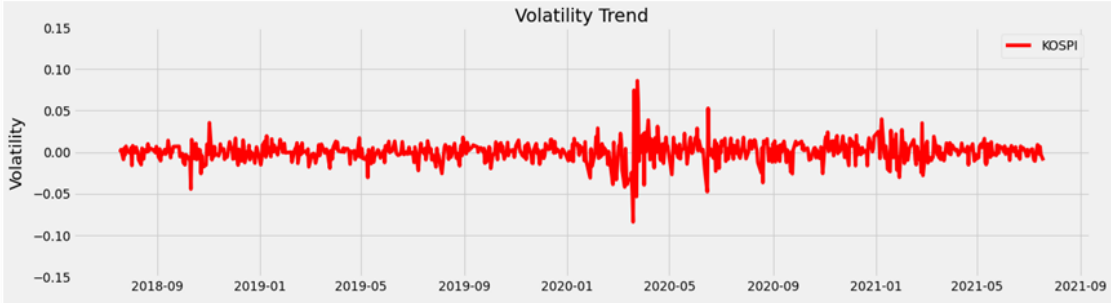
- ✓ **12개 종목으로 포트폴리오 구성**
- ✓ **기대수익률 :** 69.2%
- ✓ **변동률 :** 30 %
- ✓ **샤프비율 :** 2.24

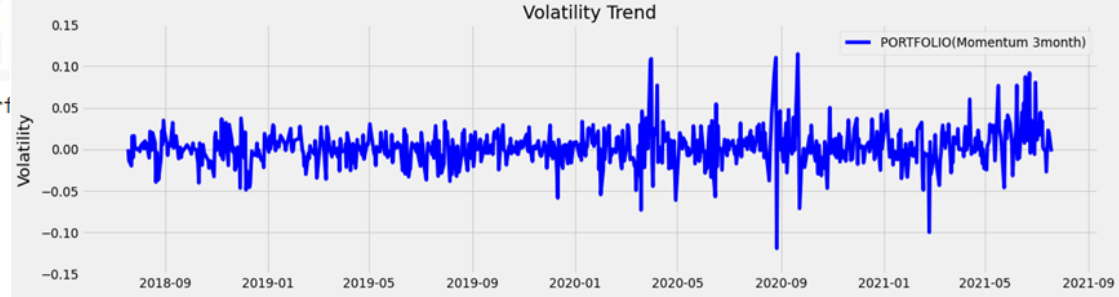


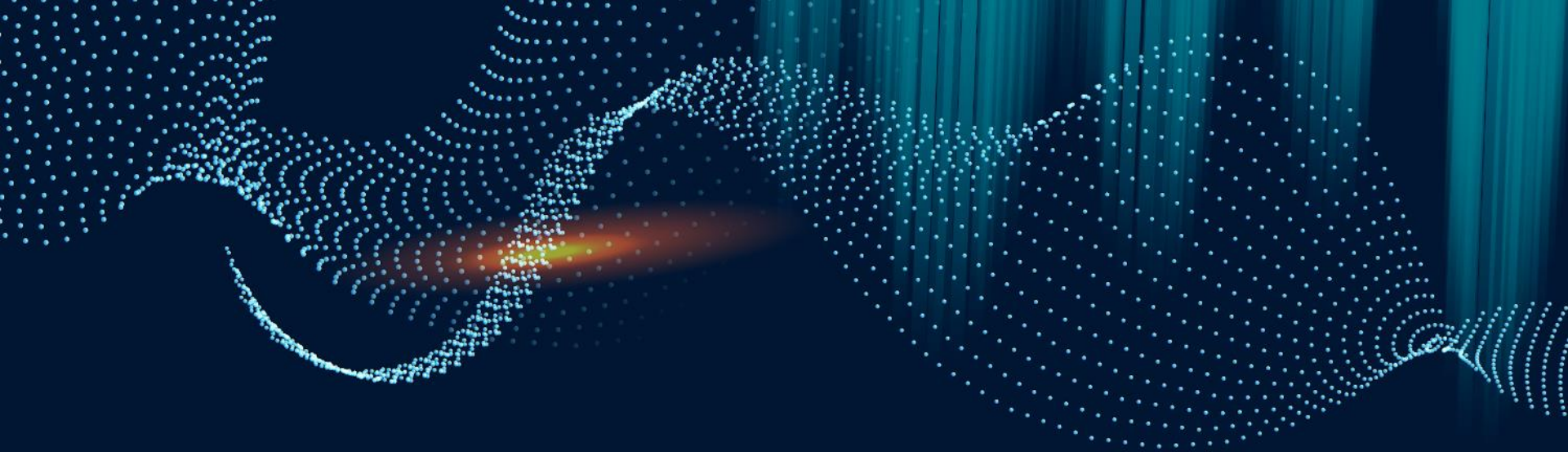
	종목명	종목코드	수량(주)	투자금액(원)	투자비중
0	한전산업	130660	268.0	4274600.0	0.285006
1	KG ETS	151860	113.0	2344750.0	0.156334
2	진원생명과학	011000	36.0	1857600.0	0.123854
3	삼성공조	006660	66.0	1775400.0	0.118373
4	한일단조	024740	238.0	1211420.0	0.080770
5	범양건영	002410	111.0	909090.0	0.060613
6	씨이랩	189330	9.0	772200.0	0.051486
7	광진윈텍	090150	62.0	462520.0	0.030838
8	카스	016920	83.0	460650.0	0.030713
9	네이처셀	007390	13.0	409500.0	0.027303
10	이루온	065440	43.0	291970.0	0.019467
11	씨미시스코	136510	6.0	228600.0	0.015242
합계	NaN	NaN	1048.0	14998300.0	1.000000

```
-----  Momentum 3month portfolio perf
Risk limit:  0.3
Funds: 15000000 KRW
Funds Remaining:  1700.0  KRW
Expected annual return: 69.2%
Annual volatility: 30.0%
Sharpe Ratio: 2.24
Allocation has RMSE: 0.000
```







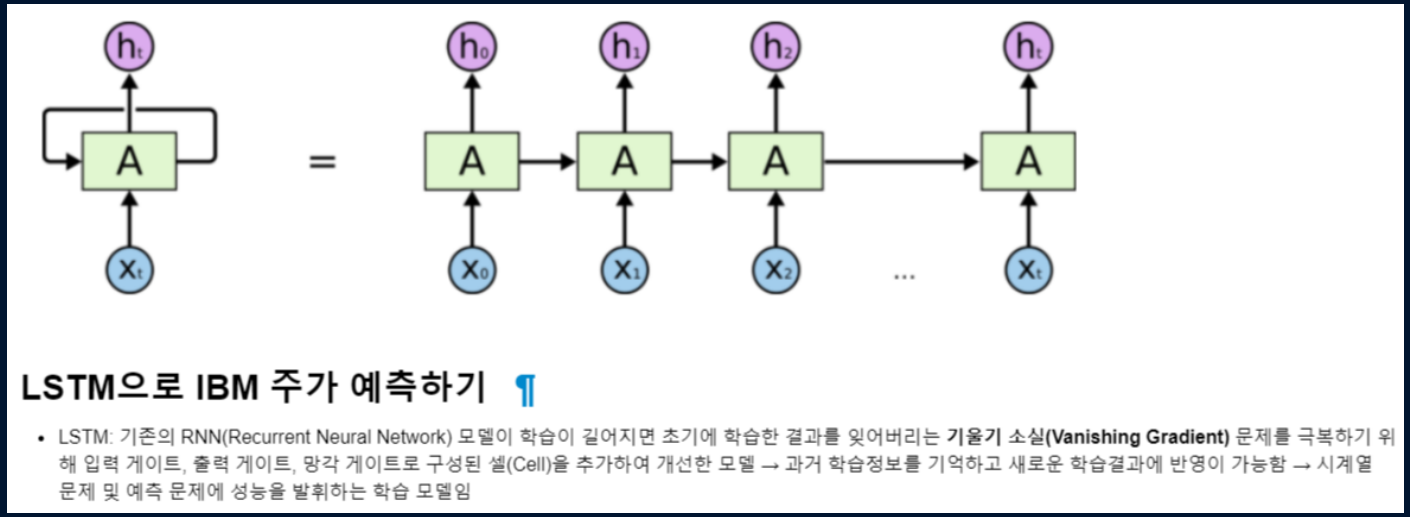


02 주가 예측 (feat. 아름다운 시도)

1. 내일의 종가 예측
2. LSTM의 한계점

1. 내일의 종가 예측 코드

- LSTM 이란?
  - 기존 RNN 모델의 학습 결과를 잊어버리는 기울기 소실 문제를 개선
  - 과거 학습 정보를 기억하고, 새로운 학습 결과에 반영이 가능 → 시계열 문제 혹은 예측에 쓰임



LSTM으로 IBM 주가 예측하기 ¶

- LSTM: 기존의 RNN(Recurrent Neural Network) 모델이 학습이 길어지면 초기에 학습한 결과를 잊어버리는 **기울기 소실(Vanishing Gradient)** 문제를 극복하기 위해 입력 게이트, 출력 게이트, 망각 게이트로 구성된 셀(Cell)을 추가하여 개선한 모델 → 과거 학습정보를 기억하고 새로운 학습결과에 반영이 가능함 → 시계열 문제 및 예측 문제에 성능을 발휘하는 학습 모델임

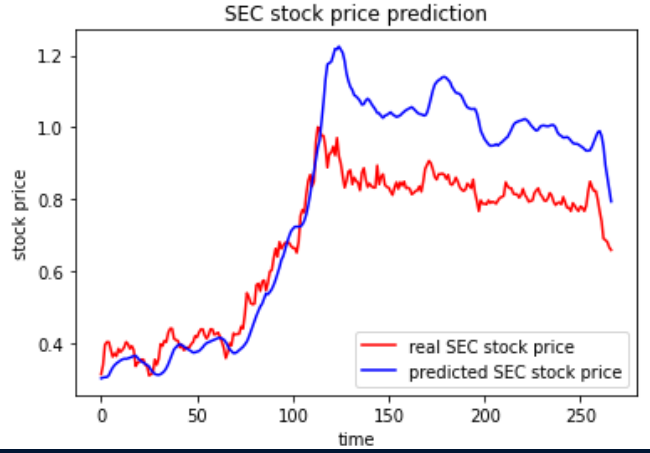
```python
def stock_prediction(ticker, start_date, end_date = datetime.datetime.now().strftime('%Y-%m-%d')):
    raw_df = fdr.DataReader(ticker, start_date, end_date)
    window_size = 10
    data_size = 5
    dfx = raw_df[['Open','High','Low','Volume', 'Close']]
    dfx = MinMaxScaler(dfx)
    dfy = dfx[['Close']]

    # feature 데이터와 label 데이터를 분리하기
    x = dfx.values.tolist()
    y = dfy.values.tolist()

    data_x = []
    data_y = []
    for i in range(len(y) - window_size):
        _x = x[i : i + window_size] # 다음 날 종가(i+windows_size)는 포함되지 않음
        _y = y[i + window_size]     # 다음 날 종가
        data_x.append(_x)
        data_y.append(_y)
    print(_x, "->", _y)

    train_size = int(len(data_y) * 0.7) # 훈련용 데이터 70%
    train_x = np.array(data_x[0 : train_size])
    train_y = np.array(data_y[0 : train_size])

    test_size = len(data_y) - train_size # 테스트용 데이터 30%
    test_x = np.array(data_x[train_size : len(data_x)])
    test_y = np.array(data_y[train_size : len(data_y)])
```

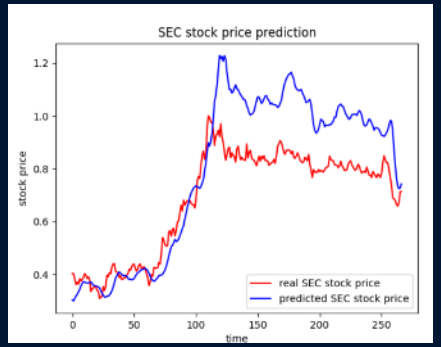
```python
# 모델 생성
model = Sequential() # 시퀀셜 모델 객체 생성
model.add(LSTM(units=10, activation='relu', return_sequences=True, input_shape=(window_size, data_size)))
# (10, 5) 입력 형태를 가지는 LSTM층을 추가함. 전체 유닛 갯수는 10개이고, 활성화 함수는 relu를 사용
model.add(Dropout(0.1)) # 드롭아웃을 10%로 지정해, 훈련 데이터의 과적합을 방지함
model.add(LSTM(units=10, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(units=1)) # 유닛이 하나인 출력층을 추가함
model.summary()

model.compile(optimizer='adam', loss='mean_squared_error')
# 손실함수는 adam을 사용하고, 손실함수는 평균 제곱 오차(MSE)를 사용함.
model.fit(train_x, train_y, epochs=60, batch_size=30)
# epochs는 전체 데이터셋에 대한 학습 횟수, batch_size는 한 번에 제공되는 훈련 데이터 갯수
pred_y = model.predict(test_x) # 테스트 데이터셋 (test_x)를 이용하여 예측치 데이터셋(pred_y)를 생성함

# Visualising the results
plt.figure()
plt.plot(test_y, color='red', label='real SEC stock price')
plt.plot(pred_y, color='blue', label='predicted SEC stock price')
plt.title('SEC stock price prediction')
plt.xlabel('time')
plt.ylabel('stock price')
plt.legend()
plt.show()

return float(raw_df.Close[-1] * pred_y[-1] / dfy.Close[-1]) # 내일 종가
```

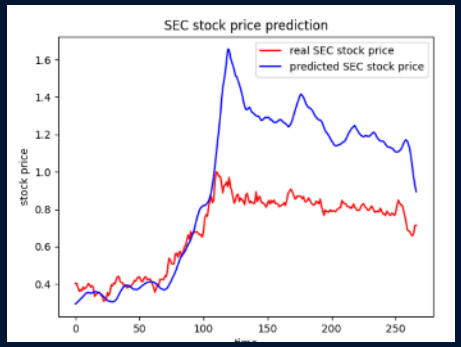
```
samsung = stock_prediction('005930', '2018-01-01') # 학습
```

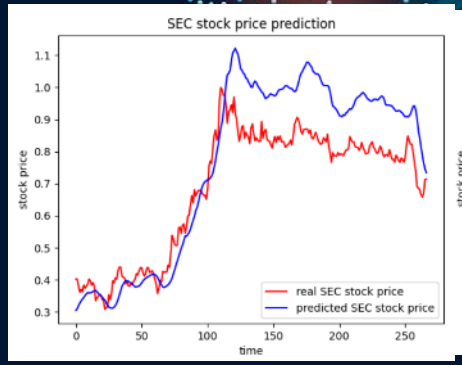
```
[[0.9224806201540172, 0.8605371900817557, 0.916201117317412, 0.204697725
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_8 (LSTM)                (None, 10, 10)            640
_________________________________________________________________
dropout_8 (Dropout)          (None, 10, 10)            0
_________________________________________________________________
lstm_9 (LSTM)                (None, 10)                840
_________________________________________________________________
dropout_9 (Dropout)          (None, 10)                0
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 11
=================================================================
Total params: 1,491
Trainable params: 1,491
Non-trainable params: 0
_________________________________________________________________
Epoch 1/60
21/21 [==============================] - 2s 9ms/step - loss: 0.0326
Epoch 2/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0109
Epoch 3/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0084
Epoch 4/60
21/21 [==============================] - 0s 9ms/step - loss: 0.0079
Epoch 5/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0076
```

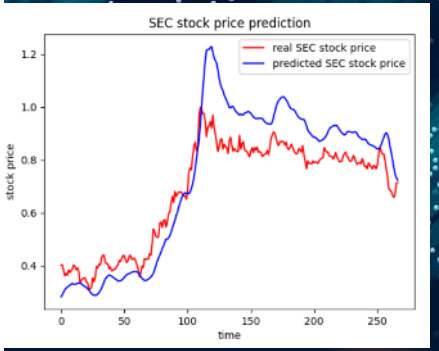
```
Epoch 56/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0014
Epoch 57/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0018
Epoch 58/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0015
Epoch 59/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0014
Epoch 60/60
21/21 [==============================] - 0s 10ms/step - loss: 0.0014
```











Adam - epoch = 60, 64

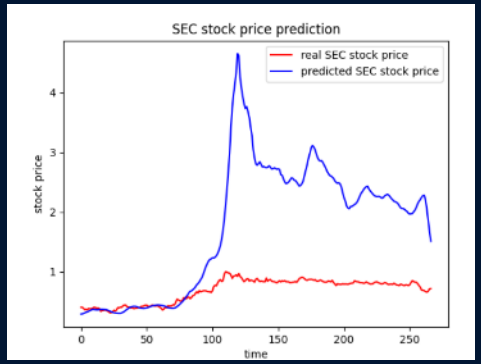
80, 128

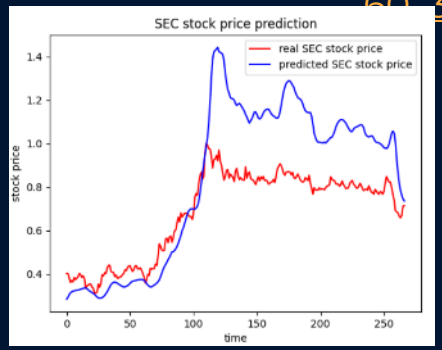
60, 128

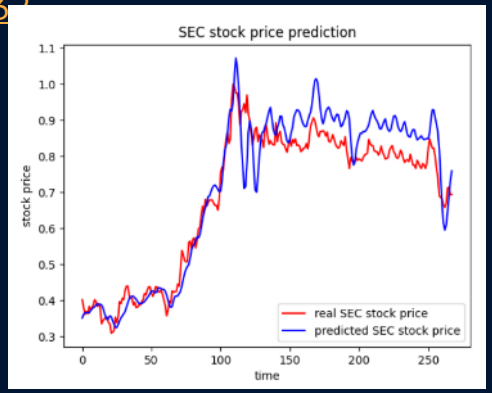
80, 32

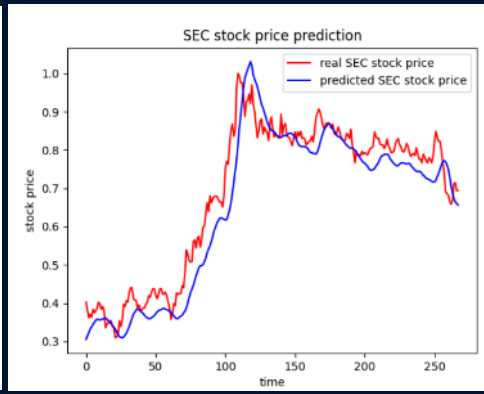
100, 32

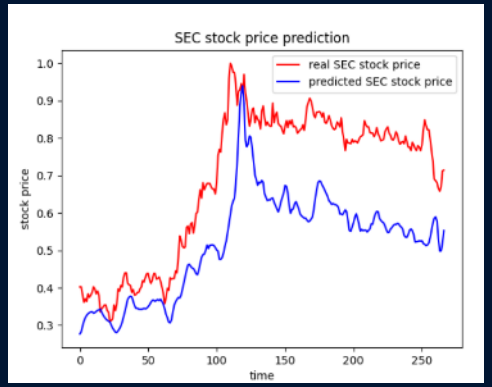
60, 32

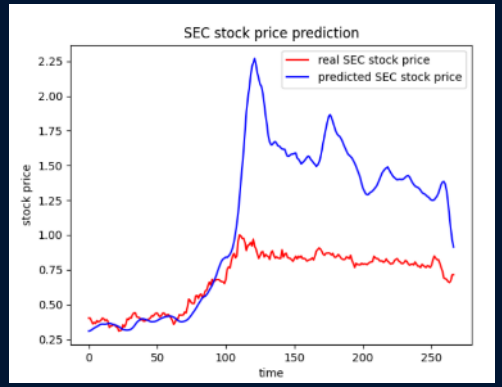


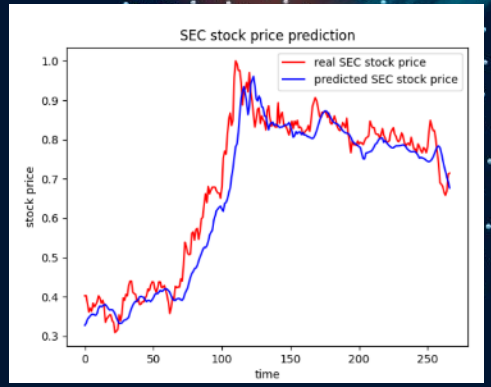


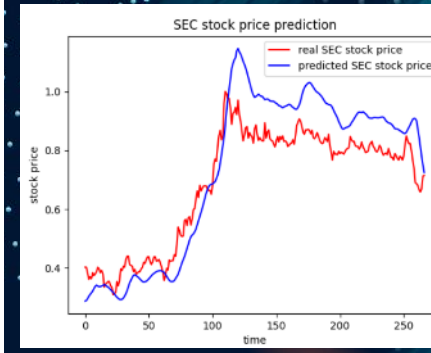












Nadam - 100, 32

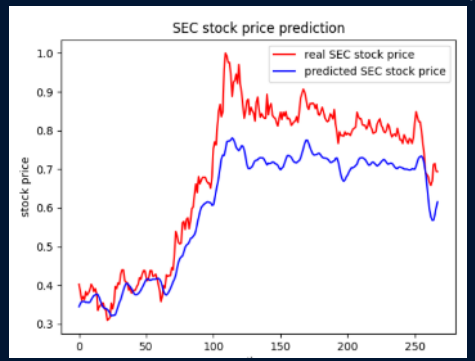
80, 32

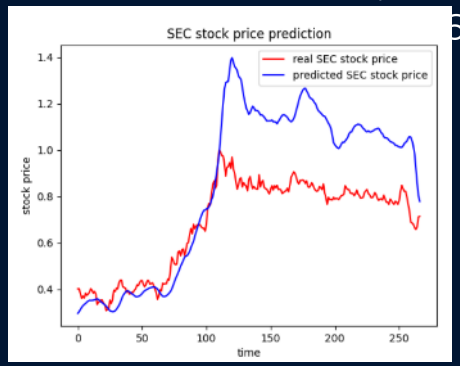
100, 64

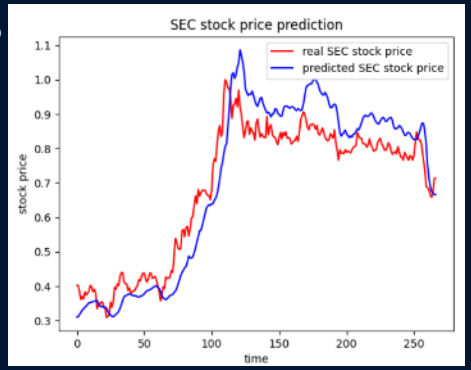
100, 128

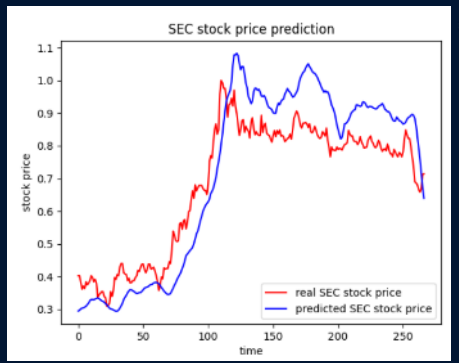
60,32

80, 64







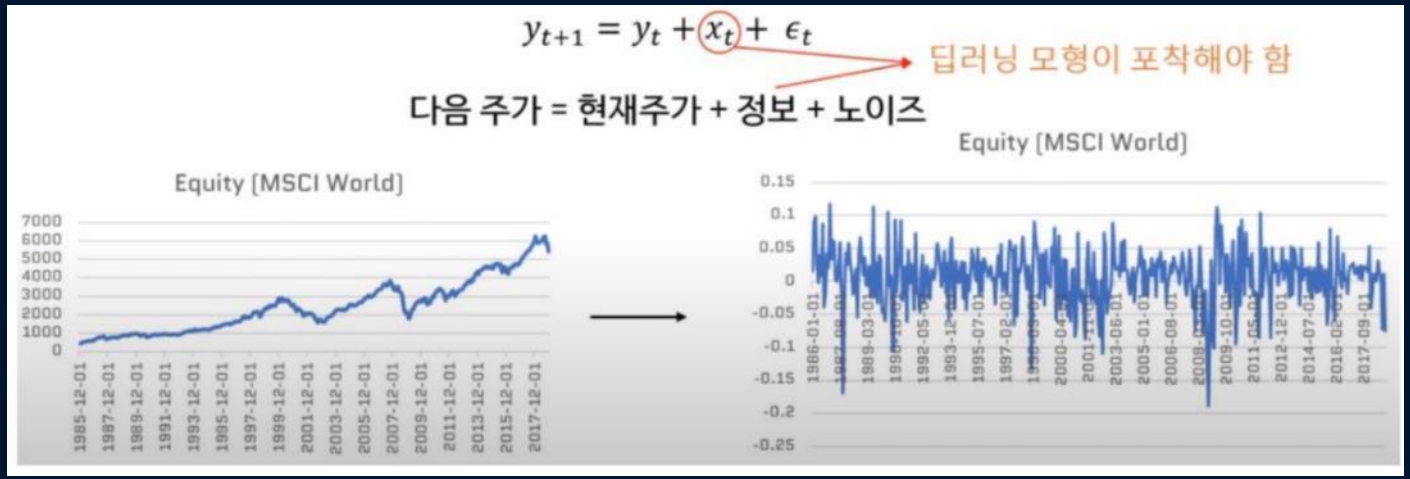


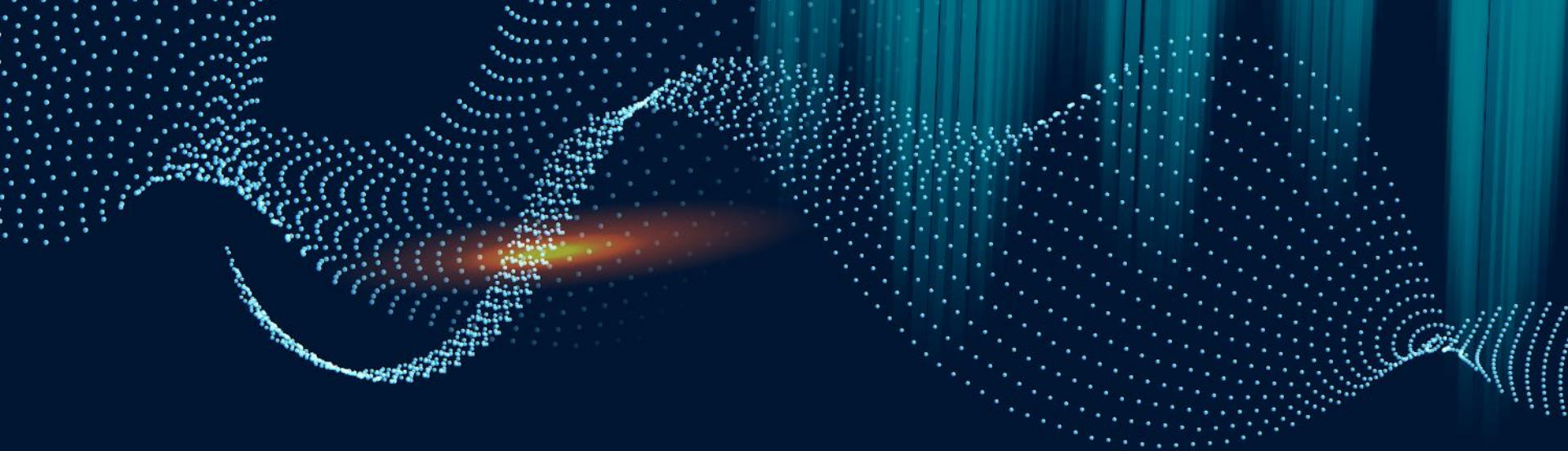
2. 금융데이터의 문제점

1) 노이즈 > 정보량 → 학습이 어려움.

- 원래는 '내일 주가 = 현재 주가 + 정보 + 노이즈'인데, 노이즈가 너무 커서, '내일 주가 = 현재 주가 + 노이즈'가 됨.
- 고려해야할 요소가 많음. 주가 데이터, 매크로 데이터(금리, 인플레이션, 장단기 금리차 등),

  High Level Feature(확장적/긴축적 통화 정책 분류, 단기/장기 부채 사이클 등)





03 웹 서비스

1. 서버 환경 구성 도커 컨테이너 구성

> 전체적인 구성

- **DB 서버 컨테이너 (_db)**
  - mysql 이미지를 토대로 생성된 컨테이너.
- **웹 어플리케이션 서버 컨테이너 (_webapp)**
  - 장고 웹 프레임워크와 웹 소스가 위치하는 컨테이너.
  - 다른 패키지로 인한 영향을 최소화하는 방향으로 설계되었다. 이를 위해서 API를 목적으로 하는 별도의 컨테이너가 필요하게 되었다.
- **API 서버 컨테이너 (_data-api)**
  - 데이터베이스 생성, 주식 금융 API, 비동기 통신을 위한 API 등을 위한 컨테이너.
  - 금융, 통계 등을 위한 패키지의 크기가 상당히 크고, 추가되는 패키지마다 용량이 상당하므로, 웹과는 별도의 컨테이너로 구성하게 되었다.

1. 서버 환경 구성

> 웹 어플리케이션 서버 컨테이너

설치된 패키지

- ➔ **Django**
  - ◆ 장고 웹 프레임워크
- ➔ **Mysql-connector-python**
  - ◆ mysql-client라는 클라이언트 도구가 있으나, 설치가 번거롭고 mysql 8.x를 지원하지 않는 이슈로 인해서, mysql에서 공식으로 제공되는 mysql-connector-python을 설치했다.
- ➔ **Django-settings-export**
  - ◆ django의 설정을 templates에서 이용할 수 있게 도와주는 패키지이다.
- ➔ **Factory-boy**
  - ◆ 테스트할 데이터나 임의의 데이터를 생성해주는 factory 패키지이다.

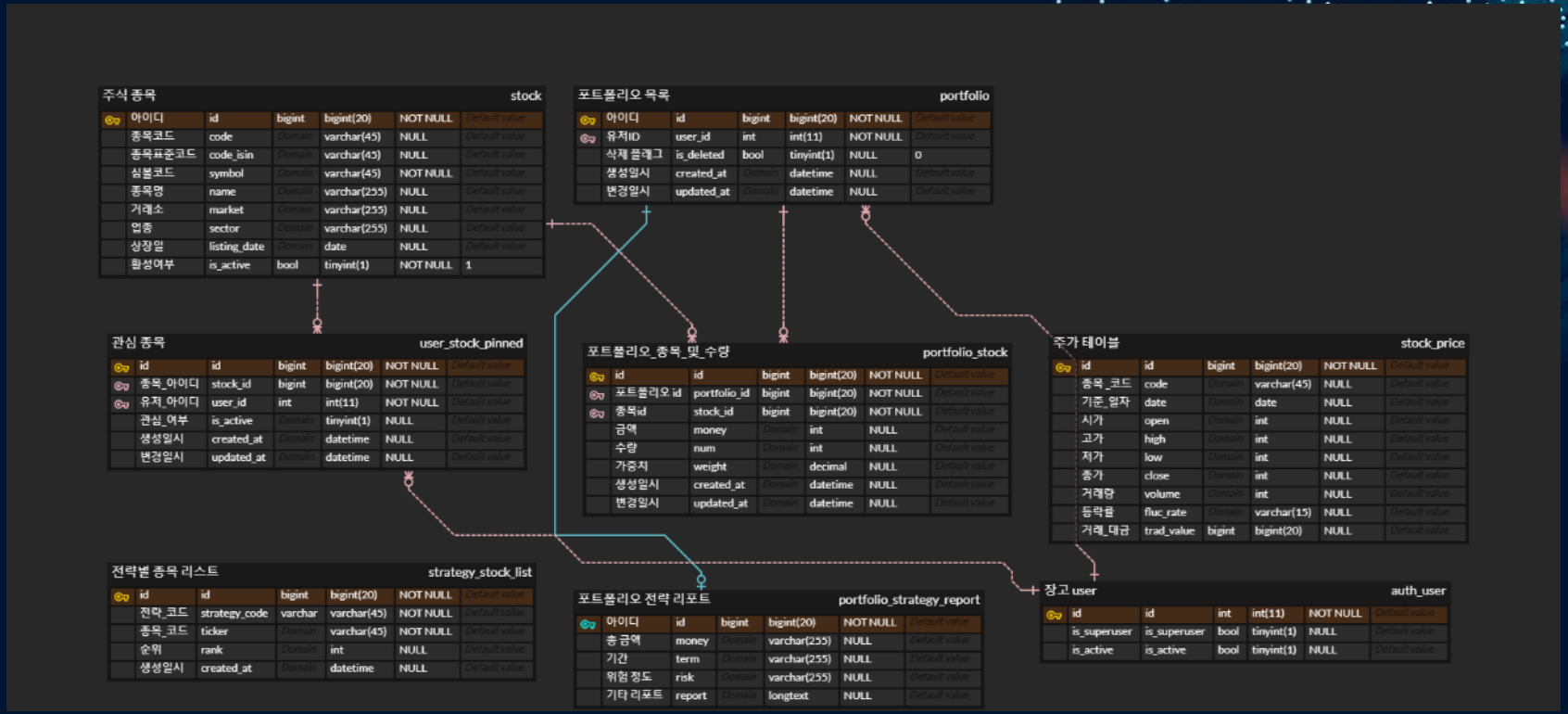
1. 서버 환경 구성

> 백엔드 + API 컨테이너

설치된 패키지

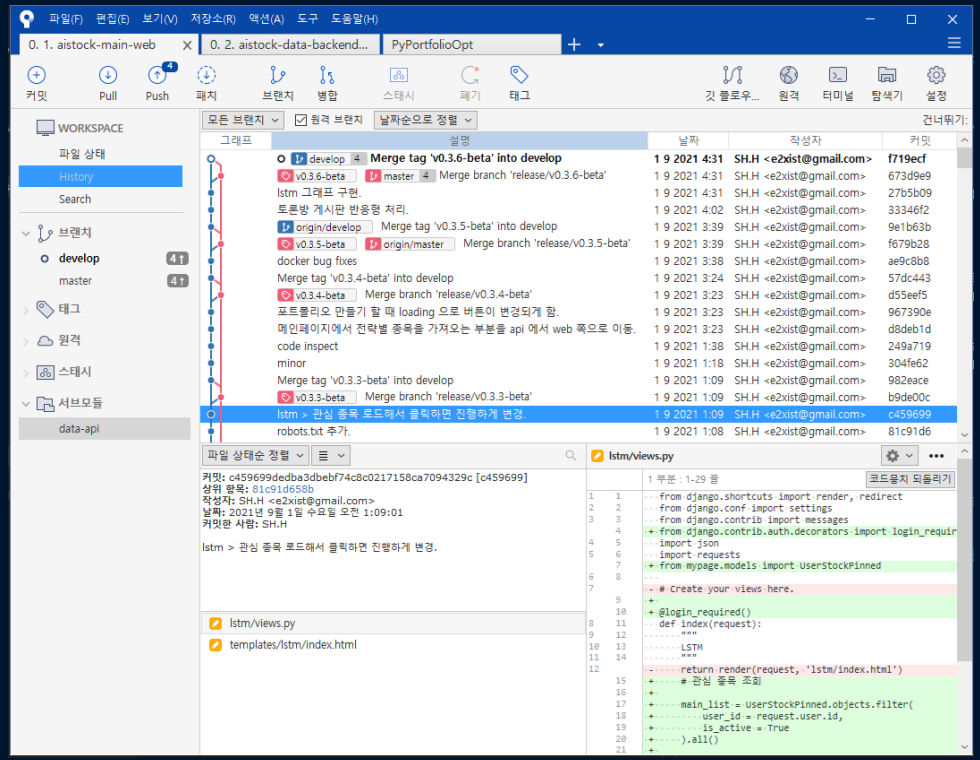
- ➔ **Flask, Flask RestX**
  - ◆ Flask RestX 를 이용하면 API 관리 및 테스트가 조금 더 수월해진다.
- ➔ **PyPorflioOpt**
  - ◆ 전략별 포트폴리오 구성 부분을 도와주는 패키지이다.
- ➔ **Finance Data Reader, pyKRX**
  - ◆ 주식 종목, 주가 정보를 크롤링하거나 데이터를 수집하는 부분을 도와주는 패키지이다.
- ➔ **TensorFlow, Matplotlib**
  - ◆ LSTM 분석 예측을 위해 이용하였다.
- ➔ **Mysql-connector-python**
  - ◆ 종목, 주가 데이터를 테이블에 적재하고, 쿼리로 활용하기 위해 사용되었다.

2. 데이터베이스 설계


주식 종목
stock
아이디 id bigint bigint(20) NOT NULL
종목코드 code varchar(45) NULL
종목표준코드 code_isin varchar(45) NULL
심볼코드 symbol varchar(45) NOT NULL
종목명 name varchar(255) NULL
거래소 market varchar(255) NULL
업종 sector varchar(255) NULL
상장일 listing_date date NULL
활성여부 is_active bool tinyint(1) NOT NULL 1
포트폴리오 목록
portfolio
아이디 id bigint bigint(20) NOT NULL
유저ID user_id int int(11) NOT NULL
삭제 플래그 is_deleted bool tinyint(1) NULL 0
생성일시 created_at datetime NULL
변경일시 updated_at datetime NULL
관심 종목
user_stock_pinned
id id bigint bigint(20) NOT NULL
종목_아이디 stock_id bigint bigint(20) NOT NULL
유저_아이디 user_id int int(11) NOT NULL
관심_여부 is_active tinyint(1) NULL
생성일시 created_at datetime NULL
변경일시 updated_at datetime NULL
포트폴리오_종목_및_수량
portfolio_stock
id id bigint bigint(20) NOT NULL
포트폴리오 id portfolio_id bigint bigint(20) NOT NULL
종목id stock_id bigint bigint(20) NOT NULL
금액 money int NULL
수량 num int NULL
가중치 weight decimal NULL
생성일시 created_at datetime NULL
변경일시 updated_at datetime NULL
주가 테이블
stock_price
id id bigint bigint(20) NOT NULL
종목_코드 code varchar(45) NULL
기준_일자 date date NULL
시가 open int NULL
고가 high int NULL
저가 low int NULL
종가 close int NULL
거래량 volume int NULL
등락률 fluc_rate varchar(15) NULL
거래_대금 trad_value bigint bigint(20) NULL
전략별 종목 리스트
strategy_stock_list
id id bigint bigint(20) NOT NULL
전략_코드 strategy_code varchar varchar(45) NOT NULL
종목_코드 ticker varchar(45) NOT NULL
순위 rank int NULL
생성일시 created_at datetime NULL
포트폴리오 전략 리포트
portfolio_strategy_report
아이디 id bigint bigint(20) NOT NULL
총 금액 money varchar(255) NULL
기간 term varchar(255) NULL
위험 정도 risk varchar(255) NULL
기타 리포트 report longtext NULL
장고 user
auth_user
id id int int(11) NOT NULL
is_superuser is_superuser bool tinyint(1) NULL
is_active is_active bool tinyint(1) NULL

3. 소스 관리

> git, sourceTree



소스 버전 관리

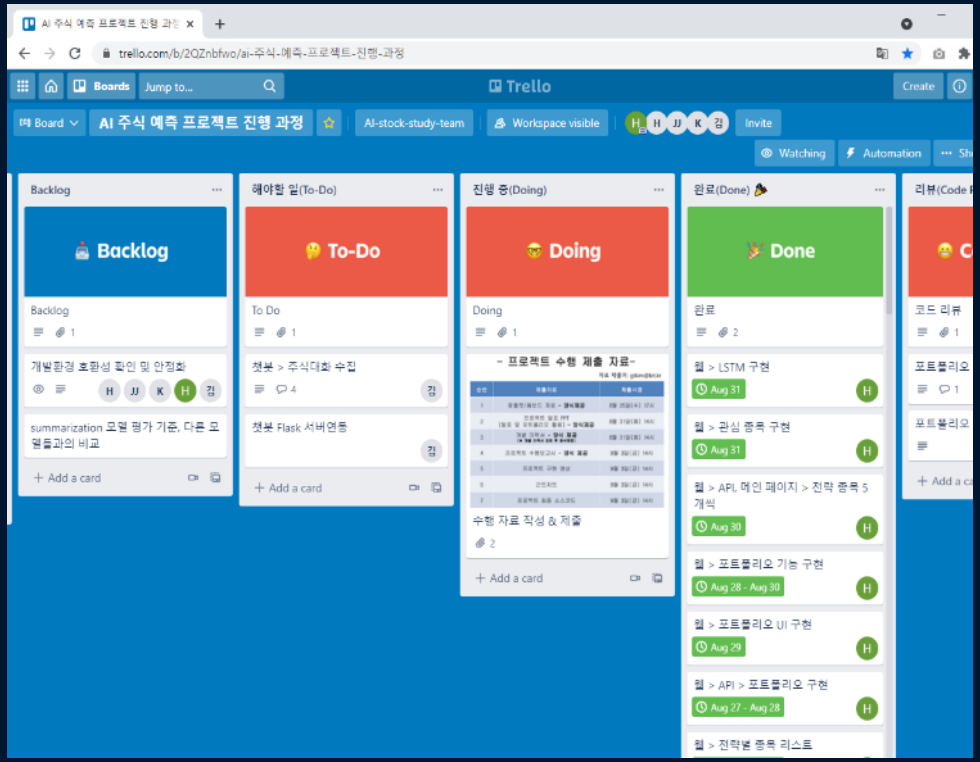
SourceTree 와 '깃플로우'를 이용하여 코드를 관리.

Master 브랜치에 업로드할 때에는 깃플로우로 버저닝 관리를 함

서버에서는 master 브랜치만 clone하여 배포 관리를 함

4. 협업

> Trello



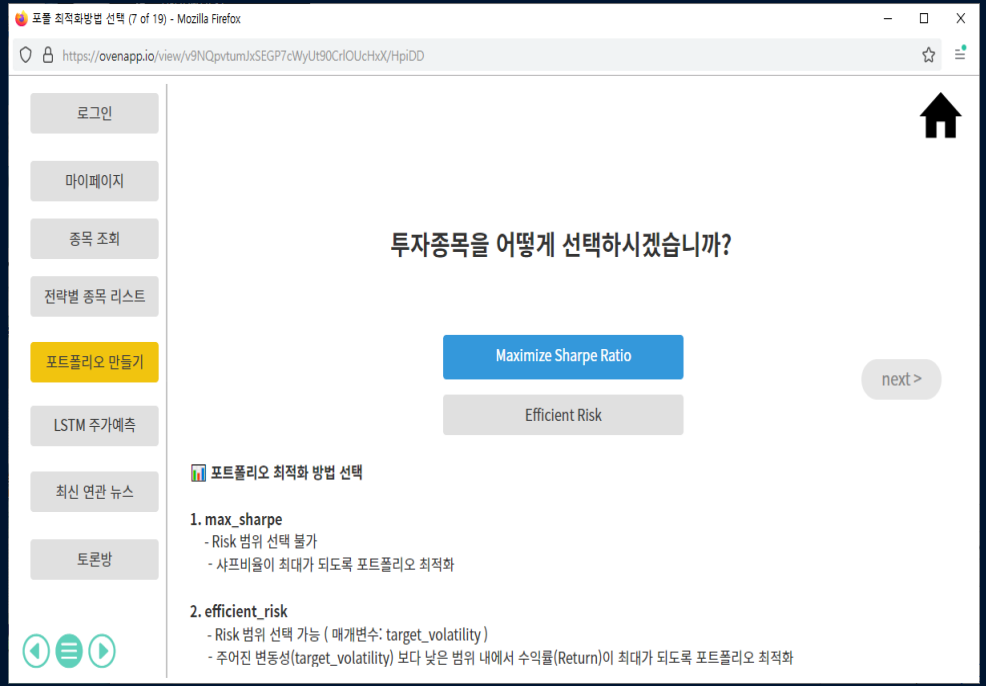
협업 Trello 사용

협업을 위해서 Trello 의 칸반보드를 이용하여, 일정과 진행사항을 공유하며 프로젝트를 진행

회의, 공유, 일정, 의논등은 메신저를 통해서 진행을 하였으나, 작업 상황의 진행도, 완성도를 확인하는 데에 Trello가 크게 도움이 되었음

4. 협업 > 화면 설계

> Oven

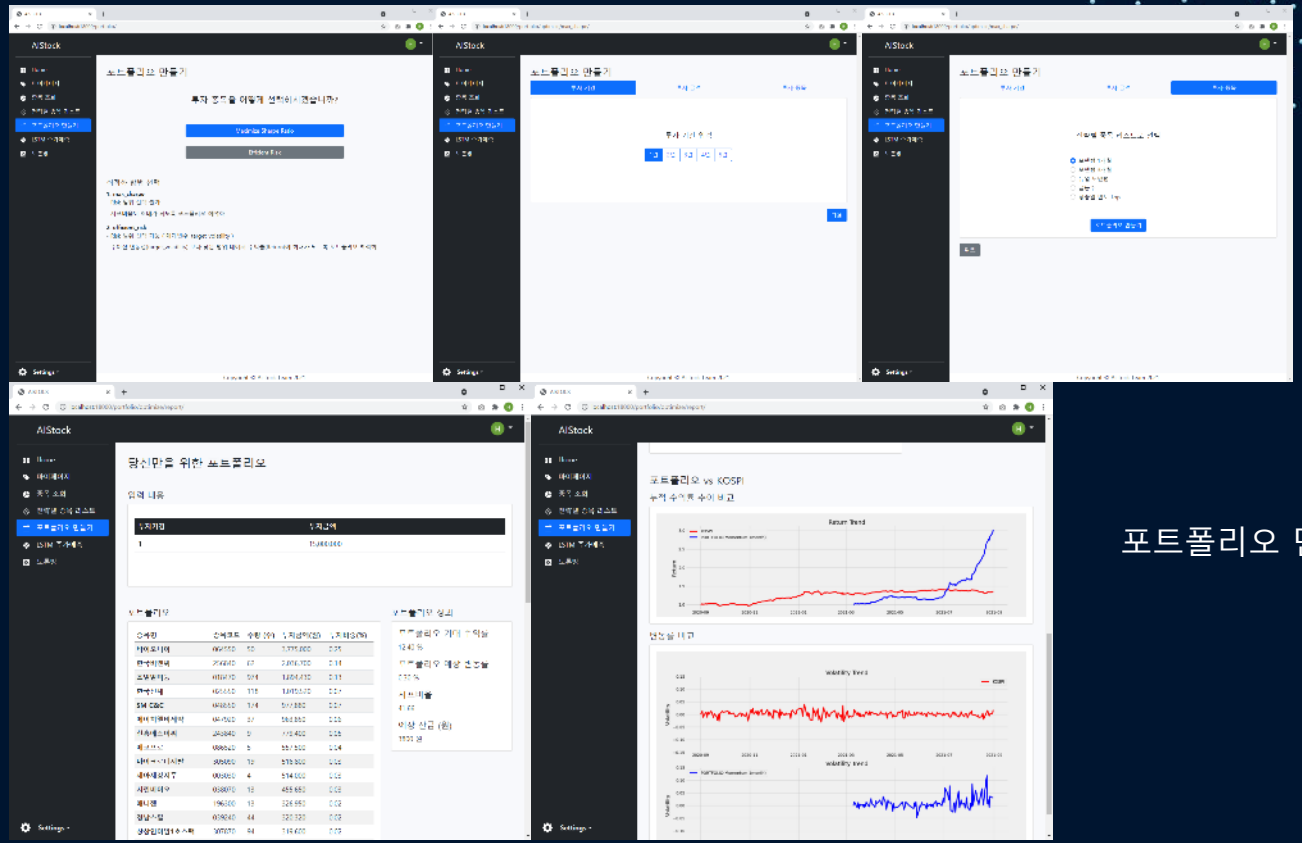


화면 설계 및 스토리보드

화면 설계를 공유하고 스토리보드를 구성하는 과정에서 Oven을 이용

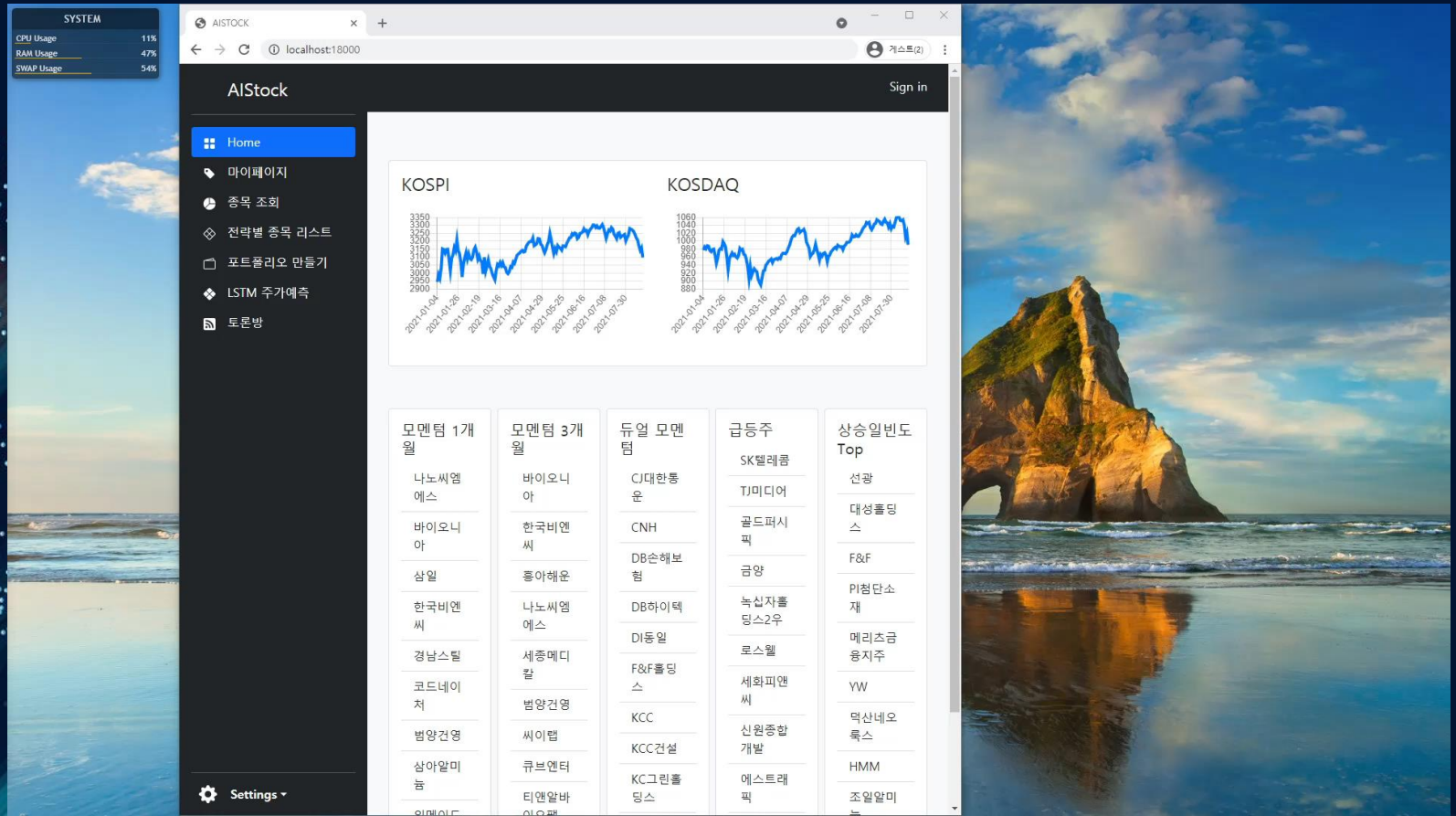
다른 팀원분들이 스토리보드를 만들어주시느라 고생 많으셨습니다.

5. 아웃풋



포트폴리오 만드는 과정의 결과물

프로젝트 구현 영상


SYSTEM
CPU Usage 11%
RAM Usage 47%
SWAP Usage 54%
AISTOCK
localhost:18000
AIStock
Sign in
Home
마이페이지
종목 조회
전략별 종목 리스트
포트폴리오 만들기
LSTM 주가예측
토론방
Settings
KOSPI
KOSDAQ
모멘텀 1개월
나노씨엠에스
바이오니아
삼일
한국비엔씨
경남스틸
코드네이처
범양건영
삼아알미늄
모멘텀 3개월
바이오니아
한국비엔씨
흥아해운
나노씨엠에스
세종메디칼
범양건영
씨이랩
큐브엔터
듀얼 모멘텀
CJ대한통운
CNH
DB손해보험
DB하이텍
DI동일
F&F홀딩스
KCC
KCC건설
KC그린홀딩스
급등주
SK텔레콤
TJ미디어
골드퍼시픽
금양
녹십자홀딩스2우
로스웰
세화피앤씨
신원종합개발
에스트래픽
상승일빈도 Top
선광
대성홀딩스
F&F
PI첨단소재
메리츠금융지주
YW
덕산네오룩스
HMM

프로젝트 종합 및 결론

프로젝트 종합 및 결론

기대효과

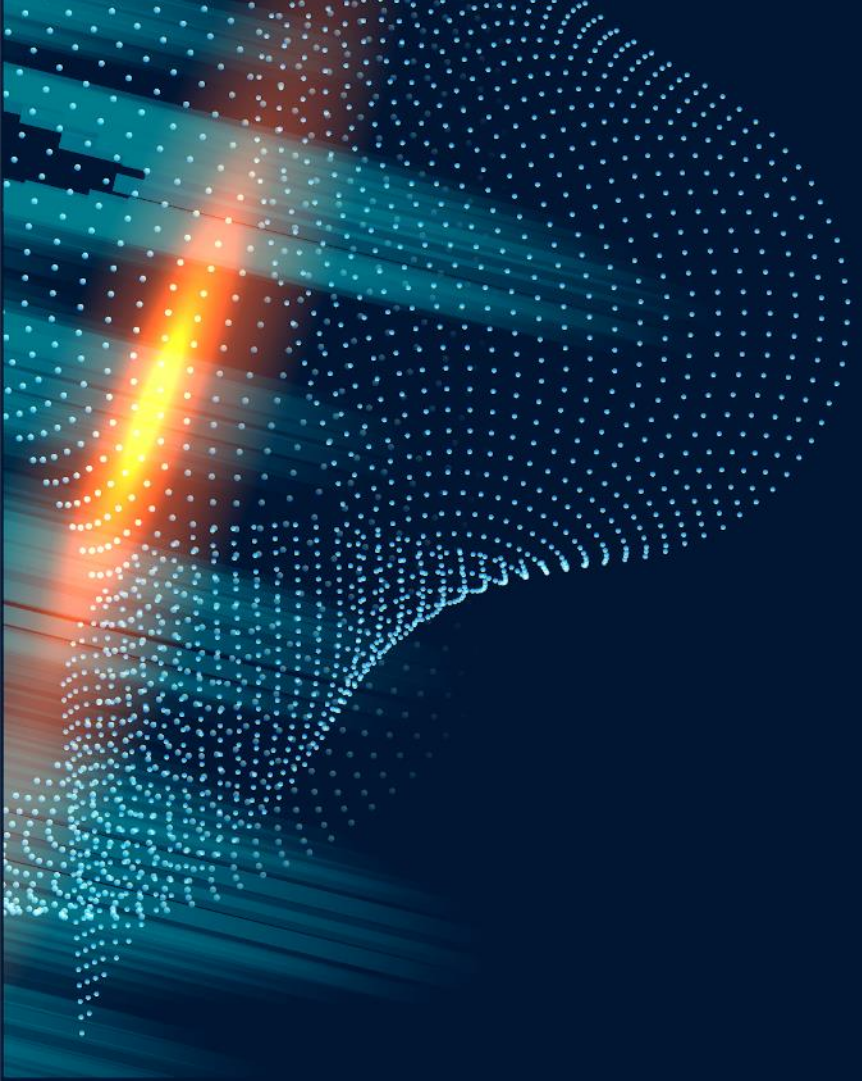
주로 고액 자산가만을 대상으로 하여 접근성이 낮았던 기존의 개인자산관리(PB) 서비스를 소액의 투자금액으로, 내가 원하는 시간과 장소에서, 데이터 기반의 객관적인 추천을 받을 수 있음

프로젝트 한계점

과거의 주가데이터를 기반으로 한 분석이며,
주가에 영향을 미치는 요인이 너무 많기 때문에 미래의 시장 상황을 정확히 예측하는데에 한계가 있음

향후계획

- 최적화된 포트폴리오를 정기적으로 재평가하여 리밸런싱하는 기능추가
- 딥러닝을 이용한 LSTM 주가 예측 개선



느낀점

1. 난관 극복 사례
2. 잘한 부분
3. 아쉬운 부분
4. 기타

한 줄평 - 샛별

" 주식 , 데이터분석, 웹, 개발.....
하나같이 모두 낯설고 어려웠지만, 어려웠기 때문에 개념부터 실전까지 익히고 배울 수 있는 시간이었다. "

— 난관 극복 사례

" 프로젝트 과정 중 난관에 부딪히거나 예상치 못한 어려움에 직면할 때마다 포기하지 않고 팀원들과 함께 해결방법을 찾아가며 끝까지 마무리를 했다는 점 "

— 잘한 부분

" 머신러닝과 딥러닝 모델을 활용해서 유의미한 예측 결과를 도출해보고 싶었는데, 시간이 조금만 더 있었더라면.. "

— 아쉬운 부분

" 3개월이라는 시간을 돌이켜보면 매 순간 최선을 다했던 나, 그리고 팀원들이 있었기에 해낼 수 있었고 많이 배울 수 있는 뜻 깊은 시간이었다. "

— 기타

한 줄평 - 지윤

"생각보다 Class/함수화 하는게 쉽지 않았지만 일일이 오류를 찾아가며 더 배울 수 있던 좋은 기회"

— 난관 극복 사례

"팀원들의 배려와 믿음 및 신뢰가 결정적, 포기하지 않고 난관을 극복한 부분 "

— 잘한 부분

"웹 디자인을 한 번 공부해보고 싶었는데"

— 아쉬운 부분

"내가 코딩인지 코딩이 나인지"

— 기타

한 줄평 - 석훈

"파이썬이 처음이었는데 개념을 이해하는 데에 초반에 어려움이 있었다. 지금은 익숙해져서 다른 언어가 오히려 헷갈린다... "

— 난관 극복 사례

"목적별로 서버를 컨테이너 구성한 점이 재밌었다.
매순간이 난관인 느낌이었는데
꾸준히 나아 갔던 점이 만족스럽다"

— 잘한 부분

"파이썬 코드, 도커, Django, Pandas에 익숙해지는 데에 시간 소모가 걸려서 기능 개선에 시간을 많이 사용하지 못한 점이 아쉽다."

— 아쉬운 부분

"많은 걸 배울 수 있었고, 스스로 한계를 넘어서는 느낌이 재밌었다."

— 기타

Reference

- 파이썬으로 배우는 포트폴리오
- 파이썬 증권 데이터 분석
- 파이썬으로 배우는 알고리즘 트레이딩 (링크)
- Pyportfolio (링크)
- Finance-datareader (링크)
- StackOverflow : 해결방법 참고용

Q & A