

C반 조주연

PORTFOLIO ANALYSIS

포트폴리오 분석

R과 Python을 활용한 빅데이터 분석 및 시각화 기획

INDEX

01
주제 설명



02
데이터 수집



03
데이터 전처리



04
변수 계산 +
그래프 그리기



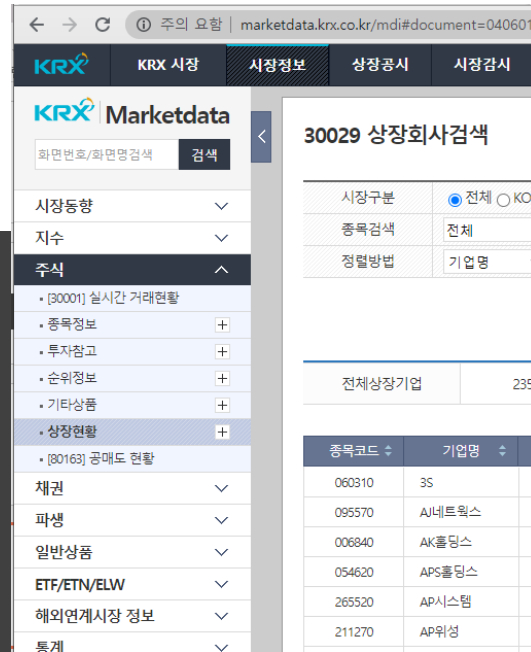
05
Shiny를
이용한 자동화

01 주제 설명



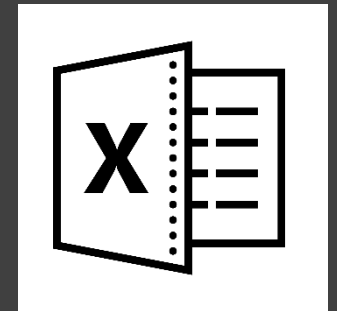
02 데이터 수집 - 종목코드 데이터

- 1 Excel 파일 다운
KRX 사이트 접속
시장구분 : KOSPI



번호	종목코드	기업명	업종코드	업종
1	95570	AJ네트웍스	147603	산업용 기
2	6840	AK홀딩스	116409	기타 금융
3	27410	BGF	116409	기타 금융
4	282330	BGF리테일	74701	종합 소매
5	138930	BNK금융지	116409	기타 금융
6	1460	BYC	31401	봉제의복
7	1040	CJ	116409	기타 금융
8	79160	CJ CGV	105901	영화, 비디
9	120	CJ대한통운	84903	도로 화물
10	11150	CJ씨푸드	31007	기타 식품
11	97950	CJ제일제당	31007	기타 식품
12	590	CS홀딩스	116409	기타 금융
13	12030	DB	106201	컴퓨터 프
14	16610	DB금융투자	116601	금융 지원
15	5830	DB소해비	116501	보험

- 3 파일로 저장



ticker_code_data.xls

- 2 데이터 가공
종목코드, 기업명 column만 남기기

02 데이터 수집 – 주가 데이터

- 1 사용 패키지: 금융 데이터 분석 패키지 `quantmod`
`getSymbol()` 함수 사용
Stock1, Stock2, 국채 주가 데이터 가져오기

- 2 기간: 2019-01-01 ~ 2019-12-31

수정주가(KS.Adjusted) 대신 **KS.Close** 사용
(한국 자료에는 오류가 많기 때문)

- 3 데이터 형태:

	x006400.KS.Open	x006400.KS.High	x006400.KS.Low	x006400.KS.Close	x006400.KS.Volume
2019-01-02	222000	224000	208500	210500	345548
2019-01-03	209000	209500	202000	203000	484354
2019-01-04	201500	204000	194000	201000	474274
2019-01-07	209000	219500	208500	217000	549913
2019-01-08	218000	219500	213000	218500	310061
2019-01-09	220500	227000	219000	226500	455066
	x006400.KS.Adjusted				
2019-01-02	210500				
2019-01-03	203000				
2019-01-04	201000				
2019-01-07	217000				
2019-01-08	218500				
2019-01-09	226500				

03 데이터 전처리 - Two Securities Analysis

일별 수익률 :

$((\text{환매시 기준가격} / \text{가입시 기준가격}) - 1) * 100$

For문 돌려서 Row별로 생성

DR_stock1 변수에 데이터프레임
형식으로 저장

Cbind() 이용 ► 원래 데이터프레임과 합치기

	x006400.KS.Open	x006400.KS.High	x006400.KS.Low	SDI_Close
2019-01-02	222000	224000	208500	210500
2019-01-03	209000	209500	202000	203000
2019-01-04	201500	204000	194000	201000
2019-01-07	209000	219500	208500	217000
2019-01-08	218000	219500	213000	218500
2019-01-09	220500	227000	219000	226500
	x006400.KS.Adjusted	Daily_Return_SDI		
2019-01-02	210500	0.00		
2019-01-03	203000	-3.56		
2019-01-04	201000	-0.99		
2019-01-07	217000	7.96		
2019-01-08	218500	0.69		
2019-01-09	226500	3.66		

New!

03 데이터 전처리 - Two Securities Analysis

일별 수익률을 활용하여 각 주식의 risk와 return을 계산

Return : 일별 수익률의 평균 / **Risk** : 일별 수익률의 표준편차

Dataframe 생성 1 : weight

두 주식의 가중치
(0~1 사이의 값)

	w_Naver	w_SDI
1	0.000	1.000
2	0.005	0.995
3	0.010	0.990
4	0.015	0.985

Dataframe 생성 2 : elements

각 주식의 통계치를 모아놓는 Dataframe

	Er_Naver	Er_SDI	Er_KOREA	sd_Naver	sd_SDI	NS_cor
1	0	0	0	0	0	0

Return (일별 수익률의 평균) - mean()

Risk (일별 수익률의 표준편차) - sd()

Stock1, Stock2 Return값의 상관계수 - cor()

Filter(!is.na()) / Summarise() 사용해서 각각 계산



	Er_Naver	Er_SDI	Er_KOREA	sd_Naver	sd_SDI	NS_cor
1	0.1661	0.0592	0.0055	1.8577	1.9254	0.09983139

04 변수 계산 – Efficient Frontier

Weight에 elements값을 각각 곱해서 포트폴리오의 return과 risk를 구하기

사용공식

$$\text{Expected return} = (w_1 * r_1) + (w_2 * r_2) + \dots + (w_n * r_n)$$

▶ 포트폴리오의 return값

$$\sigma_P = \sqrt{w_A^2 \sigma^2(k_A) + w_B^2 \sigma^2(k_B) + 2 w_A w_B R(k_A, k_B) \sigma(k_A) \sigma(k_B)}$$

▶ 포트폴리오의 risk값

	w_stock1	w_stock2
1	0.000	1.000
2	0.005	0.995
3	0.010	0.990
4	0.015	0.985
5	0.020	0.980
6	0.025	0.975
7	0.030	0.970
8	0.035	0.965
9	0.040	0.960
10	0.045	0.955
11	0.050	0.950
12	0.055	0.945
13	0.060	0.940

Weight
%>%
Mutate()

	w_Naver	w_SDI	ER_P	SD_P
1	0.000	1.000	0.0592000	1.925400
2	0.005	0.995	0.0597345	1.916723
3	0.010	0.990	0.0602690	1.908090
4	0.015	0.985	0.0608035	1.899503
5	0.020	0.980	0.0613380	1.890963
6	0.025	0.975	0.0618725	1.882469
7	0.030	0.970	0.0624070	1.874022
8	0.035	0.965	0.0629415	1.865624
9	0.040	0.960	0.0634760	1.857275
10	0.045	0.955	0.0640105	1.848974
11	0.050	0.950	0.0645450	1.840724
12	0.055	0.945	0.0650795	1.832525
13	0.060	0.940	0.0656140	1.824378

04 변수 계산 – Efficient Frontier

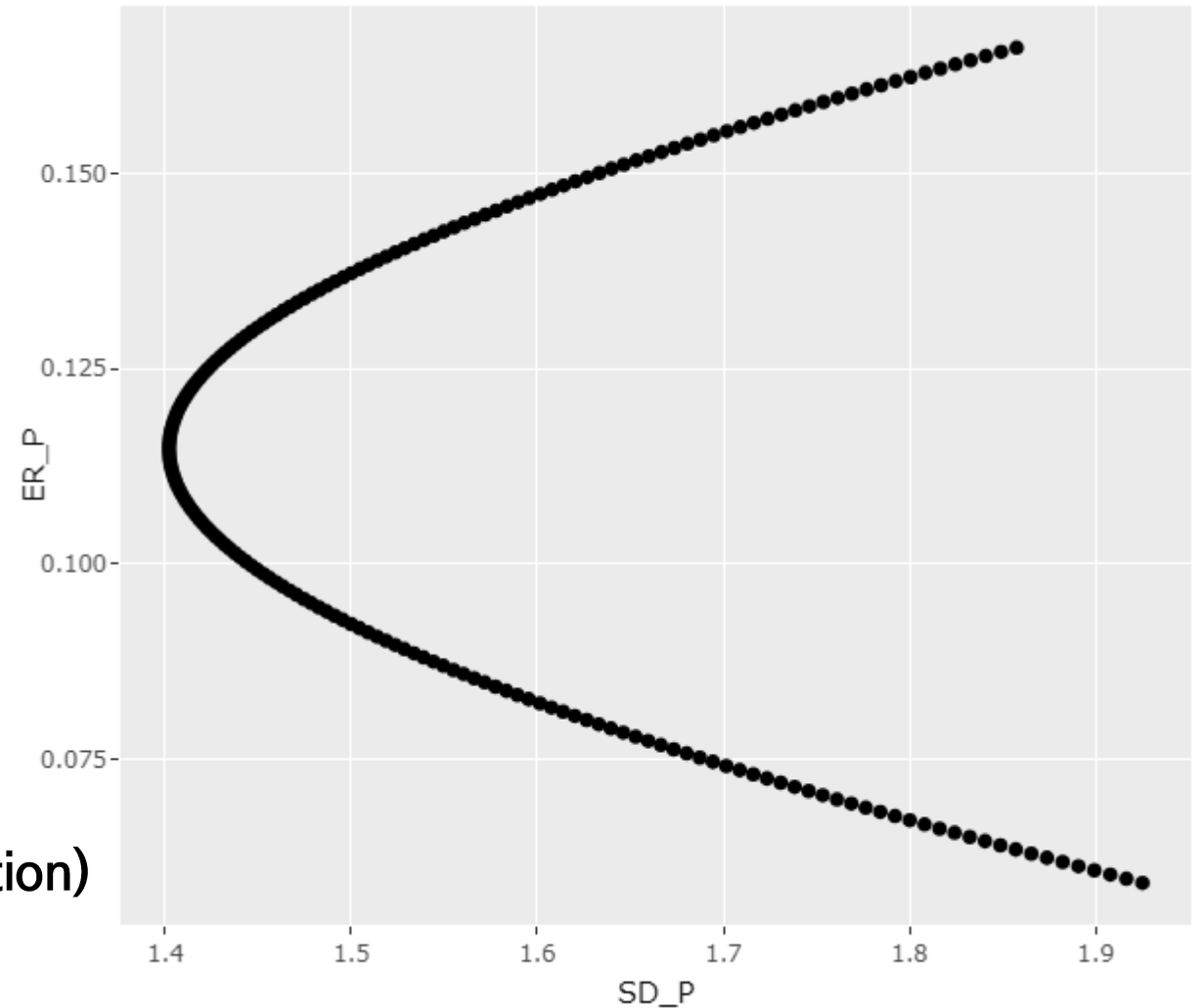
	w_Naver	w_SDI	ER_P	SD_P
1	0.000	1.000	0.0592000	1.925400
2	0.005	0.995	0.0597345	1.916723
3	0.010	0.990	0.0602690	1.908090
4	0.015	0.985	0.0608035	1.899503
5	0.020	0.980	0.0613380	1.890963
6	0.025	0.975	0.0618725	1.882469
7	0.030	0.970	0.0624070	1.874022
8	0.035	0.965	0.0629415	1.865624
9	0.040	0.960	0.0634760	1.857275
10	0.045	0.955	0.0640105	1.848974
11	0.050	0.950	0.0645450	1.840724
12	0.055	0.945	0.0650795	1.832525
13	0.060	0.940	0.0656140	1.824378

ER_P(포트폴리오의 expected return)

▶ Efficient frontier의 y축

SD_P(포트폴리오의 expected standard deviation)

▶ Efficient frontier의 x축



04 변수 계산 – CAL(Capital Allocation Line)

CAL : 위험 자산과 무위험 자산(대한민국 10년 국채) 간의 관계를 선형 그래프로 그림
CAL과 efficient frontier가 만나는 지점 ► 최적의 투자 효율 지점

CAL과 Efficient frontier가 만나는 지점에서
각 주식의 차지하는 비중 = **tangent weight**

$$w_s^{TP} = 1 - w_b^{TP}$$

$$w_b^{TP} = \frac{(E(R_b) - R_f)\sigma_s^2 - (E(R_s) - R_f)\rho_{sb}\sigma_s\sigma_b}{(E(R_b) - R_f)\sigma_s^2 + (E(R_s) - R_f)\sigma_b^2 - (E(R_b) - R_f + E(R_s) - R_f)\rho_{sb}\sigma_s\sigma_b}$$

Tangent weight를
공식에 넣어서

Tangent portfolio expected return과
**Tangent portfolio expected
standard deviation**을 구함

$$\text{Expected return} = (w_1 * r_1) + (w_2 * r_2) + \dots + (w_n * r_n)$$

$$\sigma_P = \sqrt{w_A^2 \sigma^2(k_A) + w_B^2 \sigma^2(k_B) + 2 w_A w_B R(k_A, k_B) \sigma(k_A) \sigma(k_B)}$$

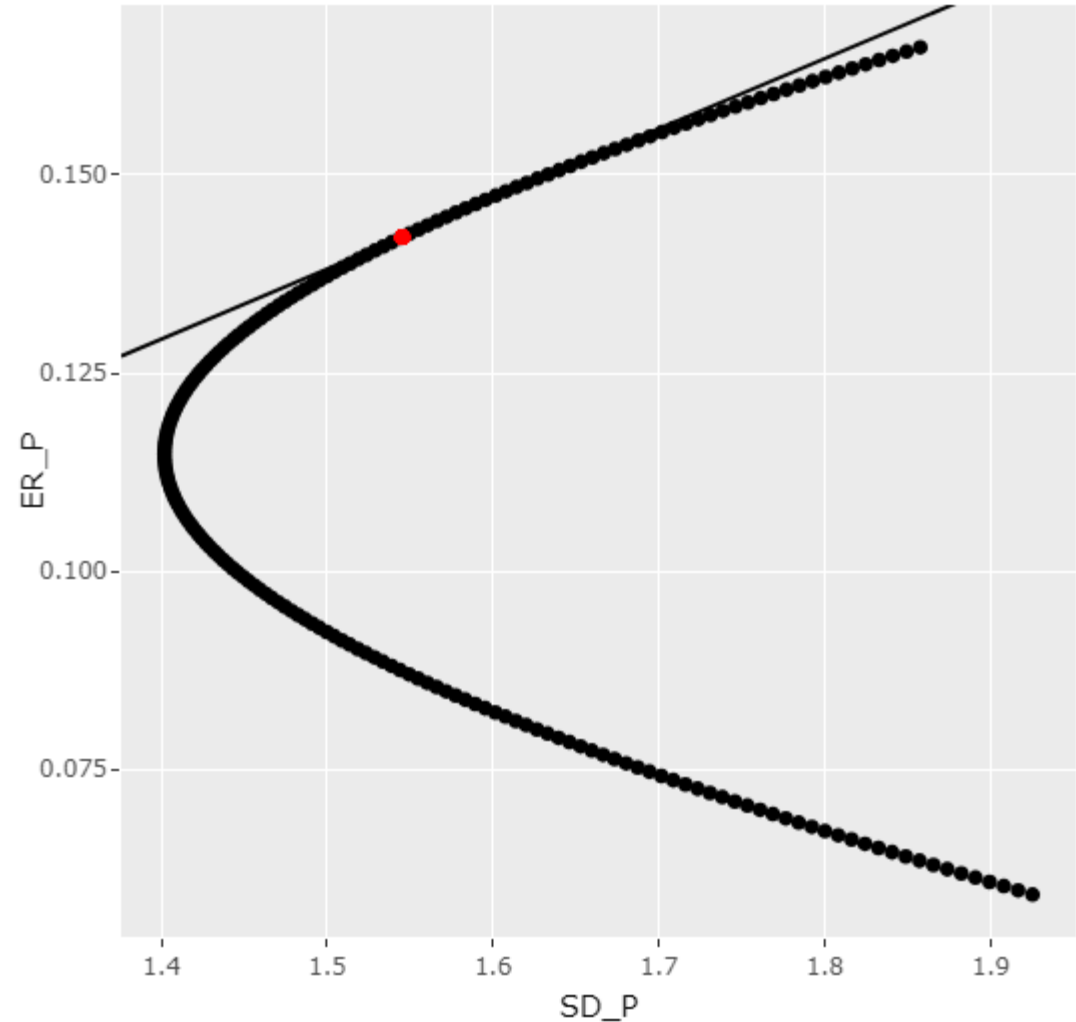
04 변수 계산 – CAL(Capital Allocation Line)

CAL

y intercept : 국채의 return값

Slope :
$$\frac{(E(R_p) - R_f)}{\sigma(R_p)}$$

ggplot2로 그래프 그리기



05 SHINY를 이용한 자동화



INPUT

종목코드 데이터를 목록상자에
넣어서 주식 종목코드를 선택할
수 있도록 변경



PROCESS

Shiny from  Studio



OUTPUT

분석 결과를 ggplotly 그래프로 출력

Tangent portfolio weight,
portfolio return, portfolio risk,
주식별 투자금을 테이블 형태로 출력

05 SHINY를 이용한 자동화

Portfolio Analysis - Efficient Frontier & CAL

stock1

034220(LG디스플레이) ▼

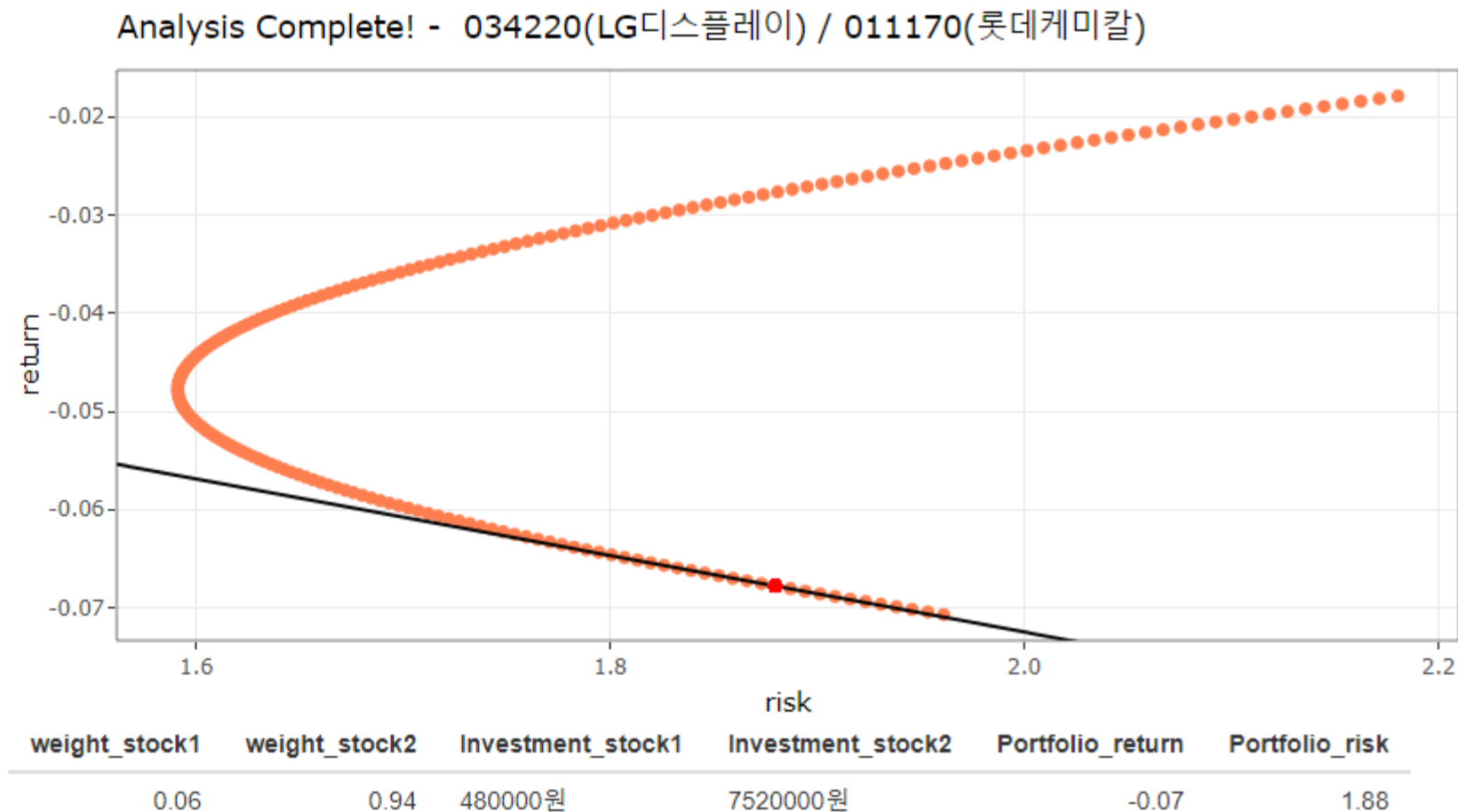
stock2

011170(롯데케미칼) ▼

How much money do you want to invest in?

8000000

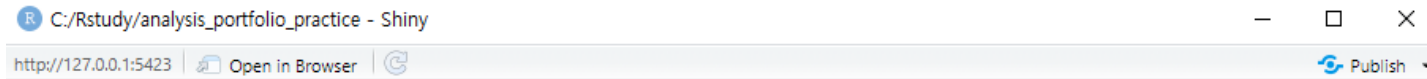
Go!



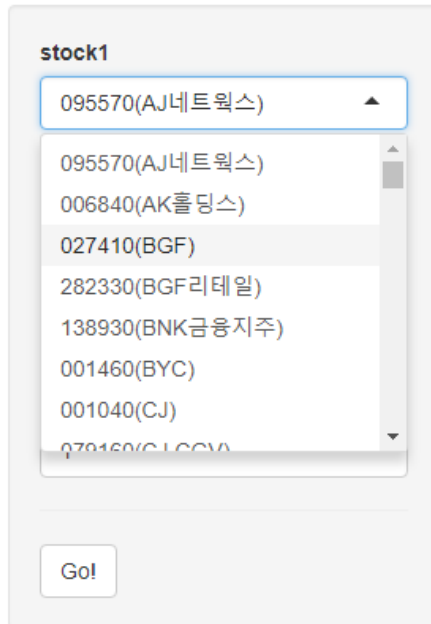
05 SHINY를 이용한 자동화

INPUT

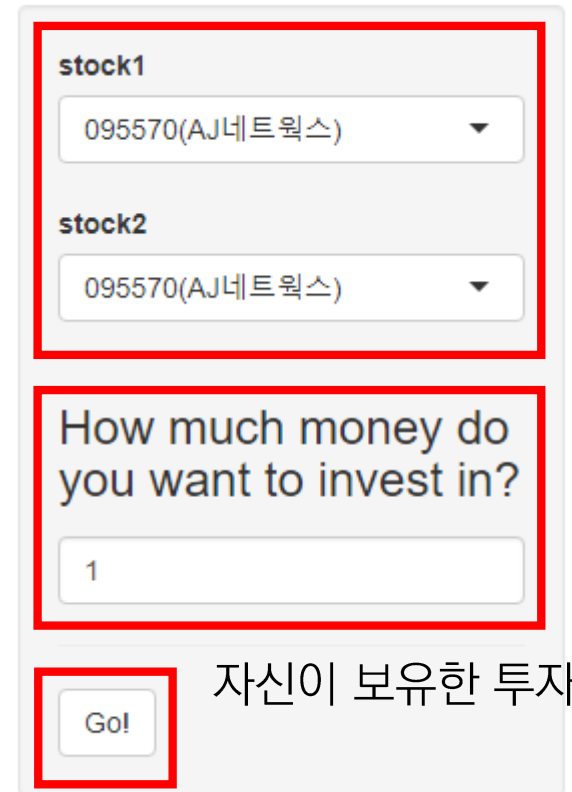
종목코드 데이터를 목록상자에 넣어서 주식 종목코드를 선택 가능



Portfolio Analysis - Efficient Frontier & CAL

A screenshot of the 'stock1' dropdown menu in the Shiny application. The menu is open, showing a list of stock codes and names: 095570(AJ네트웍스), 006840(AK홀딩스), 027410(BGF), 282330(BGF리테일), 138930(BNK금융지주), 001460(BYC), 001040(CJ), and 070160(CJCCV). A 'Go!' button is visible at the bottom of the menu.

주식 종목코드 선택 (목록상자)

A screenshot of the Shiny application input section. It features two dropdown menus labeled 'stock1' and 'stock2', both currently showing '095570(AJ네트웍스)'. Below these is a text input field with the label 'How much money do you want to invest in?' and the value '1'. At the bottom is a 'Go!' button.

자신이 보유한 투자금

Action button

05 SHINY를 이용한 자동화

OUTPUT

가중치

Weight_stock1 :
입력한 stock 1의 가중치

Weight_stock2 :
입력한 stock 2의 가중치

투자금

Investment_stock1 :
stock 1 가중치 * 투자금
(stock 1에 투자가능한 금액)

Investment_stock2 :
stock 2 가중치 * 투자금
(stock 2에 투자가능한 금액)

포트폴리오 Risk&Return

Portfolio_return :
포트폴리오 전체의 수익률

Portfolio_risk :
포트폴리오 전체의 리스크

weight_stock1

weight_stock2

Investment_stock1

Investment_stock2

Portfolio_return

Portfolio_risk

0.10

0.90

8000원

72000원

0.04

1.19

THANK YOU