

파이썬과 인공지능을 활용한 금융 자료 분석

Anaconda 설치

Download from <https://www.anaconda.com/products/individual#download-section>



Products ▾

Pricing

Solutions ▾

Resources ▾

Blog

Company ▾

Get Started



Individual Edition

Your data science toolkit

With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.

Anaconda Individual Edition

Download 

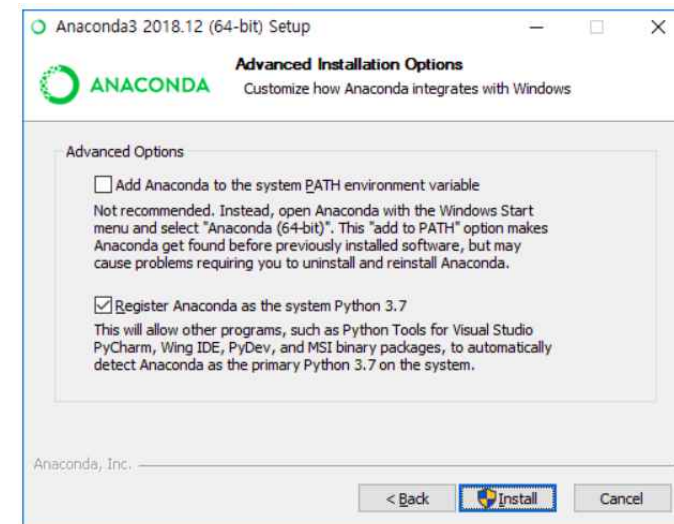
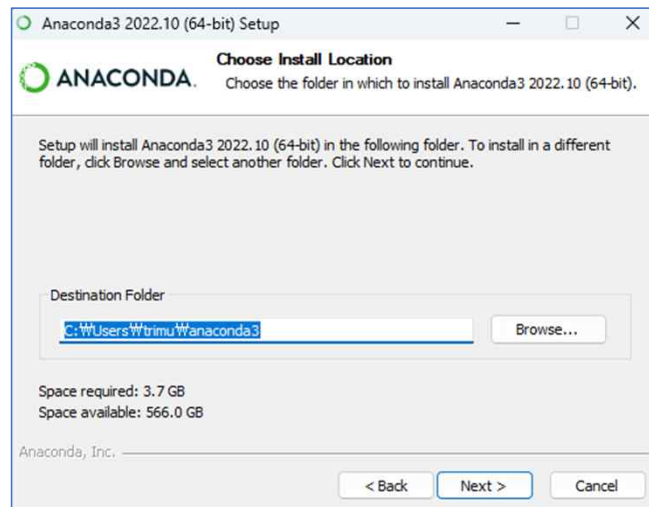
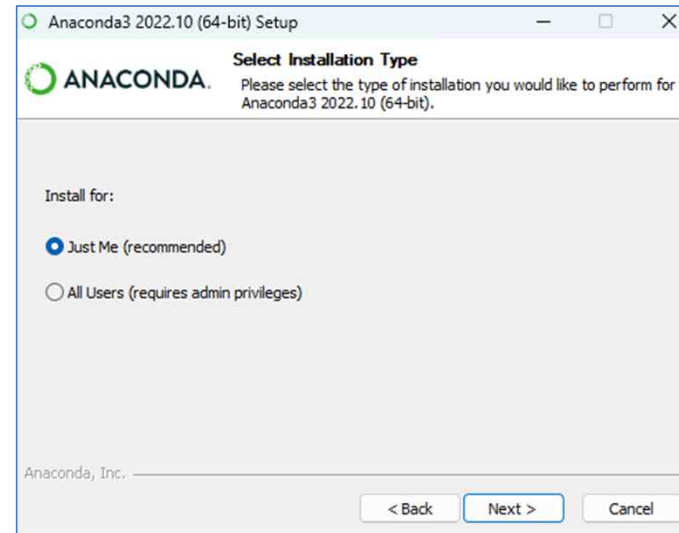
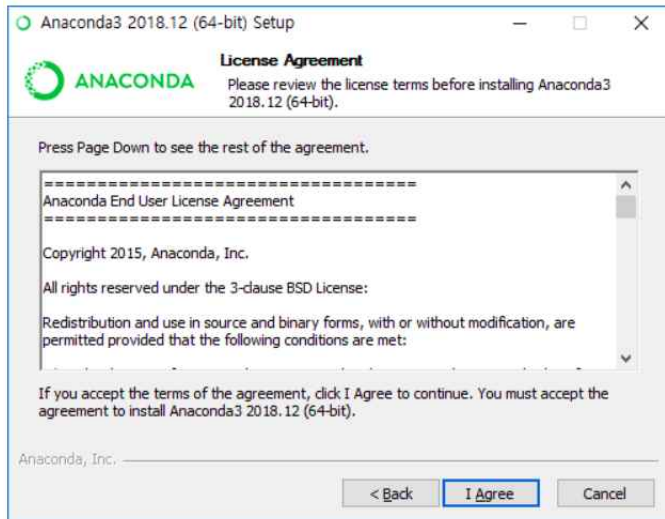
For Windows

Python 3.8 • 64-Bit Graphical Installer • 477 MB

Get Additional Installers



Next 를 눌러 설치 시작



완료

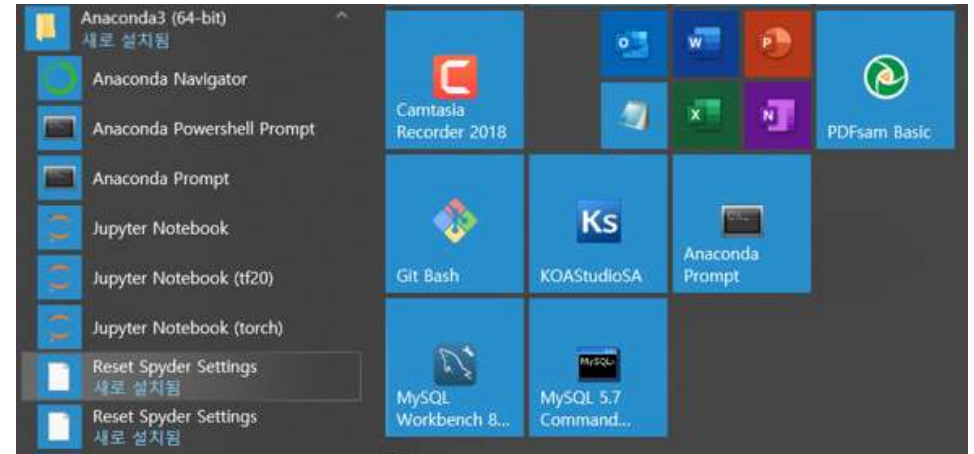
Jupyter Notebook 사용 방법

What is Jupyter Notebook ?

- 주피터 노트북(**Jupyter Notebook**)은 웹 브라우저에서 파이썬 코드를 작성하고 실행해 볼 수 있는 개발도구
- 아나콘다(Anaconda)를 설치하면 **Jupyter Notebook**이 함께 설치
- Notebook 서버 프로그램은 백그라운드에서 실행되는 파이썬 프로그램으로 웹 브라우저를 통해 interactive 하게 Python 명령 실행

Jupyter Notebook 실행 방법

1) Start menu 에서 실행



2) 명령 prompt 에서 실행
> jupyter notebook

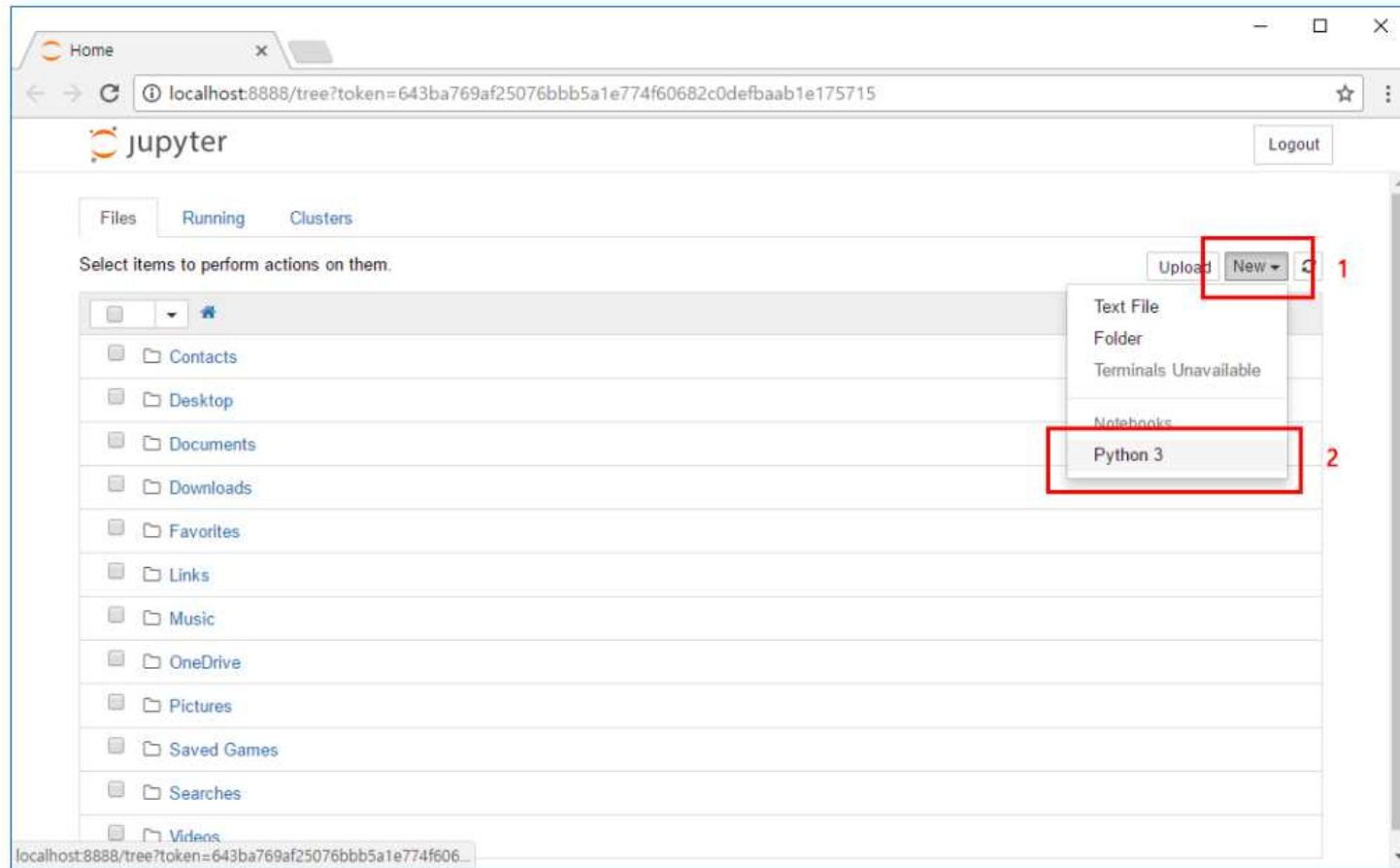
```
(base) C:\Users\trimu>jupyter notebook
[I 08:20:31.790 NotebookApp] Serving notebooks from local directory: C:\Users\trimu
[I 08:20:31.791 NotebookApp] The Jupyter Notebook is running at:
[I 08:20:31.793 NotebookApp] http://localhost:8888/?token=e1bb3f10db684cd21f9dd3c0e89f713306df271678b97d9d
[I 08:20:31.798 NotebookApp] or http://127.0.0.1:8888/?token=e1bb3f10db684cd21f9dd3c0e89f713306df271678b97d9d
[I 08:20:31.800 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation)
)
[C 08:20:31.994 NotebookApp]

To access the notebook, open this file in a browser:
    file:///C:/Users/trimu/AppData/Roaming/jupyter/runtime/nbserver-12876-open.html
Or copy and paste one of these URLs:
    http://localhost:8888/?token=e1bb3f10db684cd21f9dd3c0e89f713306df271678b97d9d
    or http://127.0.0.1:8888/?token=e1bb3f10db684cd21f9dd3c0e89f713306df271678b97d9d
```

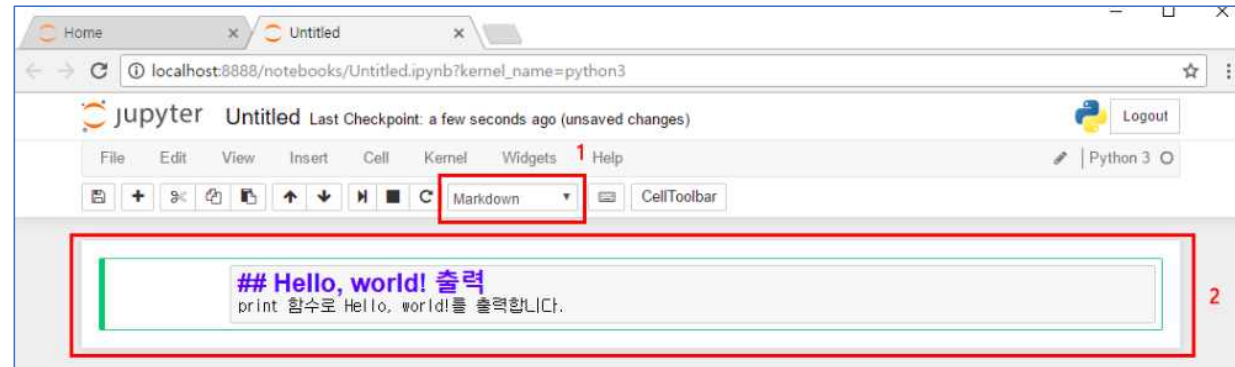
* Login 비밀번호 요구할 경우 사용자 계정의 ~/.jupyter/jupyter_notebook_config.py 삭제

Jupyter Notebook 사용방법

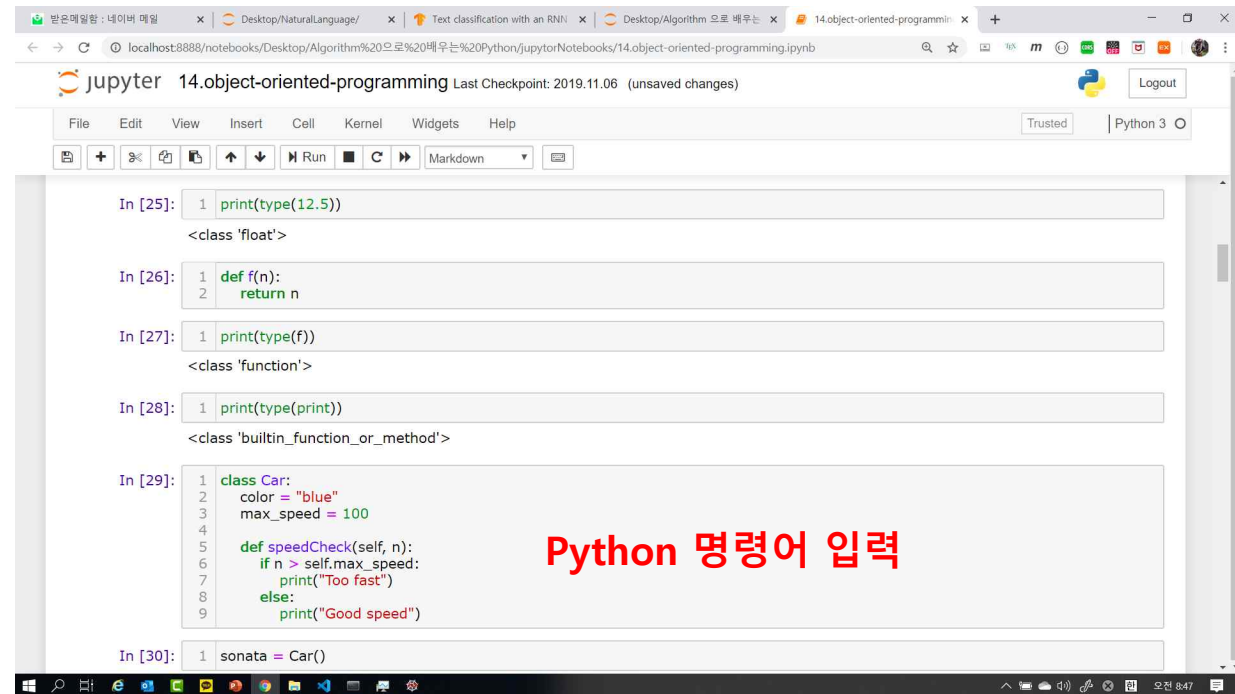
주피터 노트북 초기 화면 에서 New → Python 3 선택 → 새로운 notebook 생성



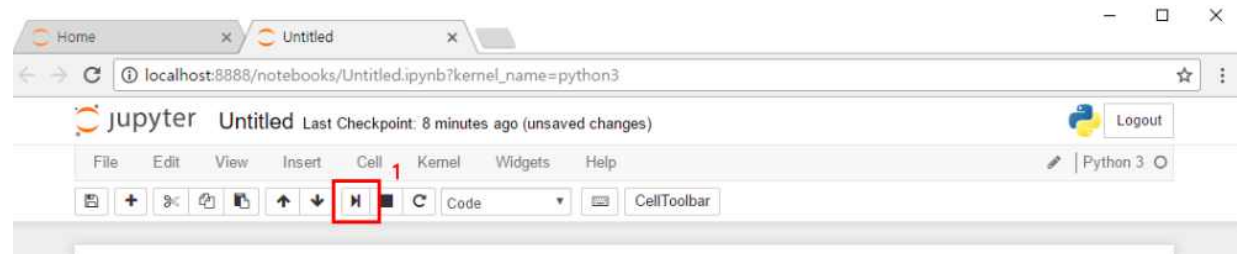
설명 추가



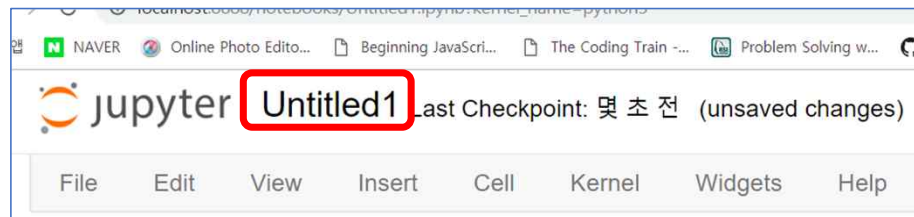
Python Code 입력
및
실행



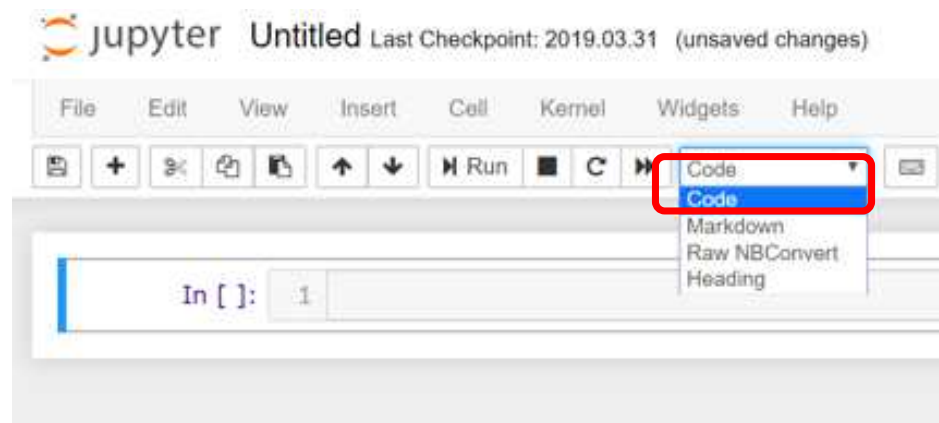
Code 실행



Notebook File Name
바꾸기

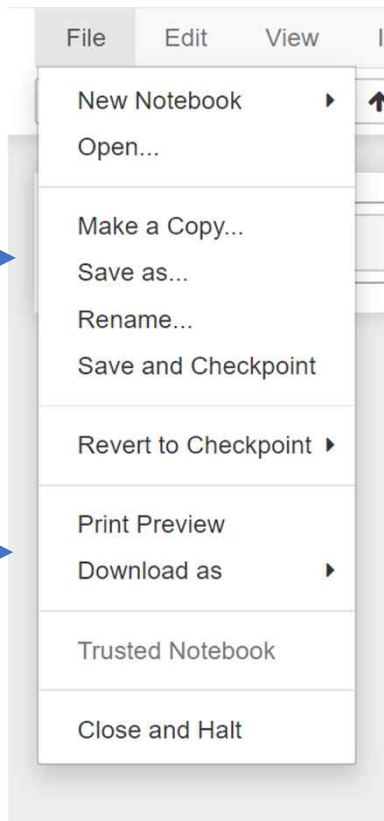


Cell Type 변경



자주 사용하는 menu 들

Notebook
저장



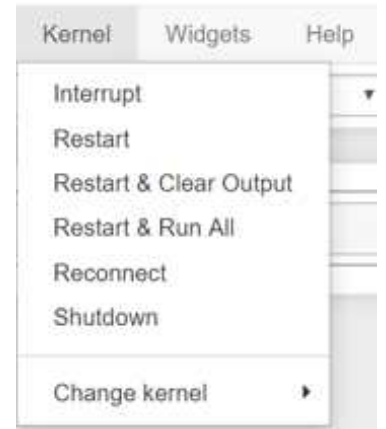
Notebook
download

Cell 삭제
취소

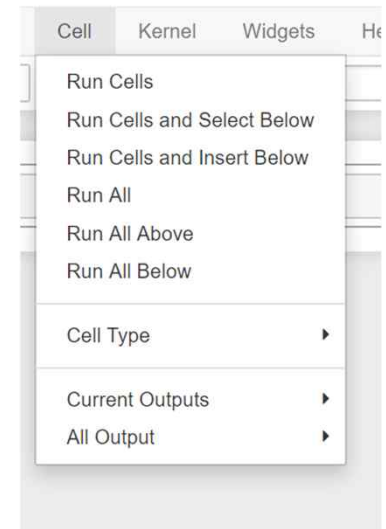


Find/
Replace

Notebook
restart



Cell 실행



Jupyter Notebook 사용방법

- 자주 사용하는 short-cut key

Shift + Enter : cell 실행 + 다음 cell 이동

Ctrl + S : save + checkpoint

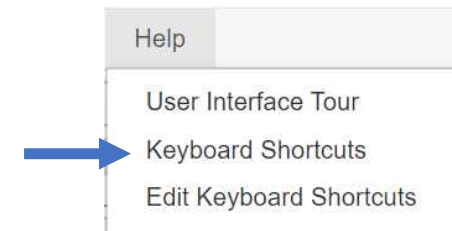
Esc+A : 위쪽에 cell 삽입

Esc+B : 아래쪽에 cell 삽입

Esc+X : cell 삭제

Esc+Z : cell 삭제 취소

Esc+M : cell 을 markdown type 으로 변경



Keyboard shortcuts

The Jupyter Notebook has two different keyboard input modes. **Edit mode** allows you to type code or text into a cell and is indicated by a green cell border. **Command mode** binds the keyboard to notebook level commands and is indicated by a grey cell border with a blue left margin.

Command Mode (press `Esc` to enable)

Edit Shortcuts

<code>F</code> : find and replace	<code>Shift-Down</code> : extend selected cells below
<code>Ctrl-Shift-F</code> : open the command palette	<code>Shift-J</code> : extend selected cells below
<code>Ctrl-Shift-P</code> : open the command palette	<code>Ctrl-A</code> : select all cells
<code>Enter</code> : enter edit mode	<code>A</code> : insert cell above
<code>P</code> : open the command palette	<code>B</code> : insert cell below
<code>Shift-Enter</code> : run cell, select below	<code>X</code> : cut selected cells
<code>Ctrl-Enter</code> : run selected cells	<code>C</code> : copy selected cells
<code>Alt-Enter</code> : run cell and insert below	<code>Shift-V</code> : paste cells above
<code>Y</code> : change cell to code	<code>V</code> : paste cells below
<code>M</code> : change cell to markdown	<code>Z</code> : undo cell deletion
<code>R</code> : change cell to raw	<code>D, D</code> : delete selected cells

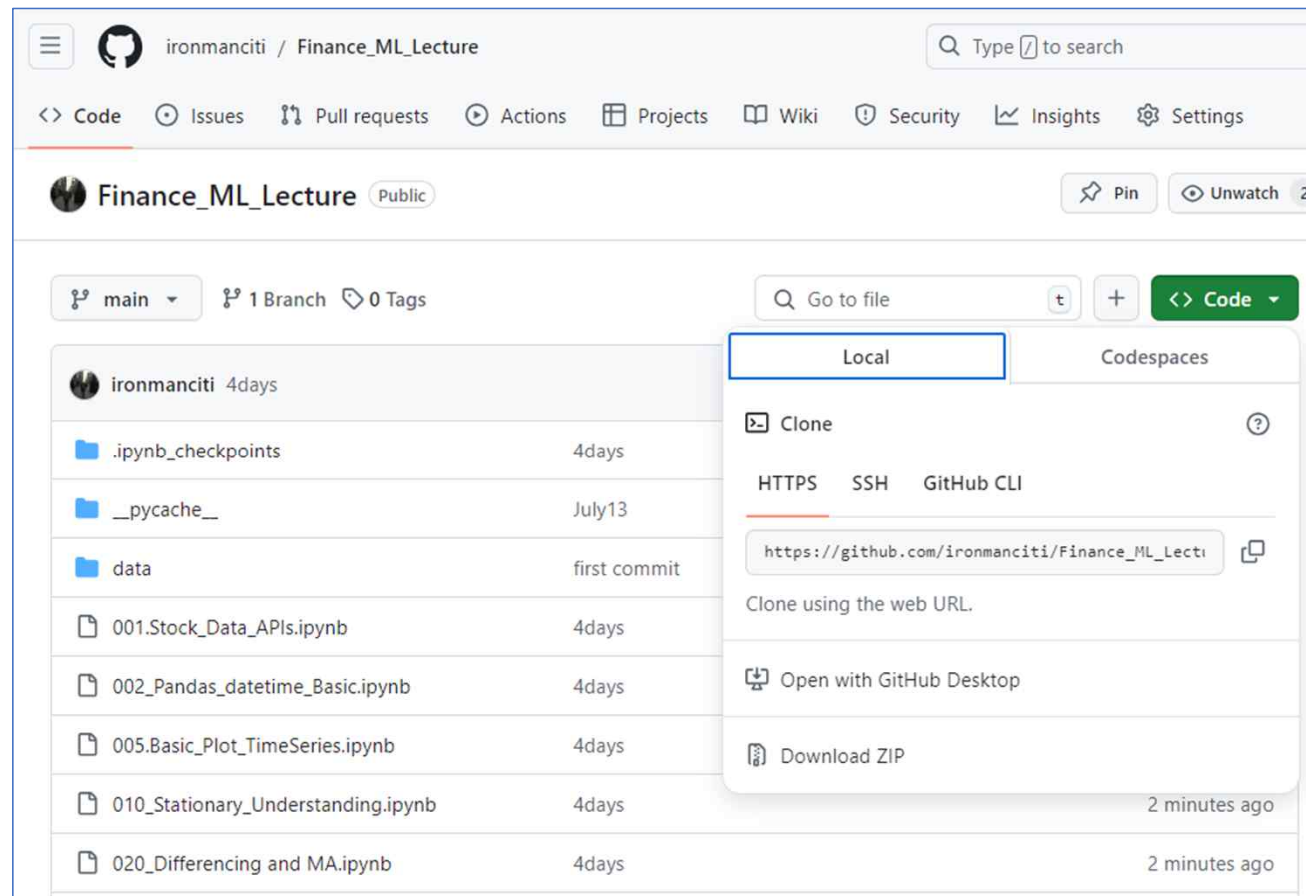
Tensorflow 설치

- pip install tensorflow

Anaconda Prompt (anaconda3)

```
(base) C:\Users\trimu>pip install tensorflow
```

교재 Source Download https://github.com/ironmanciti/Finance_ML_Lecture



Machine Learning in Finance

What is Financial Analysis (재무 분석) ?

- 비즈니스, 프로젝트, 예산 및 기타 재무 관련 거래를 평가하여 성과와 달성 수준을 측정
- 기업이 금전적 투자를 보증할 만큼 안정적인지, 유동성 또는 수익성이 있는지 분석하는 데 사용
- 경제 동향, 비즈니스 활동에 대한 장기 계획을 평가하고 투자할 프로젝트 또는 회사를 식별하는 데 사용
- 회사의 손익 계산서, 대차 대조표 및 현금 흐름표를 조사
- 기업 금융 및 투자 금융 환경 모두에서 수행

재무 분석의 종류

- 기술적 분석

- 가격 변동 및 거래량과 같은 거래 활동에서 수집한 통계적 추세를 분석하여 투자를 평가하고 거래 기회를 식별 (ex. 차트 분석)

- 펀더멘털 분석

- 관련 경제 및 재무 요인을 조사하여 유가 증권에의 내재 가치를 측정
- 펀더멘털 분석가는 경제 상황 및 산업 상황과 같은 거시경제적 요소에서 회사 경영의 효율성과 같은 미시 경제적 요소에 이르기까지 증권의 가치에 영향을 미칠 수 있는 모든 것을 연구 (ex. 계량 투자)

인공지능 응용 금융 분야

은행



- 고객 세분화
- 신용카드 연체
- 상환 불이행
- 금융 사기 탐지
- 자금 세탁
- 신용 등급 예측
- 추천 시스템, etc

자산 관리



- 시장 변화 감지
- 포트폴리오 최적화
- 파생 상품 거래, etc

퀀트 Trading



- 이익 예측
- 알고리즘 트레이딩

기술 분야 vs 금융 분야 ML 비교

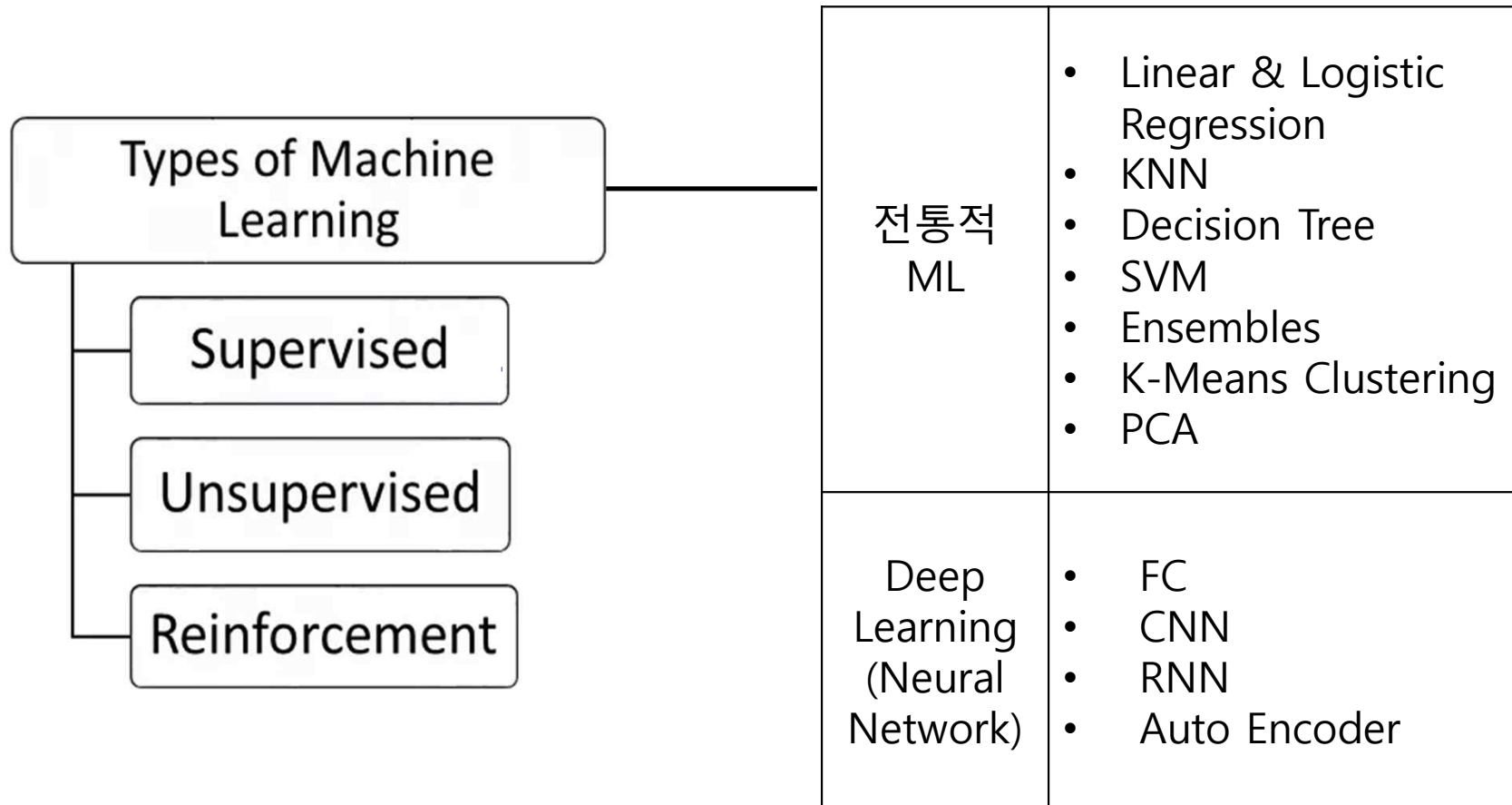
Tasks	기술 분야 ML	금융 분야 ML
Data Volume	Big Data	Medium/Small Data
Stationary Data ?	YES	Mostly No
Noisy	Low	High
설명 가능한 결과 ?	부수적 관심사	필요



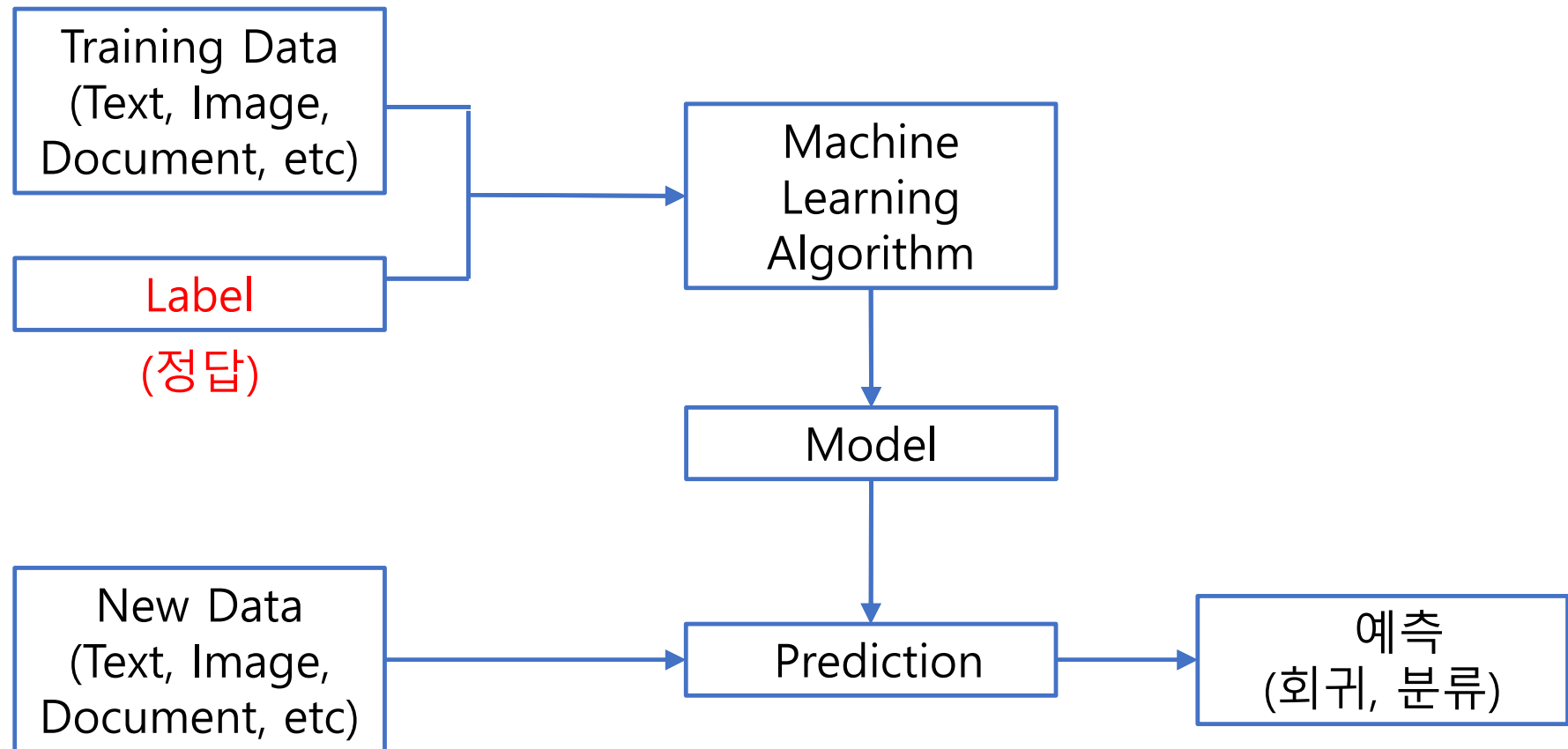
GDPR : General Data Protection Regulation

Machine Learning Models in Finance

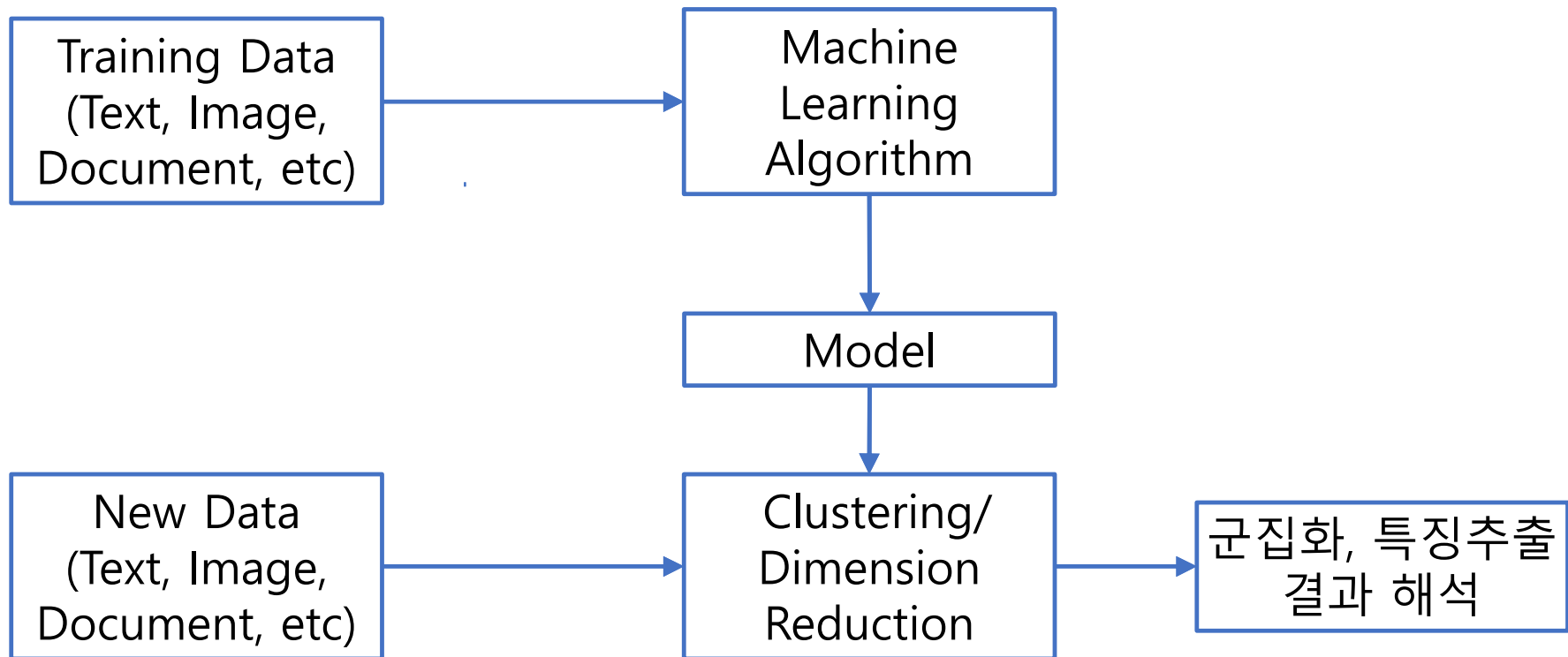
Machine Learning 기법의 종류



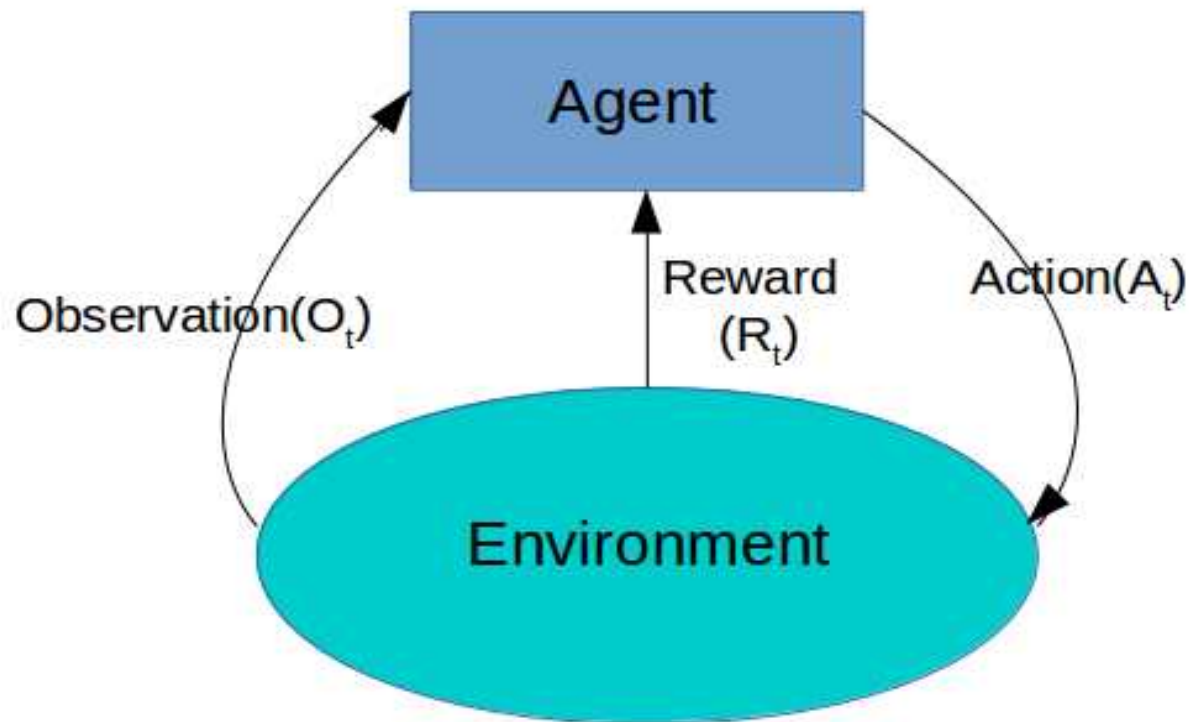
Supervised Learning (지도학습)



Unsupervised Learning (비지도학습)



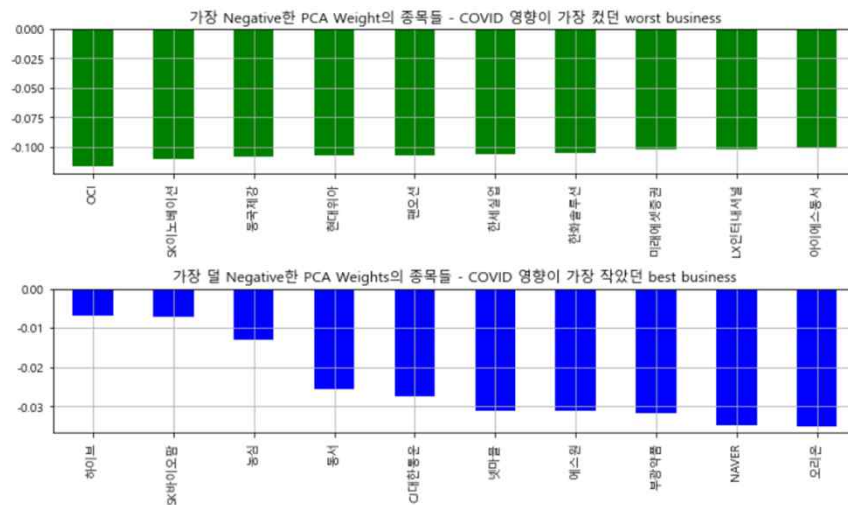
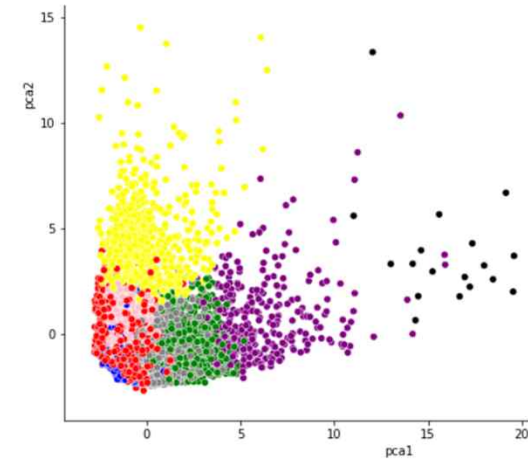
Reinforcement Learning (강화학습)



주가 예측

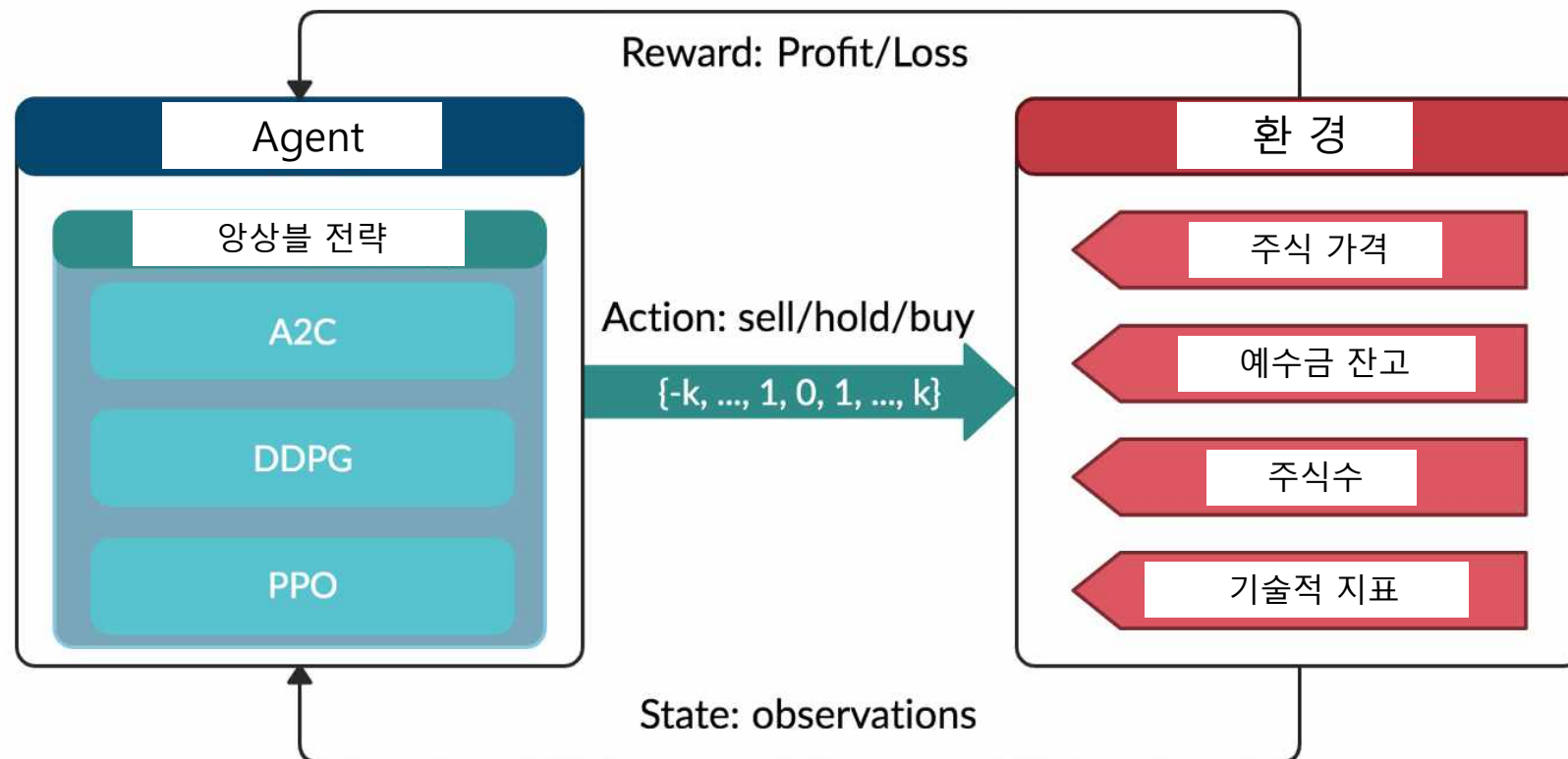


고객 segmentation



포트폴리오 구성

강화학습을 이용한 주식 자동 매매



실습 : 001. Stock Data APIs

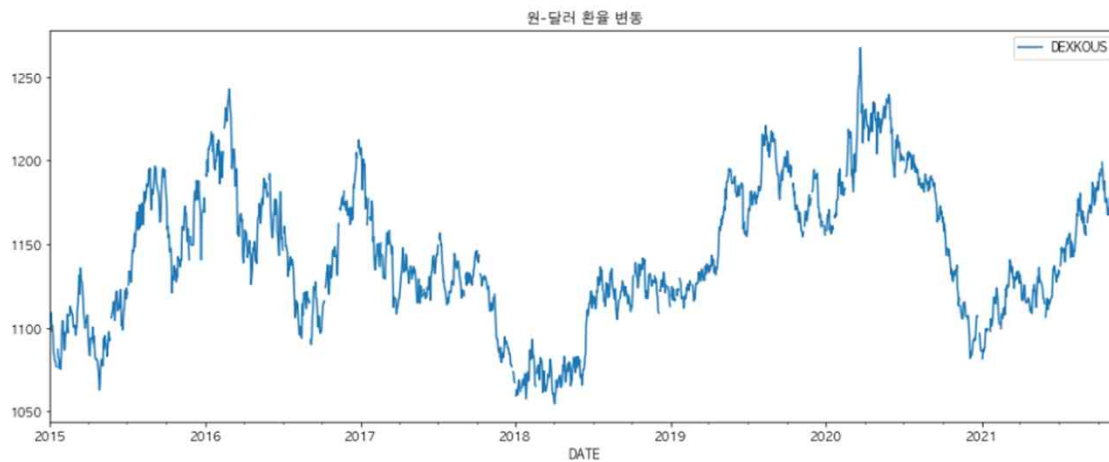
- Yahoo Finance
- Pandas data_reader
- FinanceDataReader

실습 : 002. Pandas Datetime 기본

- python, numpy, pandas 날짜 타입 비교 및 정리

라이브러리	날짜, 시간 클래스	타임델타 클래스
datetime	datetime, date, time	timedelta
numpy	datetime64	timedelta64
panads	Timestamp	Timedelta

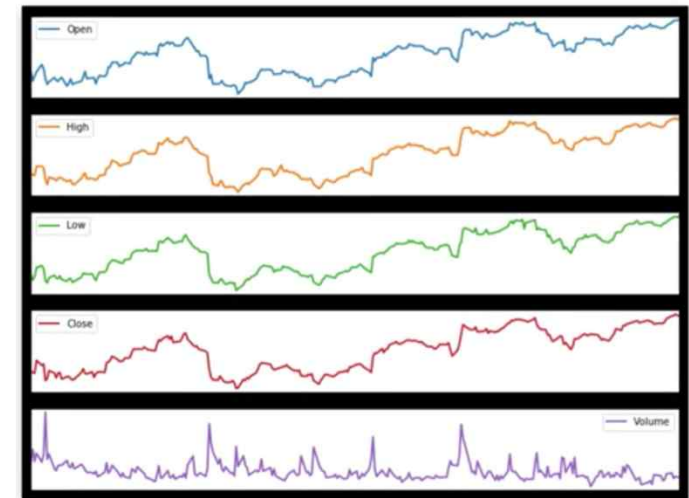
실습 : 005. Time Series data 시각화



What is
Time Series Data ?

What is Time-Series (시계열) Data ?

- Time Series는 일반적으로 시간이 지남에 따라 **일정한 간격**으로 정렬된 **값의 시퀀스**로 정의
ex) 매매동향, 주가, 일기예보 등
- **Time Series Forecasting**
 - history value 로 future value를 예측



Forecasting 응용 분야

- 에너지 산업
- 소매 산업
- 정부 예산, 계획 수립
- 금융 기관
- 농업
- 교육
- 여행업
- 기타

Why need different Approach ?

- 표준적인 regression 접근법은 time series model 에서 동작하지 않음

- Feature 와 Target 이 동일

이전 기간에 발생한 것이 다음 기간에 영향을 미침. 한 기간의 예측이 틀리면 그 이후 예측도 계속 틀림

- Data 가 time 에 대해 correlated 되어 있는 경우가 많음

- Non-stationary data – hard to model

대부분의 ML model이 stationary한 data를 가정하고 있음

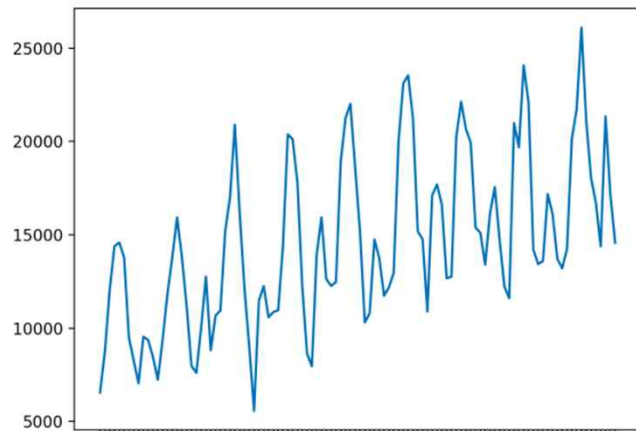
- 많은 data 필요

수년, 수십년간의 pattern을 파악하려면 장기간에 걸친 data 필요

단변수, 다변수 시계열

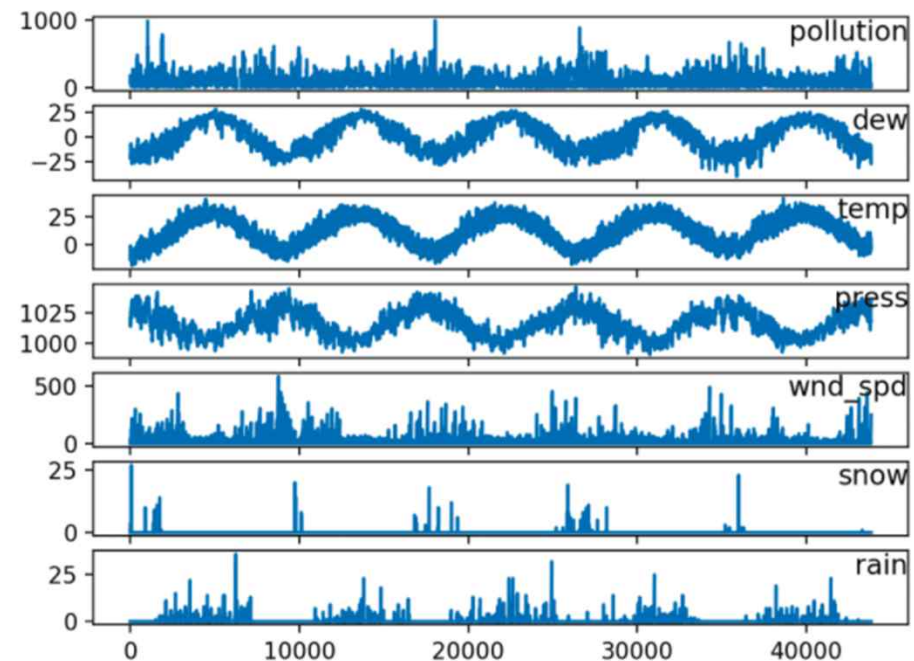
여러 변수 간의 상관
관계 파악에 유용

단변수 (univariate)



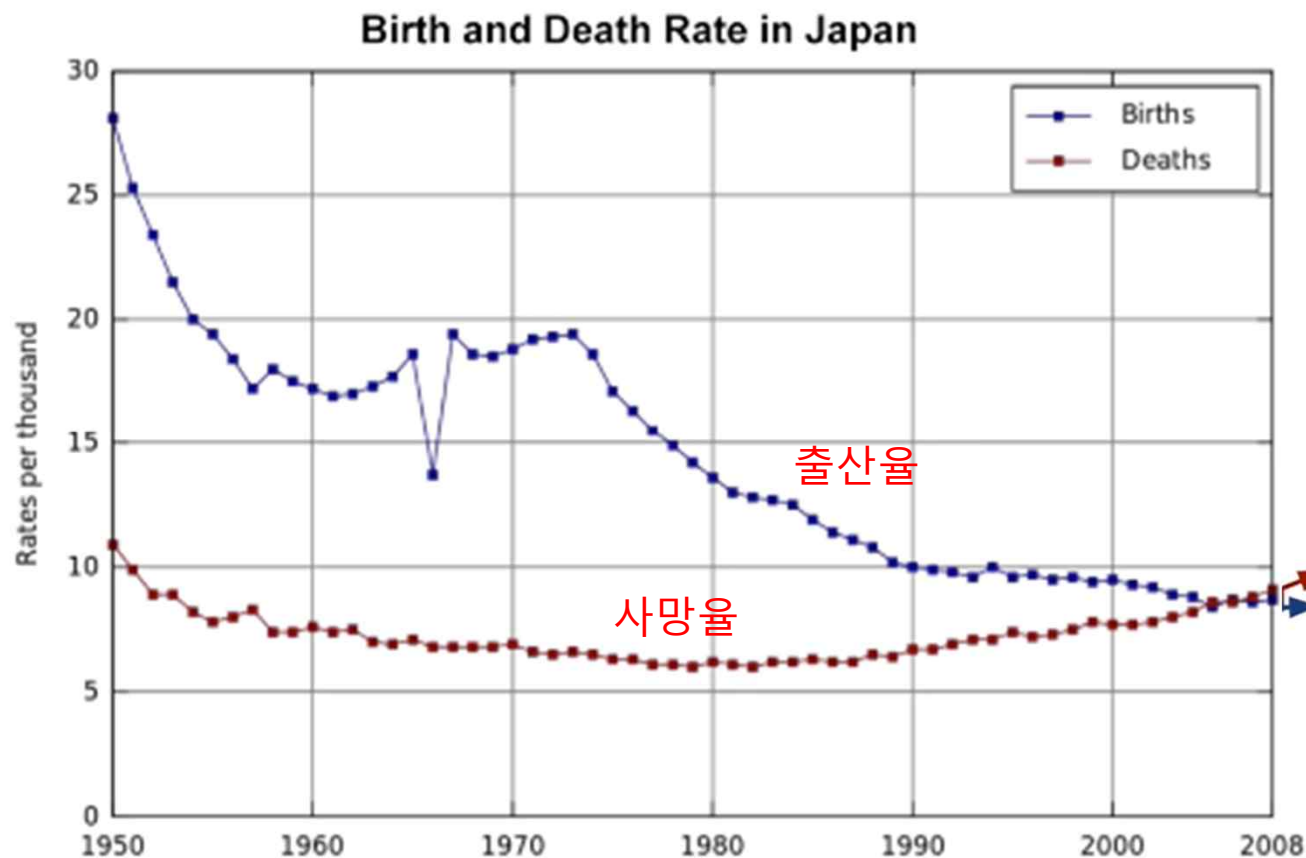
특성 time step 에 변수가 한 개

다변수 (multivariate)



특성 time step 에 변수가 여러 개

Multivariate Time Series Example



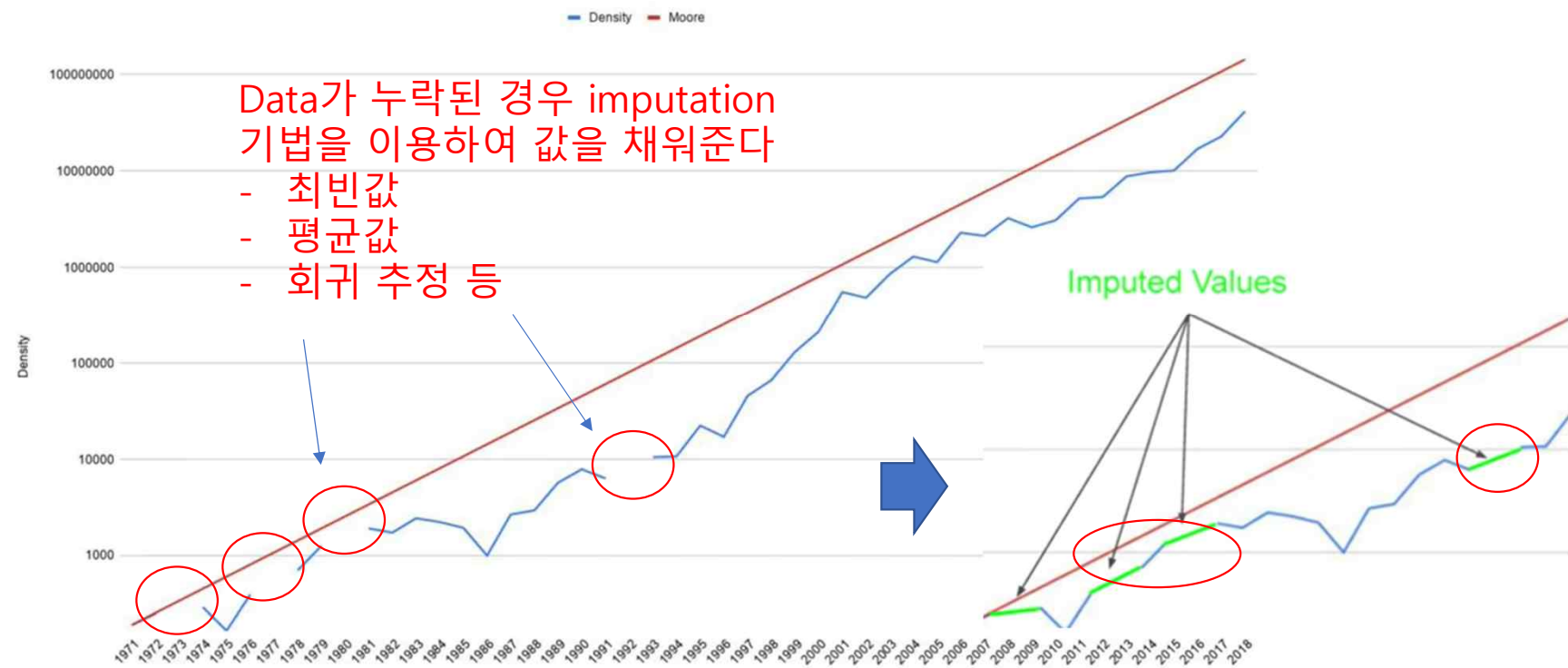
- 일본의 출산율과 사망률

1950년대에는 출생률이 압도적으로 높았으나 2000년대 중반 역전

두 변수 간의 상관 관계를 명확히 보여줌

Imputation (결측값 대체)

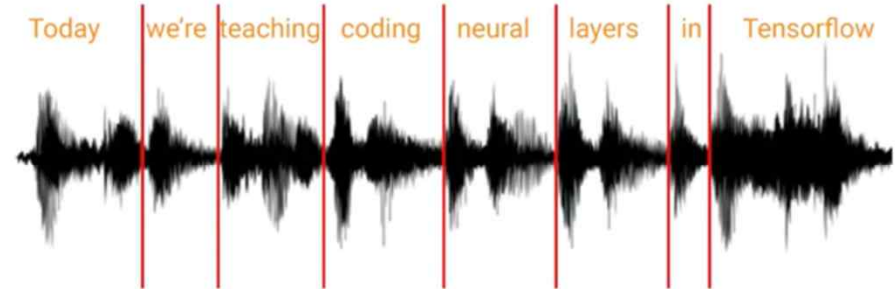
Density vs. Year



Machine Learning 적용 Example

Neural Network의
speech recognition 입력

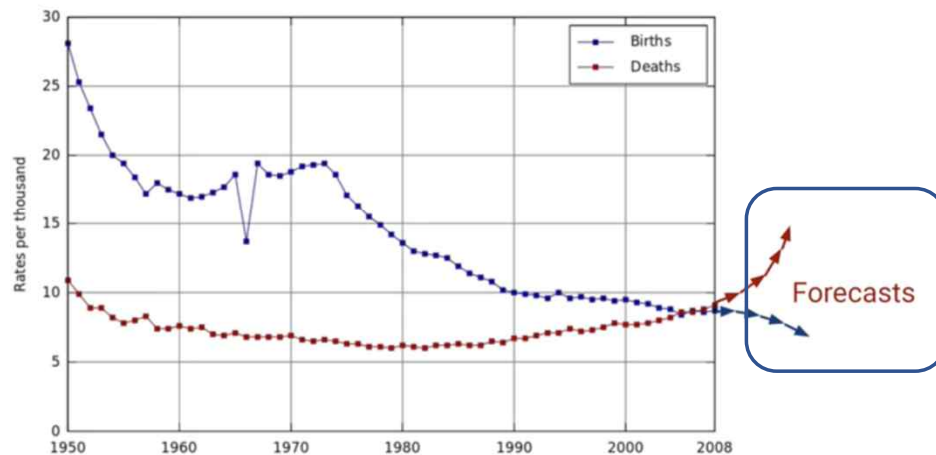
Time Series



음파가 단어 별로 구분됨

단어 구분

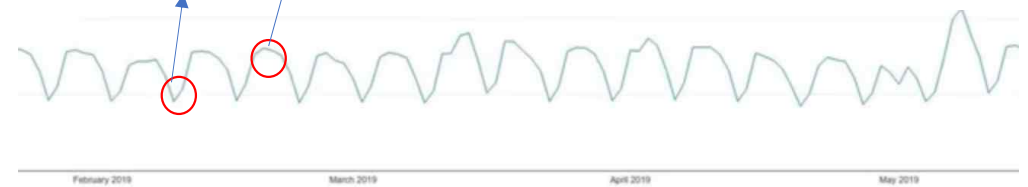
Birth and Death Rate in Japan



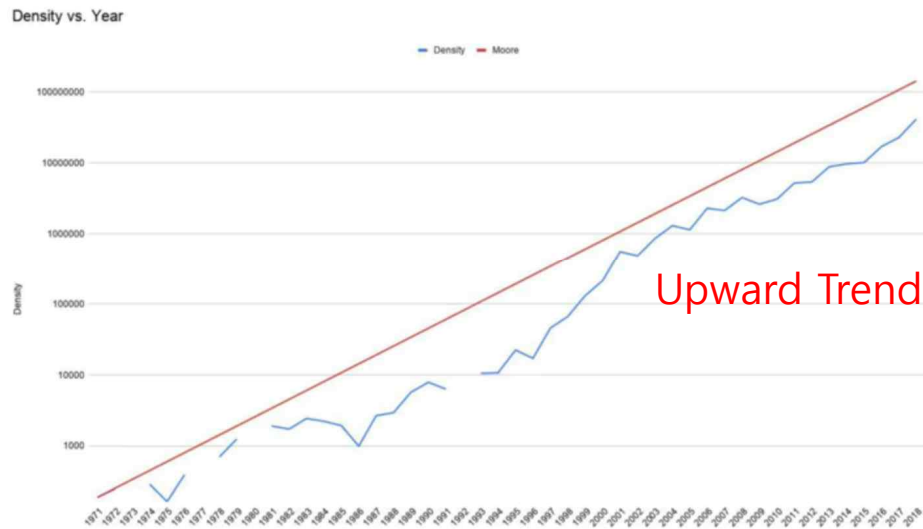
미래 값 예측

Time-Series 종류

예) 주말, 주중 전기 사용량 / 매출액 변화 등

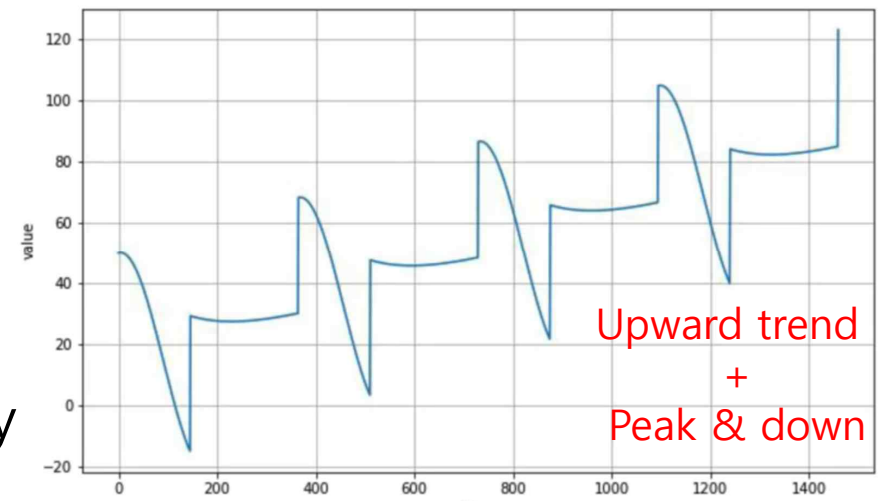


Seasonality
(예측 가능한 간격으로 반복)
Predictable interval repeat



Trend (일정한 방향성)
long-term direction

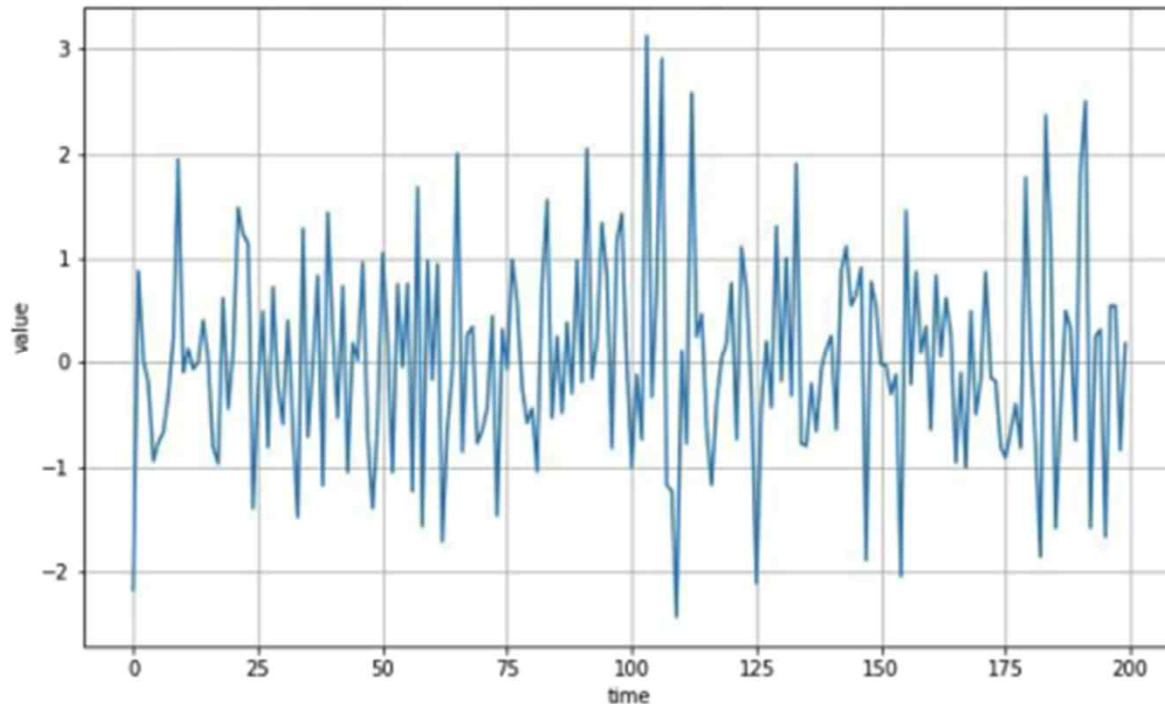
Trend + seasonality



Time-Series 종류

Random Values (White noise)

residual – irregular fluctuation



전혀 예측할 수 없음

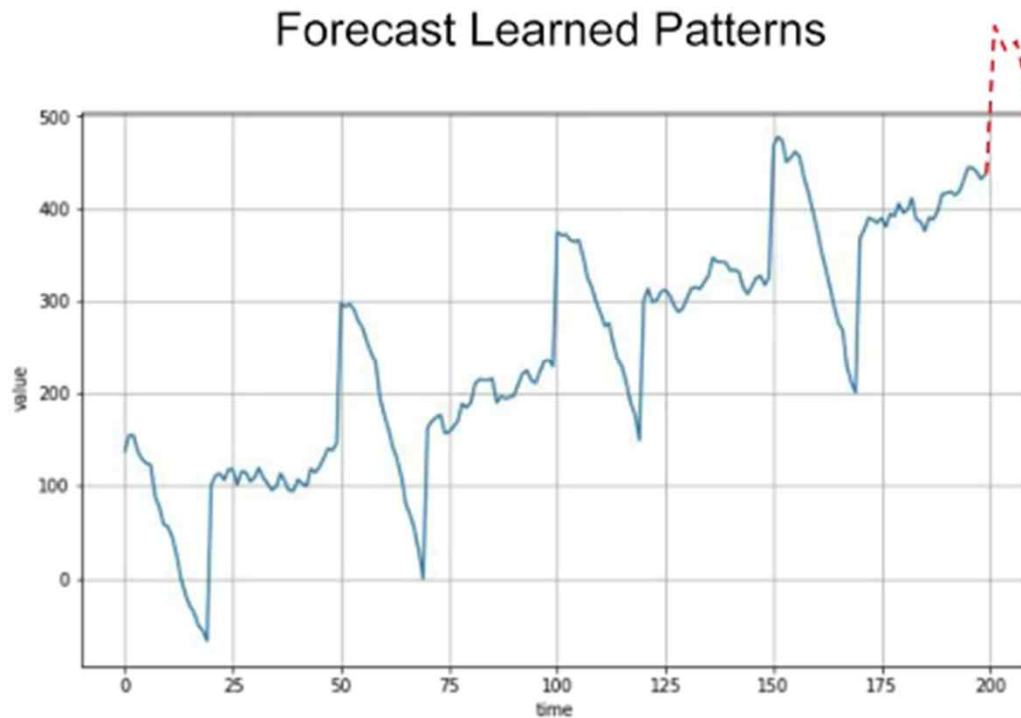
- 예측 불가 - 순전히 무작위적인 성격의 시리즈
- Mean = 0
일정한 분산. 상관성 없음
- 평균은 이 시리즈의 가장 좋은 예측

Autocorrelation(자기 상관)

- Autocorrelation 은 time series 의 key concept 이다.
- 현재(today)의 측정값은 과거 값(past value)에 highly dependent 하다.
- Correlated value 간의 time interval 을 lag 라고 부른다.
- 예) 주가는 이전 data 와 correlate 되어 있음
(전일 : lag = 1, 이틀 전: lag = 2)

실세계에서 자주 만날 수 있는 복합된 형태의 시계열 data

Trend + Seasonality + Autocorrelation + Noise

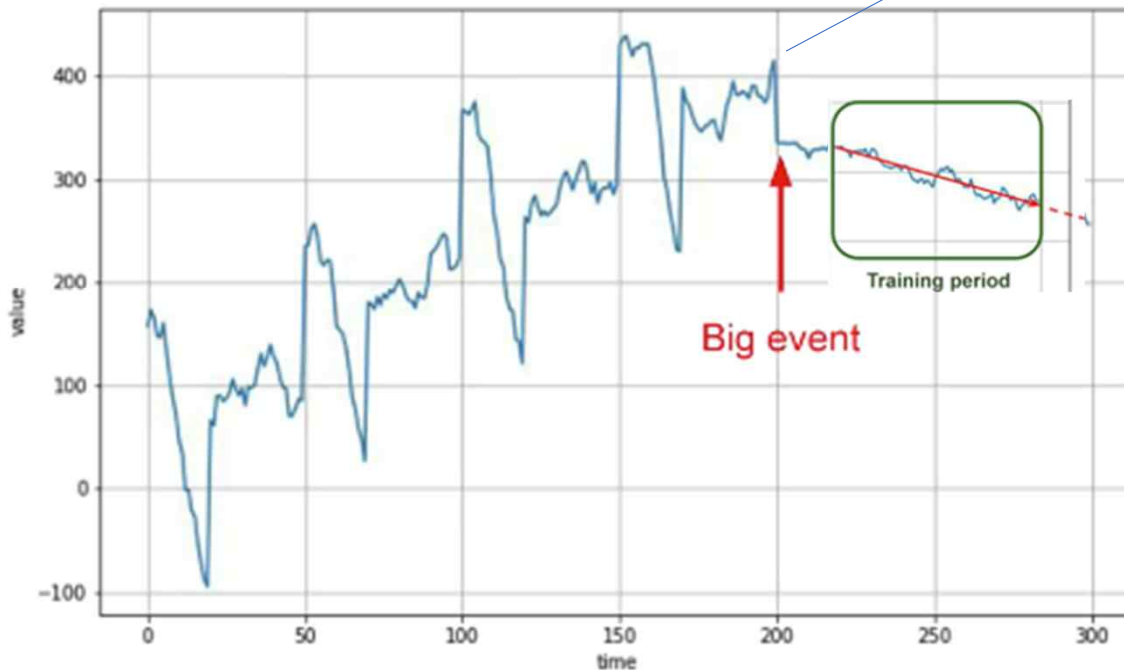


Noise를 제외하고는 Machine Learning
으로 pattern 학습 및 예측 가능.

단, 과거의 패턴이 미래에도 계속 된다는
가정을 전제로 함.

이전에는 uptrend + seasonality + autocorrelation + noise

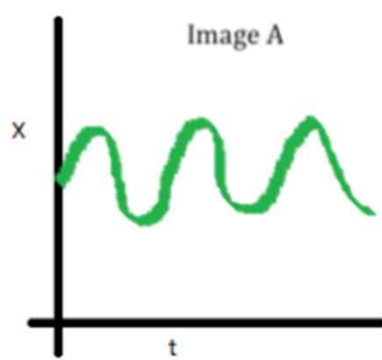
Non-Stationary Time Series



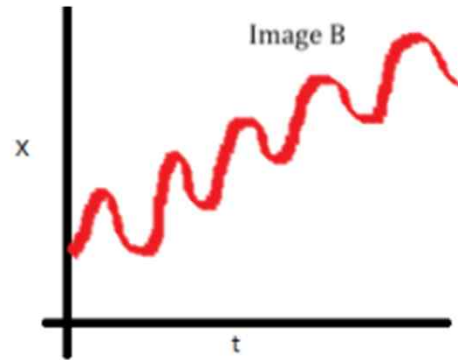
- 시간 단계 200 이후 동작이 완전히 바뀜
- 금융 위기, 파괴적인 기술 혁신 등
- 이후 시계열은 뚜렷한 계절성 없이 하향 추세
- 일반적으로 이것을 **비정상 시계열**이라고 부름
- ML Training의 기간에 제한을 두어 예측
(비정상 시계열은 기간별로 train 시켜야 한다.)

- Stationary Time-Series : 데이터가 많을수록 좋다.
- Non-Stationary Time-Series : 최적 Time Window 가 다양하게 변함.

Stationary (정상성) vs non-stationary (비정상성)

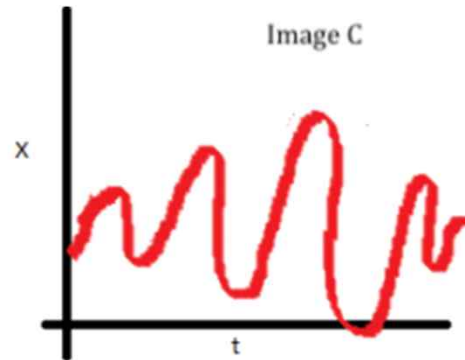


stationary



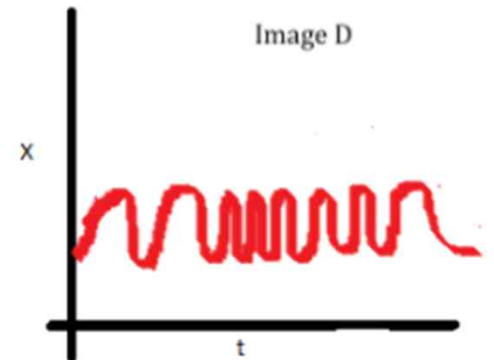
non-stationary

평균이 시간에
따라 변화



non-stationary

분산이 시간에
따라 변화

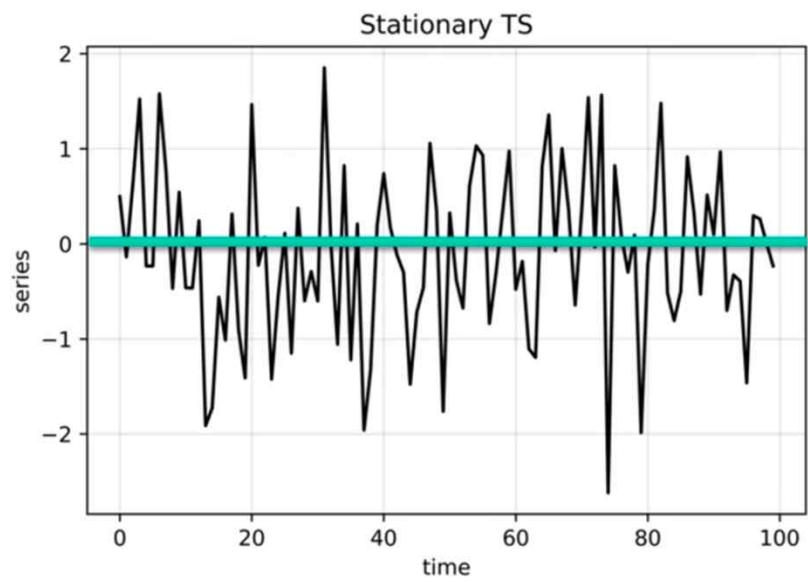


non-stationary

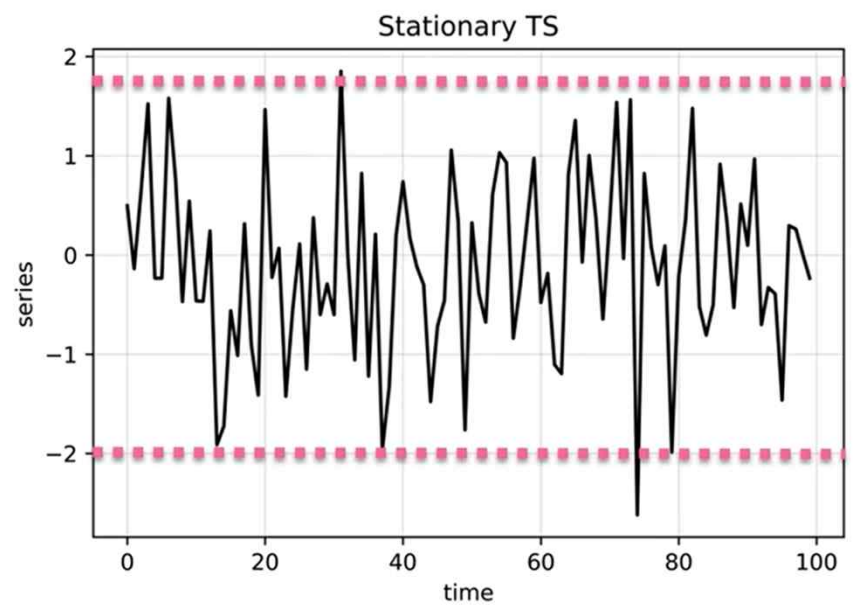
공분산이 시간에
따라 변화



정상 프로세스 : 시간에 관계 없이 평균과 분산이 일정한 시계열 데이터

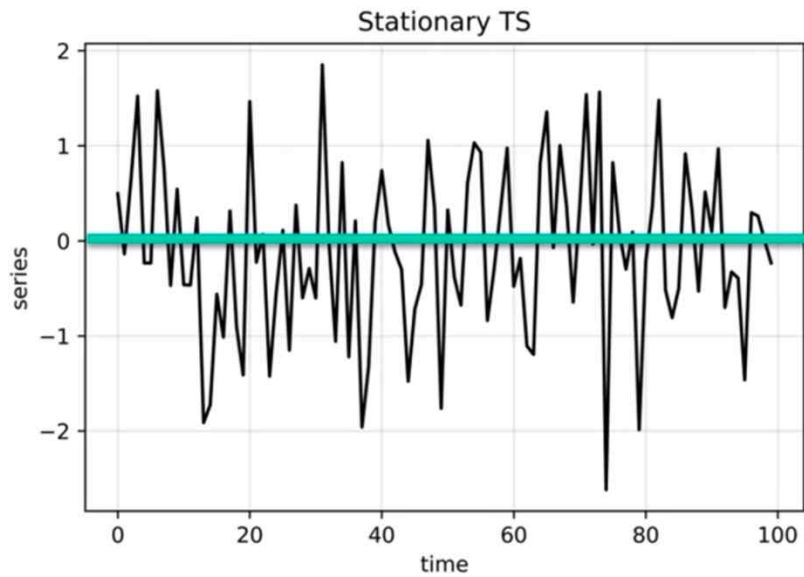


constant mean

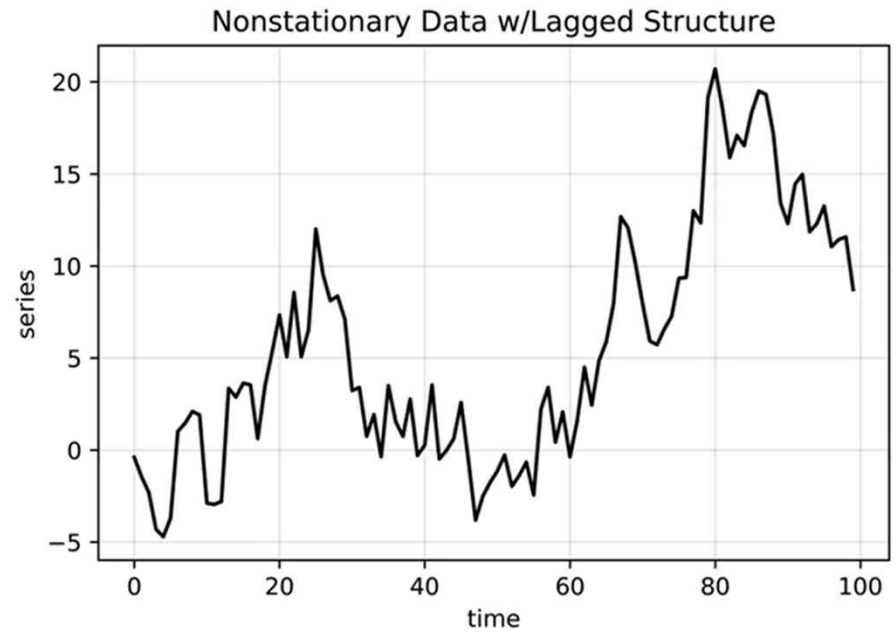


constant variance

Autocorrelation이 아주 강한 경우
→ 일정한 평균을 유지 못함




constant mean



No constant mean

간단한 transformation을 거쳐
stationary series로 변환 가능

Why stationarity (정상성) is important ?

- Time Series modeling의 중요 조건 → 통계학적 기법들은 모두 정상성을 전제로 함
 - model 의 parameter와 구조가 시간 흐름에 stable 해야 예측 가능
 - data 가 **stationary** 해야 AR(Auto-Regression) 및 MA(Moving Average) 에 사용되는 average가 time series 의 behavior 를 설명할 수 있다.
- 

Common approach

- 1) 큰 value를 squash하여 variance를 작게 만듦
- 2) Log transformation 적용

Step 1 - Non-stationary 의 source 구분

Step 2 - Time series 를 **stationary 로 transform**

- Trend 제거 (constant mean)
- 이분산성 제거 (constant variance)
- differencing 으로 autocorrelation 제거
- 계절성 제거 (no periodic component)

Step 3 - Stationary series 로 model 구축

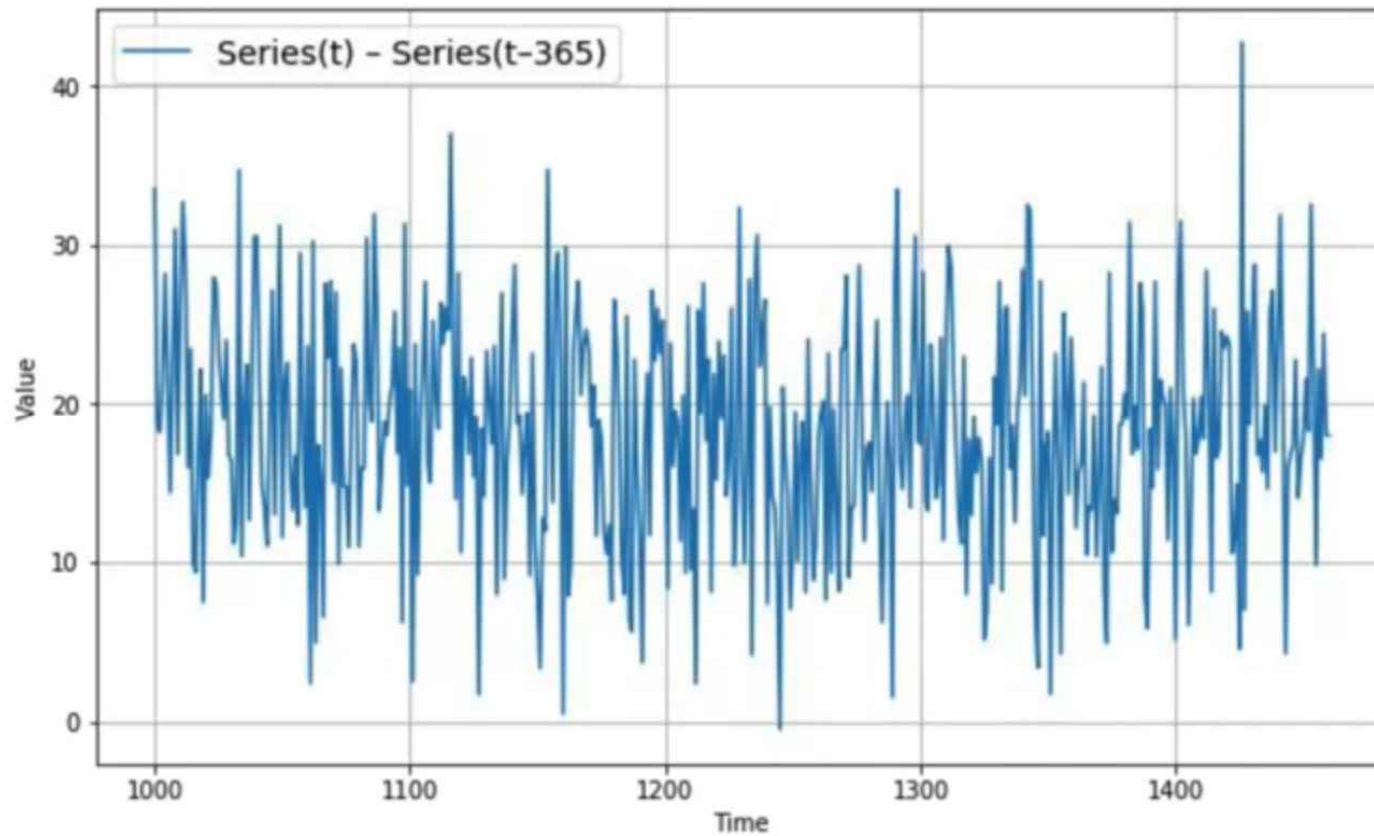


실습: 010. Stationary Time Series(정상 시계열)

- What is Stationarity (정상성)?
- Autocorrelation Structure (자기 상관 구조)
- non-stationary (비정상) data의 특징
- Differencing 으로 Autocorrelation 제거

How to make Time Series
Stationary ?

Differencing(차분) – Trend, Seasonality 제거



Differencing (차분)

- Non-stationary 한 data 를 stationary 하게 변환
- Differencing 한 data 에 대해 stationary 할 때까지 differencing 반복
- Seasonal data 의 경우 season 을 기준으로 differencing
 - Ex) 1 년 주기의 seasonality 를 갖는 월간 data 에 대해 differencing 할 때, differencing 의 시간 단위 는 12 (12차 shifting)

1차, 2차, 3차 차분

Differencing (차분)

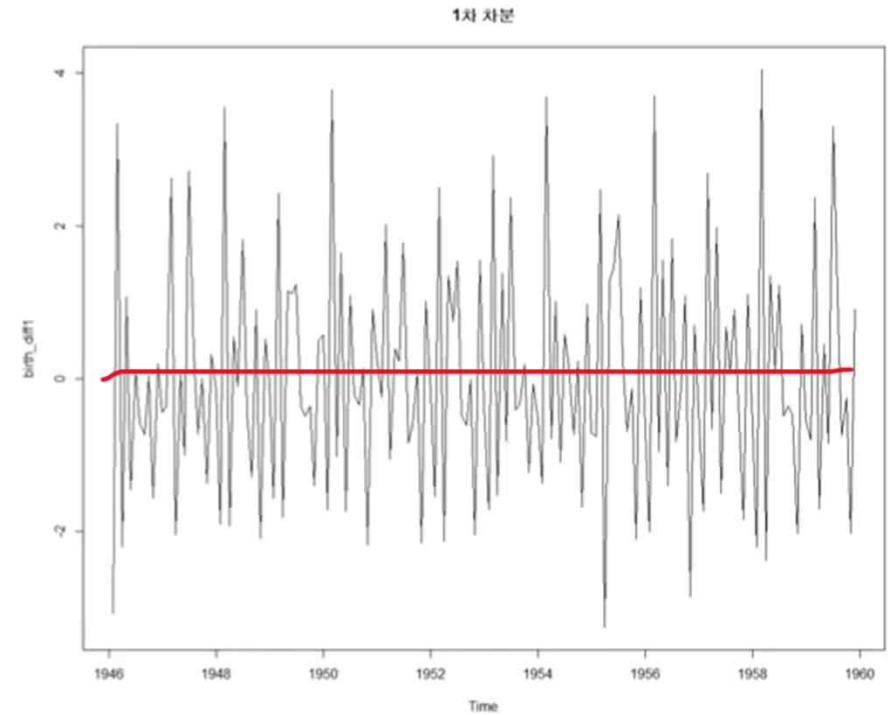
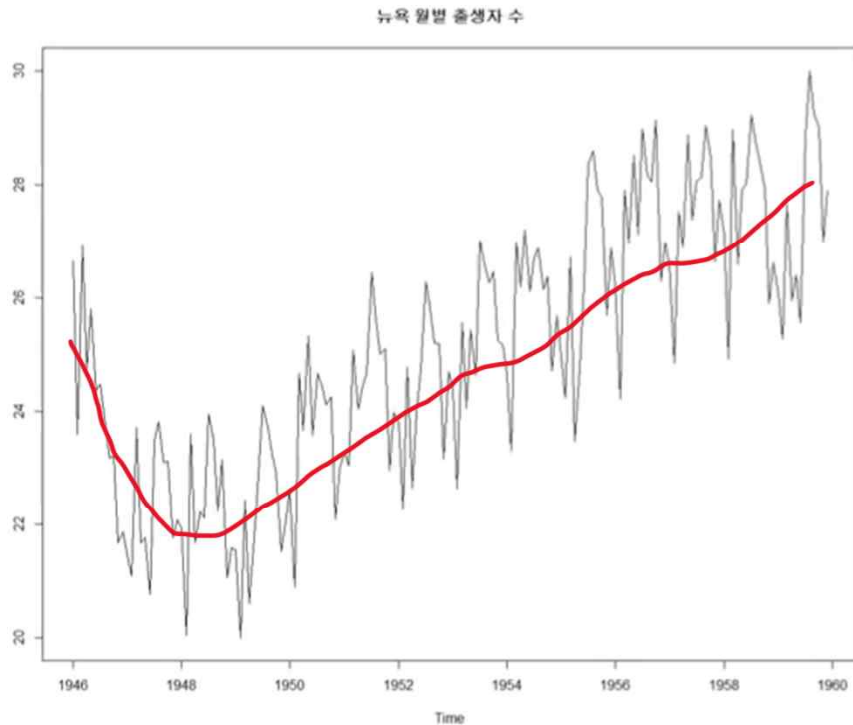
Lag 2 Difference (2차 차분)

$$Y_t - Y_{t-d}$$

	Price	Difference1	Difference2
0	10	NaN	NaN
1	12	2.0	NaN
2	8	-4.0	-6.0
3	14	6.0	10.0
4	7	-7.0	-13.0

Lag 1 Difference (1차 차분)

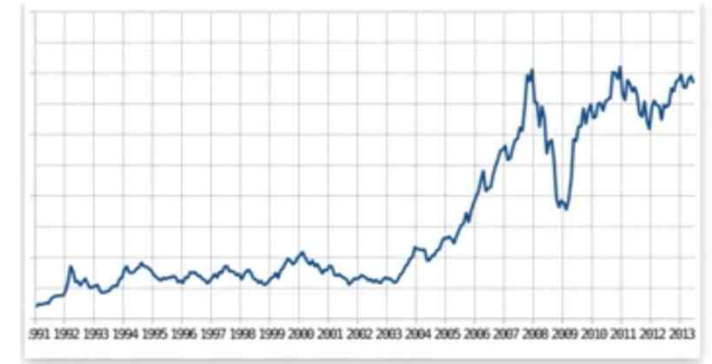
1st Differencing (1차 차분)



평균 0, 분산 일정

How to change stock prices stationary ?

- 주가는 상향, 하향 trend 가 있는 것이 일반적이고, 평균도 시간에 따라 바뀜
(ex. 시간에 따라 평균값 증가) **Non-stationary**
- 따라서, 현재 주가에서 전일, 전월 혹은 전년 주가를 차감하여 stationary 하게 만든다.



변동을 예측



Price Difference 구하기 – 종가를 1day shift

- `df['Yesterday Close'] = df['close'].shift(1)`

Date	High	Low	Open	Close	Volume	Adj Close	Yesterday Close
2017-01-17	128.339996	127.400002	128.039993	127.870003	15294500	127.870003	NaN
2017-01-18	128.429993	126.839996	128.410004	127.919998	13145900	127.919998	127.870003
2017-01-19	128.350006	127.449997	128.229996	127.550003	12195500	127.550003	127.919998
2017-01-20	128.479996	126.779999	128.100006	127.040001	19097200	127.040001	127.550003
2017-01-23	129.250000	126.949997	127.309998	128.929993	16593600	128.929993	127.040001

1 day shift



Price Difference 구하기 - 일별 수익

- PriceDiff = (당일 종가 - 전일 종가)
- `df['Price Difference'] = df['close'] - df['Yesterday Close']`
- `df['Price Difference'] = df['close'].diff()`

Date	High	Low	Open	Close	Volume	Adj Close	Yesterday Close	Price Difference
2017-01-17	128.339996	127.400002	128.039993	127.870003	15294500	127.870003	NaN	NaN
2017-01-18	128.429993	126.839996	128.410004	(1) 127.919998	13145900	127.919998	(2) 127.870003	0.049995
2017-01-19	128.350006	127.449997	128.229996	127.550003	12195500	127.550003	127.919998	-0.369995
2017-01-20	128.479996	126.779999	128.100006	127.040001	19097200	127.040001	127.550003	-0.510002
2017-01-23	129.250000	126.949997	127.309998	128.929993	16593600	128.929993	127.040001	1.889992

(1)-(2)

Return (Daily/Monthly/Yearly Return)

- Return – 일정기간 동안의 수익, 손실율
- $df['Return'] = df['Close'] / df['Close'].shift(1) - 1$
- `df['Close'].pct_change()`

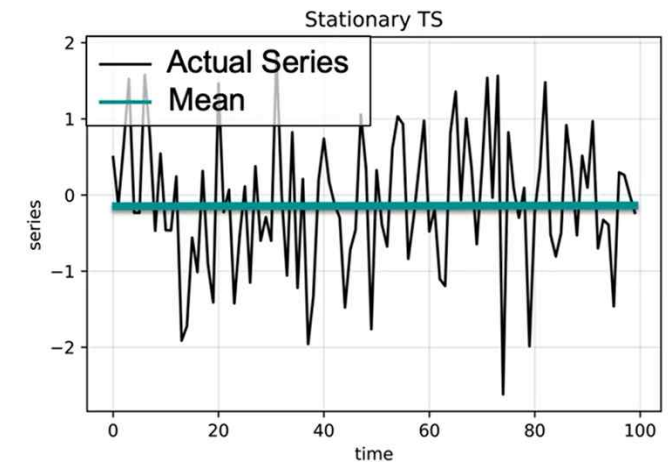


Date	High	Low	Open	Close	Volume	Adj Close	Yesterday Close	Price Difference	Return
2017-01-17	62.700001	62.029999	62.680000	62.529999	20664000.0	58.139027	NaN	NaN	NaN
2017-01-18	62.700001	62.119999	62.669998	(1) 62.500000	19670100.0	58.111130	(2) 62.529999	-0.029999	-0.000480
2017-01-19	62.980000	62.200001	62.240002	62.299999	18451700.0	57.925182	62.500000	-0.200001	-0.003200
2017-01-20	62.820000	62.369999	62.669998	62.740002	30213500.0	58.334286	62.299999	0.440002	0.007063
2017-01-23	63.119999	62.570000	62.700001	62.959999	23097600.0	58.538834	62.740002	0.219997	0.003506

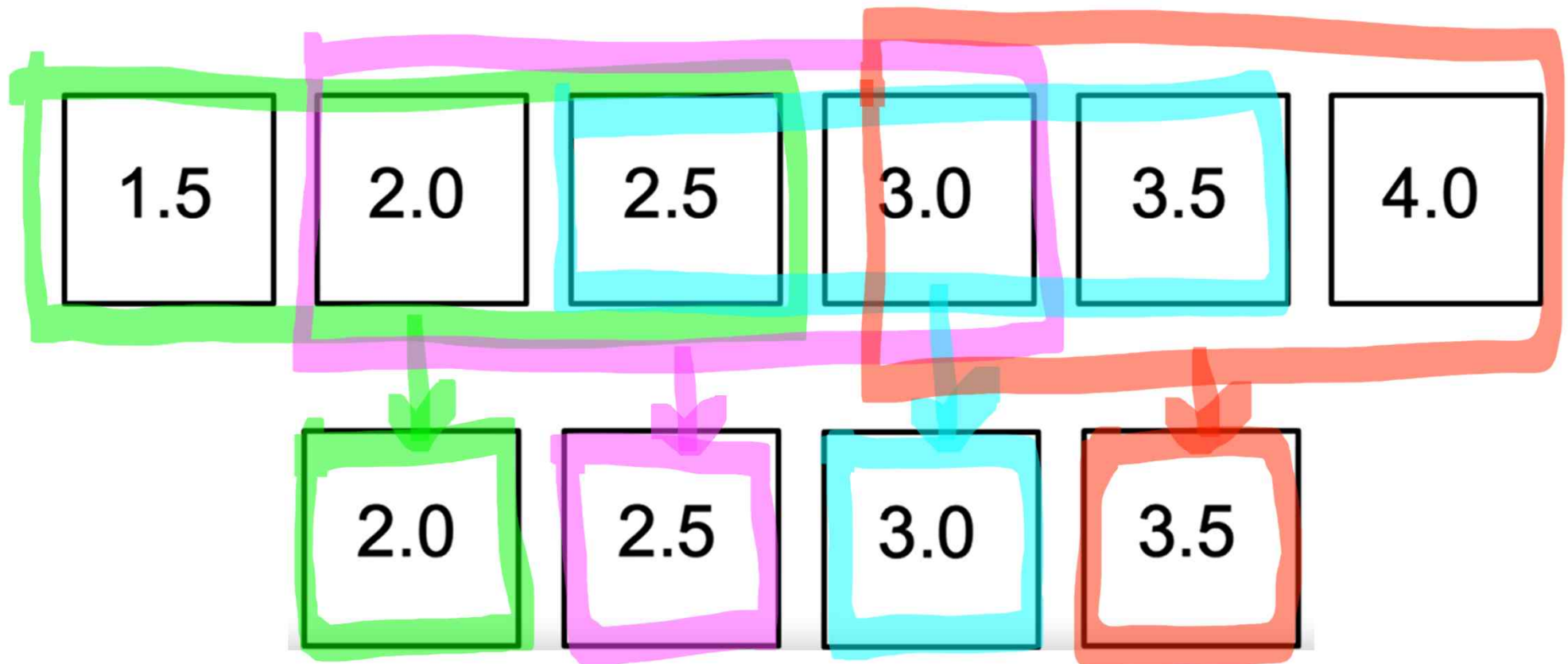
(1)/(2) - 1

Time Series Smoothing (시계열 평활화)

- Smoothing - noise 의 영향을 줄이는 방법
- smoothing 방법
 - Simple average smoothing →
 - Equally weighted moving average
 - Exponentially weighted moving average





Equally-Weighted Moving Average



Equally-Weighted Moving Average (이동 평균)

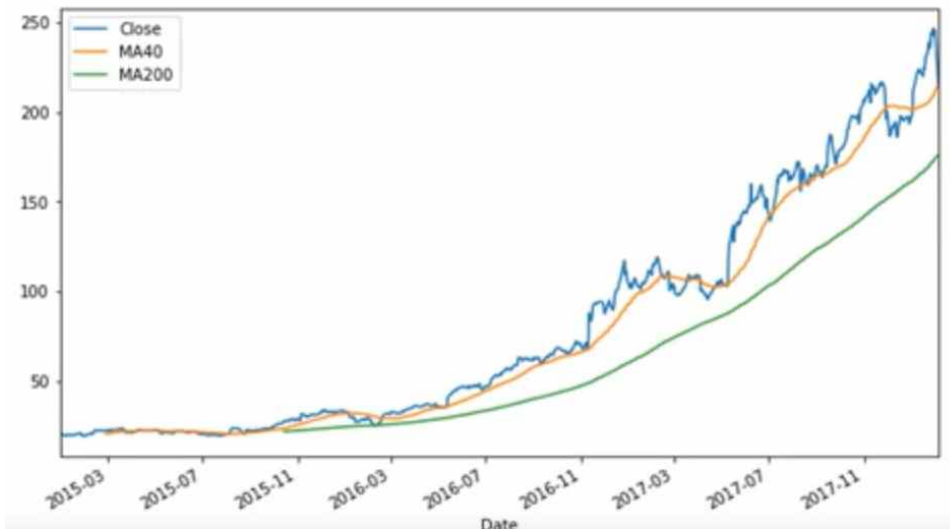
- $df['average'] = (df['close'] + df['close'].shift(1) + df['close'].shift(2)) / 3$

Date	Close		Date	Close		Date	Close
2014-12-31	20.04999		2014-12-31	NaN		2014-12-31	NaN
2015-01-02	20.12999		2015-01-02	20.04999		2015-01-02	NaN
2015-01-05	19.79001	shift(1)	2015-01-05	20.12999	shift(2)	2015-01-05	20.04999
2015-01-06	19.19001		2015-01-06	19.79001		2015-01-06	20.12999

Moving Average – 중기, 장기 이동 평균

- `df['MA30'] = df['close'].rolling(30).mean()`
- `df['MA200'] = df['close'].rolling(200).mean()`

```
fb['Close'].plot()  
fb['MA40'].plot()  
fb['MA200'].plot()
```



EWMA (Exponentially-weighted moving average)

- SMA(Simple Moving Average)의 약점
 - Window 가 작을수록 신호가 아닌 잡음이 증가
 - 항상 window 크기만큼 지연
 - 평균화로 인해 데이터의 전체 피크 또는 계곡에 도달하지 않음
 - 미래의 움직임에 대해 실제로 알려주지 않음. 실제로는 데이터의 trend를 묘사
 - 극단적인 historical value 로 인해 SMA가 크게 왜곡 될 수 있다.
- 이러한 문제 해결을 위해 EWMA (지수 가중 이동 평균)를 사용

Exponentially-Weighted Moving Average

- 지수가중이동평균
- 현재를 기준으로 오래된 값은 가중치를 낮게 부여하고, 최근 값은 가중치를 높게 부여해서 평균값을 도출
- 가중치는 Exponential Function(지수함수)에 근거하여 도출

$$y_t = \frac{\sum_{i=0}^t w_i x_{t-i}}{\sum_{i=0}^t w_i}$$

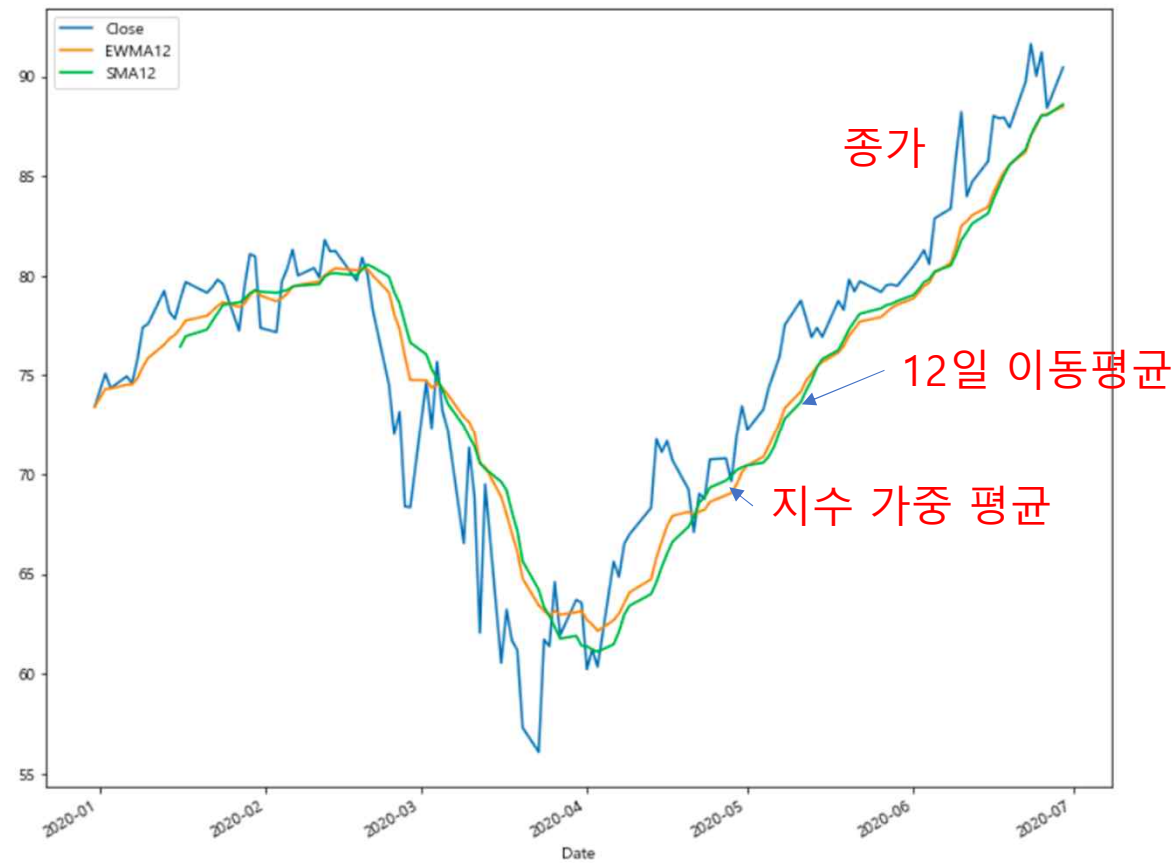
x_t - 입력값 w_i - 적용된 weight y_t - 산출된 값

EWMA (지수 가중 이동 평균)

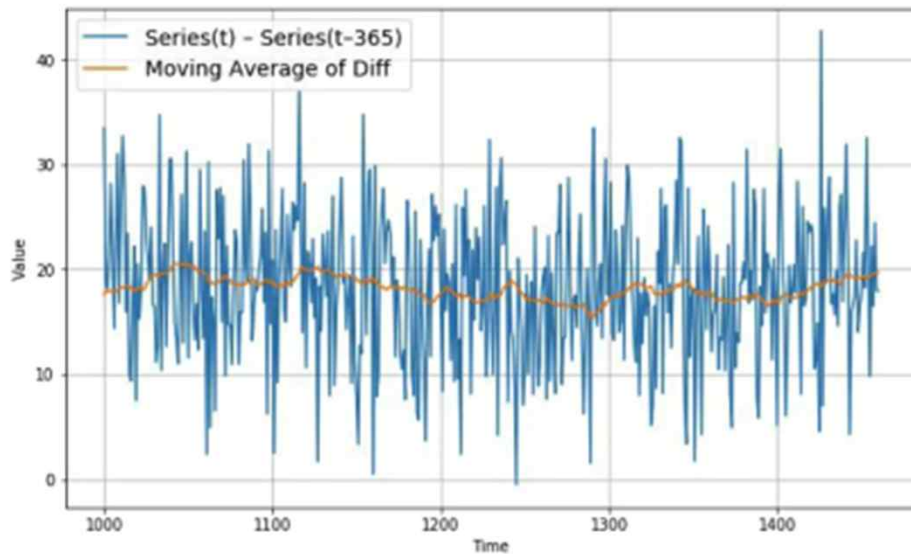
- `df['EWMA12'] = df['Close'].ewm(span=12).mean()`

	Close	SMA6	SMA12	EWMA12
Date				
2020-06-23	91.632500	88.771666	87.052082	87.036708
2020-06-24	90.014999	89.104167	87.606249	87.494907
2020-06-25	91.209999	89.656250	88.040624	88.066459
2020-06-26	88.407501	89.735416	88.057082	88.118927
2020-06-29	90.445000	90.237916	88.596249	88.476785

- Simple Moving Average 와 Exponentially Weighted Moving Average 비교



Differencing한 시계열의 Moving Average



- 추세나 계절성을 제거한 후 이동 평균 계산
- 시계열 자체 보다 time t 와 time t-1 의 value 차이를 주목. t는 연, 월, 일 모두 가능.

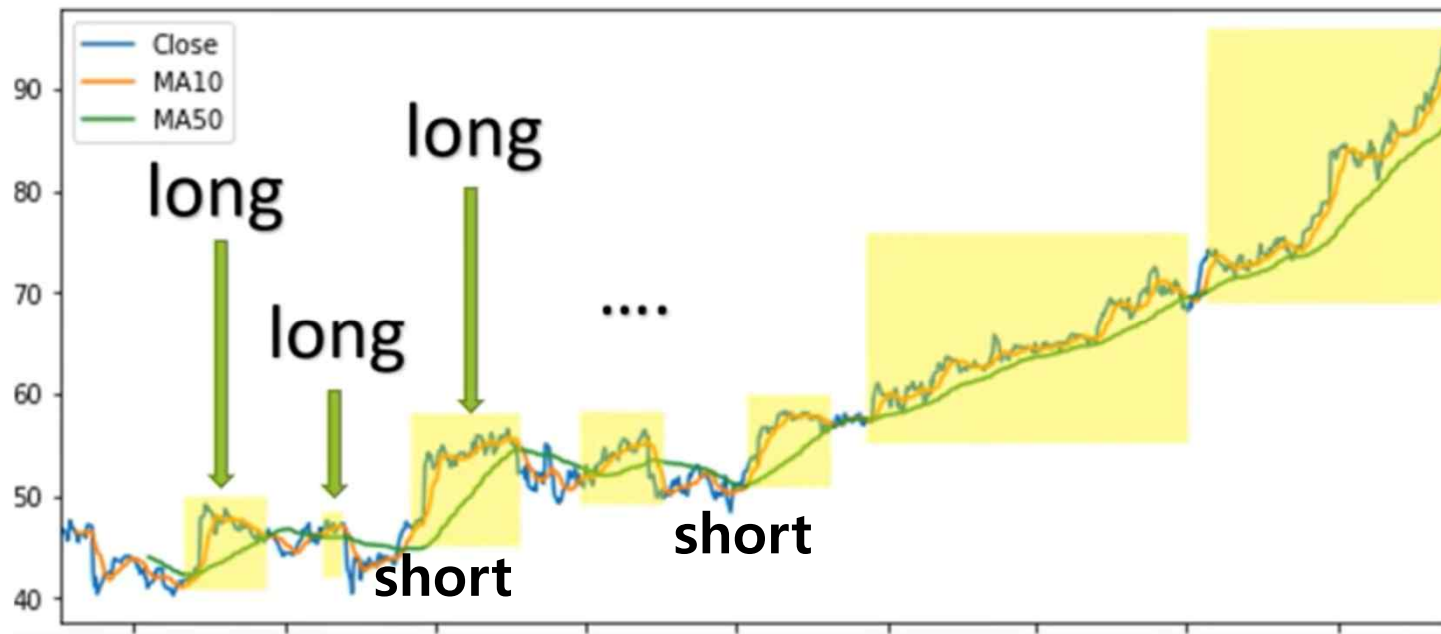
실습: 020. Differencing(차분)과 MA(이동평균)

- Price Difference 계산
- 일일 수익률 계산
- Price Up/Down Direction 생성
- 이동 평균 작성

이동평균선을 이용한 단순 매매 전략

- 장단기 이동 평균선을 이용한 Simple Trading Strategy

$MA10 > MA50 \rightarrow 1$ 주 보유, $MA10 < MA50 \rightarrow 0$ 주 보유



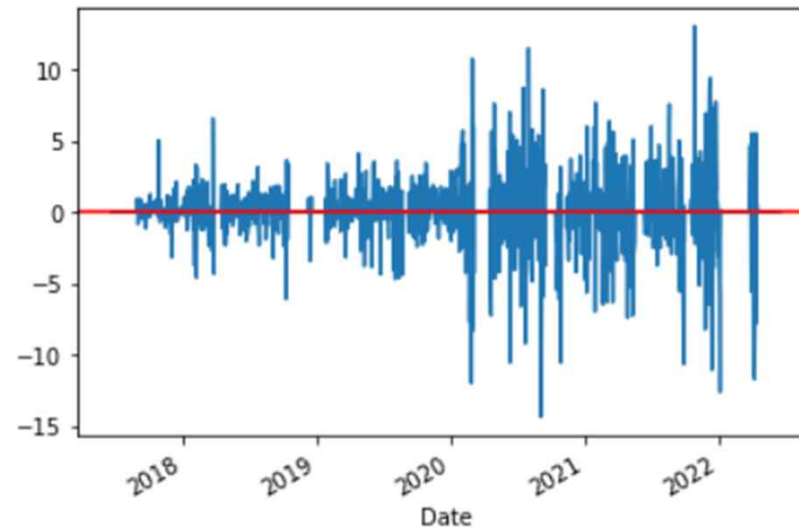
- Daily Profit(일별 수익) 계산 – step 1

- 당일 보유 주식에 있으면 (익일 종가 - 당일 종가) 를 수익으로 계산

- $df['Profit'] = [df.loc[idx, 'close1'] - df.loc[idx, 'close']$
if $df['Share'] == 1$ else 0 for idx in $df.index$]

수익 →

손실 →



- 누적 수익 (Cumulative Profit) 계산

- `df['wealth'] = df['Profit'].cumsum()`

	Close	MA10	MA50	Shares	NextDay Close	Profit	wealth
Date							
2022-06-13	242.259995	266.059000	275.324799	0	244.490005	0.0	82.299988
2022-06-14	244.490005	263.321001	274.026198	0	251.759995	0.0	82.299988
2022-06-15	251.759995	261.254999	272.761998	0	244.970001	0.0	82.299988
2022-06-16	244.970001	258.294000	271.443798	0	247.649994	0.0	82.299988
2022-06-17	247.649994	256.057001	270.406798	0	NaN	0.0	82.299988

실습 : 021. 이동평균선 활용 단순 매매 전략

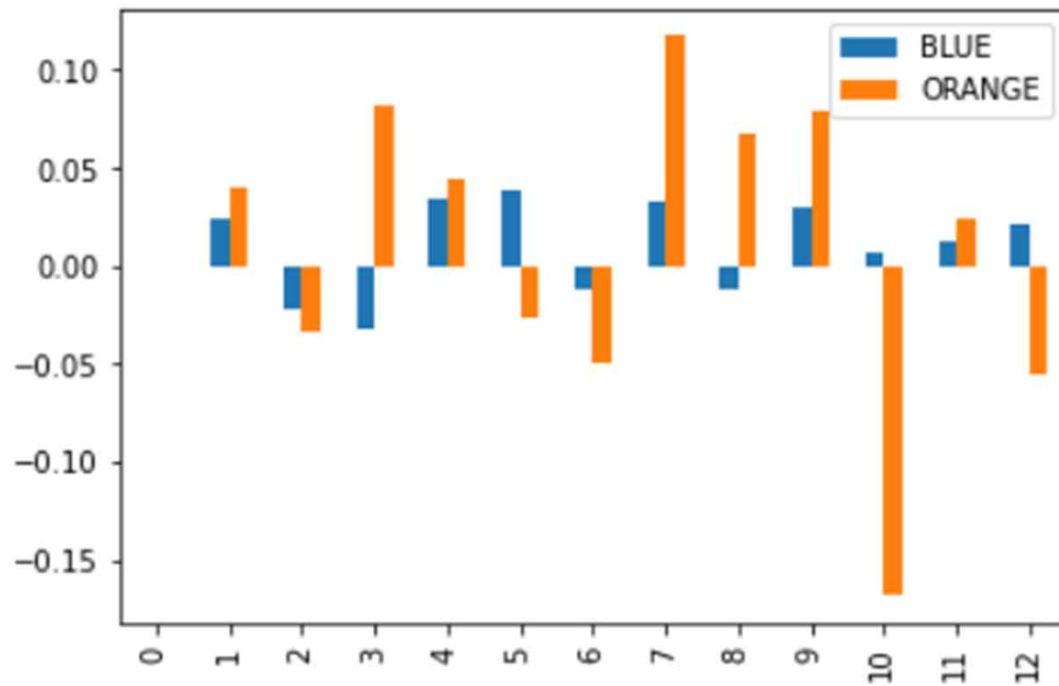
- 단기 이동평균선이 장기 이동평균선 위에 있을 때 매일 1 주씩 매입
 - 단순히 10일 이동평균선(단기, fast signal)이 50일 이동 평균선(장기, slow signal) 위에 위치하면 1 주 보유. 아래에 위치하면 0 주 보유.

```
df['Share'] = [1 if df.loc[idx, 'MA10'] > df.loc[idx, 'MA50'] else 0 for idx in df.index]
```

- 1 – Long (1 주 매입 or Hold)
 - 0 – Short (1 주 매도 or no action)
-
- Daily Profit 계산
 - 누적 수익 계산

수익률과 Risk

수익률(Return)



```
1 returns.mean()
```

```
BLUE      0.01  
ORANGE     0.01  
dtype: float64
```

```
1 returns.std()
```

```
BLUE      0.023977  
ORANGE     0.079601  
dtype: float64
```

- 두 개의 Return 은 동일
- Risk (sigma) 는 상이

따라서, 평균 Return 은 주식
평가의 좋은 지표가 아니다.

수익률(Return) 계산 – Simple Return

$$R_{t,t+1} = \frac{P_{t+1} - P_t}{P_t} = \frac{P_{t+1}}{P_t} - 1$$

R : Return
P : Price

예) 주식을 1000원에 사서 1200원에 팔았다면 수익률은,

$$\frac{1200 - 1000}{1000} = \frac{200}{1000} = 0.2 = 20\%$$

$$\frac{1200}{1000} - 1 = 1.2 - 1 = 0.2 = 20\%$$

배당금을 감안한 Return (수익률)

$$R_{t,t+1} = \frac{P_{t+1} + D_{t,t+1} - P_t}{P_t} = \frac{P_{t+1} + D_{t,t+1}}{P_t} - 1$$

D: Dividend (배당금)

R : Return

P : Price

일일수익률 (Daily Percentage Change)

- 주식을 구입하고 다음 날 팔았을 때의 이득 (손실)
- 주식의 변동성 분석에 유용
- 일일 수익률의 분포가 넓으면 변동성 큼
- $r_t = \frac{p_{t+1}}{p_t} - 1$ (당일가격 / 전일가격 - 1)

Compounding Returns (복리 수익률)

- CAGR (Compound Annual Growth Rate) – 연평균 성장률, 연복리 수익률
- 매월 1%의 수익을 낸다면 연간 수익률은 ?
 - $0.01 \times 12 = 0.12$? → No.
- 일정 기간 동안의 수익률을 연간으로 계산하려면 1년의 기간만큼 수익률을 복리화. 수익률에 1을 더한 다음 곱하기만 하면 됩니다.

$$(1 + R_m)^{12} - 1$$

$$(1 + 0.01)^{12} - 1 = 0.1268$$

누적수익률 (Cumulative Return)

- 1 불을 투자했을 때 누적적으로 가장 수익이 많이 났을 주식 파악
- 누적 수익률 = 현재 자산/과거자산 - 1
- 누적 수익률 = (1+당일 수익률) * (전일자 누적수익률)
 - $i_t = (1 + r_t) * i_{t-1}$
 - `df['cum ret'] = (1 + df['return']).cumprod() - 1`
- CAGR = (현재자산/과거자산)**(1/투자기간) - 1
= (1+누적수익률)**(1/투자기간) - 1

Log Return (로그 수익률)

- 금융분야의 수익률은 주로 log 수익률 사용 : 계산이 편리

$$R_{t,t+1} = \log\left(\frac{p_{t+1}}{p_t}\right) = \log(P_{t+1}) - \log(P_t)$$

R : Return

P : Price

당일가격 / 전일가격

Log Return vs Simple Return (Arithmetic Return)

- 산술적 수익률 = (나중가격 - 이전가격) / 이전가격
- Log 수익률 = $\ln \frac{\text{나중가격}}{\text{이전가격}} = \ln(\text{나중가격}) - \ln(\text{이전가격})$
- Ex) asset 가격 변화 : 100 → 130 → 100
 - 산술적 수익률 : 100 → 130 : +30%,
130 → 100 : -23 %
 - 누적 수익률 : 30 - 23 = 7 % (?)
 - log 수익률 : 100 → 130 : 26.24%,
130 → 100 : -26.24%
 - 누적 수익률 : 26.24 - 26.24 = 0 %

- Log 누적수익률(Cumulative Return) = $\sum \text{Log Return}$

- 단순히 daily return 을 더하면 기간 전체의 복리로 계산된 누적 수익률이 계산됨

- $1 + r = \frac{p_t}{p_0} = \frac{p_t}{p_{t-1}} + \frac{p_{t-1}}{p_{t-2}} + \dots + \frac{p_1}{p_0}$

r : logarithm return

$$r_t = \log\left(\frac{p_t}{p_{t-1}}\right) = \log(p_t) - \log(p_{t-1}) \text{ 이므로,}$$

$$\begin{aligned} r_3 + r_2 + r_1 &= (\log(p_3) - \log(p_2)) + (\log(p_2) - \log(p_1)) + (\log(p_1) - \log(p_0)) \\ &= \log(p_3) - \log(p_0) = \log\left(\frac{p_3}{p_0}\right) \end{aligned}$$

Annualized Returns (연간 수익률)

- 일정 기간의 수익률을 연간으로 환산하려면 1 년 기간 만큼의 수익률을 복리화 합니다.
- 월별 수익 R_m 을 연간으로 환산 $\rightarrow (1 + R_m)^{12} - 1$
- 분기별 수익 R_q 를 연간으로 환산 $\rightarrow (1 + R_q)^4 - 1$
- 일일 수익 R_d 를 연간으로 환산 $\rightarrow (1 + R_d)^{252} - 1$

다양한 종류의 수익률 비교

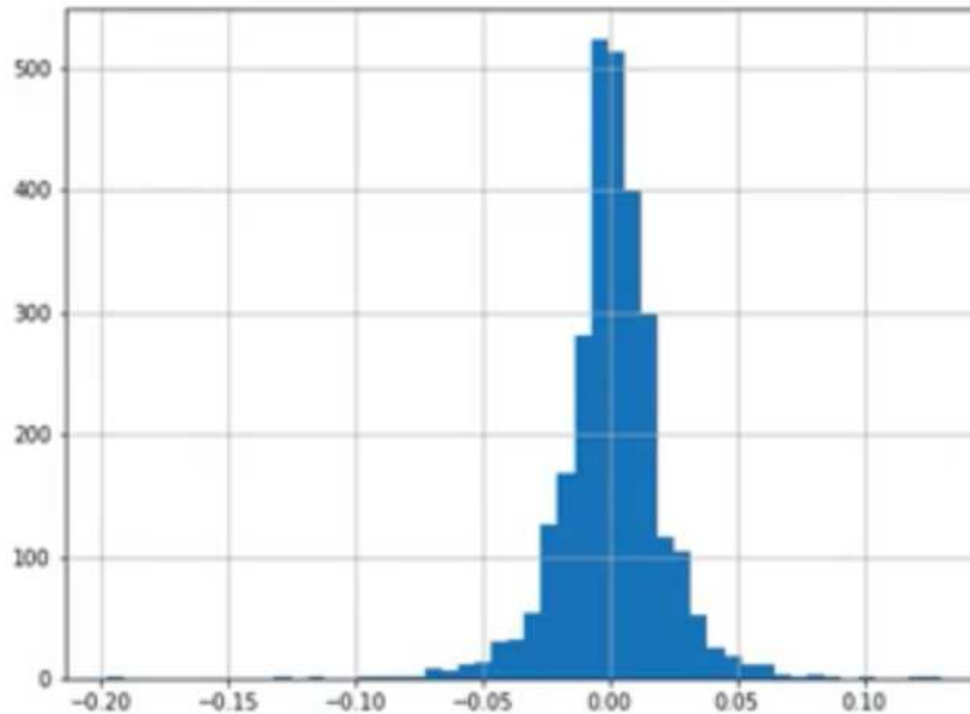
수익률 종류	정의	계산 방법	사용 상황
단순 수익률 (Simple Returns)	한 시점에서 다음 시점까지의 수익률	$(\text{후 시점의 가격} - \text{전 시점의 가격}) / \text{전 시점의 가격}$	단기 수익률 비교
복리 수익률 (Compound Returns)	여러 시점에 걸친 수익률로, 이전 시점의 수익이 다음 시점의 수익 계산에 영향을 미침	$(\text{최종값} / \text{초기값})^{1/n} - 1$ $n = \text{투자 기간(년)}$	장기 투자의 수익률 계산
연간 수익률 (Annual Returns)	투자 수익률을 연간 기준으로 환산한 수익률	$((1 + \text{복리 수익률})^{(1 / n\text{년})}) - 1$	투자 성과를 연간 기준으로 비교
누적 수익률 (Cumulative Returns)	투자의 시작부터 특정 시점까지의 전체 수익률	$\text{현재 자산} / \text{과거자산} - 1$	투자의 총 성과를 평가
로그 수익률 (Log Returns)	연속 복리 수익률이라고도 하며, 단순 수익률의 로그 값을 사용	$\ln(\text{나중가격}) - \ln(\text{이전가격})$	여러 시점의 수익률을 더할 때 사용

실습 : 040. Basics of Returns (수익률)

1. Simple Returns (단순 수익률, 산술 수익률)
2. Compounding Returns (복리 수익률)
3. Annualized Returns (연간 수익률)
4. Cumulative Returns (누적 수익률)
5. Log Returns (로그 수익률)

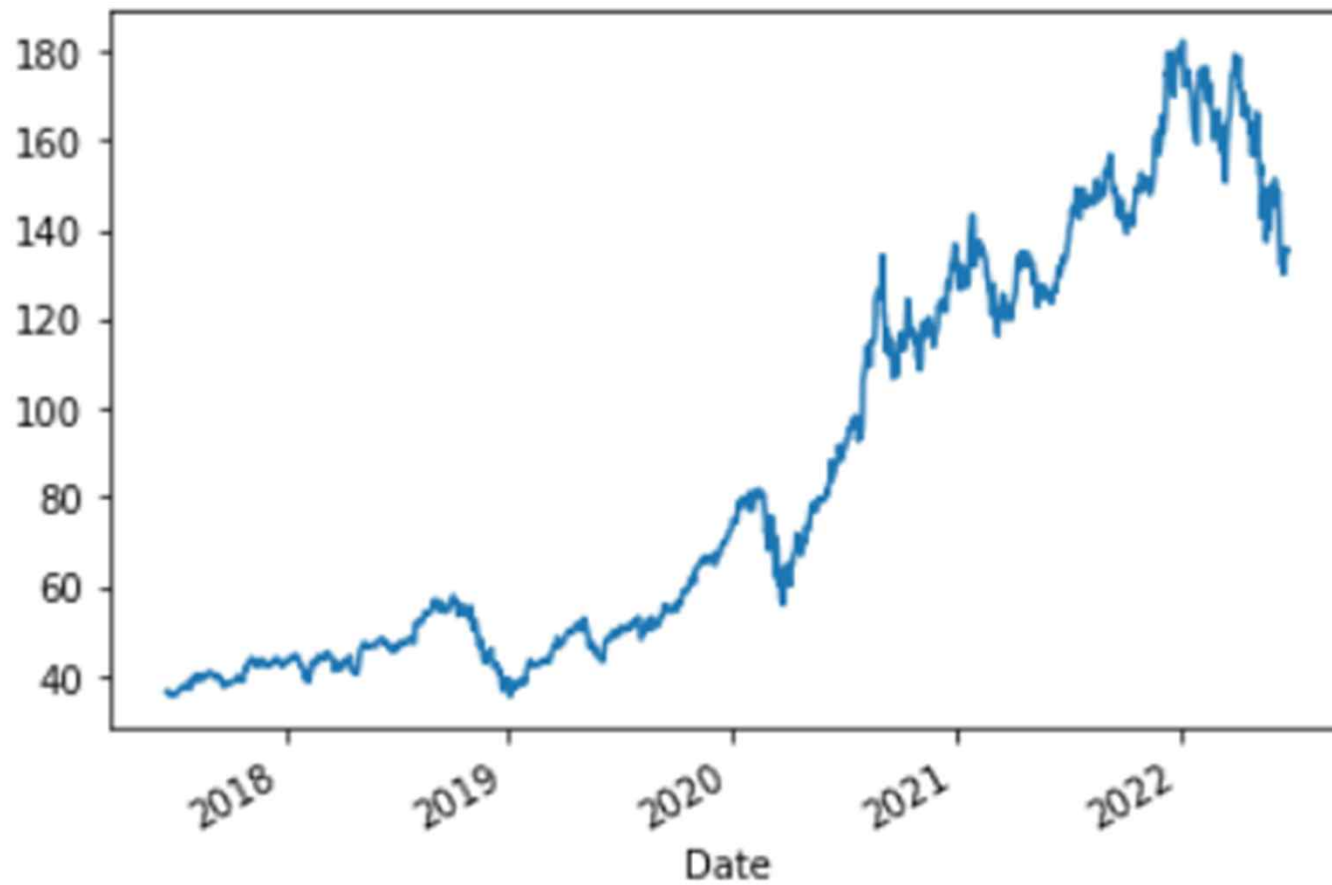
수익률의 정규분포화

- Mean(μ), variance(σ) \rightarrow historic data로부터 추정

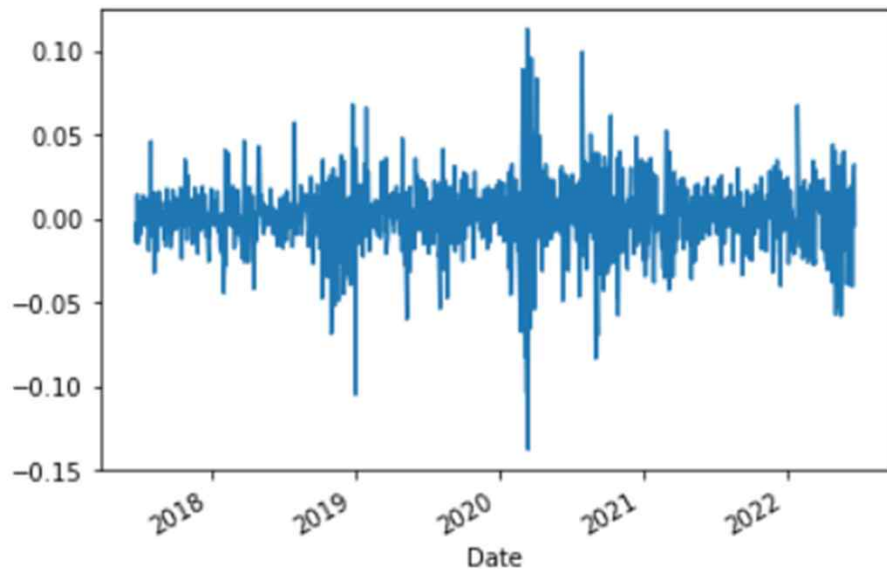


$$-0.1 < x < 0.1$$

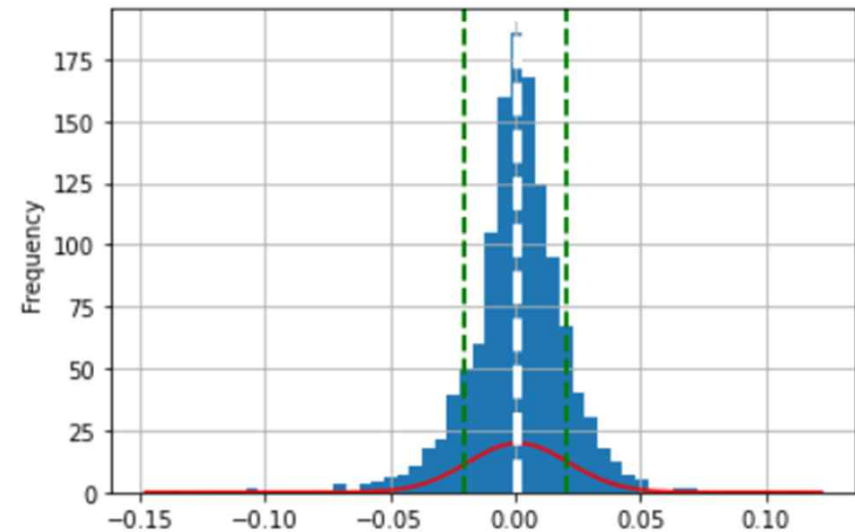
Example : Apple 주식의 주가 변동



Example : Apple 주식의 daily log return



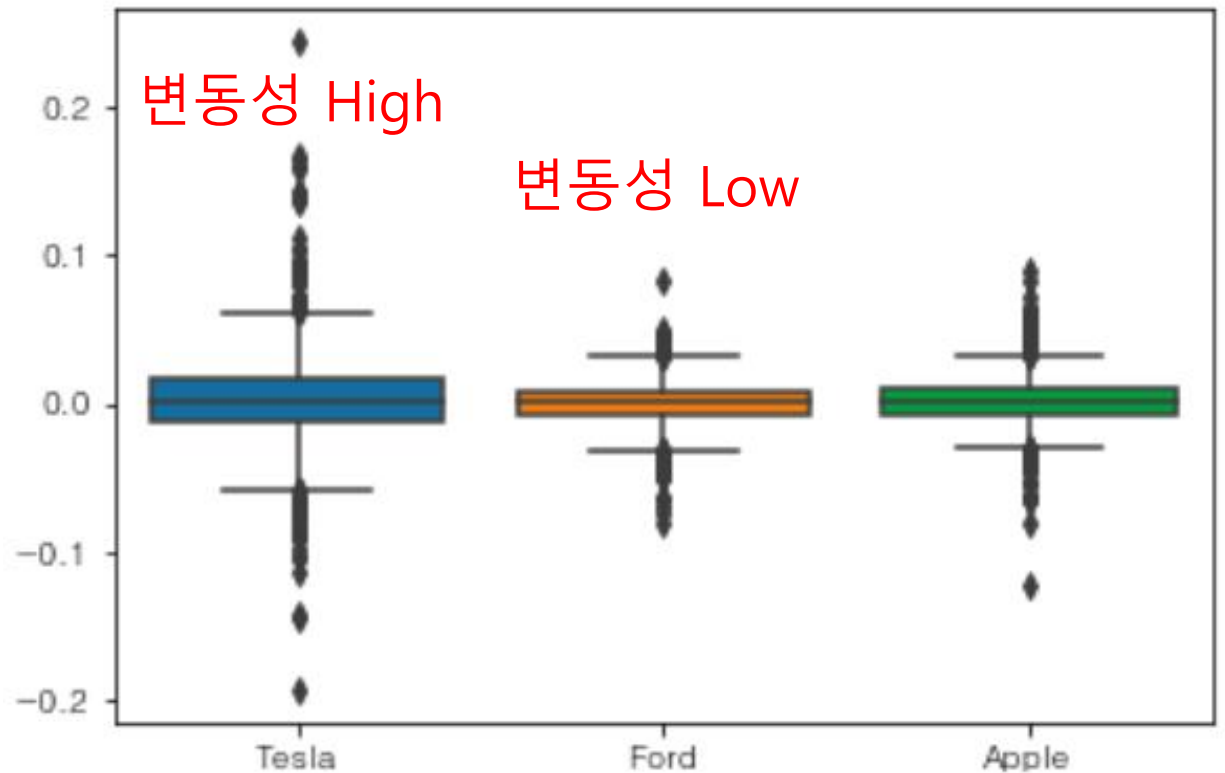
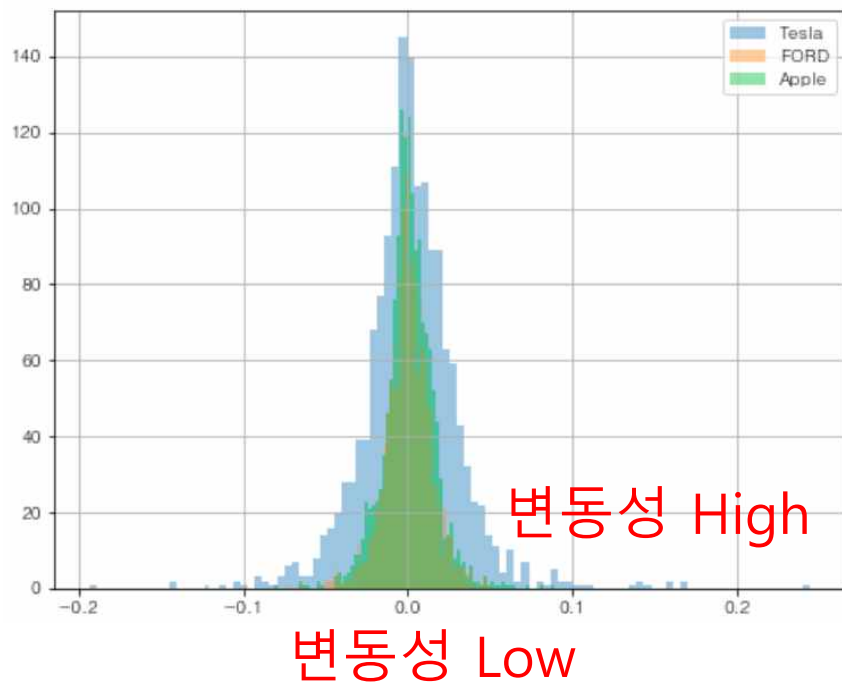
Daily Log Return 의 일일 변동



Daily Log Return 의 histogram

Histogram / Box Plot

– daily return 분포 확인



- 어떤 주식의 5 년간 수익률이 12%, 15%, 31%, 22%, 7% 인 경우 분산과 표준편차 계산
 - 평균(μ) = $(12 + 15 + 31 + 22 + 7) / 5 = 17.4$
 - 분산(variance) = $\sigma^2 = \frac{\sum(Value - \mu)^2}{n-1} = (12 - 17.4)^2 + (15 - 17.4)^2 + (31 - 17.4)^2 + (22 - 17.4)^2 + (7 - 17.4)^2 = 87.3(\%)$
 - 표준편차 = $\sigma = \sqrt{87.3} = 9.34 (\%)$
- 5 년간 평균 수익률은 17.4% 이고 상하로 9.34% 편차의 가능성이 있다. 따라서 6 년째의 수익률은 $(17.4-9.34\%) \sim (17.4+9.34\%)$ 사이 즉, 8.06%~ 26.74% 사이가 될 가능성이 있다.

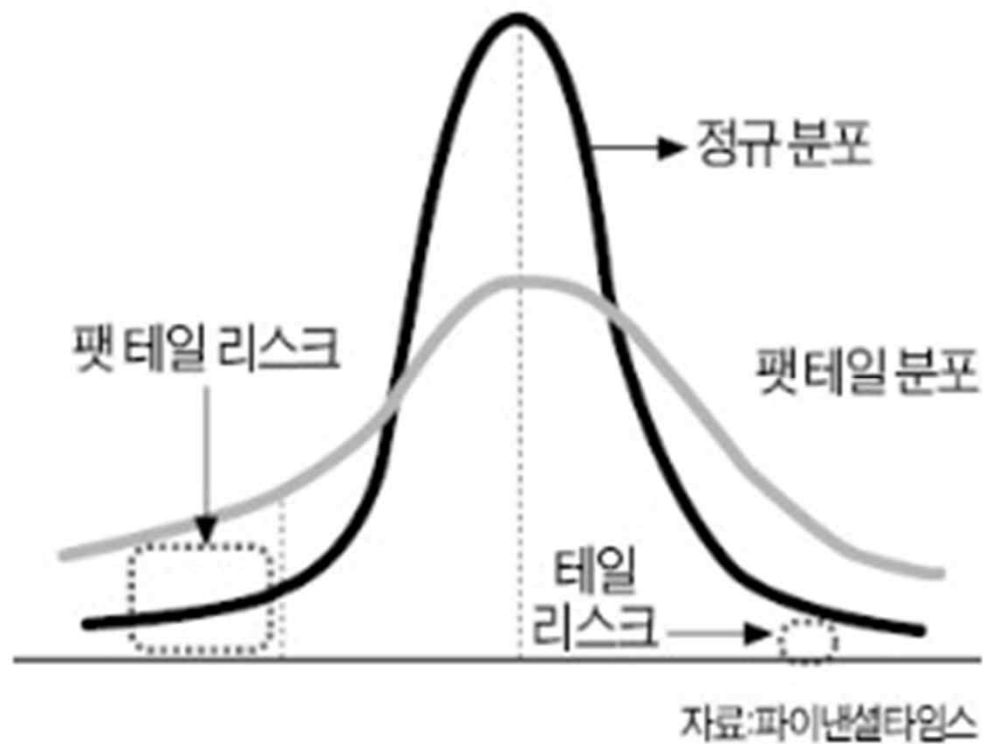
실습 : 041. 주식 시장 분석 Project

- 2012-1-1~2021-12-31 기간의 Tesla, FORD, Apple 주가 등락, 변동성, 수익률 비교
- 주가 추이 시각화
 - Close Price
 - Log Close Price
 - Normalized Price
- 거래 대금 추이 시각화
- 주식 종목간의 correlation 분석
- 일일 수익율 (Daily Percentage Change) 계산

What is Risk ?

- Risk = Volatility (변동성) = σ
 - 표본 평균 = μ
 - 표본 분산 = $(\text{변수값} - \text{평균})^2 = \sigma^2 = \frac{\sum(\text{Value} - \mu)^2}{n-1}$
 - 표본 표준편차 = $\sigma = \sqrt{\text{Sample Variance}}$
- 평균에서 더 많이 벗어나는 것이 more volatile

Stock 이 정규분포라고 가정할 때의 위험성



Annualizing Volatility (연간 환산 변동성)

$$\sigma_{ann} = \sigma_p \sqrt{p}$$

어떤 주식의 일일 수익률 series의 표준편차가 0.1% 라면
연간으로 환산한 변동성은 $0.001 \times \sqrt{252} = 1.58\%$

VaR (Value at Risk, 발생가능 최대손실)

- 투자 손실 위험을 나타냄
- 특정 포트폴리오가 일정기간 동안 보여준 변동률을 고려할 때 향후 발생할 수도 있는 최대 손실 가능 금액 (Worst Expected Loss)과 확률
 - '1일 5% VaR이 1억원' 이라 함은 특정 포트폴리오가 '하루 동안에 1 억원 이상 손해 볼 가능성이 5%' 라는 뜻
- VaR는 Exposure 의 크기, 변동성, 노출 기간에 의해 영향을 받습니다. 따라서, 노출이 커지거나 변동성이 커지면 VaR 값이는 커지게 되며,
- 최악의 상황을 정의하는 유의 수준을 낮추면 (예를 들어, 5%를 1%로) VaR 값은 증가 합니다. (→ 1% 확률로 더 큰 손실 발생 대비)

VaR의 용도

- 거래 및 투자활동에서 발생하는 위험을 최고경영자 및 주주에게 알리는데 사용
- 포지션한도를 설정하고 제한된 자원을 배분하는데 사용
- 다양한 시장에서의 투자활동을 상호 비교할 수 있는 공통의 기준치 제공
- 실적을 위험에 대해 조정하여 평가
- 감독기관(regulator) - VaR을 적절한 위험측정치로 사용
- 금융기관 및 기관투자가 - VaR를 이용하여 통합적으로 위험을 관리
(Barings, Daiwa은행들의 실패사례)

VAR (Value At Risk) 계산

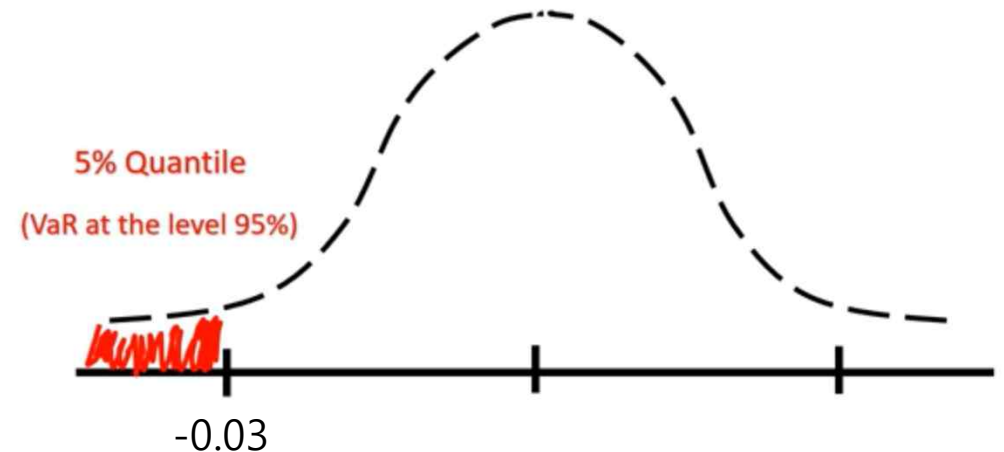
- `norm.ppf(0.05, mu, sigma)` → -0.03
 - 5 % quantile (분위수) 는 95% VaR 이라고 부름

Ex) 95% 신뢰수준에서 VaR -0.03

→ 3% 손실을 볼 확률이 5%

VaR 이 신뢰수준 99% 에서 10억

→ 10억 손실 볼 확률이 1 %



실습 : 045. VaR 계산

- 수익률 분포 시각화
- APPL 주식이 하루에 $x\%$ 이상 하락할 확률 계산
- APPL 주식이 일년(220 거래일)에 $xx\%$ 이상 하락할 확률 계산
- Value at risk (VaR) 계산
- 확률 분포 시각화

Risk Adjusted Return

- 펀드가 감수한 위험을 고려해 그 수익률을 평가하는 척도
- 비율(Ratio)이라는 단어를 달고 있는데, 기본적으로 수익 나누기 위험의 꼴을 따르기 때문
- 감수한 위험의 크기에 따라 수익률을 조정하기 때문에 이들을 위험조정 수익률(Risk-adjusted Return)이라고 부른다. 그리고 당연히 이 값들이 클수록 좋은 펀드이다.
- Sharpe Ratio 가 대표적인 위험 조정 수익률 지수

Risk Adjusted Measures Examples

- Ratio of return to risk – 단위 risk 당 수익 비율
 - ex) A 주식의 수익률 17.2%. Risk 36.8% → $17.2/36.8 = 0.47$
 - B 주식의 수익률 9.5%. Risk 18.7% → $9.5/18.7 = 0.5$

→ Which one is better ?
- Risk Free rate (미국 국채 금리, 정기예금)에 대한 초과 수익을 계산
 - Risk Free rate는 bench mark 수익률 이라고도 한다.
- Sharpe Ratio (샤프 비율) – 국채 금리(3%)를 제외한 risk 당 수익 비율
 - ex) A 주식의 수익률 17.2%. Risk 36.8% → $(17.2 - 3)/36.8 = 0.39$
 - B 주식의 수익률 9.5%. Risk 18.7% → $(9.5 - 3)/18.7 = 0.35$

Sharp Ratio (샤프비율)

- 샤프비율 = $\frac{(\text{기대되는 투자의 수익률} - \text{무위험 수익률})}{(\text{기대되는 투자의 수익률} - \text{무위험 수익률})\text{의 표준편차}}$

- $SR = \left(\frac{R_P - R_f}{\sigma_p} \right) \approx \frac{\text{Mean yearly Return}}{\sigma_p}$

$$S = \frac{E[R_p - R_f]}{\sqrt{\text{var}[R - R_f]}}$$

- R_P : Expected Portfolio Return
- R_f : Risk-free Return (정기예금, 국채금리)
- σ_p : Portfolio standard deviation

- $SR_{Year} = \sqrt{252} SR_{Day}$ - yearly Sharp Ratio

Annualized Sharpe Ratio

- $ASR = k * SR$

K : Daily sampling Rate $\rightarrow \sqrt{252}$

Weekly sampling Rate $\rightarrow \sqrt{52}$

Monthly sampling Rate $\rightarrow \sqrt{12}$

- thumb of rules

- $ASR > 1$: good

- $ASR > 2$: very good

- $ASR > 3$: excellent

실습 : 070. 변동성과 위험, Sharpe Ratio

- 변동성의 정의 이해
- 연간 변동성 계산
- 위험 조정 수익 : Sharpe Ratio 계산

Portfolio Optimization

What is Portfolio ?

- 현대 포트폴리오 이론은 Markowitz(1952)의 효율적인 프론티어 분석으로 탄생
- Portfolio 의 Sharpe Ratio (샤프 비율)

$$SR = \left(\frac{R_P - R_f}{\sigma_p} \right) \approx \frac{\text{Mean yearly Return}}{\sigma_p}$$

R_P : Expected Portfolio Return

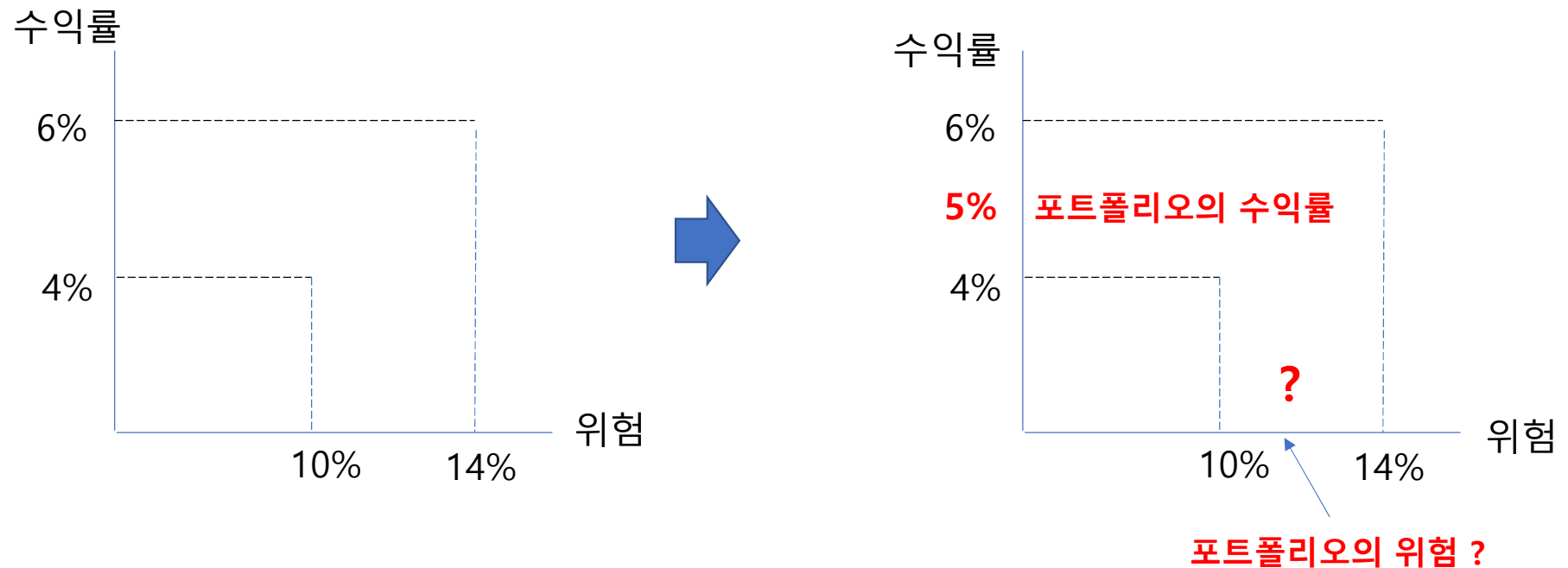
R_f : Risk-free Return (정기예금, 국채금리)

σ_p : Portfolio standard deviation

- General Rule
 - 상호간에 low / negative correlation 을 가진 asset 들로 portfolio 구성

2 asset portfolio의 수익률

- 포트폴리오의 수익률은 포트폴리오의 모든 구성 요소의 가중 평균에 불과



Two asset Portfolio 의 Variance (변동성)

- Portfolio Variance = $w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + 2w_1w_2Cov_{1,2}$

여기서, w_i - asset 의 weight

σ_i^2 - asset 의 variance

$Cov_{1,2}$ - asset 1 과 2 사이의 covariance

- Correlation = $\frac{Cov_{1,2}}{\sigma_1\sigma_2}$

- Portfolio Standard Deviation = $\sqrt{w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + 2w_1w_2Cov_{1,2}}$

Portfolio 변동성의 마법

- 2 asset portfolio의 이상한 점은 위험 측면에 있다.
A와 B의 상관관계에 의존하기 때문.
- A와 B가 완벽하게 상관되어 있다면 기본적으로 같은 자산처럼 행동할 것.
- 그러나 상관 관계가 없으면 하나는 올라가고 다른 하나는 내려가고 서로 지그재그로 움직인다.
이 경우 조합이 실제로 생각하는 것보다 변동성이 적다는 것을 알 수 있다.
- 그것은 단지 절반이 아니라 그보다 적은 어딘가에 있다. 그리고 그것들이 더 많은 상관관계를 가질수록 해당 포트폴리오의 변동성은 줄어든다.
- 이것이 바로 기본적인 종류의 미스터리 또는 포트폴리오 구성의 마법이다.

- Portfolio Variance = $w_1^2\sigma_1^2 + w_2^2\sigma_2^2 + 2w_1w_2\text{Cov}_{1,2}$

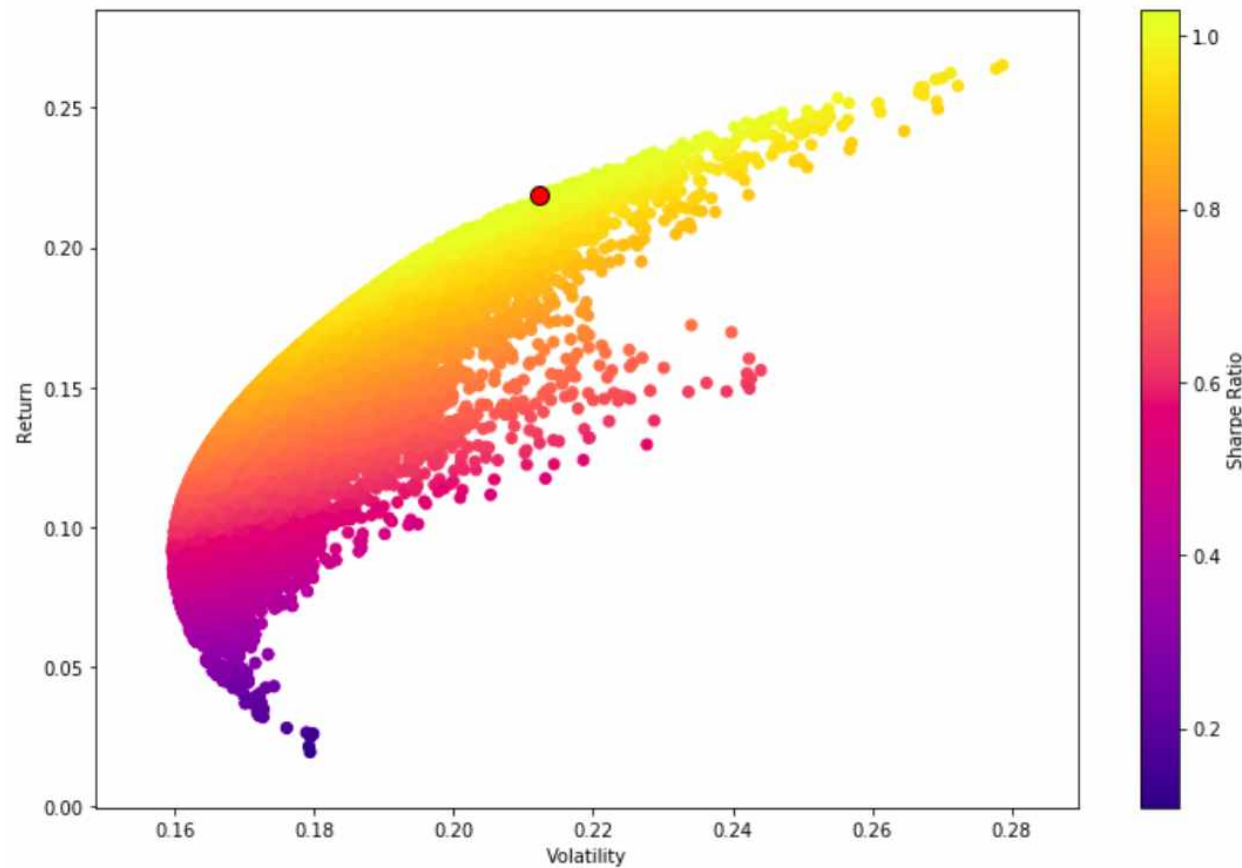
Portfolio	W1	W2	Return	Volatility
1	0.0	1.0	6.0	14.0
2	0.1	0.9	5.8	13.03
3	0.2	0.8	5.6	12.14
4	0.3	0.7	5.4	11.34
5	0.4	0.6	5.2	10.65
6	0.5	0.5	5.0	10.10
7	0.6	0.4	4.8	9.71
8	0.7	0.3	4.6	9.50
9	0.8	0.2	4.4	9.47
10	0.9	0.1	4.2	9.65
11	1.0	0.0	4.0	10.00

포트폴리오 최적화 (Portfolio Optimization)

- 위험회피적 투자자가 시장 위험의 주어진 수준에 따라 기대수익을 최적화하거나 최대화하여 보상 수준 선택
- 방법 1 – Monte Carlo Simulation 에 의한 최적 포트폴리오 탐색
- 방법 2 – scipy 를 이용한 수학적 최적화 (Mathematical Optimization)

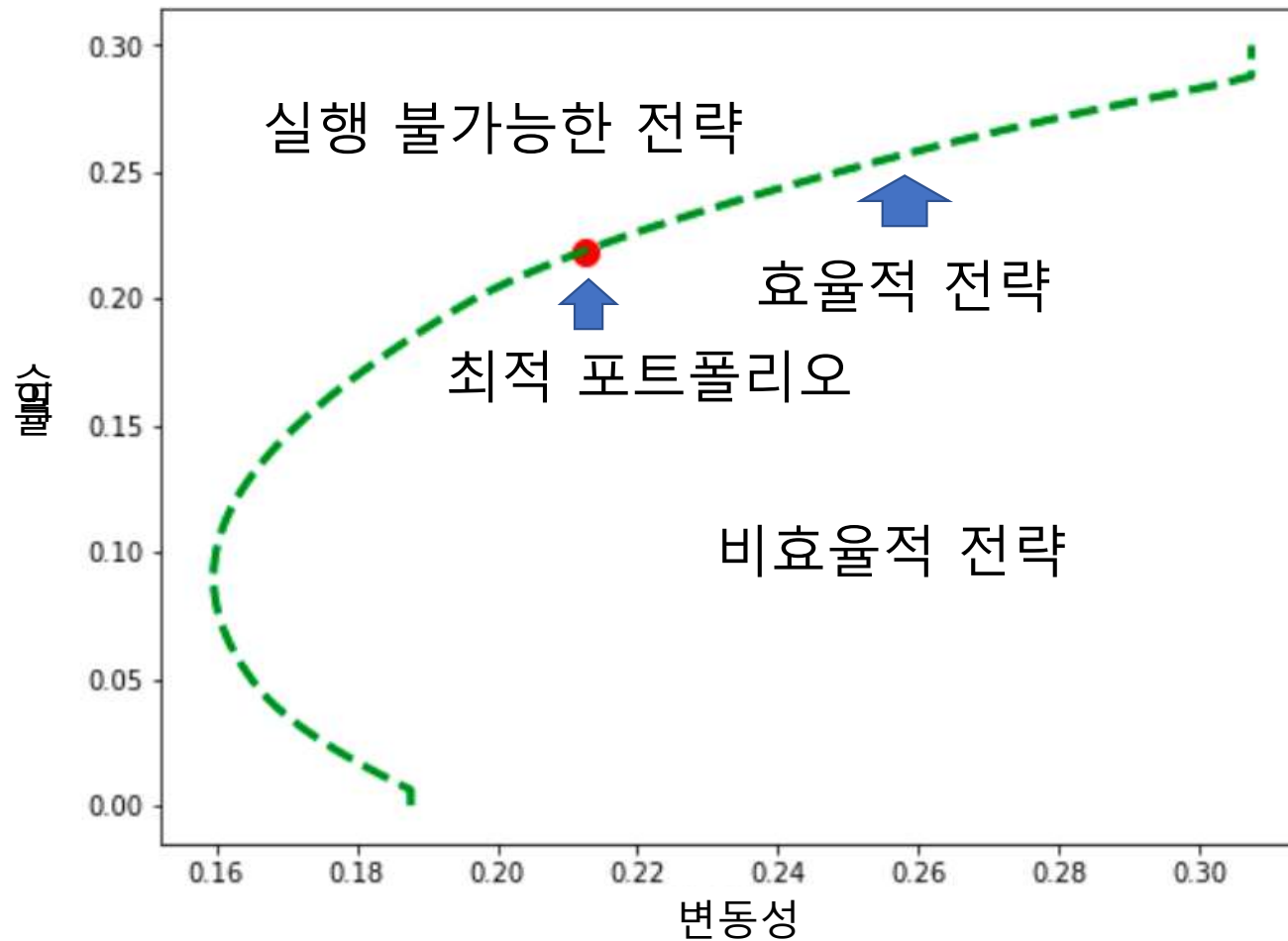
최적 Portfolio 와 효율적 투자선

포트폴리오의
기대수익률



포트폴리오의 위험

효율적 투자선 (Efficient Frontier)



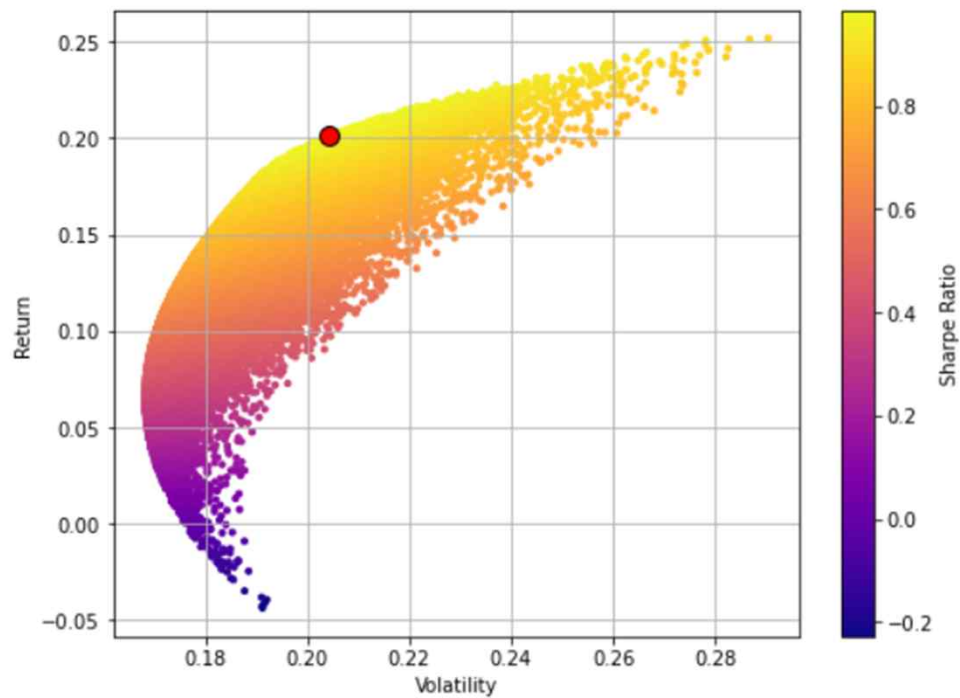
Python의 Optimization 함수

- `from scipy.optimize import minimize`
- `minimize(object-function, initial_guess, method='SLSQP', bounds=bounds, constraints=cons)`
 - Objective function : 최소화할 negative Sharpe ratio
 - Bounds : 각 weight 에 대한 (min, max)
 - Constraints : {'type': 'eq', 'fun': check_sum}

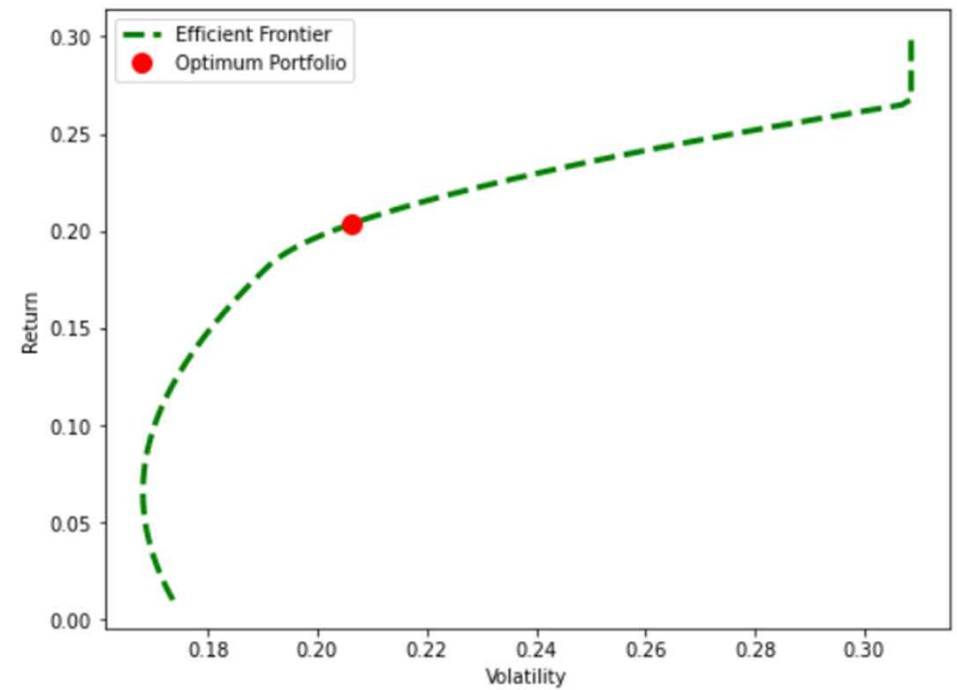
실습 : 081. Portfolio Optimization

- 방법 1 – Monte Carlo Simulation 에 의한 최적 포트폴리오 탐색
- 방법 2 – scipy 를 이용한 수학적 최적화 (Mathematical Optimization)
- MC method + 효율적 frontier 시각화
- 목표 수익률에 대한 최적 portfolio

Monte Carlo Method



수학적 Optimization



CAPM (Capital Asset Pricing Model)

What is CAPM(자본자산 가격결정 모델) ?

- Finance 분석의 중요한 개념
- 기대수익(expected return)과 risk와의 관계를 설명
- 자산(주식)의 기대수익은 무위험 수익 + risk premium (위험 프리미엄)
 - 무위험 자산
 - zero standard deviation(변동성 0)인 자산의 return $\rightarrow r_f$
ex) 10년 만기 미국 국채
 - 위험 프리미엄 – 주식 같은 위험 자산에 투자할 경우, 국채나 현금 같은 무위험 자산의 수익률에 추가하여 얻을 수 있는 초과 수익률

CAPM 공식

- $r_i - r_f = \beta_i(r_m - r_f)$

i 라고하는 위험 자산의 기대수익률은 그 i 자산의 beta 값만
알면 결정할 수 있다. 즉, i라고 하는 위험자산의 기대수익률과
i 라는 자산의 beta 값은 선형 관계이다.

- r_i : 개별자산의 기대수익
- r_m - market portfolio 의 기대수익
- $\beta_i = \frac{cov(r_i, r_m)}{var(r_m)}$
- r_f - risk free rate

r_m : market portfolio. 시장의 모든 주식을 포함하는 포트폴리오(ex. S&P500)

$r_m - r_f$: RISK PREMIUM. 위험 자산에 투자하는데 따른 incentive

What is BETA ?

- 개별 주식과 시장(포트폴리오)의 관련성 – beta 계수

$$\beta_i = \frac{\text{cov}(r_i, r_m)}{\text{var}(r_m)}$$

$$= \frac{\text{자산수익률과 시장수익률 사이의 공분산}}{\text{시장수익률의 분산}}$$

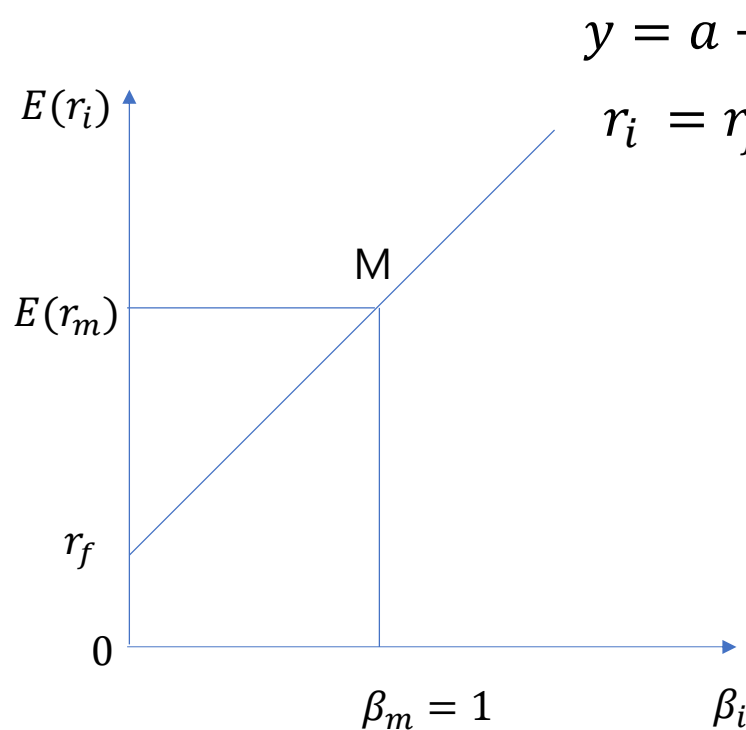
- 베타계수의 값 = 1 → 시장위험과 동일 (시장과 같이 움직임)
- 베타계수의 값 > 1 → 시장위험보다 자산위험이 큼
- 베타계수의 값 < 1 → 시장위험보다 자산위험이 작음
- 베타계수의 값 = 0 → 무위험 자산

CAPM Example

- 어떤 투자자가 애플 주식에 투자할 때의 기대수익률은 ?
 - r_m - 12.4% (S&P500 의 수익률)
 - $\beta_{APPL} = 1.11 \rightarrow$ market 보다 volatile
 - r_f - 0%
- $r_{APPL} = r_f + \beta_i(r_m - r_f) = 0 + 1.11(12.4-0) = 13.7\%$
 - Apple 주식에 투자하는 사람은 자신이 선택한 risk에 대한 보상으로 13.7%의 수익을 기대 한다.

증권시장선 (SML)

- 체계적 위험(beta 계수)과 기대수익률 간의 선형 관계식



$$y = a + b * x$$

$$r_i = r_f + b * \beta_i \quad b = \frac{r_m - r_f}{\beta_m} = r_m - r_f \text{ 이므로}$$

$$r_i = r_f + (r_m - r_f)\beta_i$$

r_i : 개별자산의 기대수익

r_m - market portfolio 의 기대수익

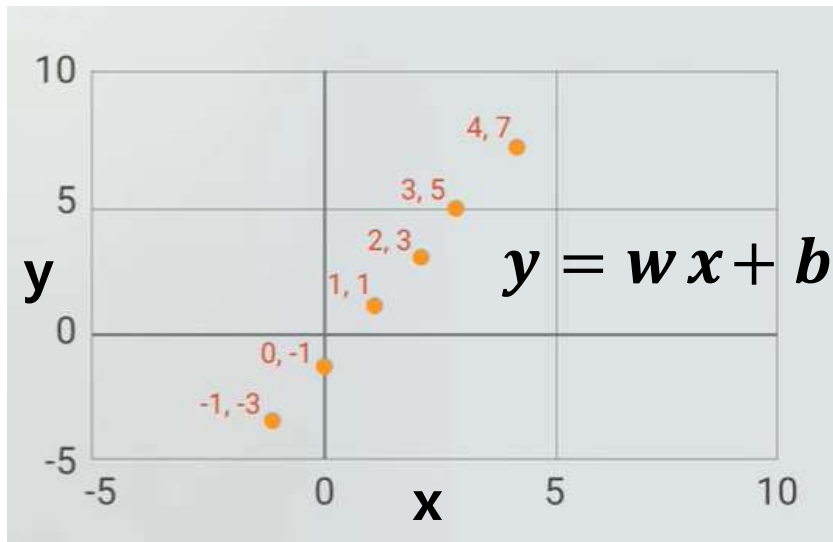
$$\beta_i = \frac{\text{cov}(r_i, r_m)}{\text{var}(r_m)}$$

r_f - risk free rate

Machine Learning 이용 Beta 계산

Univariate Linear Regression (단변수 선형회귀)

- 한개의 변수로 결과 예측 (ex. KOSPI 지수로 삼성전자 주가 예측)



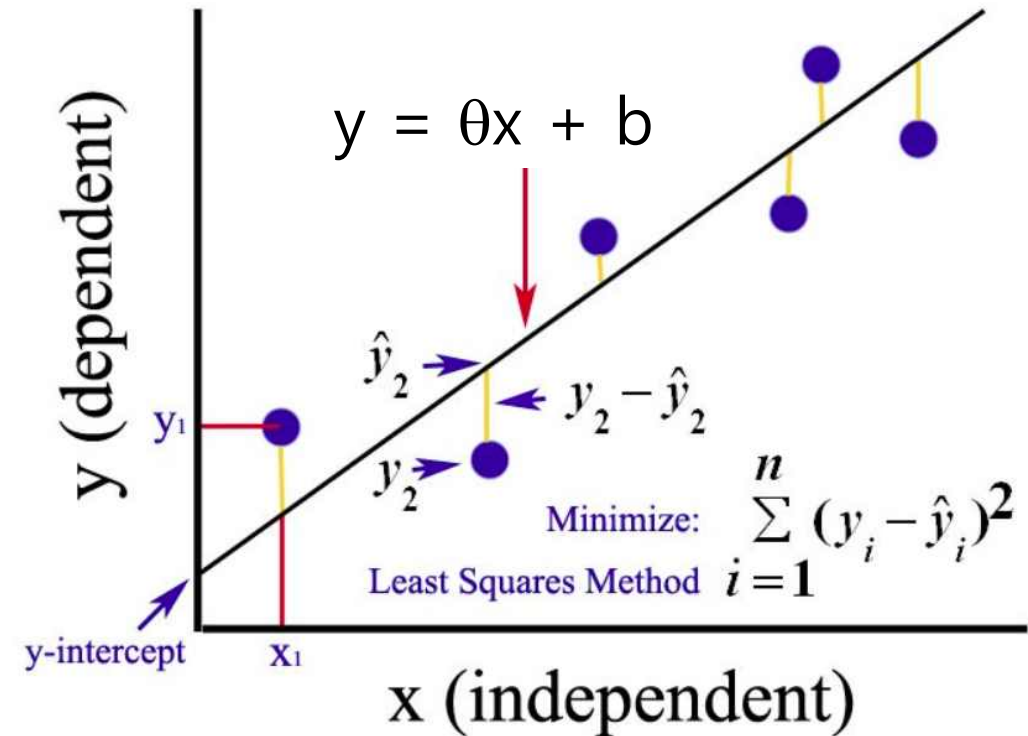
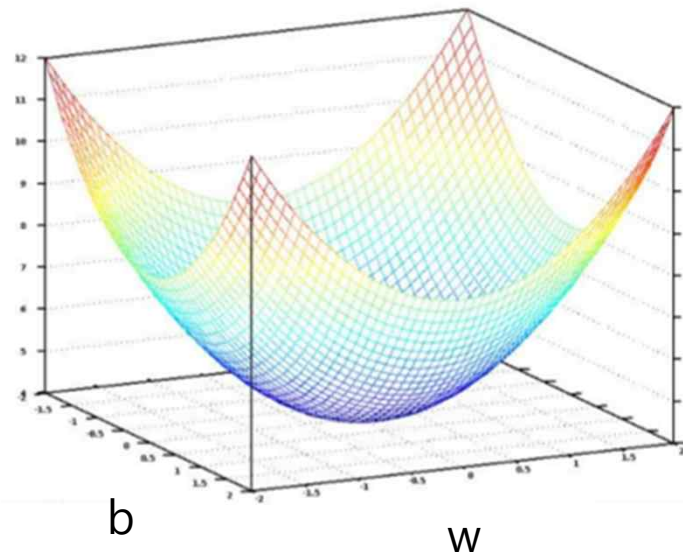
- x, y 가 주어지고
 w, b 가 미지수



- w, b 를 infer (추정)

Cost Function - Linear Regression

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{Y}_i - Y_i)^2$$



**MSE 를 최소화 하는
 θ 와 b 를 optimize**

Multivariate Linear Regression (다변수선형회귀)

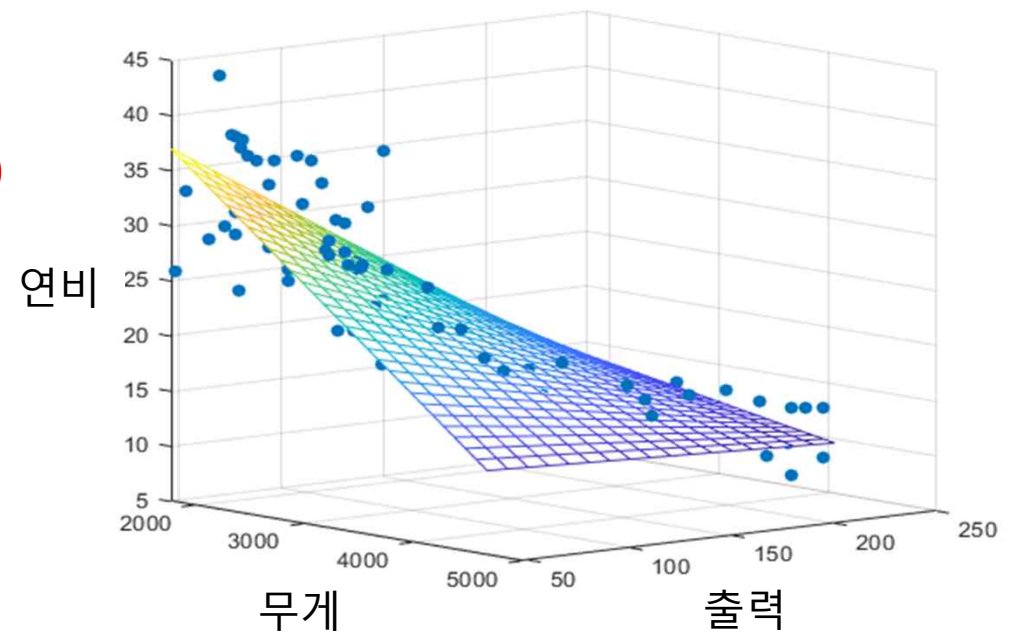
$$\hat{Y} = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \theta_3 X_3 + \dots + \theta_n X_n$$

$$\hat{Y} = \theta X$$

sklearn.linear_model.LinearRegression()

$\theta = \text{coef_}$

$\theta_0 = \text{intercept_}$

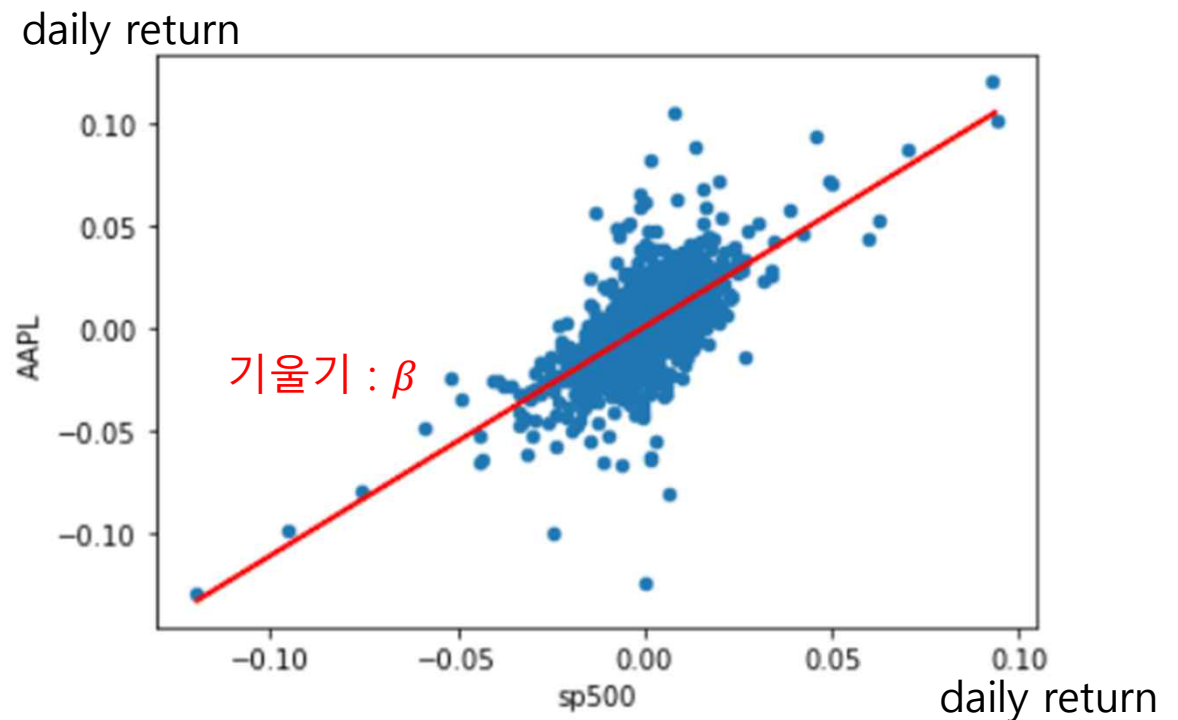


실습 : 100. Capital Asset Pricing Model

- Simple Linear Regression 을 이용한 CAPM modeling

- CAPM 공식 이용

$$r_i = r_f + \beta_i(r_m - r_f)$$



파생 상품

파생상품(Derivatives)의 정의

- 그 가치가 하나 이상의 기초 자산에 의해 결정되는 금융 계약
- 기초 자산은 주식, 채권, 상품, 이자율, 지수, 통화 등 다양
- 파생 상품의 가치는 기초 자산의 가격 변동에 따라 달라진다.
- 자산 가격 변동의 위험을 관리하고, 헤지(hedge)하기 위해 사용
- 주가, 이자율, 통화 가치 등이 미래에 어떻게 변동할지에 대한 불확실성을 줄여주는 역할
- 시장 변동에 따라 수익을 얻기 위한 투기 도구로도 사용

파생 상품의 종류

- 선물 계약 (Futures)

- 구매자와 판매자가 미래의 특정 날짜에 특정 가격으로 기초 자산을 거래하기로 합의
- 상품, 외환, 이자율 등의 가격 변동 위험을 관리하는 데 사용

- 옵션 계약 (Options)

- 구매자에게 특정 기간 동안 또는 특정 날짜에 기초 자산을 사거나 팔 수 있는 권리 제공
- 콜 옵션(Call Options)과 풋 옵션(Put Options)의 두 가지 유형

- 스왑 계약 (Swaps)

- 두 계약 당사자가 자산이나 채무, 지급 채권 등을 교환하는 계약
- 이자율 스왑, 통화 스왑, 신용 디폴트 스왑 등

선물 계약 (Future Contract)

	선물 계약의 구매자 (Long Position)	선물 계약의 판매자 (Short Position)
권리/의무	계약에 명시된 날짜에 기초 자산을 행사 가로 사는 권리/의무	계약에 명시된 날짜에 기초 자산을 행사 가로 판매하는 권리/의무
이익과 손실	기초 자산의 가격이 행사가보다 높으면 이익을 보고, 행사가보다 낮으면 손실을 보게 된다.	기초 자산의 가격이 행사가보다 낮으면 이익을 보고, 행사가보다 높으면 손실을 보게 된다.
전망	기초 자산의 가격이 상승할 것으로 예상	기초 자산의 가격이 하락할 것으로 예상

Contango/Backwardation

- 콘탱고 - 선물 가격이 현물 가격보다 높은 상태. 시간이 지남에 따라 선물 가격이 하락하여 현물 가격에 수렴합니다. 발생 이유는,
 1. 보유 비용(저장비용, 보험비용, 이자 비용 등)이 포함
 2. 시장 기대 - 시장 참가자들이 미래 가격 상승을 예상
- 백워데이션 - 선물 가격이 현물 가격보다 낮으며, 시간이 지남에 따라 선물 가격이 상승하여 현물 가격에 수렴합니다. 발생 이유는,
 1. 단기적 공급 부족 우려: 현재 수요가 높아져 현물 가격이 상승하고, 미래 공급이 회복될 것으로 기대하고 선물 가격이 상대적으로 낮게 형성
 2. 리스크 프리미엄: 미래의 불확실성 때문에 현물 자산을 미리 확보

Contango/Backwardation 비교

	Contango	Backwardation
가격	선물 가격이 현물 가격보다 높음	선물 가격이 현물 가격보다 낮음
이유	보유 비용 포함, 미래 가격 상승 예상	단기 공급 부족 예상, 미래 가격 하락 예상
특징	선물 가격이 시간이 지나면서 하락 하여 현물 가격에 수렴	선물 가격이 시간이 지나면서 상승 하여 현물 가격에 수렴
예시	금, 원유 등 장기 보유 비용이 높은 상품	원유, 농산물 등 공급 부족 우려가 있는 상품

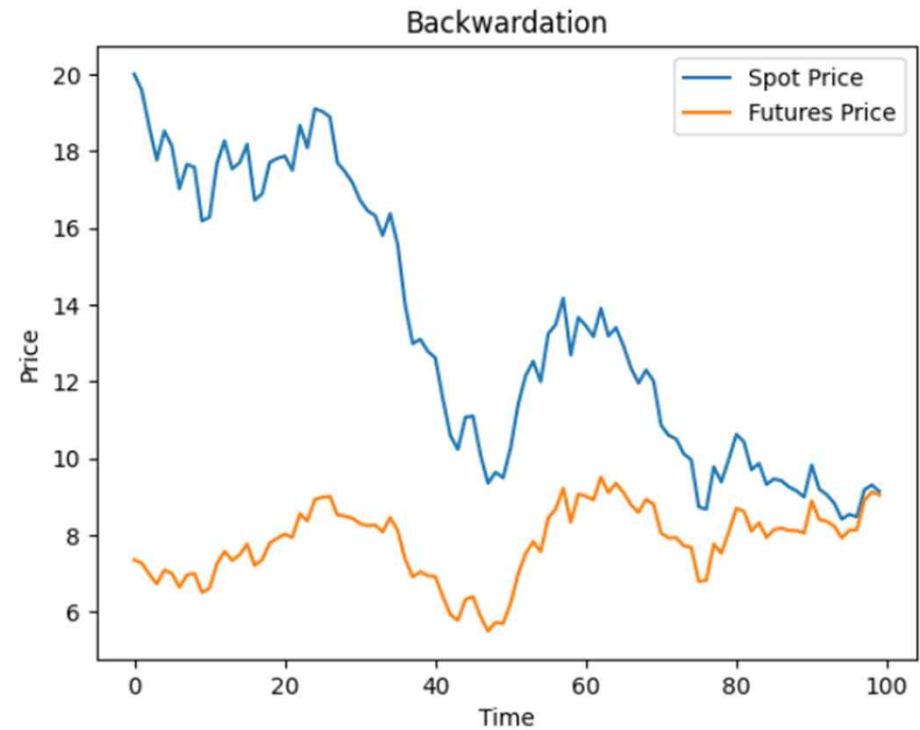
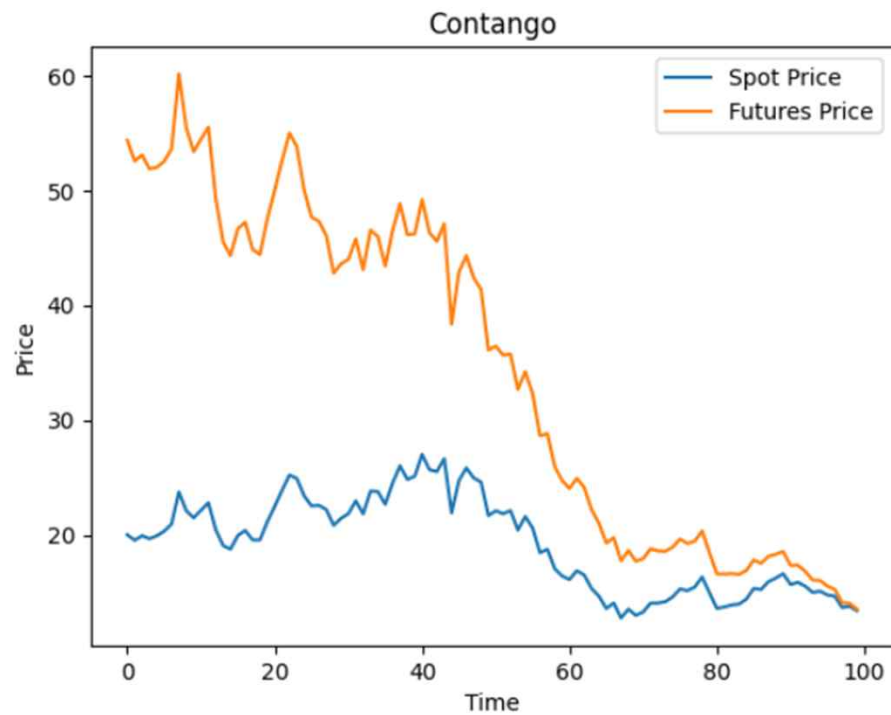
Option 계약의 종류

	콜 옵션(Call Option)	풋 옵션(Put Option)
정의	특정 가격(행사가)에서 기초 자산을 사는 권리를 부여	특정 가격(행사가)에서 기초 자산을 파는 권리를 부여
보유자의 권리	기초 자산을 행사가로 구매	기초 자산을 행사가로 판매
전망	주거나 기타 자산 가격이 상승할 것으로 예상될 때 구매	주거나 기타 자산 가격이 하락할 것으로 예상될 때 구매
손익의 한계	손실 - 지불한 프리미엄으로 제한 이익 - 이론적으로 무한정	이익 - 행사가와 프리미엄 차이로 제한 손실 - 지불한 프리미엄으로 제한
매도자의 의무	매도자는 옵션을 행사할 경우, 기초 자산을 행사가로 판매해야 한다.	매도자는 옵션을 행사할 경우, 기초 자산을 행사가로 구매해야 한다.

Option Long/Short 비교

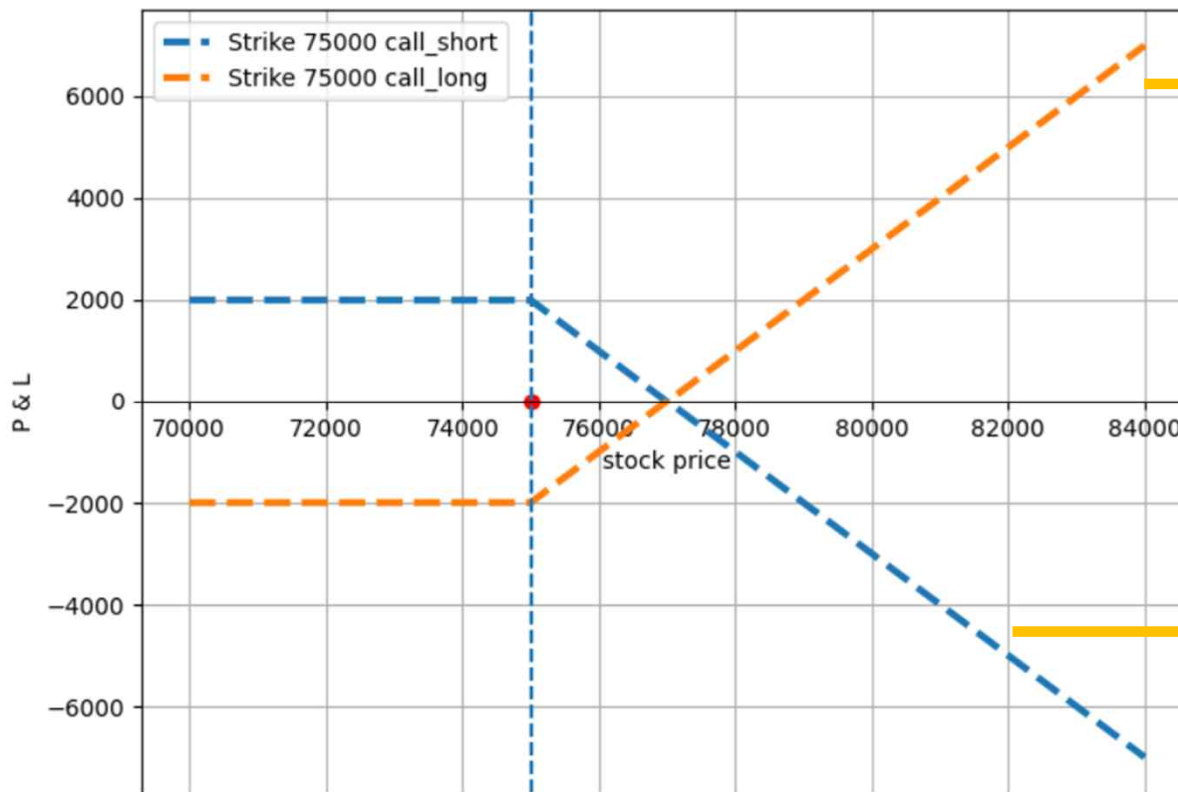
	롱콜(Long Call)	숏콜(Short Call)	롱풋(Long Put)	숏풋(Short Put)
정의	Call Option 매수	Call Option 매도	Put Option 매수	Put Option 매도
전망	주거나 기타 자산 가격이 상승 예상	주거나 기타 자산 가격이 하락 예상	주거나 기타 자산 가격이 하락 예상	주거나 기타 자산 가격이 상승 예상
이익과 손실의 한계	손실은 지불한 프리미엄으로 제한되며, 이익은 이론적으로 무한정	이익은 받은 프리미엄으로 제한되며, 손실은 이론적으로 무한정	이익은 행사가와 프리미엄 차이로 제한되며, 손실은 지불한 프리미엄으로 제한	이익은 받은 프리미엄으로 제한되며, 손실은 행사가와 프리미엄 차이로 제한
권리/의무	옵션을 행사할지 여부를 결정할 권리	옵션이 행사될 경우, 기초 자산을 행사가로 팔아야 할 의무	옵션을 행사할지 여부를 결정할 권리	옵션이 행사될 경우, 기초 자산을 행사가로 사야 할 의무

실습 – 51. Contango/Backwardation



실습 – 52. Call Put Payoff

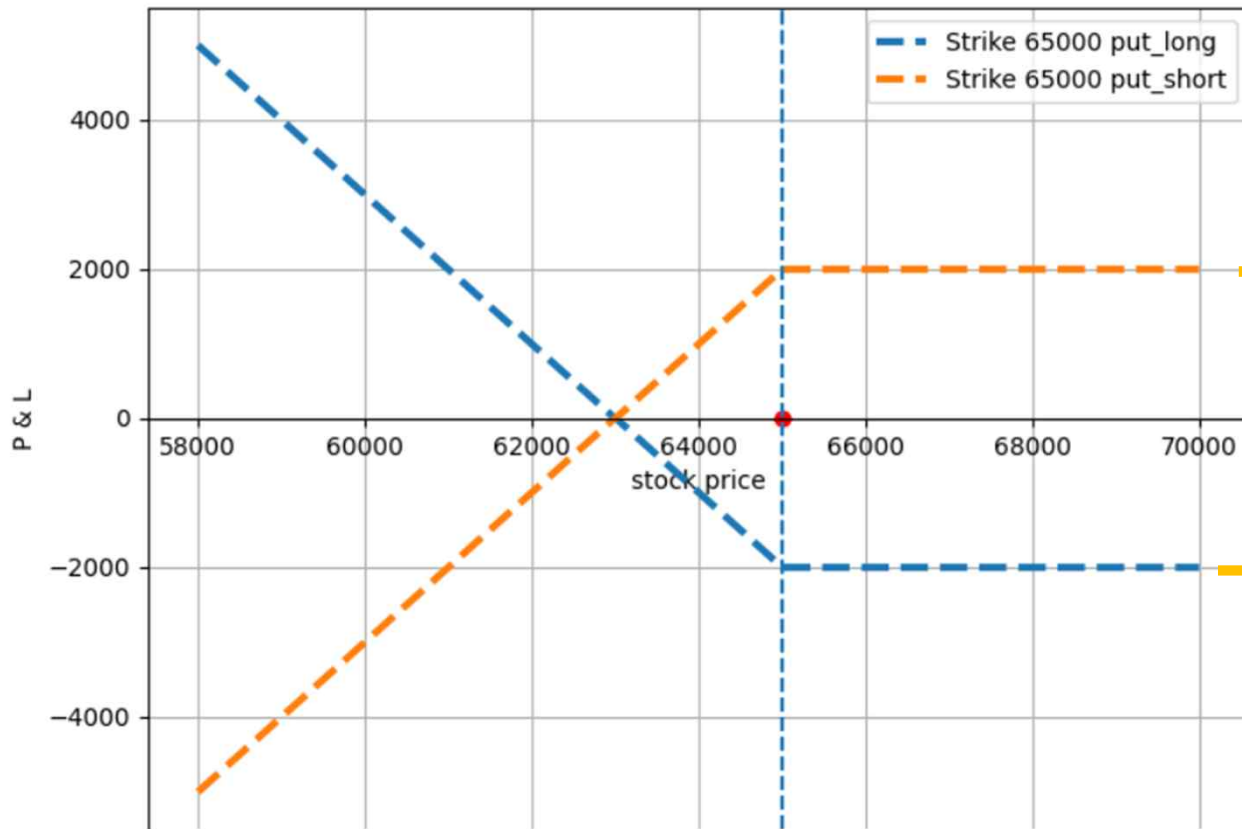
• Call Option Payoff 시각화



- 실행 가격 75,000일 때 콜 옵션을 매수 (long)한 위치의 페이오프
- 주가가 75,000 이하일 때는 구매한 프리미엄만큼의 손실을 보게 되고, 75,000을 넘어서면 수익이 증가
- 이익은 주가가 상승함에 따라 무한대로 증가할 수 있다.

- 실행 가격 75,000일 때 콜 옵션을 매도 (short)한 위치의 페이오프
- 주가가 75,000 이하일 때는 페이오프가 일정하고, 그 이상으로 올라갈수록 손실이 무한대로 증가
- 콜 옵션을 매도한 사람이 주가가 상승하면 옵션을 실행할 수 있는 권리를 매수자에게 주어야 하기 때문에 발생하는 손실

• Put Option Payoff 시각화



- 스트라이크 가격이 65,000인 풋 옵션을 매도한 경우의 페이오프
- 주가가 스트라이크 가격 아래로 떨어질수록 손실이 증가하며(풋 옵션 매도자에게 불리), 스트라이크 가격 위에서는 매도한 프리미엄만큼의 수익을 얻습니다(이 경우 그 수익은 제한적입니다).

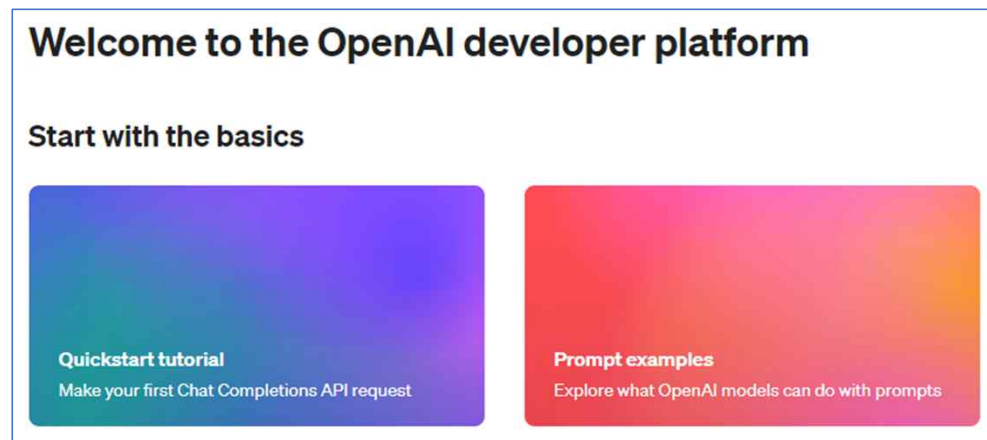
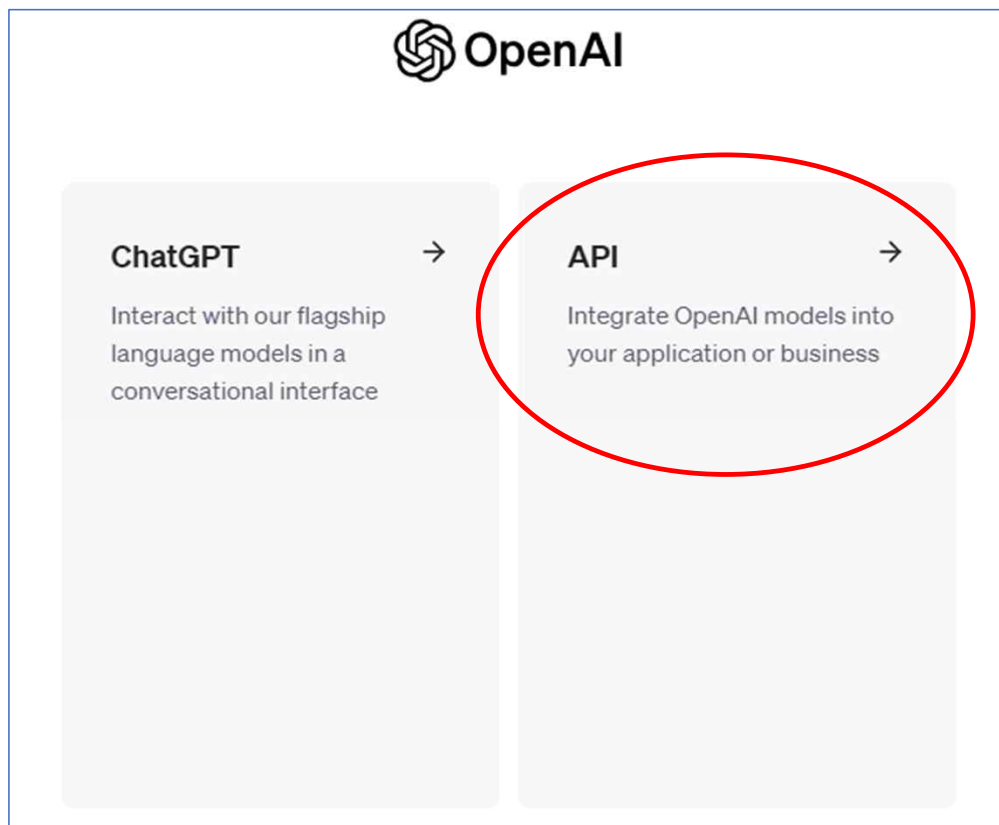
- 스트라이크 가격이 65,000인 풋 옵션을 매수한 경우의 페이오프
- 주가가 스트라이크 가격 아래로 떨어질수록 수익이 증가하며(풋 옵션 매수자에게 유리), 스트라이크 가격 위로 올라가면 매수한 프리미엄만큼 손실이 발생합니다(이 경우 그 손실은 제한적입니다).

실습 – 53. Iron Condor 전략

- Bull Put Spread 전략
- Bear Call Spread 전략
- Iron Condor 합성 옵션 전략
- Payoff 시각화

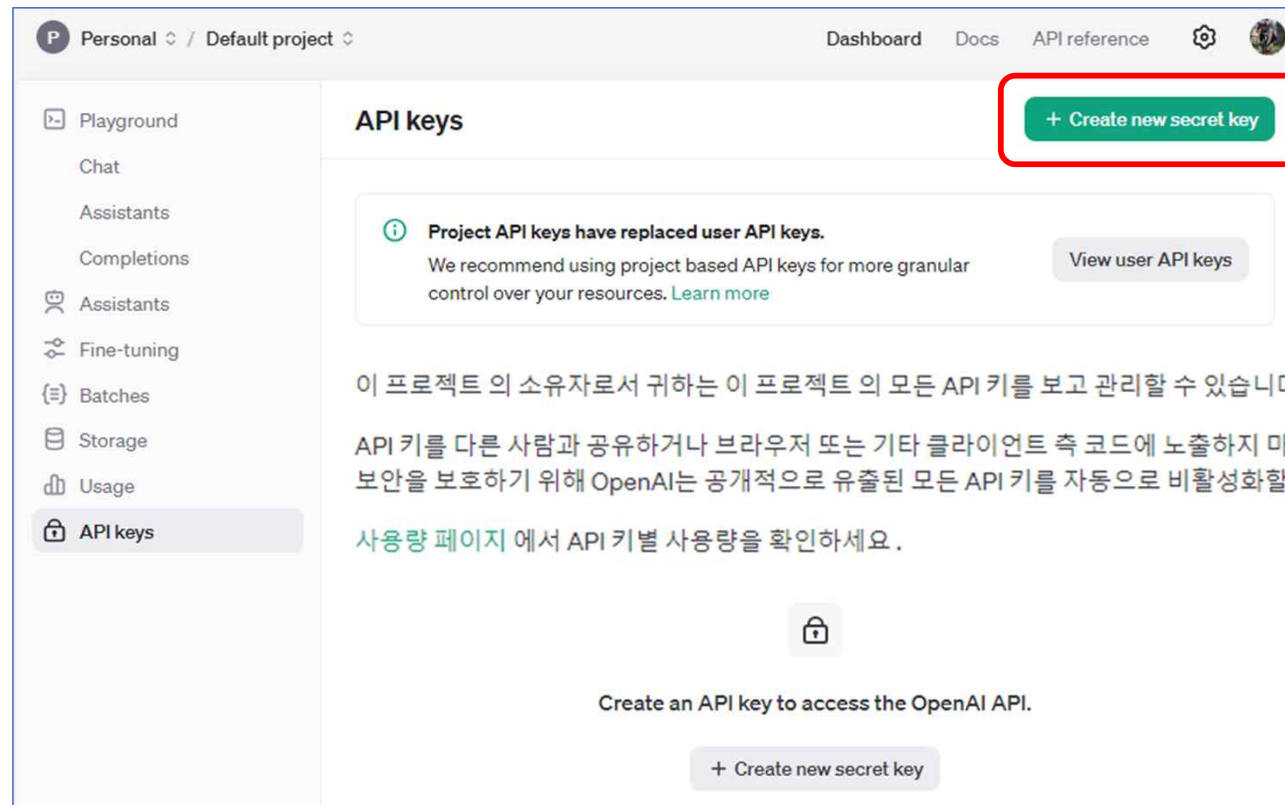
Open-AI API Program

<https://platform.openai.com/apps>



API key 생성

<https://platform.openai.com/account/api-keys>



Create new secret key

Owned by

☒ You ☐ Service account

이 API 키는 사용자와 연결되어 있으며 선택한 프로젝트에 대해 요청할 수 있습니다. 조직이 나 프로젝트에서 제거되면 이 키가 비활성화됩니다.

Name Optional

Project

Permissions

All Restricted Read Only

Cancel

Create secret key

Save your key

Please save this secret key somewhere safe and accessible. For security reasons, **you won't be able to view it again** through your OpenAI account. If you lose this secret key, you'll need to generate a new one.

3RuDSd7TQbDNViu39T3qT3B1bkFJEoTJIwcT8PhptPA

Copy

Permissions

Read and write API resources

Done

이 프로젝트의 소유자로서 귀하는 이 프로젝트의 모든 API 키를 보고 관리할 수 있습니다.

API 키를 다른 사람과 공유하거나 브라우저 또는 기타 클라이언트 측 코드에 노출하지 마십시오. 귀하의 계정 보안을 보호하기 위해 OpenAI는 공개적으로 유출된 모든 API 키를 자동으로 비활성화할 수도 있습니다.

[사용량 페이지](#)에서 API 키별 사용량을 확인하세요.





이름	비밀키	작성자:	권한	
내 테스트 키	sk-...7XGD	오영재	모두	 

생성된 API Key 관리

API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically disable any API key that we've found has leaked publicly.

NAME	KEY	CREATED	LAST USED ⓘ	
langchain1	sk-...iGKJ	2023년 7월 13일	2023년 8월 16일	 
langchainJS	sk-...m1y6	2023년 8월 6일	2023년 8월 6일	 

+ Create new secret key

Default organization

- 타인 노출 안되도록 조심
- 노출되면 반드시 삭제 후 재 생성

단일 프로젝트에 대해 API key 설정

- .env – API 키가 포함된 로컬 파일 생성

```
# .env 파일  
OPENAI_API_KEY=sk-vvDtl*****XiilEpdjLhBJaH0f
```

- .gitignore 에 .env 파일 포함

```
# .env 파일을 git에서 무시  
.env
```

- Python code

```
pip install python-dotenv
```

```
from dotenv import load_dotenv, find_dotenv
```

```
_ = load_dotenv(find_dotenv()) # local .env file을 읽어서 os.environ 에 OPENAI_API_KEY 추가
```

```
from openai import OpenAI
```

```
client = OpenAI() # os.environ.get("OPENAI_API_KEY")을 default 로 사용하여 API Key 이용
```

주요 개념

- 텍스트 생성 모델
 - 자연어와 형식 언어를 이해하도록 훈련된 모델 (예, GPT-4, GPT-3.5)
 - 프롬프트 - 입력 텍스트
 - 콘텐츠 생성, 코드 작성, 요약, 대화, 창의적 글쓰기 등
- 어시스턴트 (Assistant)
 - 대화형 AI 개발 지원 → 복잡한 대화 시나리오를 구현 가능
 - 컨텍스트 유지 → 대화의 맥락을 유지하고, 여러 턴의 대화를 자연스럽게 이어갈 수 있습니다.
 - 지시사항 및 사용자 정의 → 특정한 방식으로 응답하도록 모델을 조정할 수 있습니다.
 - 코드 실행, 파일에서 정보 검색 등을 수행할 수 있는 도구에 접근

- 토큰 (Token)

- 텍스트를 처리하는 단위로, 문자 시퀀스를 나타냄.
- 예) "tokenization"은 "token" + "ization"으로 분해됨.
1 토큰은 영어 텍스트의 경우 약 4자 또는 0.75단어.
- 토큰의 길이는 모델의 최대 컨텍스트 길이에 따라 제한됩니다.

Token 관리

- "ChatGPT is great!" → 6개의 토큰으로 인코딩
["Chat", "G", "PT", " is", " great", "!"]

`{"role": "user", "content": "2020년 코리안 시리즈를 우승한 야구팀이 어디?"}`

➔ [882], [2366, 15, 75265, 226, 3396, 66391, 29102, 31495, 230, 45618, 29102, 96064, 230, 18918, 66822, 108, 18550, 117, 24486, 24814, 120, 89359, 169, 40934, 13094, 80402, 112, 90335, 30]

- 예를 들어, API 호출이 메시지 입력에서 10개의 토큰을 사용하고 메시지 출력에서 20개의 토큰을 받은 경우 30개의 토큰에 대한 요금이 청구됩니다

OpenAI API

```
from openai import OpenAI
client = OpenAI()

completion = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a poetic assistant, 복잡한 프로그래밍 개념을 창의적인 방식으로 설명하는 데 능숙합니다."},
        {"role": "user", "content": "프로그래밍의 재귀 개념을 설명하는 시를 작성하세요."}
    ]
)
```

늘 그 자리에서 멈춰 살피던 날,
컴퓨터 속에 녹아있는 재귀의 뿌리를 알았네.

처음에는 낯설고, 이해하기 어려웠지,
함수가 자기 자신을 호출한다니 믿기 어려웠지만

어느 순간, 그 논리가 분명해지더니,
문을 열고 나무 사이로 들어가 마주친 노도,

사람도 그 곳에 심기 시작했지,
한 순간이라도 외로워하기 싫은 우리들을 위해서

끝이 없는 세계를 펼쳐놓았지,
자기 자신을 부르며 무한히 순환하는 재귀의 세계로

Chat Completion API

```
from openai import OpenAI  
client = OpenAI()
```

```
response = client.chat.completions.create(  
    model="gpt-3.5-turbo",  
    messages=[
```

```
        {"role": "system", "content": "You are a helpful assistant."},  
        {"role": "user", "content": "2020년 코리안 시리즈를 우승한 야구팀이 어디?"},  
        {"role": "assistant", "content": "2020년 한국시리즈(Korean Series) 우승 야구팀은 NC 다이노스 입니다."},  
        {"role": "user", "content": "마지막 게임이 어디에서 열렸어?"}
```

```
    ]  
)
```

메시지는 메시지 개체의 배열이어야 하며, 각 개체에는 role("system", "user" 또는 "assistant")과 content가 있습니다. 대화는 하나의 메시지만큼 짧을 수도 있고 여러 번 주고 받을 수도 있습니다.

대화는 먼저 system 메시지로 형식화되고 이어서 user 메시지와 assistant 메시지가 교대로 표시됩니다.

```
messages=[
  {"role": "system", "content": "You are a helpful assistant."},
  {"role": "user", "content": "2020년 코리안 시리즈를 우승한 야구팀이 어디?"},
  {"role": "assistant", "content": "2020년 한국시리즈(Korean Series) 우승 야구팀은 NC 다이노스 입니다."},
  {"role": "user", "content": "마지막 게임이 어디에서 열렸어?"}
]
```

- **system 메시지**는 assistant가 어떻게 행동해야 하는지에 대한 구체적인 지침을 제공. system 메시지는 선택 사항이며 system 메시지가 없는 모델의 동작은 "당신은 도움이 되는 조수입니다"와 같은 일반적인 메시지를 사용한 경우와 유사.
- **user 메시지**는 assistant가 응답할 요청이나 설명을 제공.
- **assistant 메시지**는 이전 assistant 응답을 저장. 원하는 동작의 예를 제공할 수도 있다.
- 모델에는 과거 요청에 대한 기억이 없기 때문에 사용자 지침이 이전 메시지를 참조할 때 대화 기록을 포함하는 것이 중요.
위의 예에서 사용자의 마지막 질문인 "마지막 게임이 어디에서 열렸어?" 는 이전 메시지의 context 에서만 의미가 있으므로 이전 대화 기록이 제공되어야 한다. 대화가 모델의 토큰 제한을 초과할 경우 어떤 방식으로든 줄여야 한다.

OpenAI API role 의 종류

역할(role)	설명
system	대화의 초기 설정을 정의합니다. 모델의 행동 방식을 설정하거나 대화의 전체적인 톤과 스타일을 지정하는 데 사용됩니다.
user	사용자 메시지를 나타냅니다. 모델에게 질문을 하거나 지시를 내리는 역할을 합니다.
assistant	모델 자체의 응답을 나타냅니다. 사용자의 질문이나 요청에 대한 모델의 답변을 제공합니다.
tool	함수를 호출하는 역할을 합니다. 주로 외부 API 호출이나 코드 실행 등의 작업을 위해 사용됩니다. (참고: 일부 API 버전에서만 지원될 수 있음)

이러한 역할은 대화형 세션에서 메시지의 발신자와 의도를 구별하는 데 도움을 준다. API는 이 정보를 사용하여 적절한 방식으로 반응하거나 응답.

실습: 320_Chat_Completion_Text Generation

- Chatting 역할 정의 : role parameter
- Token 관리
- 출력 제어 parameters : seed, max_token, temperature 등

실습: 60_OpenAI 종목추천

- yfinance를 이용한 주식 데이터 및 재무제표 다운로드
- 이동평균선을 이용한 상승 가능성 판단
- OpenAI API를 이용한 종목 분석 및 추천 요청