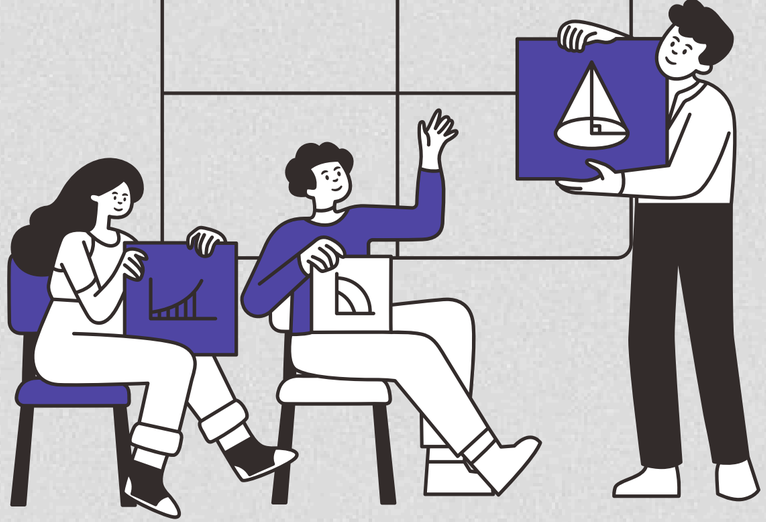
삼정KPMG Future Academy 27I

S&P 500 Analysis

노호현, 박찬영

Contents

01 프로젝트 개요



주제

S&P 500 지수 및 개별 주식 데이터 분석 솔루션 개발

목적

- 초보 투자자가 시장을 더 잘 이해할 수 있도록 간단한 분석 툴 제공
- 자동화된 데이터 처리와 예측 모델링
- 주식 및 지수 데이터를 자동으로 수집, 정제, 시각화하며 실시간 분석 가능

02 데이터 소개

S&P 500 지수의 특징

효율적 시장

주가에는 이미 모든 공개된 정보가 즉각적으로 반영되어 있으므로, 과거 데이터나 공시된 정보(X 변수)를 기반으로 미래를 예측하기 어려움

이질성

개별 주식이나 산업군이 각기 다른 요인과 관계를 가지며, 동일한 변수라도 주식마다 다른 영향을 미칠 수 있음

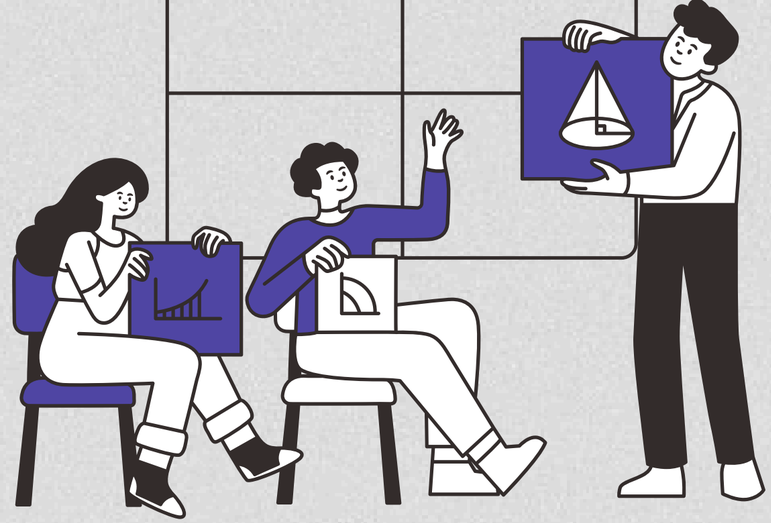
랜덤 노이즈

뉴스, 투자 심리, 정치적 사건, 자연재해 같은 비정형적이고 갑작스러운 요인들 존재

→ 고전적인 시계열 방법론 활용

- S&P 500과 같은 주가 데이터는 대개 강한 시계열 특성을 가짐
- 시계열 특성을 고려하지 않은 모델은 성능이 낮을 가능성이 높음

03 서비스 구조도



서비스 구조도

데이터 연동

yfinance 모듈로 S&P 500 및 개별 주식 데이터 실시간 로딩

세팅 설정

종목 선택
ex) S&P 500, 애플 등

기간 설정
ex) 2024.01.01 ~ 현재

파라미터 범위 설정
-각 모델에 필요한 파라미터 범위 사전 설정

데이터 개요

각 변수별 히스토그램 시각화 및 정규성 검정

시계열 분해
데이터가 시계열 모형에 적합한지 판단

데이터 분할

train/test data 분할
8:2 비율 적용

모델 학습

시계열 분석
ex) 시계열 분해, 이동평균, 지수평활, ARIMA

회귀 /ML 모델
Lasso, Ridge, Linear
XGB

AI 챗봇 어드바이저

해석이 어려운 시계열 방법론에 대해 설명하는 챗봇 어드바이저

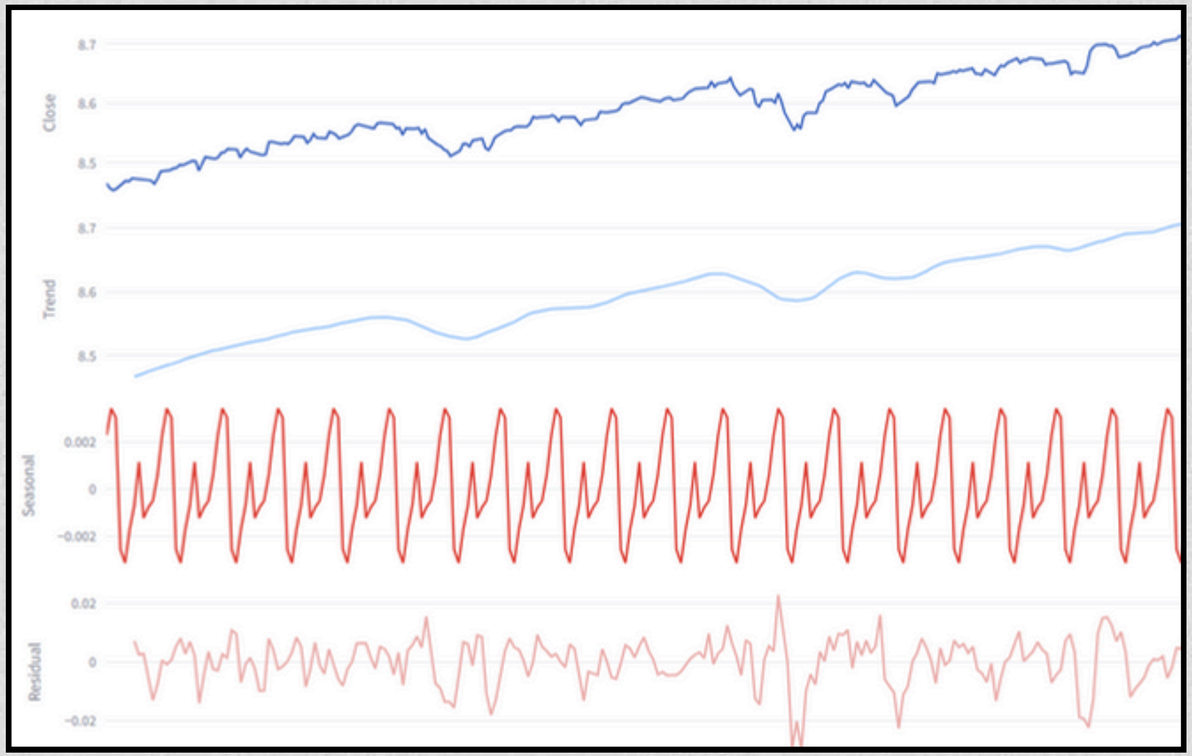
모델 적합

Loss Function 계산
최적의 모델 선택
ex) RMSE, MSE, R2 ...

Test data 예측 및 시각화

04 시계열 분해

시계열 데이터를 구성하는 요소(추세, 계절성, 잔차)를 분리하여 데이터의 구조를 이해하고, 각각의 요소를 분석하는 기법



- **추세(trend)**

  데이터의 장기적인 증가, 감소 또는 일정한 방향성을 나타내는 요소

- **계절성(Seasonal)**

  데이터의 일정 주기(예: 월별, 분기별)마다 반복되는 패턴.

- **잔차(random/residual)**

  추세와 계절성을 제거한 후 남은 데이터로, 비정상적 변동(Noise)이나 예측 불가능한 요인은 포함됨.

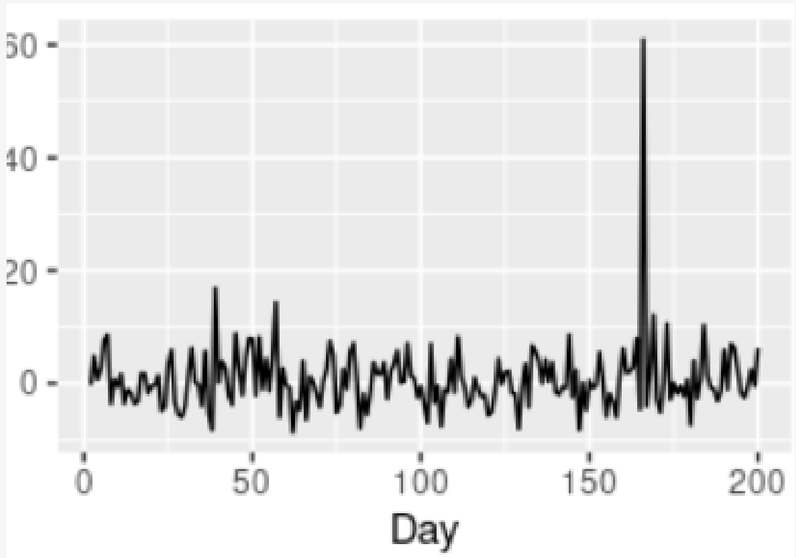
  랜덤하게 나타나지 않으면 데이터에 존재하는 특정 패턴이 모델에 의해 설명되지 않고 잔차에 존재!

04 정상성 판단(그래프)

정상성(Stationarity)이란?

시계열 데이터의 평균과 분산, 자기상관이 시간에 따라 변하지 않는 특성

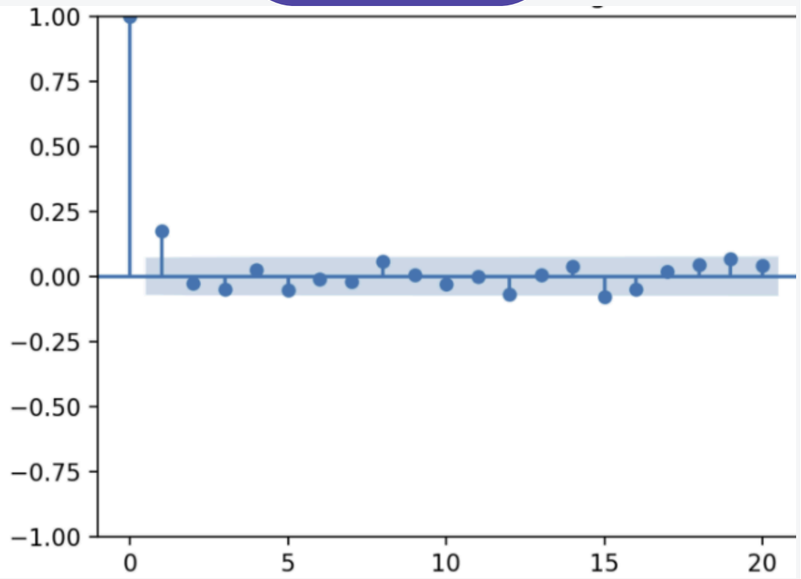
〈1〉 그래프



직관적으로 데이터 그래프를 보고 판단

=> 그래프 상에서 시간에 따라 평균과 분산이 크게 달라지지 않는 것으로 보이면 정상성으로 판단 가능

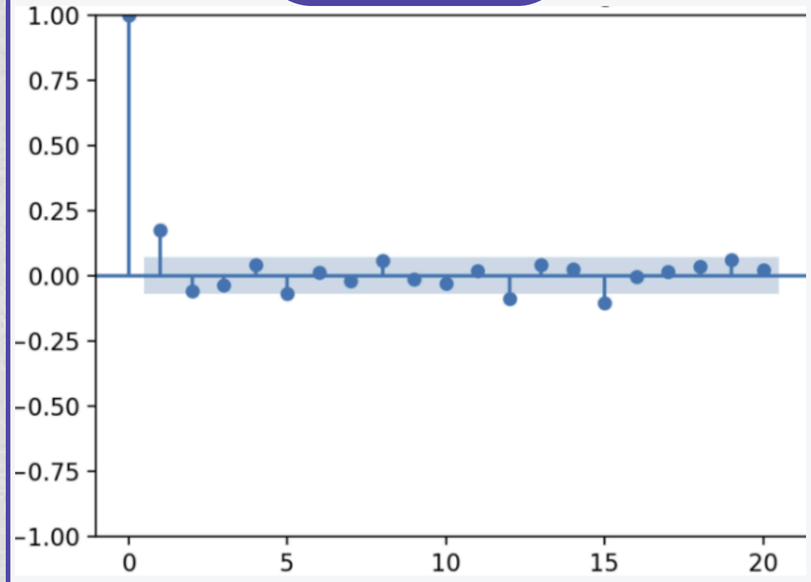
〈2〉 ACF



현재 시점의 자료와 시점 차이(Lag)를 가진 자료와의 자기상관계수

=> 값이 빠르게 0으로 수렴하면 정상성으로 판단!

〈3〉 PACF



중간에 있는 다른 시차의 영향을 제거한 상태에서 자기상관성 계산

=> 값이 빠르게 0으로 수렴하면 정상성으로 판단!

04 정상성 판단(검정)

〈4〉 ADF TEST

귀무가설(H0) : 데이터는 비정상적이다 (단위근이 존재).

대립가설(H1) : 데이터는 정상적이다 (단위근이 없다).

유의확률과 유의수준을 비교하여 검정 판단

=> P-value <= 유의수준이면 정상성으로 판단 가능

〈5〉 KPSS TEST

귀무가설(H0) : 데이터는 정상적이다 (단위근이 없다).

대립가설(H1) : 데이터는 비정상적이다 (단위근이 존재).

유의확률과 유의수준을 비교하여 검정 판단

=> P-value > 유의수준이면 정상성으로 판단 가능

ADF Test와 KPSS Test를 함께 사용하여 정상성 판단

- ADF Test는 데이터가 비정상적인지 확인하고, KPSS Test는 데이터가 정상적인지 확인

04 정상성 변환

정상성 변환

시계열 분석론은 정상성 데이터를 가정하고 수행되는 경우가 많기 때문에 정상성을 만족하도록 변환 수행이 필요함

차분

1차 차분(1st differencing)

$$X_t' = X_t - X_{t-1}$$

2차 차분(2nd differencing)

$$X_t'' = X_t' - X_{t-1}'$$

추세(trend)를 제거
보통 1차 또는 2차 차분 사용

변환

로그 변환

$$X_t' = log\,(X_t)$$

제곱근 변환

$$X_t = \sqrt{X_t}$$

차분과 로그 변환을 함께 사용하여 추세 제거와
분산 안정화를 동시에 수행
로그 변환 후 차분 진행이 일반적인 순서

평활화

이동평균

윈도우 평활화

지수평활법

시계열 데이터에서 노이즈(Noise)를 제거하여 데이터의
전반적인 추세(Trend)나 패턴(Pattern)을
더 명확하게 드러냄

04 단순이동평균

1> 단순 이동 평균(SMA)

지정된 기간 동안의 데이터 값의 평균을 계산하여 시계열 데이터의 추세를 파악하거나 노이즈를 제거하는 방법

수식

$$SMA_t = \frac{P_t + P_{t-1} + P_{t-2} + \cdots + P_{t-n+1}}{n}$$

- SMA_t: 현재 시점 t에서의 단순이동평균
- P_t: 시점 t에서의 값
- n: 이동평균에 사용할 기간(예: 5일, 10일 등)

- 모든 관측치가 같은 가중치를 가진다

n = 3일 때의 단순이동평균

t	1	2	3	4	5
P_t	100	102	101	99	98
A_t			101	100.66	99.33

$$A_3 = \frac{(P_1 + P_2 + P_3)}{3} = \frac{(100 + 102 + 101)}{3} = 101$$

$$A_4 = \frac{(P_2 + P_3 + P_4)}{3} = \frac{(102 + 101 + 99)}{3} = 100.66$$

04 가중이동평균

2> 가중 이동 평균(WMA)

지정된 기간 동안의 데이터 값에 서로 다른 가중치를 부여하여 평균을 계산함으로써, 시계열 데이터의 추세를 파악하고 최신 데이터의 변화를 더 민감하게 반영하는 방법.

수식

$$WMA_t = \frac{\sum_{i=1}^{n} w_i \cdot P_{t-i+1}}{\sum_{i=1}^{n} w_i}$$

- WMA_t: 현재 시점 t에서의 가중이동평균
- P_{t-i+1}: 시점 $t - i + 1$에서의 값
- w_i: 각 데이터 포인트에 부여된 가중치
- n: 이동평균에 사용할 기간

- 각 관측치가 서로 다른 가중치를 가진다

W_i 설정

선형 가중치	w1=5.w2=4.w3=3.w4=2.w5=1 회신 데이터에 가장 높은 가중치 부여. 과거로 갈수록 점진적으로 감소.
지수 가중치	wi=(1−α)·αi−1 가중치가 지수적으로 감소. 회신 데이터에 가장 큰 가중치를 부여.
삼각 가중치	w1=1.w2=2.w3=3.w4=2.w5=1 중앙값 근처에 가장 높은 가중치를 부여하고, 양쪽으로 갈수록 대칭적으로 가중치가 감소.
Reverse 가중치	w1=1.w2=2.w3=3.w4=4.w5=5. 선형 가중치의 Reverse

04 지수평활법

3> 지수평활법

최근 데이터에 더 높은 가중치를 부여하면서 과거 데이터를 지수적으로 감소시키며, 시계열 데이터의 추세를 예측하거나 평활화(Smoothing)하기 위한 방법

수식

$$S_t = \alpha \cdot P_t + (1 - \alpha) \cdot S_{t-1}$$

- S_t: 시점 t에서의 평활화된 값(예측값)
- P_t: 시점 t에서의 실제 데이터 값
- S_{t-1}: 시점 $t-1$에서의 평활화된 값(이전 예측값)
- α: 평활화 상수(Smoothing Constant, $0 < \alpha \leq 1$)

a = 0.5일 때의 지수평활법

t	1	2	3	4	5
P_t	100	102	101	99	98
S_t	100	101	101	99.5	99.25

$$S_2 = 0.5 \times P_2 + (1 - 0.5) \times S_1 = 0.5 \times 102 + 0.5 \times 100 = 101$$

$$S_3 = 0.5 \times P_3 + (1 - 0.5) \times S_2 = 0.5 \times 101 + 0.5 \times 101 = 101$$

04 ARIMA

4> ARIMA

AR(AutoRegressive) Model + MA(Moving Average) + differencing(차분)을 진행한 시계열 예측 모델

AR Model

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \epsilon_t$$

- X_t: 현재 시점 t의 값
- ϕ_i: 자기회귀 계수
- p: 과거 값의 개수 (AR 차수)
- ϵ_t: 백색 잡음 (랜덤 오차)

현재 값이 과거 값들의 선형 결합으로 설명된다는 가정

- 데이터 간의 자기상관을 모델링.
- 과거 데이터가 현재 데이터에 영향을 준다고 가정.

+

MA Model

$$X_t = \mu + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

- X_t: 현재 시점 t의 값
- μ: 평균
- θ_i: 이동평균 계수
- q: 과거 오차 항의 개수 (MA 차수)
- ϵ_t: 백색 잡음

현재 값이 과거 오차 항의 선형 결합으로 설명된다는 가정

SMA와 다른 모델

- 데이터의 불규칙성을 설명.
- 노이즈가 시계열 데이터에 미치는 영향을 모델링.

+ 차분

04 ARIMA

4> ARIMA(p, d, q)

정상성 검정 → 정상성 변환 ex) 차분 → 여러 조합에 대해 테스트 **or** ACF/PACF로 p와 q 결정 → AR 모델 + MA 모델 → ARIMA(p,d,q)

ARIMA

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$

AR MA

- c: 상수 항
- ϕ: 자기회귀 계수
- θ: 이동평균 계수
- ϵ_t: 백색잡음(white noise)

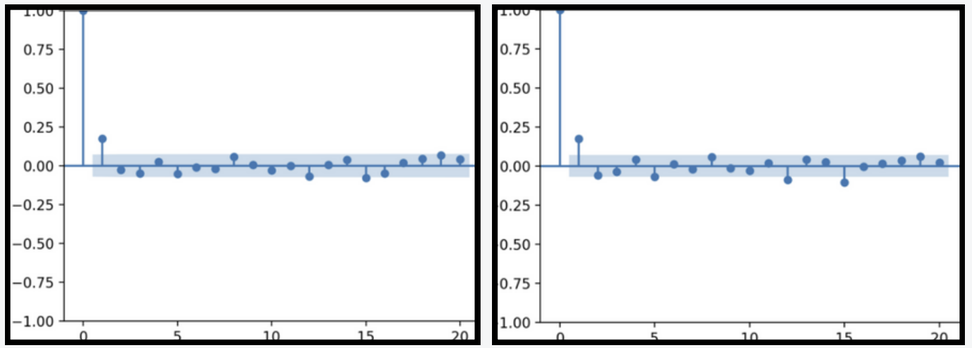
p : AR 모델 차수

d : 차분의 Lag

q : MA 모델 차수

p, q 결정

1.ACF/PACF로 판단



처음 신뢰 구간 밖으로 벗어나는 Lag를 p, q를 각각 설정

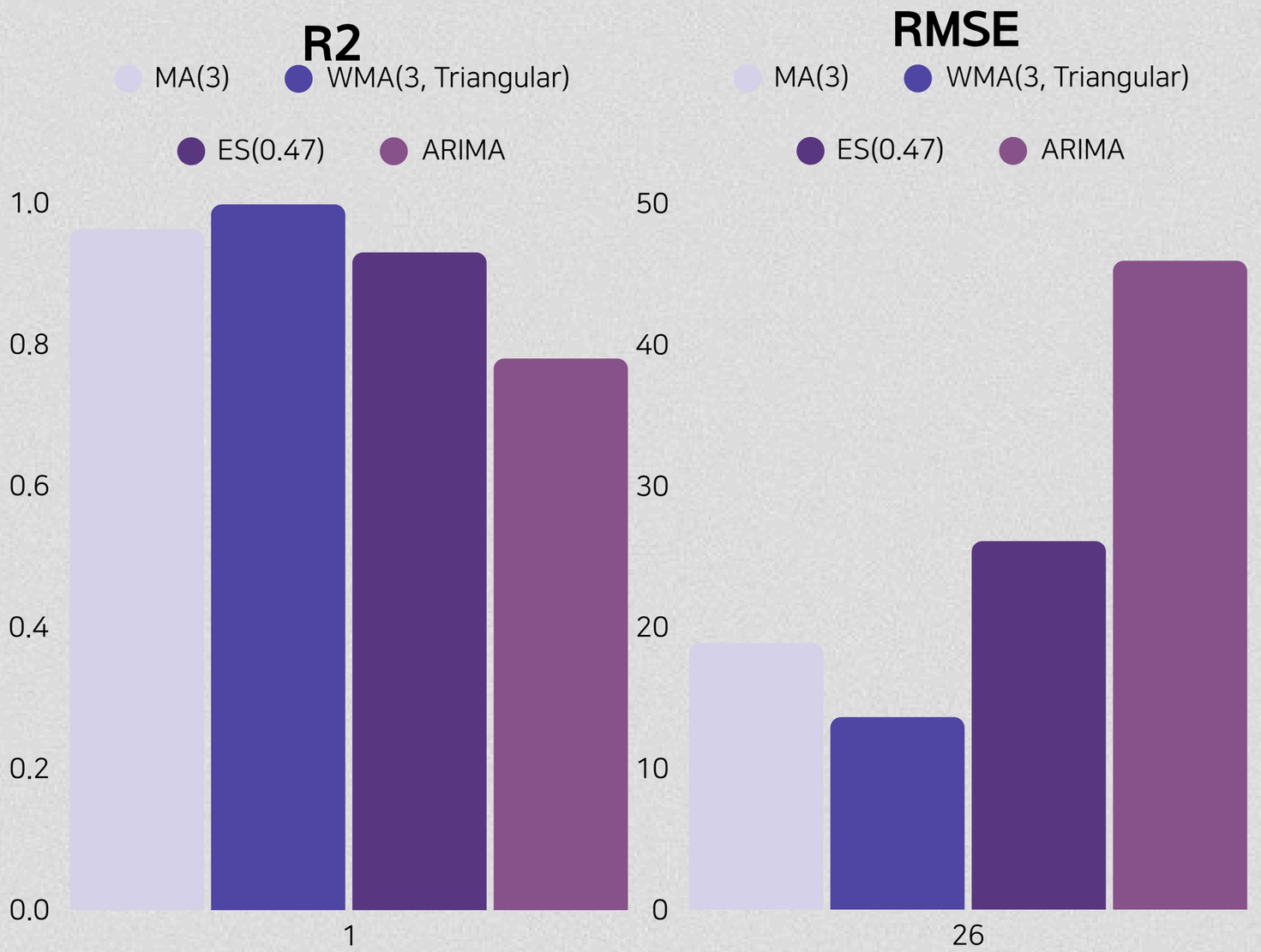
2. 파라미터 조합에 따른 AIC 값으로 판단

AIC가 가장 작은 조합의 p, q로 모델 적합

04

시계열 모델 예측 성능

기간 : 1927.12.30 ~ 2025.01.10
Loss Function : R2, RMSE



05 선형 회귀

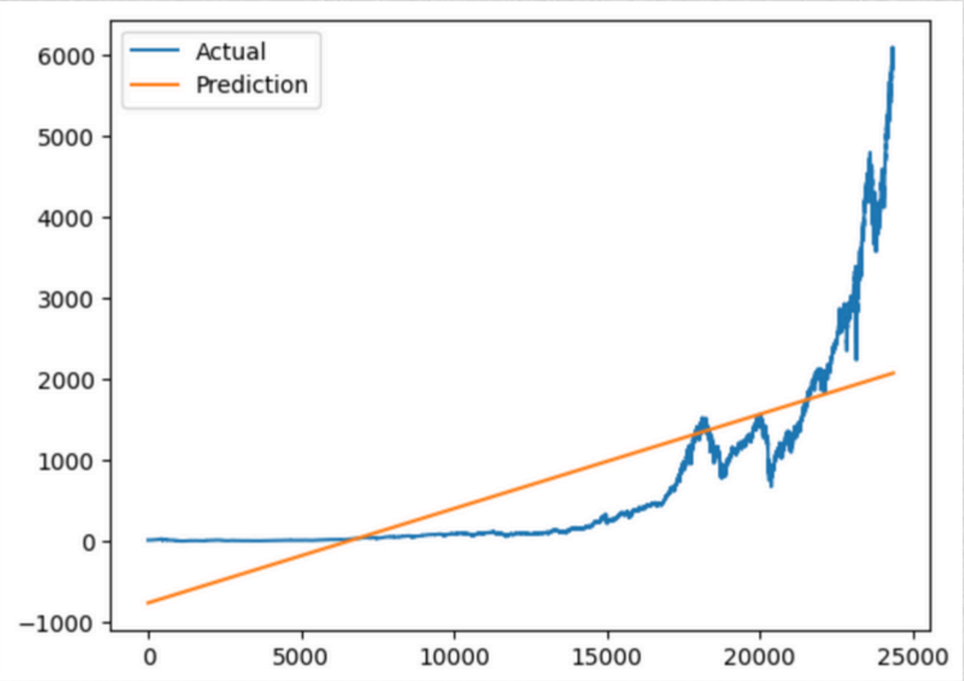
	Close
0	17.660000
1	17.760000
2	17.719999
3	17.549999
4	17.660000
5	17.500000
6	17.370001
7	17.350000
8	17.469999
9	17.580000

	t	Close
0	0	17.660000
1	1	17.760000
2	2	17.719999
3	3	17.549999
4	4	17.660000
5	5	17.500000
6	6	17.370001
7	7	17.350000
8	8	17.469999
9	9	17.580000

```
data = data.reset_index(drop=False)
data = data.rename(columns={'index':'t'})
```

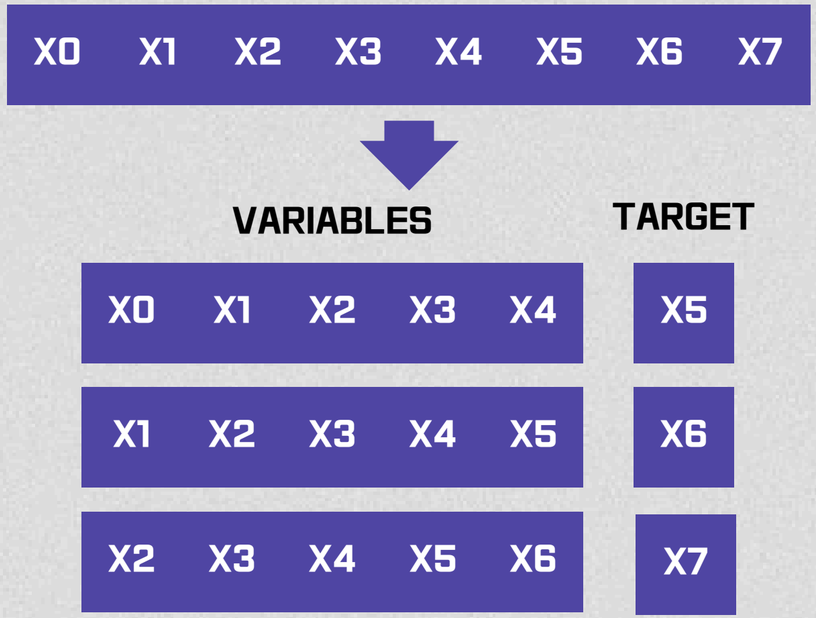
- 선형 회귀

$$X_t = wt + b$$




```
회귀 계수 (Slope):0.116
절편 (Intercept): -756.904
R^2 train 0.565
```

05 머신 러닝



- **ML model**

$$X_t = f(\underbrace{X_{t-1}, X_{t-2}, X_{t-3}, X_{t-4}, \cdots}_{Window\_size})$$

```python
def create_windows_with_labels(data, window_size):
    X, y = [], []
    for i in range(len(data) - window_size):
        X.append(data[i:i+window_size])  # 윈도우 데이터
        y.append(data[i+window_size])    # 윈도우 다음 값
    return np.array(X), np.array(y)
```

	X_t-5	X_t-4	X_t-3	X_t-2	X_t-1
0	17.660000	17.760000	17.719999	17.549999	17.660000
1	17.760000	17.719999	17.549999	17.660000	17.500000
2	17.719999	17.549999	17.660000	17.500000	17.370001
3	17.549999	17.660000	17.500000	17.370001	17.350000
4	17.660000	17.500000	17.370001	17.350000	17.469999

	X_t
0	17.500000
1	17.370001
2	17.350000
3	17.469999
4	17.580000

시연

06

질의 응답

Q & A

감사합니다.