거시경제 데이터 전처리

방법 1. 피처를 lag 피처로 사용

변수별 최적 Lag 및 상관계수:

	macro_variable	lag	corr
0	Brazil_Agricultural raw materials exports (% o...	3	0.942776
1	Brazil_Cereal yield (kg per hectare)	3	0.559131
2	Brazil_Export unit value index (2015 = 100)	1	0.545481
3	Brazil_Food production index (2014-2016 = 100)	3	0.759359
4	Brazil_GDP per capita (current US$)	2	-0.883611
5	Brazil_GDP per capita growth (annual %)	3	0.458918
6	Brazil_IMF repurchases and charges (TDS, curre...	3	0.838904
7	Brazil_Merchandise exports to low- and middle-...	2	-0.904045
8	Brazil_Merchandise trade (% of GDP)	1	0.798242
9	Brazil_Permanent cropland (% of land area)	3	-0.600309
10	Brazil_Political Stability and Absence of Viol...	3	-0.703351
11	Brazil_Production	3	0.530129
12	Brazil_Rural population	3	-0.818579
13	Brazil_Unemployment, male (% of male labor for...	3	0.315422
14	Colombia_Agricultural raw materials exports (%...	2	0.641080
15	Colombia_Cereal yield (kg per hectare)	2	0.926166
16	Colombia_Export unit value index (2015 = 100)	3	0.493691
17	Colombia_Food production index (2014-2016 = 100)	1	0.615077
18	Colombia_GDP per capita (current US$)	2	-0.670936
19	Colombia_GDP per capita growth (annual %)	1	0.387284
20	Colombia_IMF repurchases and charges (TDS, cur...	3	0.838913

최대 3으로 lag를 설정하여 상관계수 분석을 통해 피처별 최적 lag를 설정하여 피처를 생성하였다.

Year	Brazil_Agricultural raw materials exports (% of merchandise exports)_lag3	Brazil_Cereal yield (kg per hectare)_lag3	Brazil_Export unit value index (2015 = 100)_lag1	Brazil_Food production index (2014-2016 = 100)_lag3	Brazil_GDP per capita (current US$)_lag2	Brazil_GDP per capita growth (annual %)_lag3	Brazil_IMF repurchases and charges (TDS, current US$)_lag3	Brazil_Merchandise exports to low- and middle-income economies within region (% of total merchandise exports)_lag2	Brazil_Merchandise trade (% of GDP)_lag1
2015	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2016	NaN	NaN	100.0	NaN	NaN	NaN	NaN	NaN	20.377290
2017	NaN	NaN	93.3	NaN	8936.196618	NaN	NaN	14.451894	18.086272
2018	4.809225	5000.6	102.9	101.68	8836.286526	-4.306398	2024533.3	15.155146	18.456033
2019	4.959097	4181.7	108.3	99.34	10080.509280	-4.010127	3390352.7	15.790355	22.156737
2020	4.735486	5209.6	104.3	106.98	9300.661649	0.587669	2141124.2	13.378489	22.115604
2021	5.425976	4806.5	97.2	107.51	9029.833267	1.090409	3864346.2	11.936232	25.439615
2022	5.761630	5321.4	125.9	109.12	7074.193783	0.562971	39632203.4	10.350316	30.856601
2023	5.662896	5255.4	143.1	111.97	7972.536650	-3.835505	12608650.8	10.350316	32.090441

일간 데이터인 커피가격데이터와 merge를 하면 같은 년도에 대해서는 동일한 값이므로 정보량이 부족해진다.

따라서 학습이 제대로 되지 않아 모델의 설명력이 떨어져 예측이 제대로 진행되지 않으므로 해당 방법을 사용하지 않았다.

방법 2. 스플라인 보간법 사용 -> 현재 사용한 방법

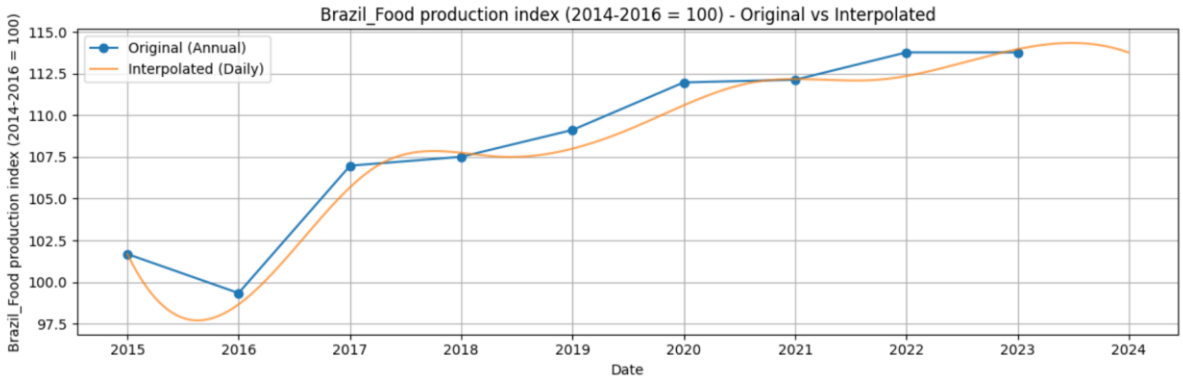
스플라인 보간법은 데이터 포인트 사이를 부드러운 곡선으로 이어주는 방식이다.

연간 값 사이의 간격을 일정한 곡률을 가진 곡선으로 채워넣어 연간 값 사이의 추세 또는 변화 방향을 일정하게 반영하는 효과를 가질 수 있다.

위에 lag 피처를 사용하는 것 처럼 년도별로 같은 값을 사용하게 되는 문제를 해결하기

위해 선형보간을 사용하게 되면 년도 사이에 계속 일정하게 증가 혹은 감소를 하게 된다.

하지만 실제 거시경제지표는 년도사이에 일정하게 증가, 감소를 하지 않으므로 데이터를 오히려 왜곡시키게 된다. 따라서 스플라인 보간법을 사용하여 연간 추세를 더 부드럽게 연결하여 일간 수준의 데이터로 추정하는 방법을 사용하였다.

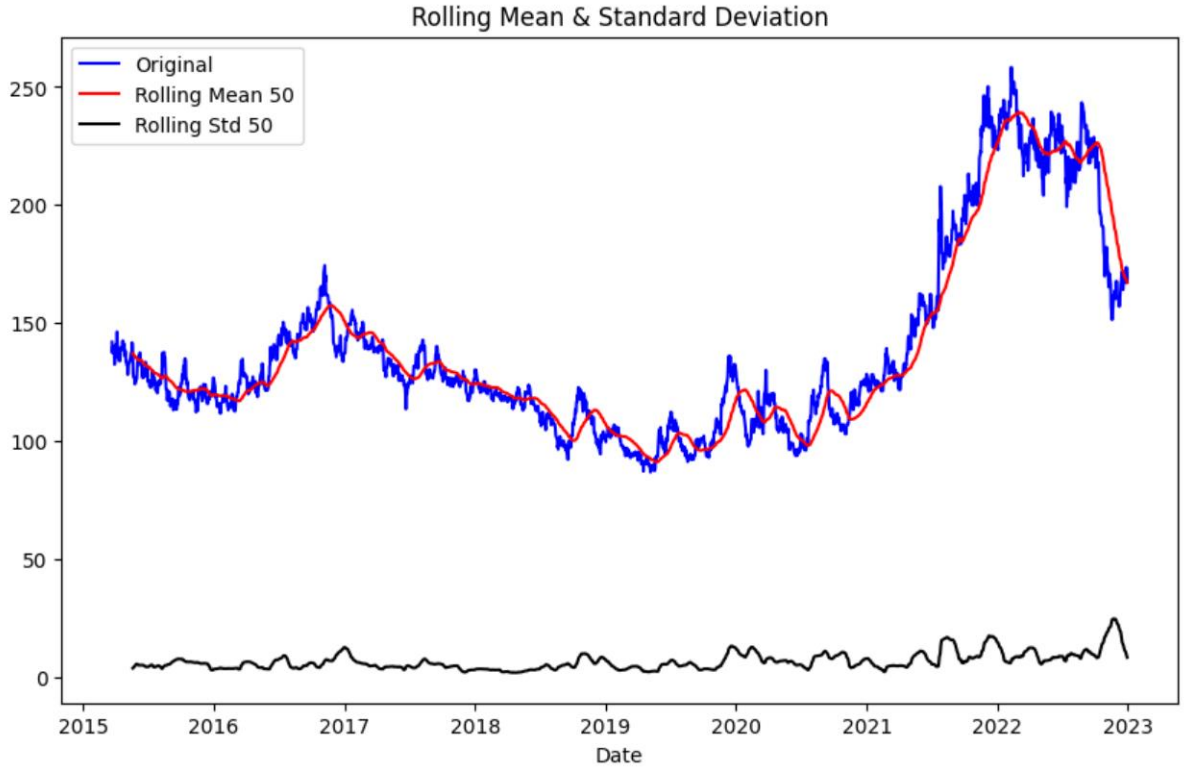


해당 방식 또한 갑작스러운 변동은 잘 반영하지 못하고 계단식 변화인 것을 오히려 스무스하게 만들어 왜곡을 발생시킬 수 있다.

커피 가격 예측을 위한 모델 학습

- ARIMA 모델

원본시계열, 이동평균, 이동표준평균 시각화



상승과 하락이 반복되지만 이동표준편차가 큰 차이 없이 유지된다.

➔ 가격의 변동폭이 크게 차이 없음을 의미함 데이터의 계절성이 딱히 없다.
➔ 임의 보행 모형으로 추정된다.

이동 평균선이 실제 데이터의 흐름을 잘 따라가므로 이동평균이 데이터의 주요 추세를 잘 반영하고 있음을 의미한다.

1. 시계열 정상성 확인(ADF 테스트)

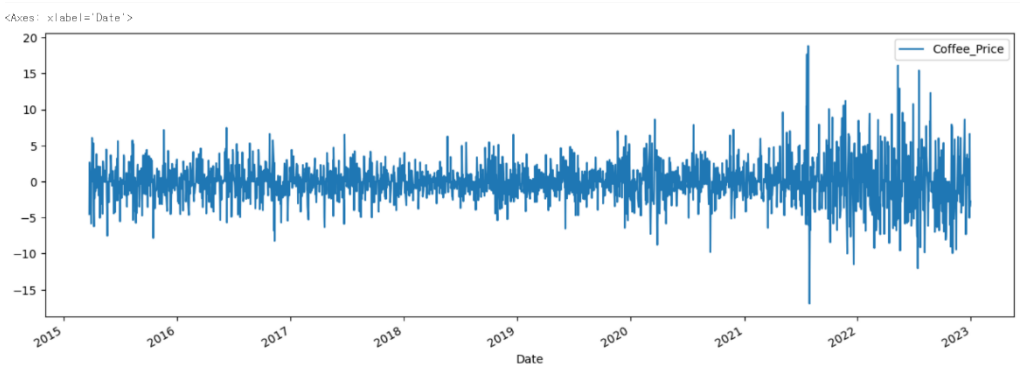
시계열 데이터의 정상성이 가정되어야하므로 정상성 검증을 실행한다.

```
ADF TEST 결과
ADF Statistics: -1.594614
p-value: 0.486298
num of lags: 0.000000
num of observations: 2265.000000
Critical values:
        1%: -3.433
        5%: -2.863
        10%: -2.567
```

p-value 가 0.05를 넘어 귀무가설을 기각할 수 없으므로 비정상성 데이터임을 의미한다.

따라서 1차 차분을 진행한다.

- 1차 차분 진행 후 ADF 테스트



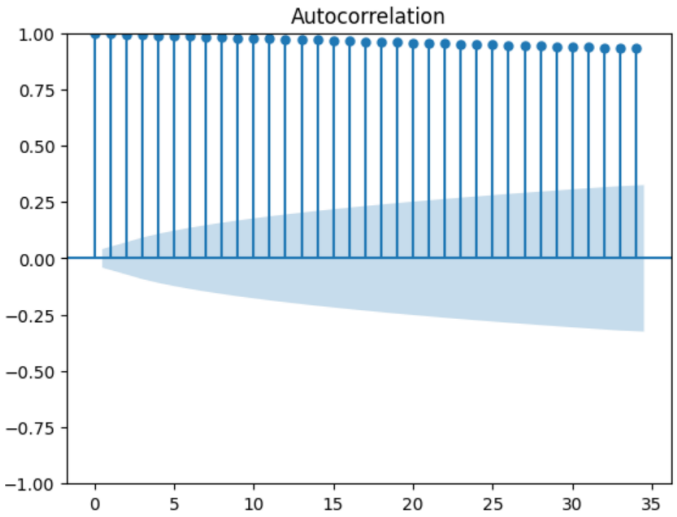
ADF TEST 결과
ADF Statistics: -46.966049
p-value: 0.000000
num of lags: 0.000000
num of observations: 2264.000000
Critical values:
1%: -3.433
5%: -2.863
10%: -2.567

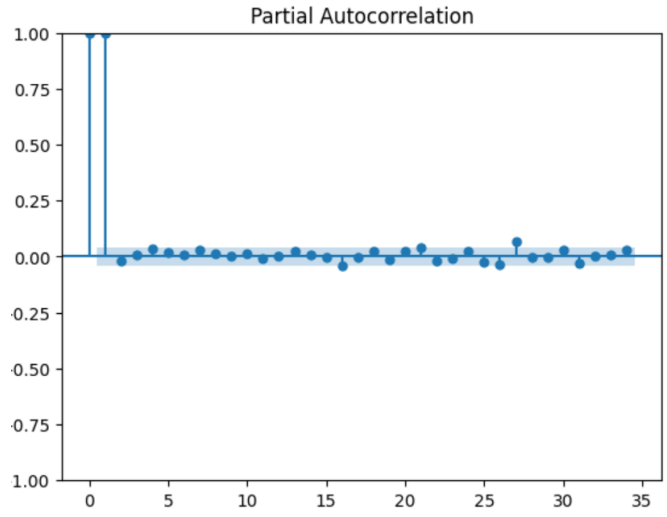
1차 차분을 통해 시계열 정상성을 확보한다.

2. ARIMA(p,d,q) 모수 추정

- ACF, PCAF 검정

<원본 데이터에 대한 검정>

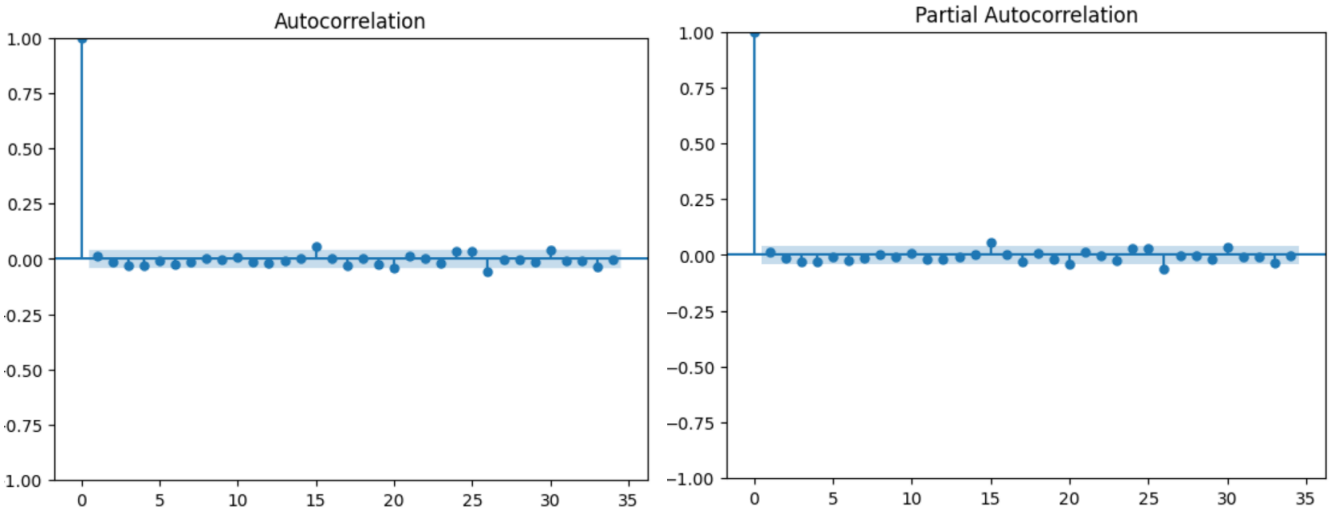




- ACF plot가 천천히 감소되는 것으로 주기에 따라 일정하지 않은 비정상성 데이터다 >> 차분이 필요하다는 의미

- PACF plot에서 첫 번째 시점에서만 강한 상관관계를 보이고, 이후 급격히 0으로 수렴하므로 이는 첫 번째 시점까지만 과거 값이 영향을 미치고 이후에는 직접적인 영향이 없다는 의미다.

<1차 차분 데이터에 대한 검정>



1차 차분 데이터에 대해 검증해보았을 때 d는 1이며 시자 p와 q이후 바로 0에 가까워지므로 p와 q모두 0이라고 추정할 수 있다.

- auto_arima 사용을 통한 차수 결정 및 모델 학습

```
Best model:  ARIMA(0,1,0)(0,0,0)[0]
Total fit time: 8.162 seconds
```

ARIMA(0,1,0)이 가장 AIC값이 낮게 나옴 즉, 임의 보행 모형이다.

SARIMAX Results

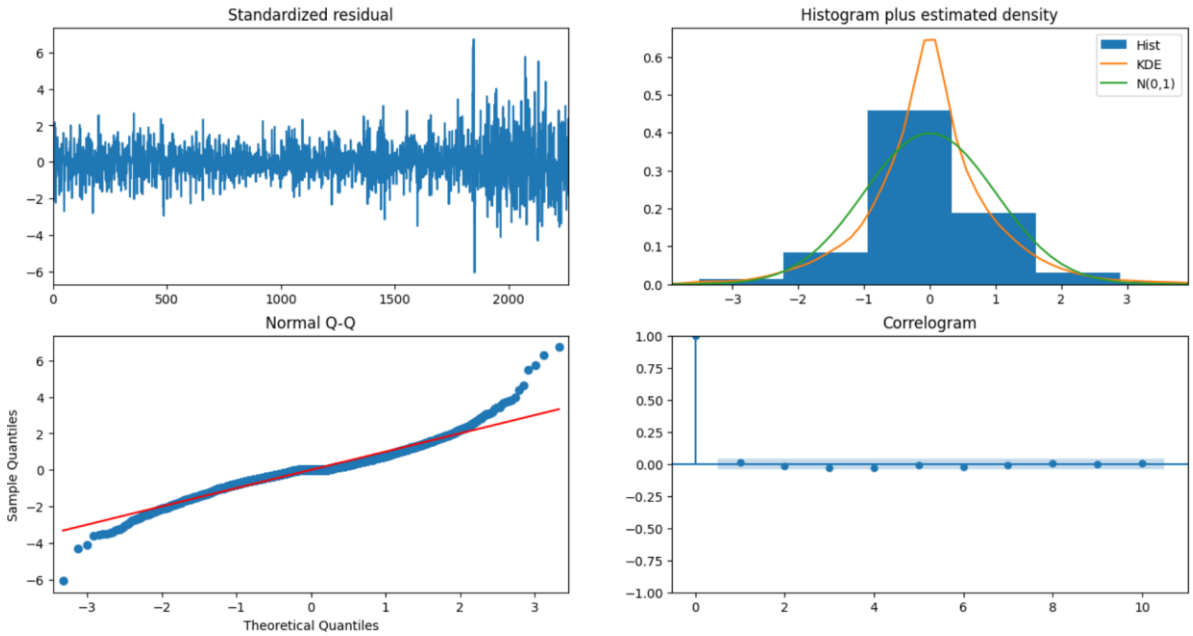
Dep. Variable:	y	No. Observations:	2266
Model:	SARIMAX(0, 1, 0)	Log Likelihood	-5542.794
Date:	Mon, 31 Mar 2025	AIC	11087.587
Time:	11:32:56	BIC	11093.313
Sample:	0	HQIC	11089.676
	- 2266		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
sigma2	7.8179	0.127	61.494	0.000	7.569	8.067

Ljung-Box (L1) (Q):	0.38	Jarque-Bera (JB):	2111.89
Prob(Q):	0.54	Prob(JB):	0.00
Heteroskedasticity (H):	2.95	Skew:	0.37
Prob(H) (two-sided):	0.00	Kurtosis:	7.67

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step)

임의 보행 모형이 잘 적합되었고 남은 잔차는 더이상 자기상관을 가지지 않는 백색 잡음이다. 하지만 잔차가 정규성을 따르지 않으며 이분산성을 가지고 있다.

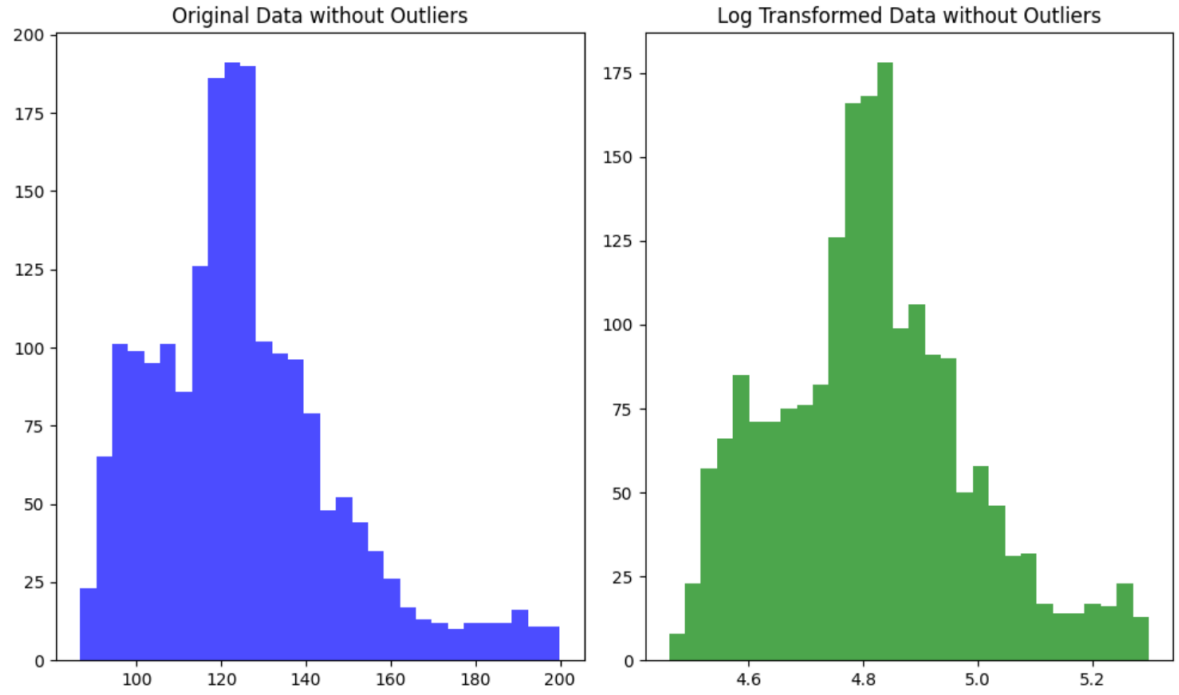


Histogram plus estimated density와 normal q-q plot을 보았을 때 정규분포에 벗어난 것을 확인할 수 있다. ARIMA는 잔차의 정규성을 가정하고 있으므로 잔차의 정규성을 최대한 만족시키기 위한 변형을 시도해보았다.

3. 데이터 변형 후 변형 데이터에 대한 모델 학습

a) 이상치 제거 후 log 변환

- 원본 데이터와 비교한 데이터 분포 변화

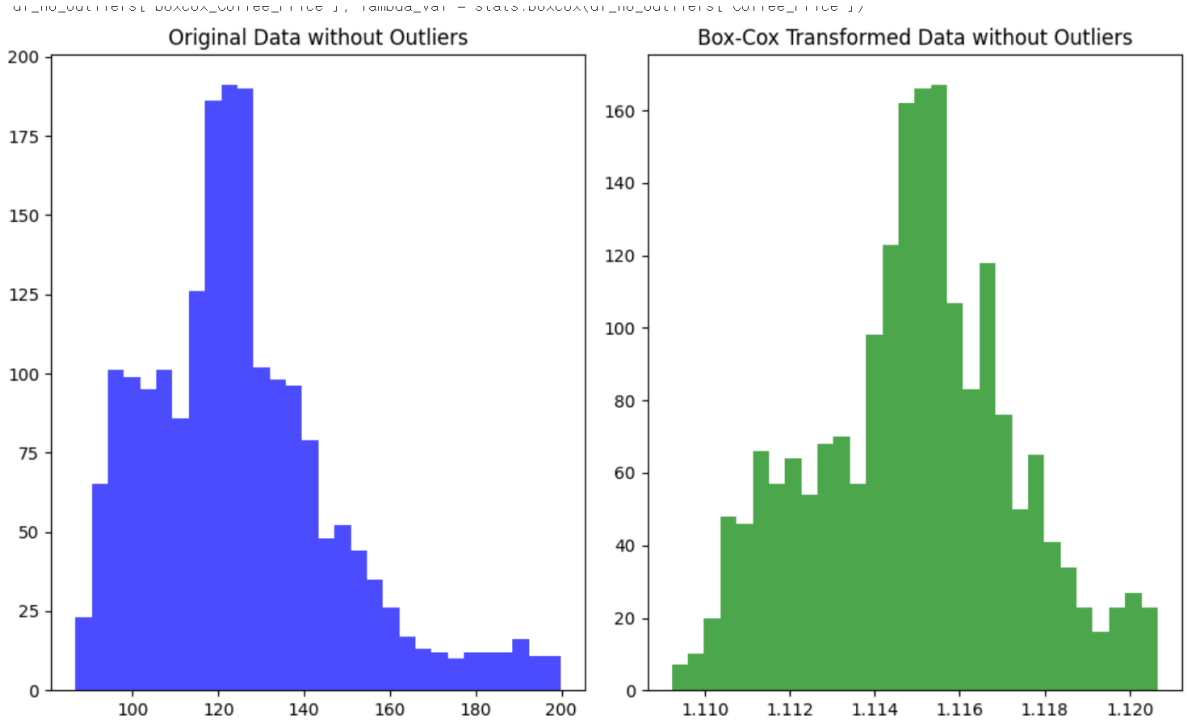


분포가 이전에 비해 조금 덜 치우쳐지게 되었다.

이전 모델에 비해서는 정규성이 회복되었지만 log변환이 아닌 box-cox변환을 시도하였다.

b) 이상치 제거 후 Box-Cox 변환

- 원본 데이터와 비교한 데이터 분포 변화



log변환 했을 때에 비해 분포가 더 골고루 퍼치게 된다.

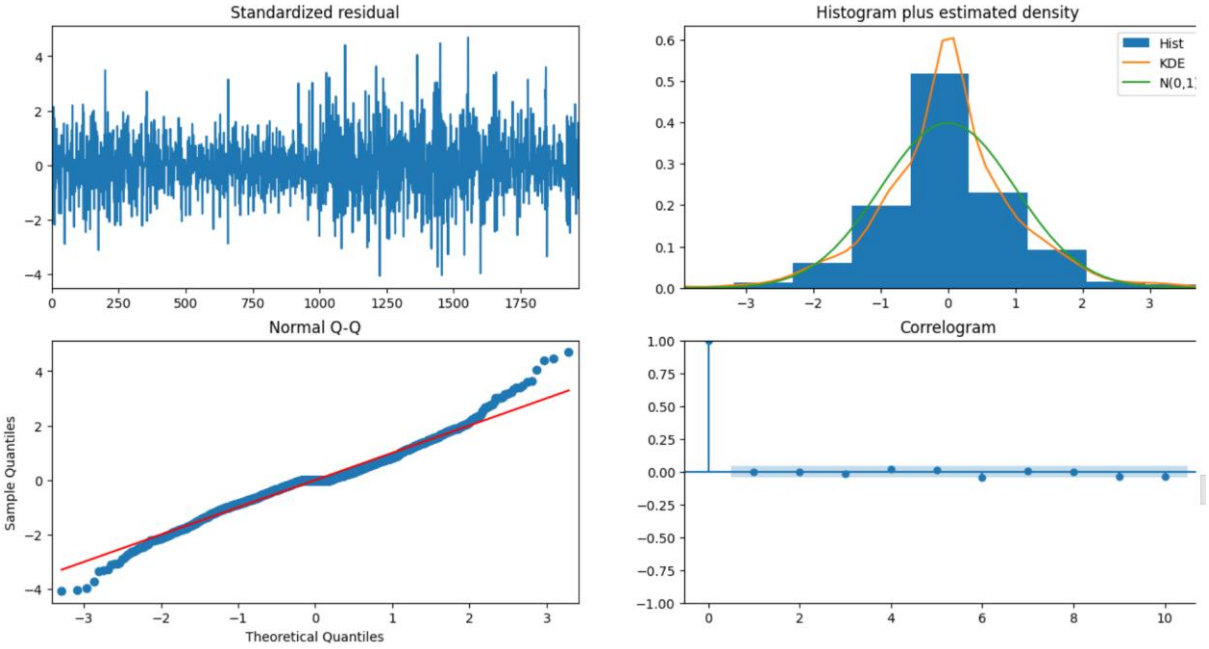
이상치 제거 후 Box-Cox 변환을 해준 데이터에 대해 모델 학습을 진행해준다.

SARIMAX Results

Dep. Variable:	y	No. Observations:	1969
Model:	SARIMAX(0, 1, 0)	Log Likelihood	13426.042
Date:	Mon, 31 Mar 2025	AIC	-26850.083
Time:	11:53:07	BIC	-26844.499
Sample:	0	HQIC	-26848.031
	- 1969		
Covariance Type:	opg		

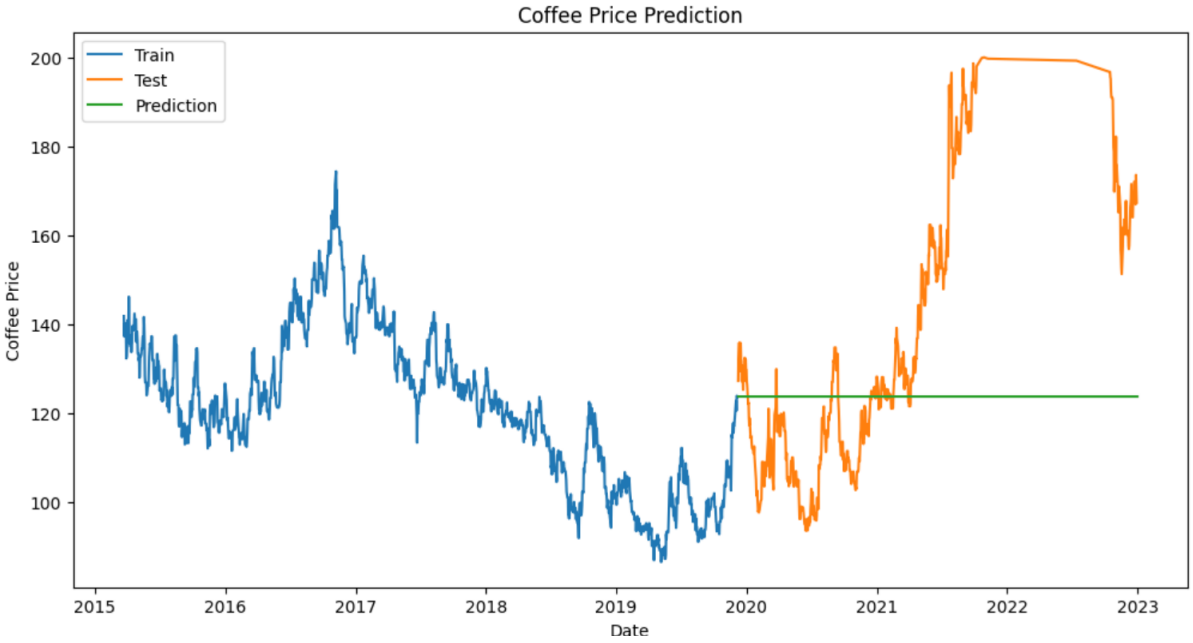
	coef	std err	z	P>|z|	[0.025	0.975]
sigma2	6.944e-08	1.59e-09	43.678	0.000	6.63e-08	7.26e-08

Ljung-Box (L1) (Q):	0.04	Jarque-Bera (JB):	300.08
Prob(Q):	0.83	Prob(JB):	0.00
Heteroskedasticity (H):	1.76	Skew:	0.16
Prob(H) (two-sided):	0.00	Kurtosis:	4.89



여전히 Prob(JB)가 0.05를 넘지 못해 정규성을 만족하지 못하지만 Histogram plus estimated density를 보았을 때 이전에 비해서는 정규분포와 비슷한 대칭을 보이고 있다는 것을 파악할 수 있다. 정규성은 따르지 않지만 데이터 변환을 통해 이전에 비해 정규분포와 비슷한 양상을 가지게 되었다. 따라서 잔차의 정규성을 따른다고 해서 모델의 성능이 꼭 높아진다는 것 또한 없으므로 일단 이 상황에서 예측을 실행하고 결과를 확인해보겠다.

4. 모델 학습 방법 수정 및 결과 확인



ARIMA(0,1,0)은 자기회귀, 이동 평균을 하지 않고 1차 차분만을 진행함. 따라서 현재 값은 이전 값에 오차(백색 잡음)이 더해진 것 뿐이다

데이터에 특정한 주기나 추세가 없기 때문에, AIC로 모형을 최적화를 하는 과정에서 의미있는 자기 상관 (AR)이나 이동 평균 (MA)를 찾기 어려웠기 때문에 최선책으로 임의 보행 모형(특별한 패턴 없이 랜덤하게 움직이는 시계열)이 데이터를 가장 잘 설명한다는 결론을 내렸다. 이는 데이터에서 어떠한 구조를 보기 어렵기 때문에, 가장 마지막 관측치가 가장 좋은 예측치다라고 판단하므로 예측값이 수평으로 나오게 된다.

그러나, 실제로 ARIMA로 양질의 예측을 하려면 위 방법을 사용할 수는 없다.

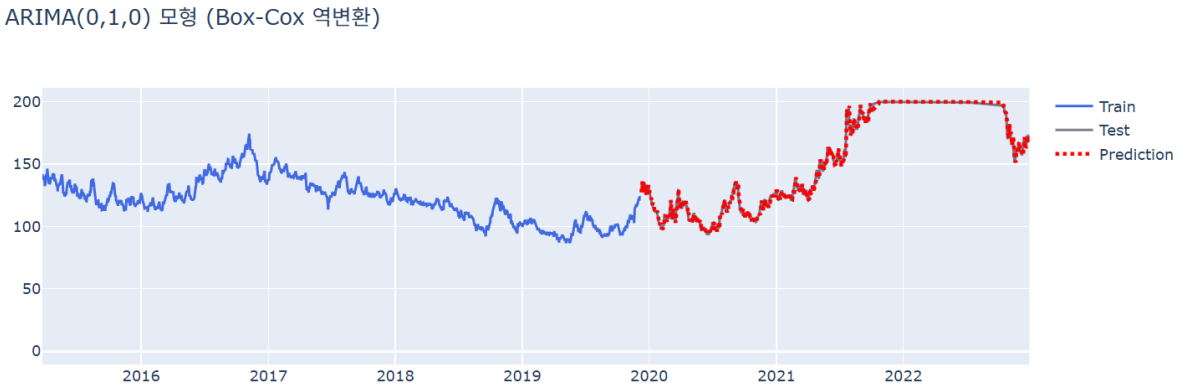
따라서 이를 해결하기 위해서는 한 스텝식 예측하고 update하는 refresh를 사용한다.

1. 한 개의 값만 예측 -> 예측 할 때마다 n_periods = 1로 설정
2. 새로운 값이 들어오면 모델 업데이트 -> model.update(new_ob)
3. 이 과정을 반복하며 계속해서 모델을 조정함

**** 즉 새로운 데이터가 들어올 때마다 예측을 업데이트>> 가장 마지막 관측치가 업데이트가 되기 때문에 관측치가 추가됨으로써 임의 보행 모형에서의 예측치도 업데이트 ****

Date	test	pred
2019-12-09	1.115532	13.000000
2019-12-10	1.116028	1.115532
2019-12-11	1.116208	1.116028
2019-12-12	1.116407	1.116208
2019-12-13	1.115761	1.116407
...	...	...
2022-12-26	1.119172	1.119172
2022-12-27	1.118853	1.119172
2022-12-28	1.119266	1.118853
2022-12-29	1.119051	1.119266
2022-12-30	1.118876	1.119051

Refresh방식을 이용해서 test와 predict를 비교해보면 오늘의 실제값이 내일의 예측값이 된다는 것을 알 수 있다. 이는 모델이 새로운 데이터가 들어올 때마다 모델을 업데이트하고 그 값이 다음 시점의 예측값이 되기 때문에 오늘의 실제값이 내일의 예측값이 된다.



결국 **ARIMA(0,1,0)모델로 예측을 해주기 위해서는 1일전 데이터 값을 새로운 데이터가 들어올 때 마다 업데이트 해주면 된다는 의미임**

모델이 단순히 기존의 예측 값을 기반으로 계속 업데이트 해나가기 때문에 모델의 실제 학습 효과가 부족하다고 볼 수 있음 -> ***장기적인 예측에는 적합하지 않음***

- VAR 모델

Train, test 데이터 분리

```
target_lag = 14

df_train = df.iloc[:-target_lag, :]
df_test = df.iloc[-target_lag:, :]
```

14일 후를 예측할 것이므로 14일을 기준으로 데이터를 나눠준다.

1. 시계열 정상성 확인

- ADF 테스트

시계열 정상성을 확보하기 위해 ADF 테스트를 진행한다.

```
Checking stationarity for Colombia_Permanent cropland (% of land area)
ADF Statistics: -1.582490
p-value: 0.492383
num of lags: 25.000000
num of observations: 2516.000000
Critical values:
        1%: -3.433
        5%: -2.863
        10%: -2.567
Checking stationarity for Ethiopia_Permanent cropland (% of land area)
ADF Statistics: -1.991710
p-value: 0.290220
num of lags: 27.000000
num of observations: 2514.000000
Critical values:
        1%: -3.433
        5%: -2.863
        10%: -2.567
Checking stationarity for Brazil_Cereal yield (kg per hectare)
ADF Statistics: -5.662000
p-value: 0.000001
num of lags: 25.000000
num of observations: 2516.000000
Critical values:
        1%: -3.433
        5%: -2.863
        10%: -2.567
Checking stationarity for Colombia_Cereal yield (kg per hectare)
ADF Statistics: -2.921966
p-value: 0.042853
num of lags: 27.000000
```

데이터에 대해 ADF 테스트를 진행하였을 때 p-value값이 0.05보다 작아 정상성을 확보하는 컬럼과 그렇지 않은 컬럼이 섞여 있다.

따라서 비정상성인 컬럼에 대해서 차분을 진행하여 정상성을 확보하는지를 확인해보았다.

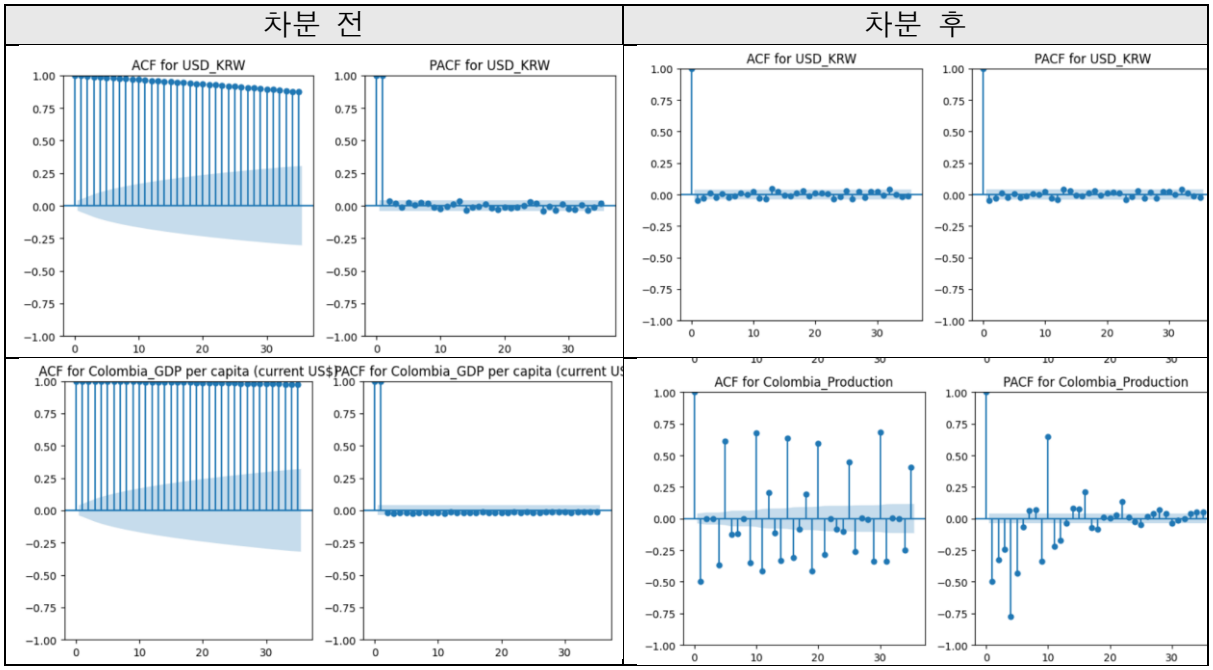
확인해보았을 때 1차 차분만 해도 되는 컬럼과 2차 차분까지 해야 하는 컬럼이 있었다.

Date	Coffee_Price	Crude_Oil_Price	USD_KRW	USD_BRL	USD_COP	Brazil_Production	Colombia_Production	Ethiopia_Production	Brazil_Agricultural raw materials exports (% of merchandise exports)
2015-03-23	NaN	NaN	NaN	NaN	NaN	2.952419e+06	NaN	NaN	NaN
2015-03-24	-4.550003	0.059998	-8.349976	-0.096300	-52.000000	2.955238e+06	NaN	NaN	NaN
2015-03-25	2.649994	1.700001	-1.760010	0.007900	-2258.000000	2.958036e+06	1.305542	-1.078881	-0.000012
2015-03-26	0.300003	2.220001	-2.690063	0.061100	2310.000000	2.960812e+06	1.302343	-1.076174	-0.000012
2015-03-27	-2.050003	-2.560001	3.969971	-0.017800	-5.000000	2.963566e+06	1.299143	-1.073467	-0.000012
...	...	...	...	...	...	...	...	...	...
2023-12-05	4.950012	-0.720001	8.610107	0.040941	45.618408	3.430930e+06	1153.192802	-756.540598	0.006633
2023-12-06	-8.600006	-2.940002	5.569946	-0.015400	-8.250000	3.430005e+06	-4.980547	1.378235	-0.000015
2023-12-07	0.250000	-0.040001	0.349976	-0.026500	-10.399902	3.429068e+06	-4.992069	1.379855	-0.000015
2023-12-08	-0.350006	1.890007	-0.959961	0.008900	-2.370117	3.428118e+06	-5.003592	1.381474	-0.000015
2023-12-11	8.850006	0.089996	3.750000	-0.016814	-49.565918	3.425195e+06	-1243.050051	781.377980	-0.006911

2542 rows × 47 columns

따라서 데이터를 차분을 할 필요 없는 컬럼, 1차 차분만 한 컬럼, 2차 차분만 한 컬럼의 조합으로 만들어주었다.

- ACF, PCAF 그래프



차분을 진행한 후 ACF, PCAF 컬럼을 그려보았을 때 0에 가깝게 그려지는 변수가 있지만 불규칙하게 나타나는 변수가 있다. ACF/PACF가 불규칙하게 나타나는 것은 단기적 상관관계를 설명하는 데 어려움을 겪을 수 있지만, VAR 모델은 각 시계열 변수들 간의 상호 작용을 고려하기 때문에, 장기적 관계를 잘 모델링할 수 있다.

```
정상적인 변수: ['Coffee_Price', 'Crude_Oil_Price', 'USD_KRW', 'USD_BRL', 'USD_COP', 'Brazil_Production', 'Colombia_Production', 'Ethiopia_Pro
비정상 변수: []
```

차분을 다 진행한 후 데이터에 대해 ADF 테스트를 다시 해보았을 때 모든 컬럼이 p-value값이 0.05보다 낮게 나왔다.

따라서 불규칙한 패턴이 있다는 것이 반드시 VAR 모델의 부적합을 의미하는 것은 아니므로 차분 후 정상성만 확보되었다면, VAR 모델이 여전히 상호작용을 잘 설명할 수 있다.

2. 다중공선성 제거

VAR 모델은 다중공선성이 높은 변수가 있으면 학습이 되지 않으므로 다중공선성 제거를 해줘야한다.

- VIF 계산

다중공선성이 높은 변수들(VIF > 10):

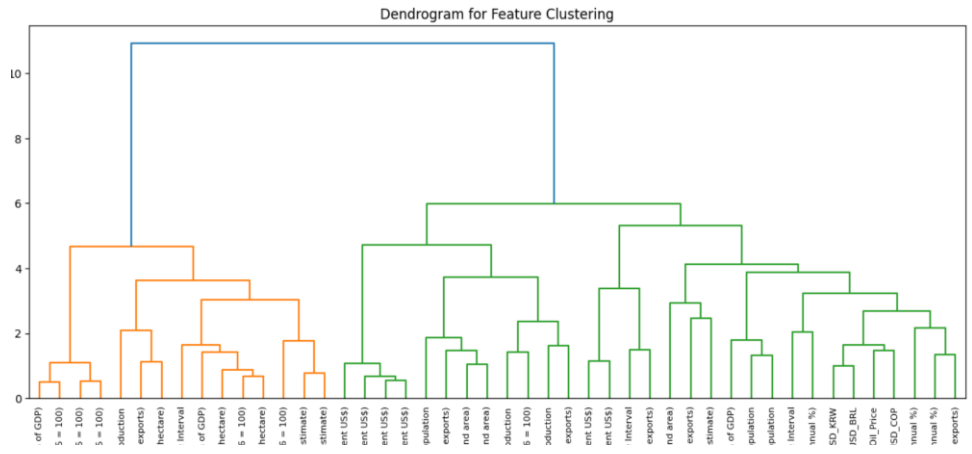
	feature	VIF
5	Brazil_Production	inf
6	Colombia_Production	inf
7	Ethiopia_Production	2.251800e+15
8	Brazil_Agricultural raw materials exports (% o...	9.007199e+15
9	Colombia_Agricultural raw materials exports (%...	inf
10	Ethiopia_Agricultural raw materials exports (%...	4.503600e+15
11	Brazil_Merchandise trade (% of GDP)	inf
12	Colombia_Merchandise trade (% of GDP)	inf
13	Ethiopia_Merchandise trade (% of GDP)	9.007199e+15
14	Brazil_Unemployment, male (% of male labor for...	inf
15	Colombia_Unemployment, male (% of male labor f...	inf
16	Ethiopia_Unemployment, male (% of male labor f...	inf
17	Brazil_GDP per capita (current US$)	inf
18	Colombia_GDP per capita (current US$)	inf
19	Ethiopia_GDP per capita (current US$)	inf
20	Brazil_IMF repurchases and charges (TDS, curre...	inf
21	Colombia_IMF repurchases and charges (TDS, cur...	inf
22	Ethiopia_IMF repurchases and charges (TDS, cur...	9.007199e+15
23	Brazil_Food production index (2014-2016 = 100)	inf
24	Colombia_Food production index (2014-2016 = 100)	inf
25	Ethiopia_Food production index (2014-2016 = 100)	inf
26	Brazil_Political Stability and Absence of Viol...	inf
27	Colombia_Political Stability and Absence of Vi...	inf
28	Ethiopia_Political Stability and Absence of Vi...	inf
29	Brazil_GDP per capita growth (annual %)	8.188363e+14
30	Colombia_GDP per capita growth (annual %)	inf
31	Ethiopia_GDP per capita growth (annual %)	inf
32	Brazil_Merchandise exports to low- and middle-...	inf
33	Colombia_Merchandise exports to low- and middl...	inf
34	Ethiopia_Merchandise exports to low- and middl...	inf
35	Brazil_Export unit value index (2015 = 100)	inf
36	Colombia_Export unit value index (2015 = 100)	inf
37	Ethiopia_Export unit value index (2015 = 100)	inf
38	Brazil_Rural population	6.166524e+02
39	Colombia_Rural population	inf
40	Ethiopia_Rural population	6.026131e+02
41	Brazil_Permanent cropland (% of land area)	inf
42	Colombia_Permanent cropland (% of land area)	inf
43	Ethiopia_Permanent cropland (% of land area)	inf
44	Brazil_Cereal yield (kg per hectare)	inf
45	Colombia_Cereal yield (kg per hectare)	inf
46	Ethiopia_Cereal yield (kg per hectare)	inf

VIF는 각 피처가 다른 피처와 얼마나 강하게 선형적으로 설명되는지를 나타내는 지표이다. VIF 계산을 해보았을 때 47개의 변수 중 43개의 변수가 아주 높은 값이 나온다.

따라서 대표 변수만 남겨주기 위해 상관계수 기반 클러스터링을 진행하였다.

- 계층적 클러스터링

상관계수를 이용하여 target feature과의 거리(1-상관계수)를 계산하여 주고 이 거리를 기반으로 계층적 클러스터링을 해준다. 이를 통해 각 클러스터마다 대표 변수 하나만을 선택해준다.



크게 두가지의 대형 클러스터가 존재하며 세부적으로 들어가면 6-7개 정도로 나눌 수 있다.

```
['Brazil_Rural population',
 'Colombia_Cereal yield (kg per hectare)',
 'Ethiopia_Permanent cropland (% of land area)',
 'Colombia_Agricultural raw materials exports (% of merchandise exports)',
 'Brazil_Export unit value index (2015 = 100)',
 'Brazil_GDP per capita (current US$)',
 'Colombia_IMF repurchases and charges (TDS, current US$)']
```

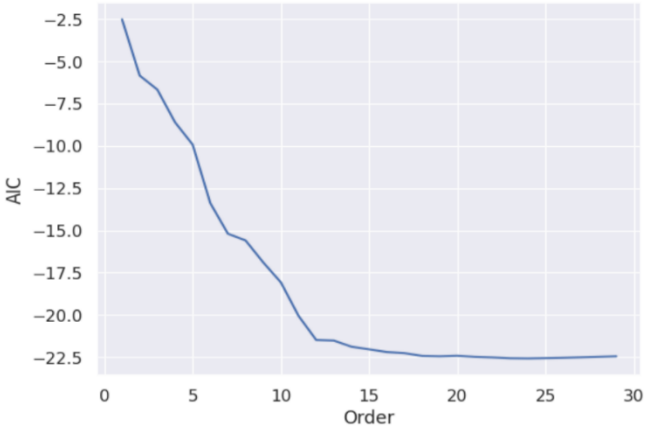
선택된 대표 변수와 target feature만 존재하는 데이터프레임을 만들어준다.

3. VAR 모델 학습 및 예측

- 최적의 lag수 파악

```
model = VAR(df_reduced)
results = model.fit(maxlags=30, ic='aic')
optimal_lag = results.k_ar
print("Selected Optimal Lag:", optimal_lag)
```

```
/usr/local/lib/python3.11/dist-packages/stats
  self._init_dates(dates, freq)
Selected Optimal Lag: 24
```



Maxlag를 모델 학습 후 k_ar를 이용해 최적의 lag를 파악해보았을 때 24라는 것을 알 수 있다.

- 모델 학습 후 잔차의 정규성 검증

Lag를 12로 설정한 후 모델을 학습해준 후 잔차의 정규성을 확인한다.

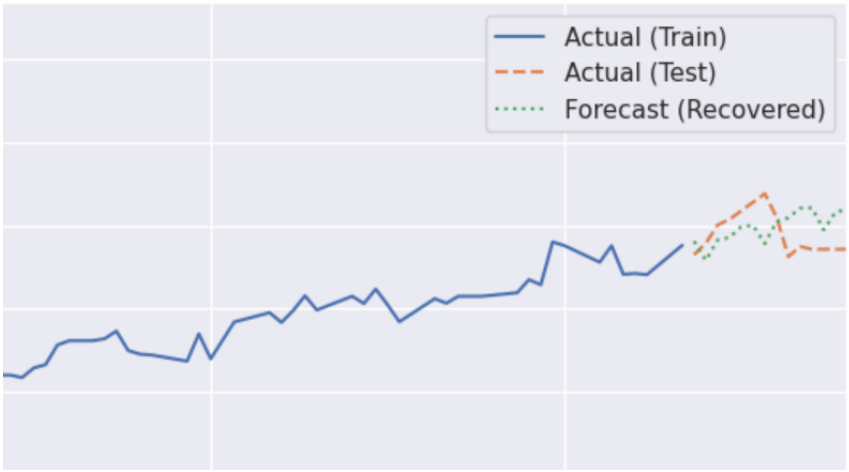
Coffee_Price	2.0
Brazil_Rural population	2.0
Colombia_Cereal yield (kg per hectare)	2.01
Ethiopia_Permanent cropland (% of land area)	2.0
Colombia_Agricultural raw materials exports (% of merchandise exports)	2.01
Brazil_Export unit value index (2015 = 100)	2.02
Brazil_GDP per capita (current US$)	2.01
Colombia_IMF repurchases and charges (TDS, current US$)	2.0

모두 2에 가까운 값이므로 모델은 잔차간의 상관성이 유의하지 않는 수준이며 충분한 예측 설명력을 갖췄음을 확인할 수 있다.

- Train 데이터의 마지막 date기준으로 2주후 후까지를 예측

train데이터의 마지막 시점을 기준으로 2주후를 예측하여 실제 값과 비교해보았다.

실제값	예측값
191.149994	195.231766
194.649994	189.443313
200.100006	195.602396
201.800003	196.351014
209.550003	199.599804
202.399994	200.021616
190.600006	194.505974
193.600006	201.160486
192.800003	202.252858
192.800003	205.176276
194.350006	205.282204
197.750000	198.645083
198.000000	203.983334
188.300003	204.914414



초기 몇 step은 비교적 잘 맞지만, 예측이 진행되면서 오차가 점점 커진다. 또한 예측값이 실제값보다 많이 튀거나, 패턴을 잘 따라가지 못한다.

4. 모델 평가

모델 성능평가 지표로 MAE와 RMSE를 사용한다.

MAE	RMSE
7.09	8.22

가격 수준이 200 정도라면, 약 4% 정도 오차가 나타난다.

- LSTM 모델

Train, test 데이터 분리

```
train_size = int(len(df) * 0.8)
train_df, test_df = df.loc[:df.index[train_size-1]], df.loc[df.index[train_size:]]
```

1. 데이터 정규화 및 입력 데이터셋 구축

- 데이터 정규화

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler(feature_range=(0, 1))
train_df = pd.DataFrame(scaler.fit_transform(train_df),
                        columns=train_df.columns,
                        index=train_df.index)
```

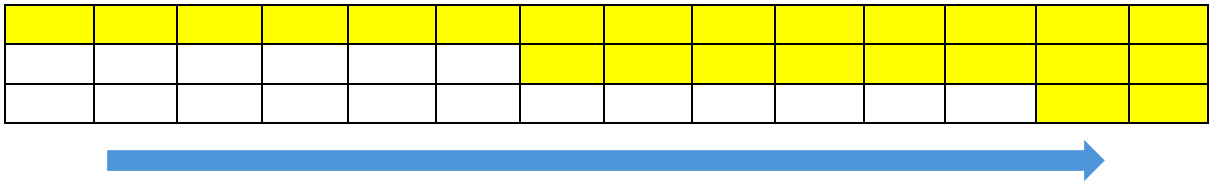
Train 데이터에 대해 MinMaxScaler을 통해 정규화를 진행한다.

- 입력 데이터셋 구축

시계열 예측에 맞게 슬라이딩 윈도우 방식으로 입력 데이터셋이 만들어진다.

```
data_window = 100  # 최근 100개 데이터를 입력으로 사용
future_target = 14  # 14개 미래 값 예측
step = 6  # 6개 단위로 샘플링
train_dataset = MultiStepTimeSeriesDataset(X_train, y_train, data_window, future_target, step)
```

Multi Step LSTM 모델링을 해주기 위해서는 예측할 데이터 길이를 선택해준다. 14일 후까지를 예측하므로 예측할 데이터 길이를 14로 설정해주고 최근 100개의 데이터를 입력으로 사용하도록 하였다. 그리고 데이터 샘플을 만들 때 6시점을 건너 뛰고 만든다.



```
X_train = train_dataset.data
y_train = train_dataset.labels
```

```
X_train.shape
```

```
torch.Size([1930, 17, 47])
```

```
train_loader = torch.utils.data.DataLoader(train_dataset, batch_size=10, shuffle=True)
```

LSTM 학습을 위해 3D 배열로 변경해준다.

2. 모델 학습 및 예측

- LSMT 모델 정의 및 학습

다변량 데이터로 미래의 여러 스텝을 한번에 예측하기 위해 Multi Step LSTM 모델을 정의한다.

Hyper-parameters 값	
Learning rate	0.001
Mini-batch size	10
Number of iteration	100
Drop-out value	0.2

표와 같이 하이퍼파라미터를 정의하였다.

```
device = 'cuda' if torch.cuda.is_available() else 'cpu'

input_size = X_train.shape[2]
target_size = 14
model = LSTMModel(input_size=input_size, target_size=target_size).to(device)

# 손실 함수 및 최적화 함수 설정
criterion = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001, weight_decay=1e-5)
```

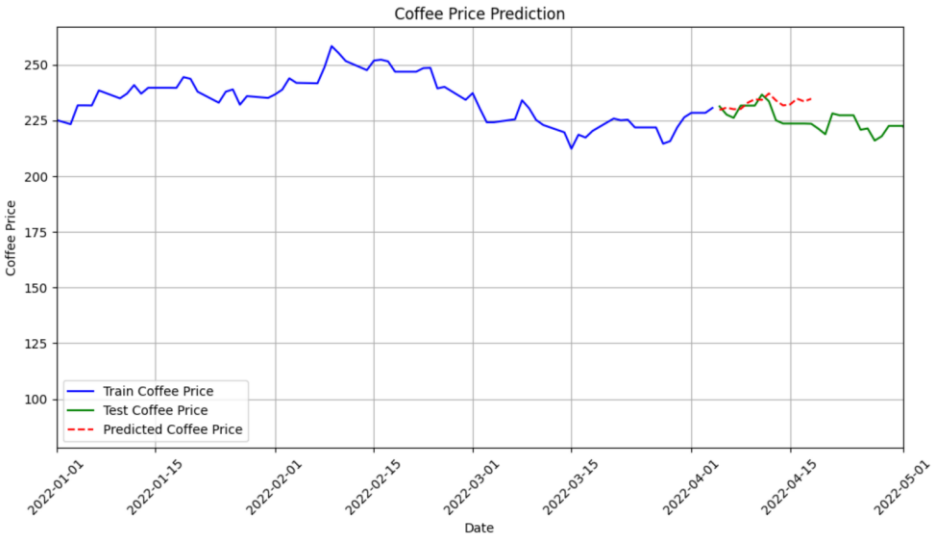
optimizer로는 일반적으로 많이 사용되는 Adam optimizer을 사용하였다.

```
num_epochs = 100
hidden_states = None  # 한 에폭 내에서 유지

for epoch in range(num_epochs):
    for batch_idx, (x_batch, y_batch) in enumerate(train_loader):
        x_batch, y_batch = x_batch.to(device), y_batch.to(device)
        optimizer.zero_grad()

        if hidden_states is None or x_batch.shape[0] != hidden_states[0].shape[1]:
            hidden_states = (
                torch.zeros(model.num_layers, x_batch.shape[0], model.hidden_size).to(device),
                torch.zeros(model.num_layers, x_batch.shape[0], model.hidden_size).to(device)
            )

        hidden_states = tuple(h.detach() for h in hidden_states)

        # 모델 예측
        y_pred, hidden_states = model(x_batch, hidden_states)

        # `y_pred`가 `(batch_size, target_size, 1)`이면 `(batch_size, target_size)`로 변환
        if y_pred.shape[-1] == 1:
            y_pred = y_pred.squeeze(-1)

        # Loss 계산
        loss = criterion(y_pred, y_batch)

        # Backpropagation
        loss.backward()
        torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=5)  # Gradient Clipping
        optimizer.step()

    print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}")
```

각 에폭의 첫 배치에서만 hidden state를 초기화해주었다. 이후 배치에서는 이전에 계산된 hidden state를 유지하여 LSTM의 시계열 특성을 살리는 방식으로 학습을 진행시켰다.

- Train 데이터의 마지막 시점을 기준으로 14일 후까지 예측

실제값	예측값
231.300003	229.699808
227.600006	230.722248
226.149994	230.035129
231.649994	230.149588
231.649994	232.931344
236.600006	234.518287
233.600006	234.186415
225.050003	237.090540
223.600006	233.964723
223.600006	231.732376
223.500000	232.148301
221.300003	234.751410
218.800003	233.509848
228.149994	234.708795



VAR에 비해서는 전체적으로 패턴을 따라간다. 또한 VAR에 비해 마지막 시점에서의 예측값과 실제값의 차이가 작다.

3. 모델 평가

MAE	RMSE
6.28	7.88

VAR모델과 비교하였을 때 전반적으로 수치가 적게 나온다. 가격 수준이 200 정도라면, 약 4% 정도 오차가 나타난다. LSTM이 장기 예측에 더 적합한 만큼 실제로 확인해보았을 때 후반 시점에서 LSTM이 VAR보다 실제값과 예측값 간의 차이가 작다.