1) 시계열 분석

- 시간에 따라 관측된 데이터를 분석하여 미래의 값을 예측하거나 데이터의 패턴을 이해하는 것

```
import pandas as pd
df = pd.read_csv('./temperature_data.csv')
df
```

	Date	Temperature
0	2023-01-01	-1.161276
1	2023-01-02	-0.910024
2	2023-01-03	2.839168
3	2023-01-04	3.788326
4	2023-01-05	1.115325
...	...	...
295	2023-10-23	9.410780
296	2023-10-24	11.203737
297	2023-10-25	8.912885
298	2023-10-26	10.205989
299	2023-10-27	11.384230

300 rows × 2 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Date         300 non-null    object 
 1   Temperature  300 non-null    float64
dtypes: float64(1), object(1)
memory usage: 4.8+ KB
```

```
df['Date'] = pd.to_datetime(df['Date'], format = '%Y-%m-%d')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype         
---  ------       --------------  -----         
 0   Date         300 non-null    datetime64[ns]
 1   Temperature  300 non-null    float64       
dtypes: datetime64[ns](1), float64(1)
memory usage: 4.8 KB
```

```
df.set_index('Date',inplace =True)
df
```

Date	Temperature
2023-01-01	-1.161276
2023-01-02	-0.910024
2023-01-03	2.839168
2023-01-04	3.788326
2023-01-05	1.115325
...	...
2023-10-23	9.410780
2023-10-24	11.203737
2023-10-25	8.912885
2023-10-26	10.205989
2023-10-27	11.384230

300 rows × 1 columns

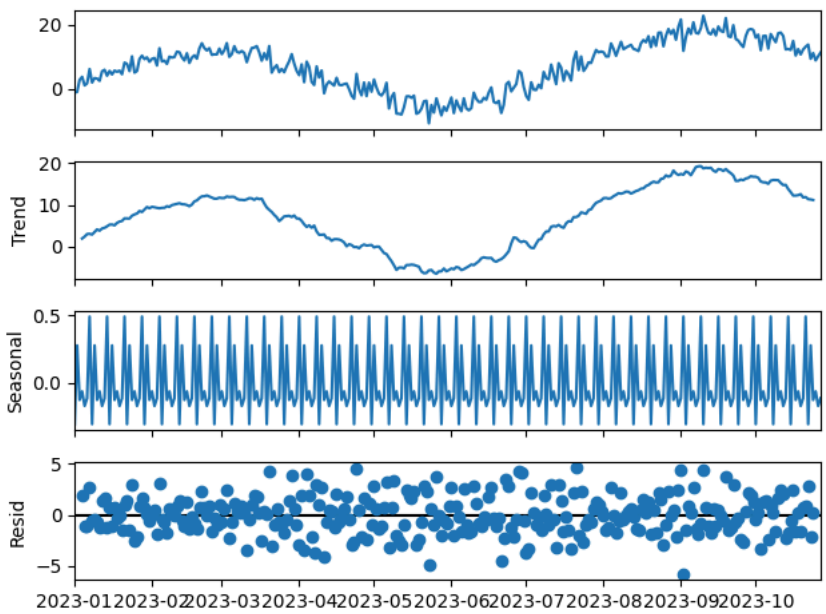
2)시계열 분해

시계열 데이터를 구성하는 주요 요소들을 분리하여 시계열 데이터를 살펴보는 방법

주요 구성요소

- 추세(Trend) : 데이터의 장기적인 방향성
- 계절성(Seasonality) : 일정 주기로 반복되는 패턴
- 불규칙성(잔차) : 추세 및 계절성을 제거 후 남는 불규칙 변동

```
from statsmodels.tsa.seasonal import seasonal_decompose
import matplotlib.pyplot as plt
result = seasonal_decompose(df, model = 'additive')
result.plot()
plt.show()
```



3) 시계열 예측 모델 생성

- 데이터가 정상성을 만족해야 함
  - 차분, 로그변환등이 있음

시계열 예측 모형

- ARIMA : AR(자기회귀) 모델과 MA(이동평균) 모델의 결합에 더해, 데이터의 비정상성을 제거하기 위한 차분을 포함하는 모델

- SARIMA : ARIMA 모델의 확장하여 계절성 또한 포함하는 모델

```
# 귀무가설 : 시계열 데이터가 정상성을 가지지 않는다
# 대립가설 : 시계열 데이터가 정상성을 가진다.
from statsmodels.tsa.stattools import adfuller
adf = adfuller(df['Temperature'])
print('adf 통계량 : ',adf[0])
print('p-value : ',adf[1])
```

```
adf 통계량 :  -1.4891316512616641
p-value :  0.5388807410140046
```

```
# 정상성 만족을 위한 차분
diff = df.diff(1)
diff.dropna(inplace = True)
```

```
from statsmodels.tsa.stattools import adfuller
adf = adfuller(diff['Temperature'])
print('adf 통계량 : ',adf[0])
print('p-value : ',adf[1])
```

```
adf 통계량 :  -11.59980365440354
p-value :  2.665741836029786e-21
```

```
# 트레인 데이터 7, 테스트 데이터 3
cut = int(len(df) * 0.7)
train_data = df[:cut]
test_data = df[cut:]
```

```
train_data
```

Date	Temperature
2023-01-01	-1.161276
2023-01-02	-0.910024
2023-01-03	2.839168
2023-01-04	3.788326
2023-01-05	1.115325
...	...
2023-07-25	6.720818
2023-07-26	8.305848
2023-07-27	8.102038
2023-07-28	10.266580
2023-07-29	12.180896

210 rows × 1 columns

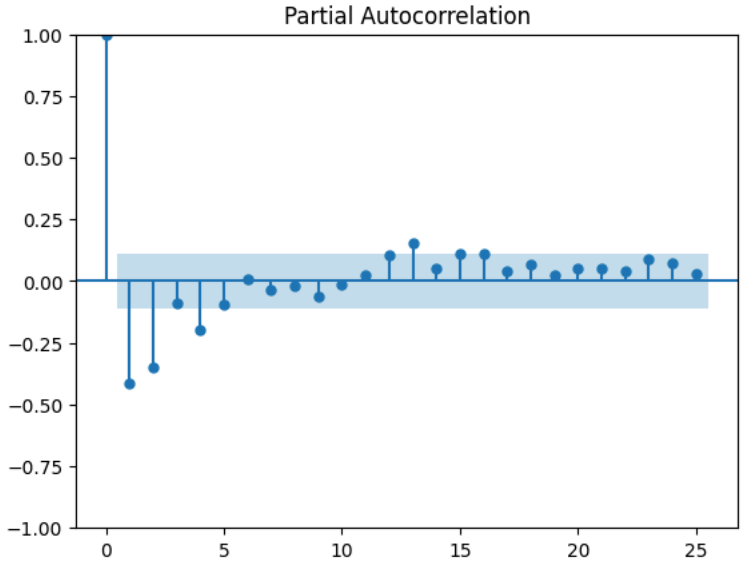
```
from sklearn.model_selection import train_test_split
train_sk, test_sk = train_test_split(df, train_size = 0.7, shuffle = False)
train_sk
```

Date	Temperature
2023-01-01	-1.161276
2023-01-02	-0.910024
2023-01-03	2.839168
2023-01-04	3.788326
2023-01-05	1.115325
...	...
2023-07-25	6.720818
2023-07-26	8.305848
2023-07-27	8.102038
2023-07-28	10.266580
2023-07-29	12.180896

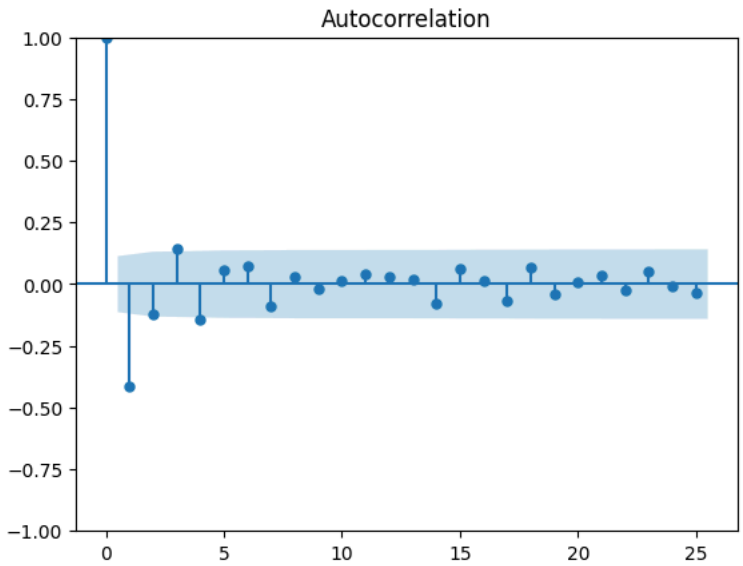
210 rows × 1 columns

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

import matplotlib.pyplot as plt
plot_pacf(diff)
plt.show()
```




```
plot_acf(diff)
plt.show()
```



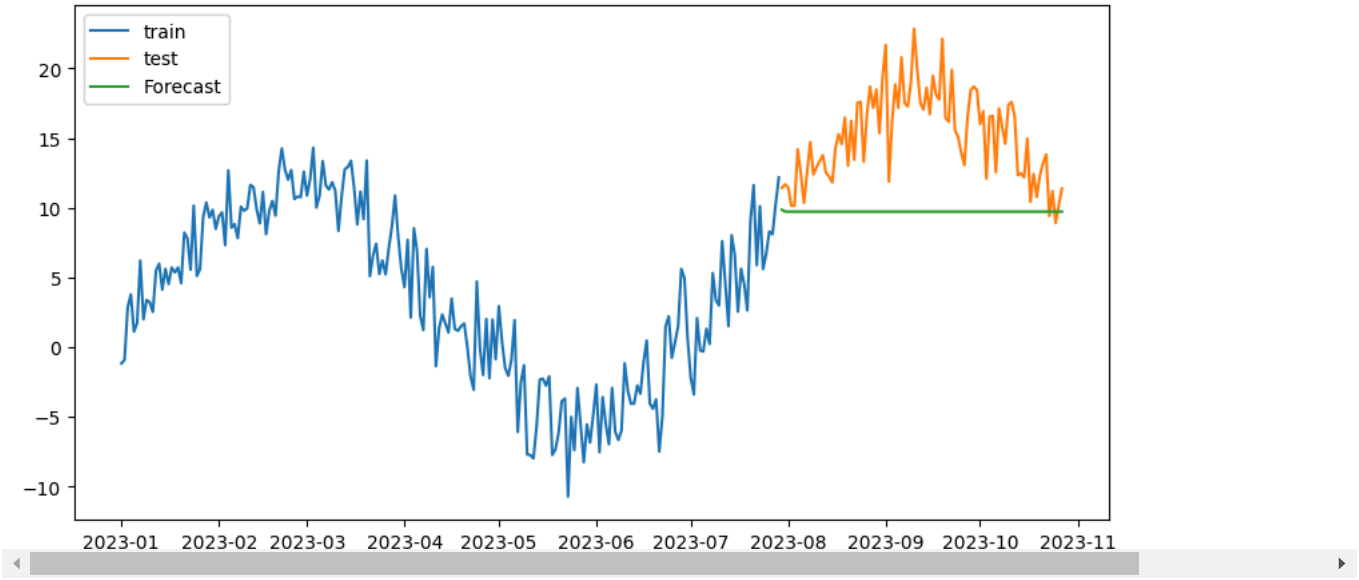
```
import pandas as pd
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt

p,d,q = 1,1,1
model=ARIMA(train_data, order= (p,d,q))
model_fit = model.fit()

# 예측
forecast = model_fit.forecast(steps = len(test_data))

plt.figure(figsize = (10,5))
plt.plot(train_data, label = 'train')
plt.plot(test_data, label = 'test')
plt.plot(forecast, label='Forecast')
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/statsmodels/tsa/base/tsa_model.py:473: ValueWarning: No frequency information was provided, so inferred
  self._init_dates(dates, freq)
```




```
from sklearn.metrics import r2_score
r2_score(test_data.values, forecast)
```

```
-2.998711837172147
```

```
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(forecast, test_data.values)
```

```
mse
```

```
39.365882163733104
```

```
!pip install pmdarima
```

```
Collecting pmdarima
  Downloading pmdarima-2.0.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl (2.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.1/2.1 MB 7.3 MB/s eta 0:00:00
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (1.4.2)
Requirement already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (3.0.10)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (1.25.2)
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (2.0.3)
Requirement already satisfied: scikit-learn>=0.22 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (1.2.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (1.11.4)
Requirement already satisfied: statsmodels>=0.13.2 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (0.14.2)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (2.0.7)
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (67.7.2)
Requirement already satisfied: packaging>=17.1 in /usr/local/lib/python3.10/dist-packages (from pmdarima) (24.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.19->pmdarima) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.19->pmdarima) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.19->pmdarima) (2024.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.22->pmdarima) (3.5.0)
Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.10/dist-packages (from statsmodels>=0.13.2->pmdarima) (0.5.6)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from patsy>=0.5.6->statsmodels>=0.13.2->pmdarima) (1.16.0)
Installing collected packages: pmdarima
Successfully installed pmdarima-2.0.4
```

```
import pmdarima as pm
model = pm.auto_arima(train_data, seasonal = True, m = 12, information_criterion = 'aic', stepwise = True, trace = True)
print(model.summary())
```

```
Performing stepwise search to minimize aic
 ARIMA(2,1,2)(1,0,1)[12] intercept   : AIC=977.833, Time=2.34 sec
 ARIMA(0,1,0)(0,0,0)[12] intercept   : AIC=1043.789, Time=0.08 sec
 ARIMA(1,1,0)(1,0,0)[12] intercept   : AIC=1008.099, Time=0.44 sec
 ARIMA(0,1,1)(0,0,1)[12] intercept   : AIC=972.885, Time=0.65 sec
 ARIMA(0,1,0)(0,0,0)[12]             : AIC=1041.889, Time=0.06 sec
 ARIMA(0,1,1)(0,0,0)[12] intercept   : AIC=976.604, Time=0.24 sec
 ARIMA(0,1,1)(1,0,1)[12] intercept   : AIC=974.675, Time=1.30 sec
 ARIMA(0,1,1)(0,0,2)[12] intercept   : AIC=974.571, Time=1.88 sec
 ARIMA(0,1,1)(1,0,0)[12] intercept   : AIC=973.580, Time=0.41 sec
 ARIMA(0,1,1)(1,0,2)[12] intercept   : AIC=inf, Time=6.77 sec
 ARIMA(0,1,0)(0,0,1)[12] intercept   : AIC=1044.773, Time=0.14 sec
 ARIMA(1,1,1)(0,0,1)[12] intercept   : AIC=974.653, Time=0.32 sec
 ARIMA(0,1,2)(0,0,1)[12] intercept   : AIC=974.591, Time=0.25 sec
 ARIMA(1,1,0)(0,0,1)[12] intercept   : AIC=1007.632, Time=0.16 sec
 ARIMA(1,1,2)(0,0,1)[12] intercept   : AIC=973.480, Time=0.63 sec
 ARIMA(0,1,1)(0,0,1)[12]             : AIC=971.521, Time=0.11 sec
 ARIMA(0,1,1)(0,0,0)[12]             : AIC=975.179, Time=0.04 sec
 ARIMA(0,1,1)(1,0,1)[12]             : AIC=973.288, Time=0.23 sec
 ARIMA(0,1,1)(0,0,2)[12]             : AIC=973.164, Time=0.41 sec
 ARIMA(0,1,1)(1,0,0)[12]             : AIC=972.239, Time=0.10 sec
 ARIMA(0,1,1)(1,0,2)[12]             : AIC=inf, Time=3.91 sec
 ARIMA(0,1,0)(0,0,1)[12]             : AIC=1042.873, Time=0.17 sec
 ARIMA(1,1,1)(0,0,1)[12]             : AIC=973.292, Time=0.33 sec
 ARIMA(0,1,2)(0,0,1)[12]             : AIC=973.230, Time=0.32 sec
 ARIMA(1,1,0)(0,0,1)[12]             : AIC=1005.852, Time=0.25 sec
 ARIMA(1,1,2)(0,0,1)[12]             : AIC=972.122, Time=0.62 sec

Best model:  ARIMA(0,1,1)(0,0,1)[12]
Total fit time: 22.327 seconds
                                     SARIMAX Results
==========================================================================================
Dep. Variable:                                   y   No. Observations:                  210
Model:             SARIMAX(0, 1, 1)x(0, 0, 1, 12)   Log Likelihood                -482.761
Date:                            Tue, 21 May 2024   AIC                             971.521
Time:                                    03:04:55   BIC                             981.548
```