Prophet 시계열 분석 예제 - 삼성전자 주가 예측

`yfinance` 패키지 설치가 필요하다

#01. 패키지 참조

```
import yfinance as yf
from pandas import DataFrame
from matplotlib import pyplot as plt
import seaborn as sb
import datetime as dt
```

```
plt.rcParams['font.family'] = 'Malgun Gothic'
```

#02. 데이터 불러오기

```
#origin = yf.download('005930.KS', start='2020-01-01', end='2020-12-31')
origin = yf.download('005930.KS', start='2020-01-01')
origin
```

```
[*********************100%%**********************]  1 of 1 completed
```

	Open	High	Low	Close	Adj Close	Volume
Date						
2020-01-02	55500.0	56000.0	55000.0	55200.0	50037.410156	12993228
2020-01-03	56000.0	56600.0	54900.0	55500.0	50309.359375	15422255
2020-01-06	54900.0	55600.0	54600.0	55500.0	50309.359375	10278951
2020-01-07	55700.0	56400.0	55600.0	55800.0	50581.292969	10009778
2020-01-08	56200.0	57400.0	55900.0	56800.0	51487.765625	23501171
...	...	...	...	...	...	...
2023-10-13	68000.0	68500.0	67700.0	68000.0	68000.000000	9724086
2023-10-16	67900.0	68500.0	66800.0	67300.0	67300.000000	12599299
2023-10-17	67700.0	69900.0	67400.0	69400.0	69400.000000	17299253
2023-10-18	68900.0	70500.0	68800.0	70500.0	70500.000000	16493184
2023-10-19	69700.0	70000.0	69400.0	69700.0	69700.000000	8869733

936 rows × 6 columns

#03. 데이터 전처리

필요한 필드만 추출하기

```
target_df = origin[['Close']]
target_df.head()
```

	Close
Date	
2020-01-02	55200.0
2020-01-03	55500.0
2020-01-06	55500.0
2020-01-07	55800.0
2020-01-08	56800.0

Prophet 라이브러리 형식에 맞추기

날짜 인덱스를 일반 컬럼으로 변환

```
df = target_df.reset_index()
df.head()
```

	Date	Close
0	2020-01-02	55200.0
1	2020-01-03	55500.0
2	2020-01-06	55500.0
3	2020-01-07	55800.0
4	2020-01-08	56800.0

필드 이름 변경

```
df.rename(columns={'Date':'ds', 'Close':'y'}, inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 936 entries, 0 to 935
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ds      936 non-null    datetime64[ns]
 1   y       936 non-null    float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 14.8 KB
```

#04. 데이터 시각화

최고가, 최저가 확인

```
df['y'].max(), df['y'].min()
```

```
(91000.0, 42500.0)
```

```
minmax = df.query("y == @df['y'].max() | y == @df['y'].min()")
minmax
```

	ds	y
55	2020-03-23	42500.0
253	2021-01-11	91000.0

라인 그래프

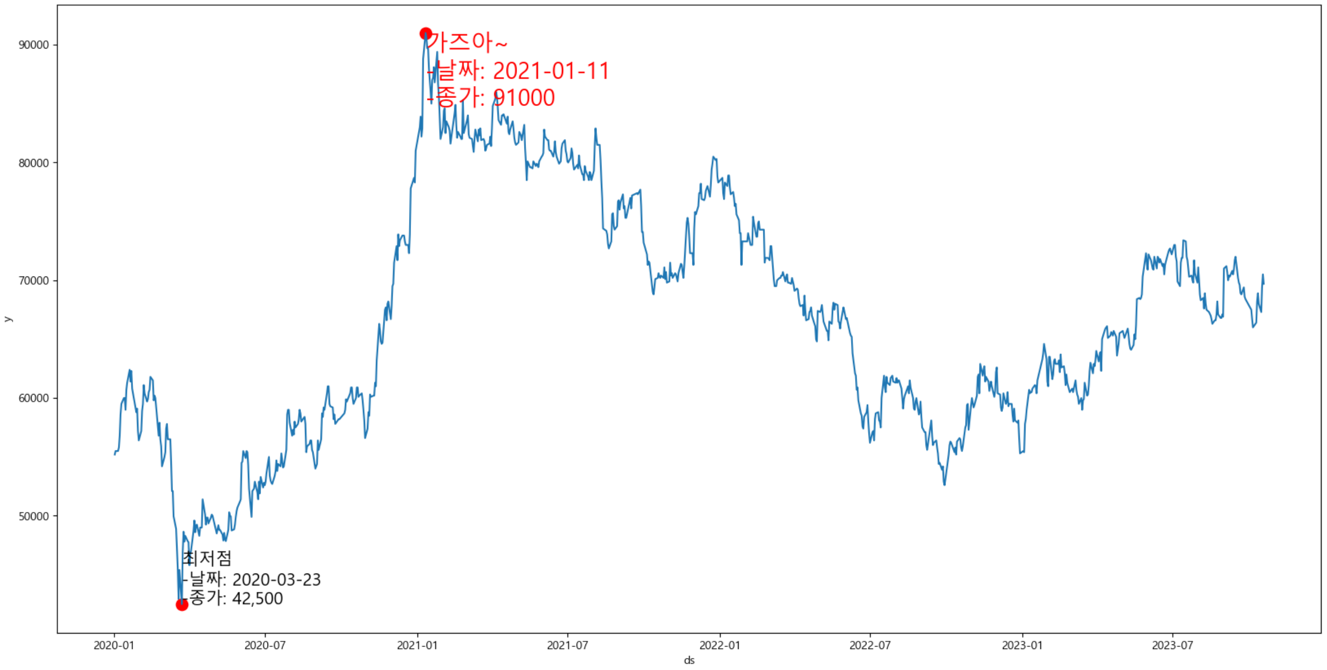
```
min_date = dt.datetime(2020, 3, 23)
max_date = dt.datetime(2021, 1, 11)

plt.figure(figsize=(20, 10))
sb.lineplot(data=df, x='ds', y='y')
sb.scatterplot(data=minmax, x='ds', y='y', color='red', s=150, marker='o')

plt.text(min_date, 42500, '최저점 \n-날짜: 2020-03-23 \n-종가: 42,500',
         fontsize=15)

plt.text(max_date, 91000, '가즈아~ \n-날짜: 2021-01-11 \n-종가: 91000',
         fontsize=20, color='red', verticalalignment='top',
         horizontalalignment='left')

plt.show()
plt.close()
```


가즈아~
날짜: 2021-01-11
종가: 91000
최저점
날짜: 2020-03-23
종가: 42,500
90000
80000
70000
60000
50000
y
2020-01
2020-07
2021-01
2021-07
2022-01
2022-07
2023-01
2023-07
ds