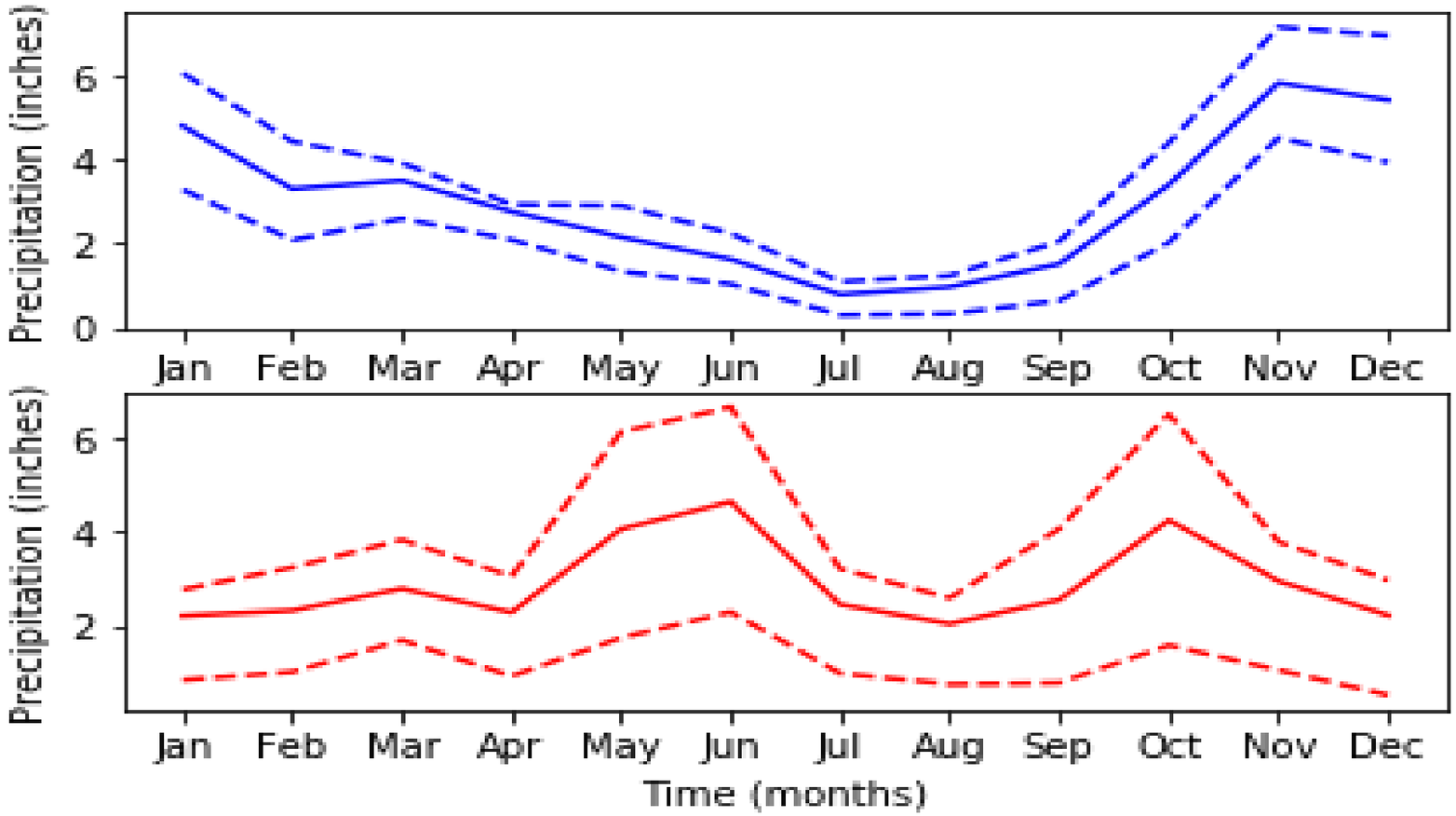
# Chapter 2
# Plotting time-series

Introduction to data visualization with matplotlib

# Part 1 .
# Plotting time-series data

# Time-series data



# DateTimeIndex

```
climate_change.index          # 1958년 ~ 2016년까지 매달 6일을 값으로 가지는 시간 변수
```

```
DatetimeIndex(['1958-03-06', '1958-04-06', '1958-05-06', '1958-06-06',
               '1958-07-06', '1958-08-06', '1958-09-06', '1958-10-06',
               '1958-11-06', '1958-12-06',
               ...
               '2016-03-06', '2016-04-06', '2016-05-06', '2016-06-06',
               '2016-07-06', '2016-08-06', '2016-09-06', '2016-10-06',
               '2016-11-06', '2016-12-06'],
              dtype='datetime64[ns]', name='date', length=706, freq=None)
```

# Time-series data

```
climate_change['relative_temp']
```

```
0       0.10
1       0.01
2       0.08
3      -0.05
4       0.06
5      -0.06
6      -0.03
7       0.04
       ...
701     0.98
702     0.87
703     0.89
704     0.93
705     0.81
Name:co2, Length: 706, dtype: float64
```

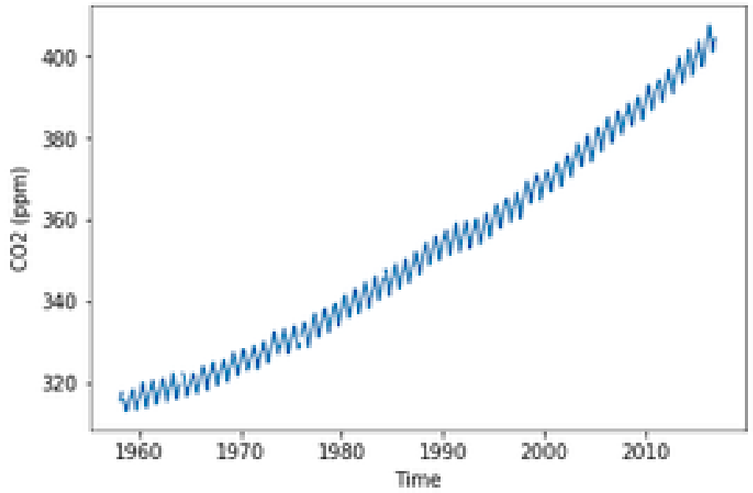
```
climate_change['co2']    # 시계열 데이터에서 y variable에 활용할 데이터 값
```

```
0      315.71
1      317.45
2      317.50
3         NaN
4      315.86
5      314.93
6      313.20
7         NaN
      ...
701    402.27
702    401.05
703    401.59
704    403.55
705    404.45
Name:co2, Length: 706, dtype: float64
```

# Plotting time-series data

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
```

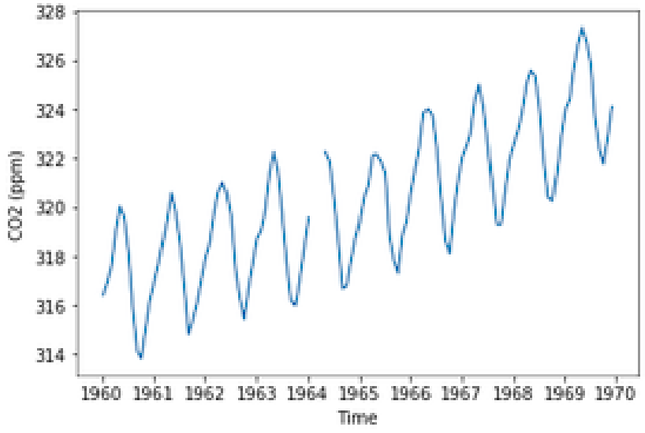
```
ax.plot(climate_change.index, climate_change['co2'])  # time-series 그래프 : x축 :time 변수, y축 :관심 데이터 값
ax.set_xlabel('Time')
ax.set_ylabel('CO2 (ppm)')
plt.show()
```



# Zooming in on a decade

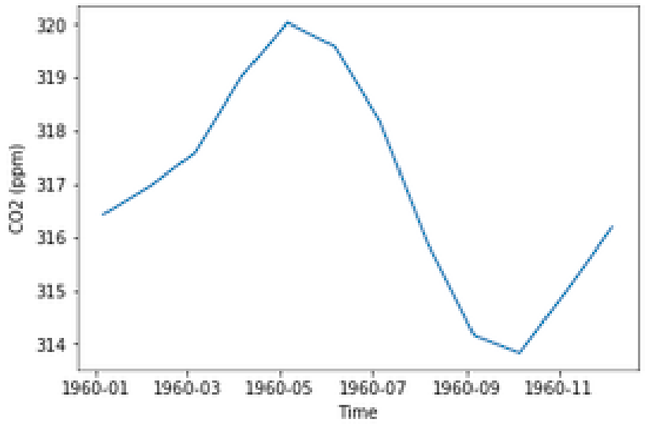
```
sixties =  climate_change["1960-01-01":"1969-12-31"]   # a decade로 시간 범위를 한정한 경우 , 새로운 변수 지정
```

```
fig, ax = plt.subplots()
ax.plot(sixties.index, sixties['co2'])
ax.set_xlabel('Time')
ax.set_ylabel('CO2 (ppm)')
plt.show()
```



# Zooming in on one year

```
sixty_nine = climate_change["1969-01-01":"1969-12-31"]  # specific one year로 시간 범위를 한정한 경우 , 새로운 변수 지정
fig, ax = plt.subplots()
ax.plot(sixty_nine.index, sixty_nine['co2'])
ax.set_xlabel('Time')
ax.set_ylabel('CO2 (ppm)')
plt.show()
```



# Part 2 .
# Plotting time-series with different variables

# Plotting two time-series together

```
import pandas as pd
climate_change = pd.read_csv('climate_change.csv',
                             parse_dates=["date"],
                             index_col="date")          #date 를 기준으로 data 불러오기
```

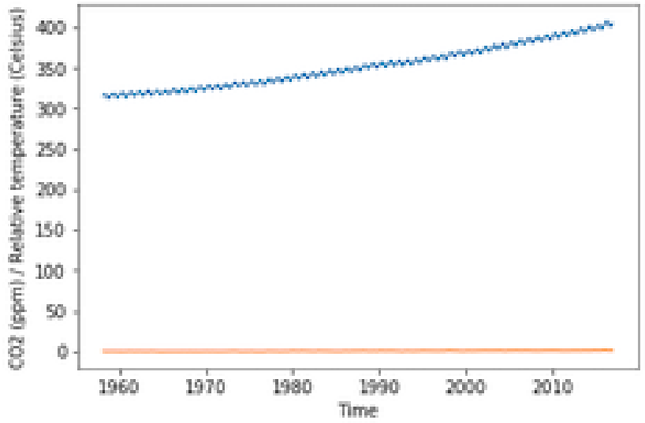
```
climate_change
```

```
              co2   relative_temp
date
1958-03-06  315.71           0.10
1958-04-06  317.45           0.01
1958-07-06  315.86           0.06
...            ...            ...
2016-11-06  403.55           0.93
2016-12-06  404.45           0.81

[706 rows x 2 columns]
```

# Plotting two time-seriestogether

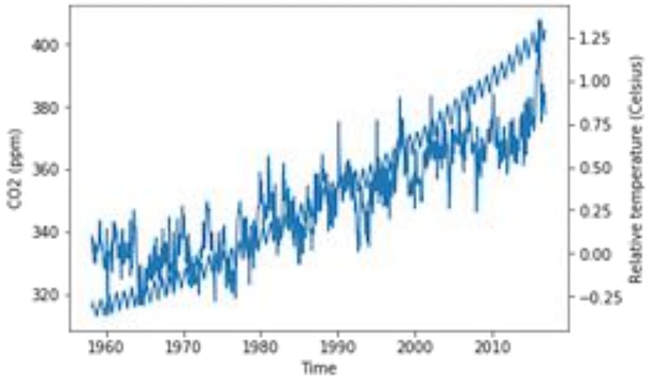
```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.plot(climate_change.index, climate_change["co2"])
ax.plot(climate_change.index, climate_change["relative_temp"])
ax.set_xlabel('Time')
ax.set_ylabel('CO2 (ppm) / Relative temperature')
plt.show()
```




#이 경우 co2과 relative_temp의 값의 범위가 크게 달라서
하나의 그래프 상에서 y값들의 변화 양상이 효과적으로 드러나지 않음

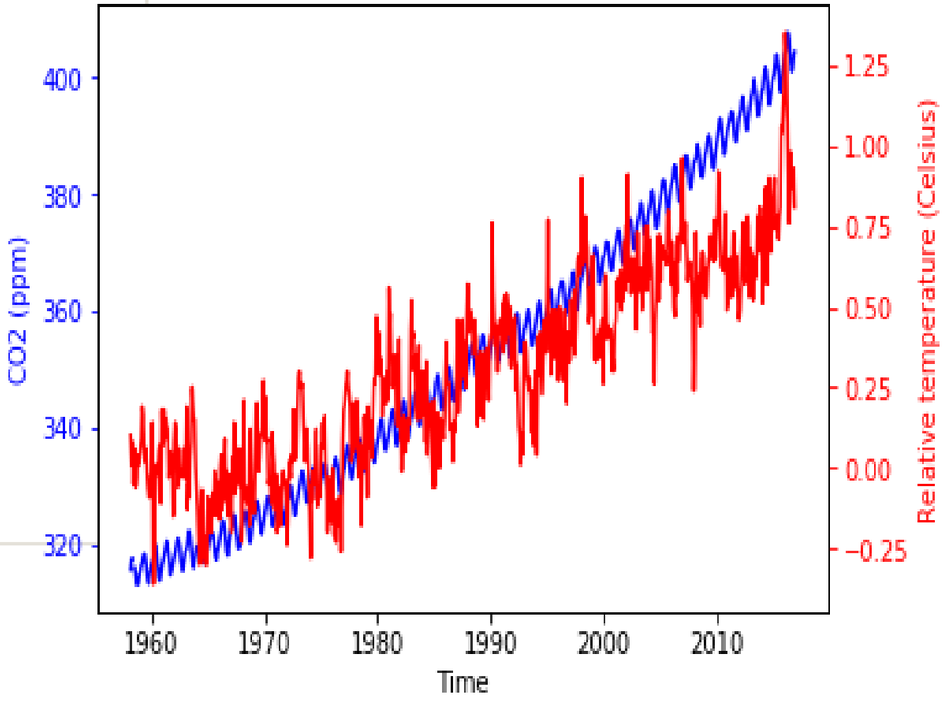
# Using twin axes

```
fig, ax = plt.subplots()
ax.plot(climate_change.index, climate_change["co2"])
ax.set_xlabel('Time')
ax.set_ylabel('CO2 (ppm)')
ax2 = ax.twinx()          #ax.twinx() : y축 추가
ax2.plot(climate_change.index,  climate_change["relative_temp"])
ax2.set_ylabel('Relative temperature (Celsius)')
plt.show()
```



# Coloring the ticks

```
fig, ax = plt.subplots()
ax.plot(climate_change.index, climate_change["co2"],color='blue')  #color option을 추가하여 데이터 구분
ax.set_xlabel('Time')
ax.set_ylabel('CO2 (ppm)', color='blue')  #color option을 추가하여 데이터 구분
ax.tick_params('y', colors='blue')  #tick(y축 위의 기준 값들) option에 color을  지정하여 데이터 구분
ax2 = ax.twinx()
ax2.plot(climate_change.index, climate_change["relative_temp"], color='red')
ax2.set_ylabel('Relative temperature (Celsius)', color='red')
ax2.tick_params('y', colors='red')
plt.show()
```



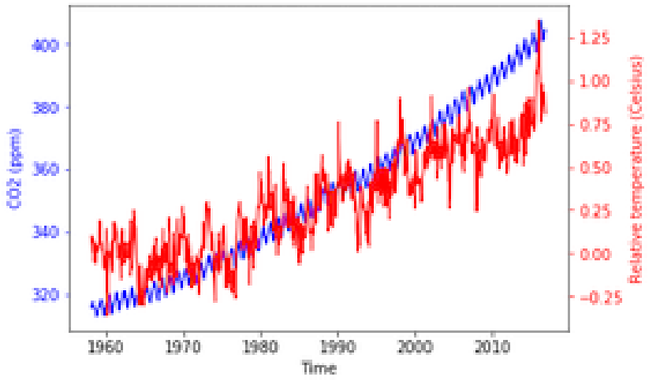
# A function that plots time-series

```
def plot_timeseries(axes, x, y, color, xlabel,ylabel):  #def function() : 사용자 지정 함수 생성
    axes.plot(x, y, color=color)
    axes.set_xlabel(xlabel)
    axes.set_ylabel(ylabel, color=color)
    axes.tick_params('y', colors=color)
```

# Using our function

```
fig, ax = plt.subplots()
plot_timeseries(ax, climate_change.index, climate_change['co2'],  'blue', 'Time', 'CO2 (ppm)')
ax2 = ax.twinx()
plot_timeseries(ax, climate_change.index,
                          climate_change['relative_temp'],
                        'red', 'Time', 'Relative temperature  (Celsius)')
plt.show()
```

#사용자 지정 함수 생성시 반복 최소화 가능

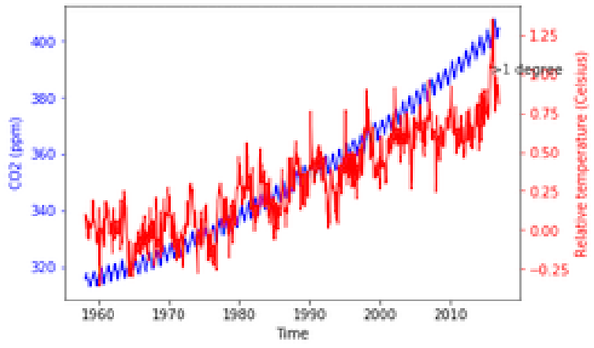


# Part 3 .
# Annotating time-series data

# Annotation(주석 추가)

```
fig, ax = plt.subplots()
plot_timeseries(ax, climate_change.index, climate_change['co2'],  'blue', 'Time', 'CO2 (ppm)')
ax2 = ax.twinx()
plot_timeseries(ax2, climate_change.index, climate_change['relative_temp'],
                        'red', 'Time', 'Relative temperature (Celsius)')
ax2.annotate(">1 degree",xy=[pd.TimeStamp("2015-10-06"), 1])
plt.show()
```

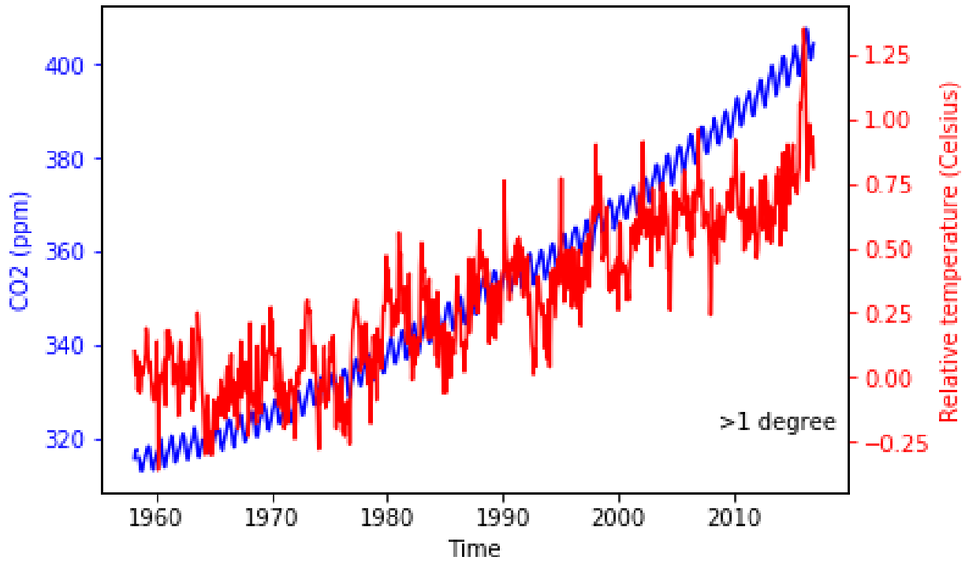
#annotate("text", xy = [x 위치, y 위치]) :
특정 text 가 그래프 위의 (x,y) 위치에 나타나도록 지정하는 옵션



# Positioning the text

```
ax2.annotate(">1 degree",
             xy=(pd.Timestamp('2015-10-06'), 1),
             xytext=(pd.Timestamp('2008-10-06'), -0.2))
```

#annotate("text", xy = [pd.Timestamp("date"), y값]) :
xy 값을 만족하는 데이터 값(date와 y값)에
"text"로 지정해준 문자열 주석을 다는 메소드

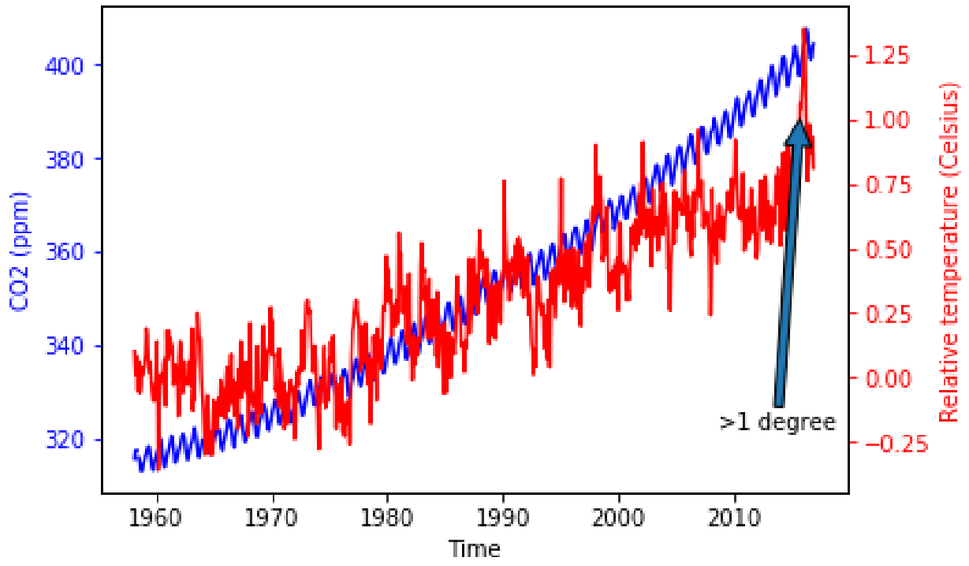


# Adding arrows to annotation

```
ax2.annotate(">1 degree",
             xy=(pd.Timestamp('2015-10-06'), 1),
             xytext=(pd.Timestamp('2008-10-06'), -0.2),
             arrowprops={})
```

#xytext(pd.Timestamp('date'),y)
: annotate("text")로 지정해준 "text" 가 그래프상에서 나타날 위치 지정

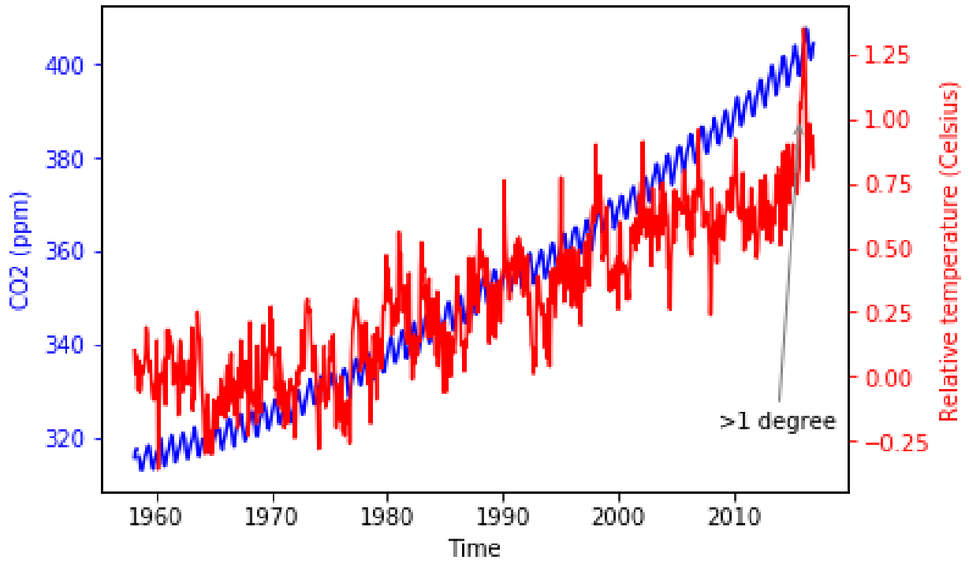
#arrowprops{} :
xy로 지정한 data point와 xytext로 위치를 지정한 "text 사이에 arrow를 추가함



# Customizing arrow properties

```
ax2.annotate(">1 degree",
                    xy=(pd.Timestamp('2015-10-06'), 1),
                    xytext=(pd.Timestamp('2008-10-06'), -0.2),
                    arrowprops={"arrowstyle":"->", "color":"gray"})
```

#arrowprops {} 의 option
(1) “arrowstyle” : arrow의 모양 지정
(2) “color”: arrow의 색상 지정



# Customizing annotations

h ps://matplotlib.org/users/annotations.html #더 많은 option 참고하기