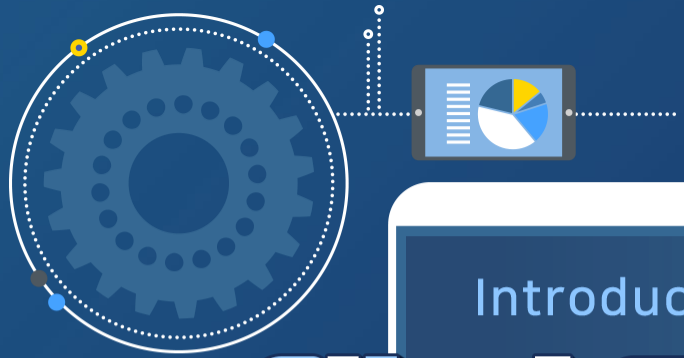
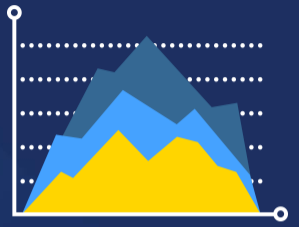


Introduction To Data Analysis

# 데이터분석입문

## Lecture 13. 서울의 크리스마스 기온 분석

인공지능소프트웨어학과 강환수 교수



## 학습개요

- 원천 자료에서 기온 데이터 변환
- 문자열에서 날짜 데이터 추출
- 기온 변화 데이터 시각화
- 연도별 평균 기온
- 추세선 그리기

## 학습목표

- 서울의 12월 25일 일별 최저, 평균, 최고 기온 변화 그래프를 그릴 수 있다.
- 서울의 연도별 평균 기온을 계산해 변화 그래프를 그릴 수 있다.
- 서울의 연도별 평균 기온 변화의 추세선을 그릴 수 있다.

LESSON 01

# 크리스마스의 기온 변화 그래프

## 데이터에 질문하기 (1/2)

- 매년 크리스마스의 최고 기온을 그래프로 그린다면 어떤 모양일까요?
- 데이터를 읽어와서 최고 기온 데이터를 출력합니다.

```
import csv

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

for row in data:
    print(row[4])

f.close()
```

| 0(-5) | 1(-4) | 2(-3) | 3(-2) | 4(-1) |
|---|---|---|---|---|
| 날짜 | 지점 | 평균기온(℃) | 최저기온(℃) | 최고기온(℃) |

## 데이터에 질문하기 (2/2)

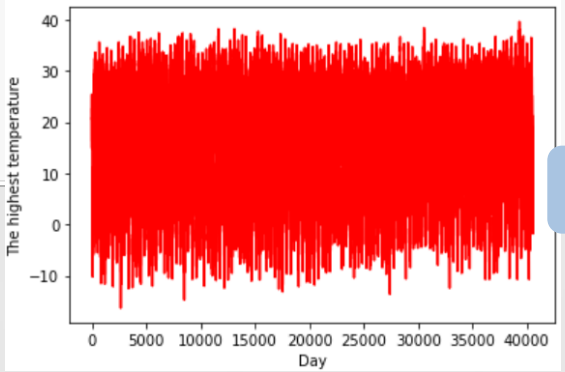
### 최고 기온 데이터를 리스트에 저장합니다.

```
import csv

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

result = []  # 최고 기온 데이터를 저장할 리스트 생성

for row in data:
    if row[4] != '':   # 최고 기온 데이터가 존재한다면
        result.append(float(row[4]))  # 리스트 result에 최고 기온 값을 실수형 (float)으로 추가

f.close()
print(len(result))
```

약 4100여개: 1907년 10월 1일부터 ~

# 데이터 시각화하기 (1/2)

```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

result = []  # 최고 기온 데이터를 저장할 리스트 생성

for row in data:
    if row[4] != '':   # 최고 기온 데이터가 존재한다면
        result.append(float(row[4]))  # 리스트 result에 최고 기온 값을 실수형 (float)으로 추가

f.close()
# print(len(result))

plt.plot(result, 'r')
plt.xlabel('Day')
plt.ylabel('The highest temperature')
plt.show()
```



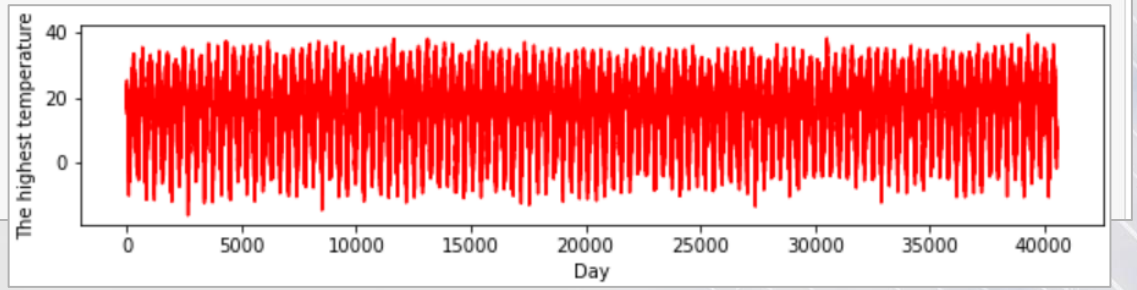

그래프가 뭉쳐 있어서 가시성이 떨어지네요.

# 데이터 시각화하기 (2/2)

```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

result = []  # 최고 기온 데이터를 저장할 리스트 생성

for row in data:
    if row[4] != '':   # 최고 기온 데이터가 존재한다면
        result.append(float(row[4]))  # 리스트 result에 최고 기온 값을 실수형 (float)으로 추가

f.close()
# print(len(result))

plt.figure(figsize=(10, 2))
plt.plot(result, 'r')
plt.xlabel('Day')
plt.ylabel('The highest temperature')
plt.show()
```

figsize=(가로 길이, 세로 길이)

길이의 단위 인치 (Inch)
1 inch = 2.54 cm




그래프를 크게 해봐도 의미를 파악하는 것이 어렵네요.

## 날짜 데이터 추출하기 (1/6)

- 매년 크리스마스 (12월 25일)의 최고 기온 데이터를 추출하기 위해서 우선 '2020-12-25' 같은 형태의 날짜 데이터를 '-'를 기준으로 년, 월, 일로 분리

- 내장 함수 split( ) 이용

```
s = 'Happy New Year!'
print(s.split())
```

```
['Happy', 'New', 'Year!']
```

split( ) 함수는 사용자가 설정하는 특정 문자가 없다면 기본적으로 공백 문자를 기준으로 문자열을 분리합니다.

## 날짜 데이터 추출하기 (2/6)

### '2020-12-25' 같은 형태의 날짜 데이터를 '-'를 기준으로 년, 월, 일로 분리

```
date = '2020-12-25'
print(date.split('-'))
```

```
['2020', '12', '25']
```

### 리스트의 인덱싱 (Indexing) 기능을 활용하여 날짜의 년, 월, 일 정보를 각각 추출

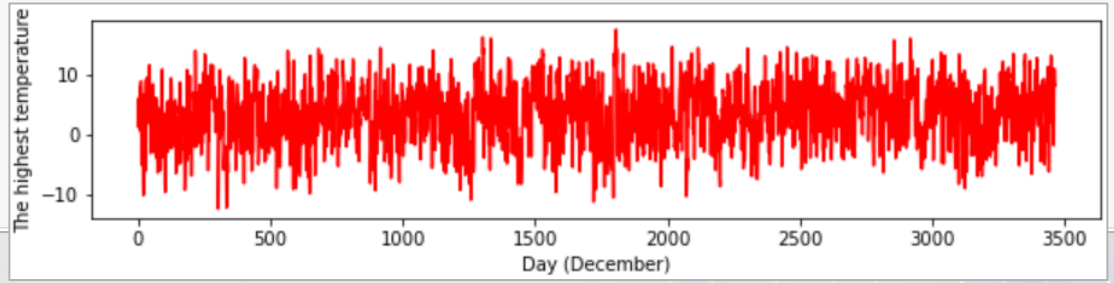
```
date = '2020-12-25'

print(date.split('-')[0])
print(date.split('-')[1])
print(date.split('-')[2])
```

```
2020
12
25
```

## 날짜 데이터 추출하기 (3/6)

### split( ) 함수를 이용하여 매년 12월의 최고 기온 데이터만 추출하여 그래프 그리기

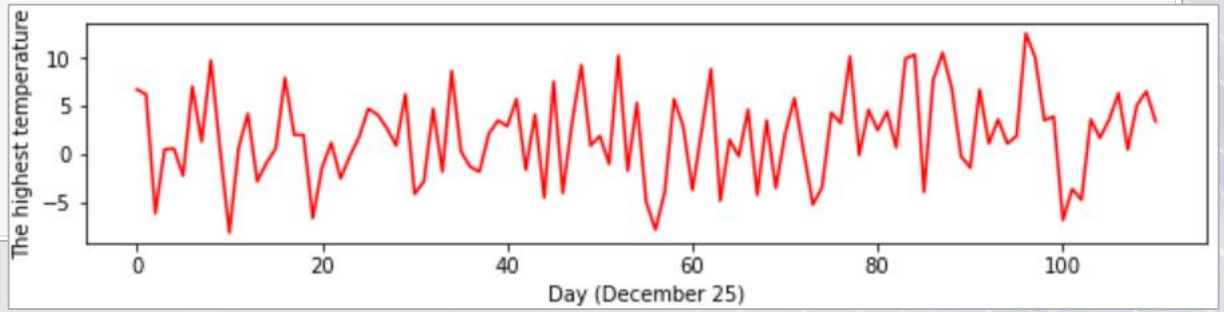
```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

result = []  # 최고 기온 데이터를 저장할 리스트 생성

for row in data:
    if row[4] != '':   # 최고 기온 데이터가 존재한다면
        if row[0].split('-')[1] == '12':  # 12월에 해당하는 값이라면
            result.append(float(row[4]))  # 리스트 result에 최고 기온 값을 실수형 (float)으로 추가

f.close()

plt.figure(figsize=(10, 2))
plt.plot(result, 'r')
plt.xlabel('Day (December)')
plt.ylabel('The highest temperature')
plt.show()
```



## 날짜 데이터 추출하기 (4/6)

### split( ) 함수를 이용하여 매년 12월 25일의 최고 기온 데이터만 추출하여 그래프 그리기

```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

result = []  # 최고 기온 데이터를 저장할 리스트 생성

for row in data:
    if row[4] != '':   # 최고 기온 데이터가 존재한다면
        if (row[0].split('-')[1] == '12') and (row[0].split('-')[2] == '25'):
            result.append(float(row[4]))  # 리스트 result에 최고 기온 값을 실수형 (float)으로 추가

f.close()

plt.figure(figsize=(10, 2))
plt.plot(result, 'r')
plt.xlabel('Day (December 25)')
plt.ylabel('The highest temperature')
plt.show()
```



# 날짜 데이터 추출하기 (5/6)

## split( ) 함수를 이용하여 매년 12월 25일의 최고 기온 데이터만 추출해 그리기

## 연산자 in

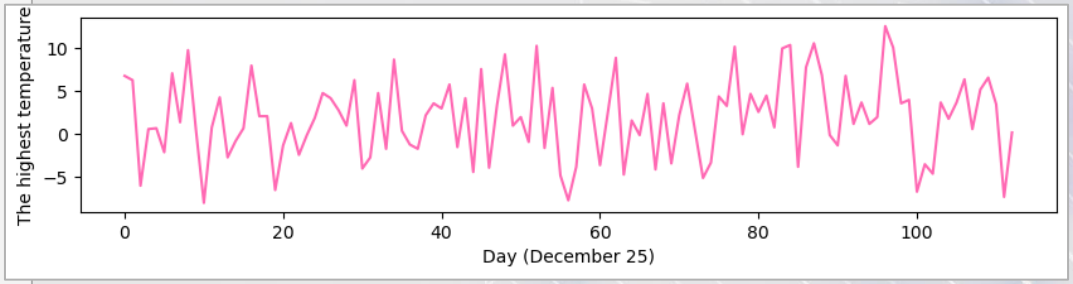
```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)
next(data)
result = []

for row in data :
    if row[-1] != '' :
        if '12-25' in row[0]:
            result.append(float(row[-1]))

plt.figure(figsize = (10,2))
plt.plot(result, 'hotpink')
plt.xlabel('Day (December 25)')
plt.ylabel('The highest temperature')
plt.show()
```

item in container
item not in container

```
data = ' 2021-12-25'
'12-25' in data
```

✓ 0.0s

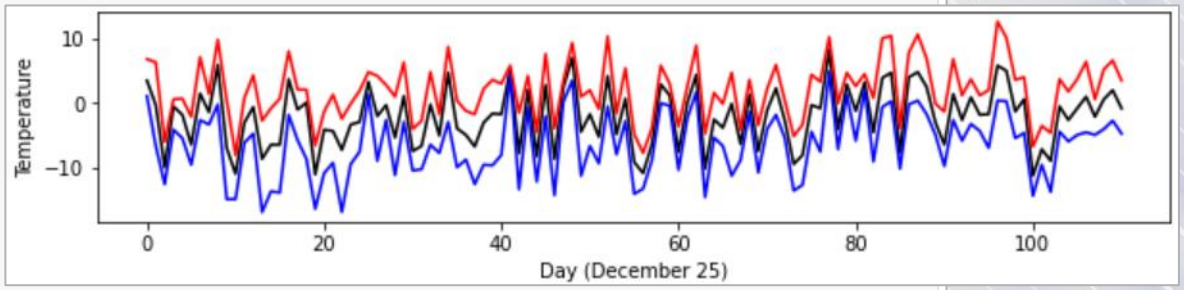
```
True
```



# 날짜 데이터 추출하기 (6/6)

## 매년 12월 25일의 평균, 최저, 최고 기온 데이터로 그래프 그리기

```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)

header = next(data)

mean = []  # 평균 기온 데이터를 저장할 리스트 생성
low = []   # 최저 기온 데이터를 저장할 리스트 생성
high = []  # 최고 기온 데이터를 저장할 리스트 생성

for row in data:
    if (row[2] != '') and (row[3] != '') and (row[4] != ''):
        if (row[0].split('-')[1] == '12') and (row[0].split('-')[2] == '25'):
            mean.append(float(row[2]))
            low.append(float(row[3]))
            high.append(float(row[4]))

f.close()

plt.figure(figsize=(10, 2))
plt.plot(mean, 'k')
plt.plot(low, 'b')
plt.plot(high, 'r')
plt.xlabel('Day (December 25)')
plt.ylabel('Temperature')
plt.show()
```



## 한글 폰트 사용하기 (Windows 기준)

- Malgun Gothic은 “맑은 고딕“
- 만약 macOS 운영체제를 사용하고 있다면 “AppleGothic”

```
plt.rc('font', family='Malgun Gothic')
plt.title('크리스마스의 기온 변화 그래프')
```

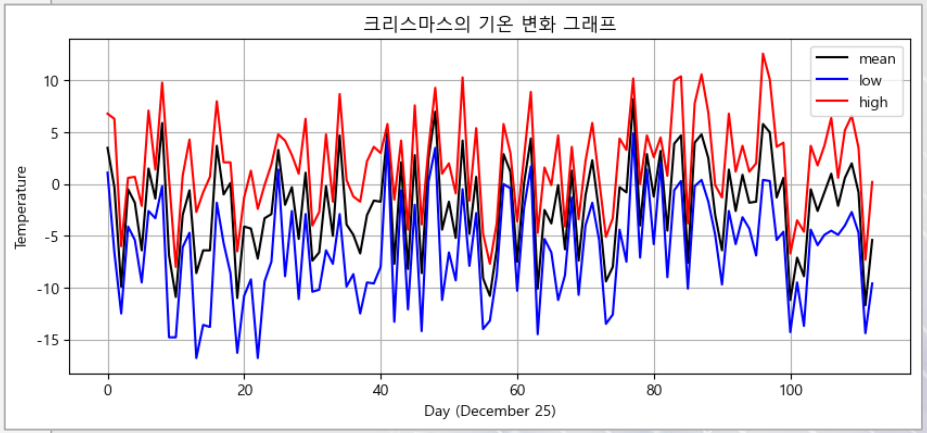
- 한글 폰트 사용시 마이너스 부호 표현하기

```
plt.rcParams['axes.unicode_minus'] = False
```

```
plt.rc('font', family='Malgun Gothic')
plt.rcParams['font.family'] ='Malgun Gothic'
plt.rc('axes', unicode_minus=False)
plt.rcParams['axes.unicode_minus'] = False
```

## 그래프 다듬기

```
plt.rc('font', family='Malgun Gothic')
plt.rcParams['axes.unicode_minus'] = False

plt.figure(figsize=(10, 4))
plt.plot(mean, 'k', label='mean')
plt.plot(low, 'b', label='low')
plt.plot(high, 'r', label='high')

plt.title('크리스마스의 기온 변화 그래프')
plt.xlabel('Day (December 25)')
plt.ylabel('Temperature')
plt.legend()
plt.grid()
plt.show()
```



LESSON 02

# 서울 연도별 평균 기온 변화 그래프

## 연도별 평균 기온 계산이 필요

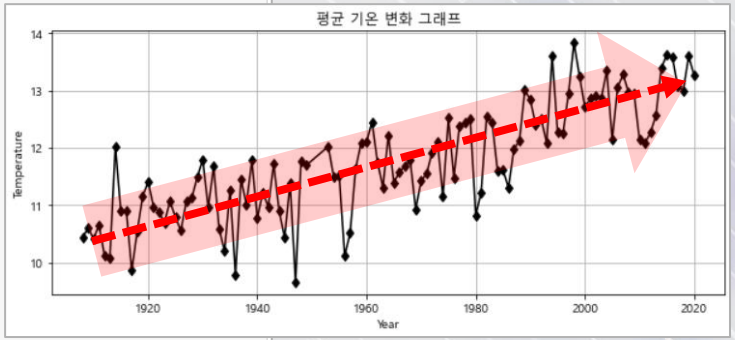
서울의 평균 기온은 계산이 필요: 365일의 평균기온합 / 365

```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding = 'cp949')
data = csv.reader(f)

header = next(data)

temp_day = []     # 1년 단위로 365일의 일별 평균 기온 데이터를 저장할 리스트 생성
temp_mean = []    # 연도 별로, temp_day에 저장된 평균 기온 데이터의 평균 값을 저장할 리스트 생성
year = []         # 연도 데이터를 저장할 리스트 생성 -> 그래프의 x축 데이터로 활용

for row in data:
    # 1907년도에는 10월부터 기온 데이터가 기록되어 있어서, 평균 기온이 매우 낮음 (분석에서 제외함)
    if int(row[0].split('-')[0]) > 1907:
        if row[2] != '':   # 평균 기온 데이터가 결측치가 아니라면 아래를 수행
            # 1월 1일부터 12월 31일까지의 일별 평균 기온 데이터를 저장
            temp_day.append(float(row[2]))
            if (row[0].split('-')[1] == '12') and (row[0].split('-')[2] == '31'):
                year.append(int(row[0].split('-')[0])) # 연도 데이터를 저장
                # 12월 31일이 되면 지금까지 temp_day에 저장했던 평균 기온 데이터의 평균 값을 계산
                temp_mean.append(sum(temp_day) / len(temp_day))
                temp_day = []   # 다음 년도의 평균 기온 데이터를 1월 1일부터 새롭게 저장해야 하므로
                # temp_day 리스트를 [] 비어있는 리스트로 만들어 줌 (이를 "초기화"라고 부름)

f.close()
```

# 평균 기온 상승 확인

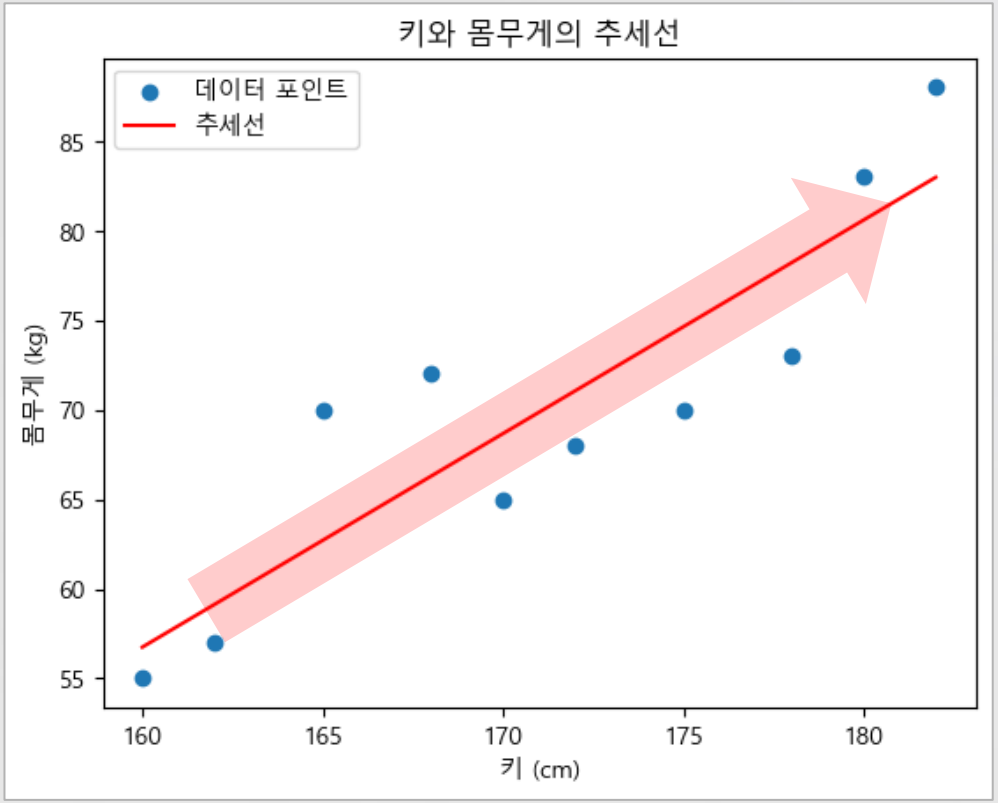
```
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding = 'cp949')
data = csv.reader(f)

header = next(data)

temp_day = []     # 1년 단위로 365일의 일별 평균 기온 데이터를 저장할 리스트 생성
temp_mean = []    # 연도 별로, temp_day에 저장된 평균 기온 데이터의 평균 값을 저장할 리스트 생성
year = []         # 연도 데이터를 저장할 리스트 생성 -> 그래프의 x축 데이터로 활용

for row in data:
    # 1907년도에는 10월부터 기온 데이터가 기록되어 있어서, 평균 기온이 매우 낮음 (분석에서 제외함)
    if int(row[0].split('-')[0]) > 1907:
        if row[2] != '':   # 평균 기온 데이터가 결측치가 아니라면 아래를 수행
            # 1월 1일부터 12월 31일까지의 일별 평균 기온 데이터를 저장
            temp_day.append(float(row[2]))
            if (row[0].split('-')[1] == '12') and (row[0].split('-')[2] == '31'):
                year.append(int(row[0].split('-')[0])) # 연도 데이터를 저장
                # 12월 31일이 되면 지금까지 temp_day에 저장했던 평균 기온 데이터의 평균 값을 계산
                temp_mean.append(sum(temp_day) / len(temp_day))
                temp_day = []   # 다음 년도의 평균 기온 데이터를 1월 1일부터 새롭게 저장해야 하므로
                # temp_day 리스트를 [] 비어있는 리스트로 만들어 줌 (이를 "초기화"라고 부름)

f.close()

plt.figure(figsize=(10, 4))
plt.rc('font', family='Malgun Gothic')
plt.title('평균 기온 변화 그래프')
plt.plot(year, temp_mean, color='k', marker='d')
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.grid()
plt.show()
```

서울의 평균 기온이
상승하고 있음을 알 수 있습니다.



## 추세선 개요

### 기존 데이터로 가장 근접한 1차원 직선 그리기



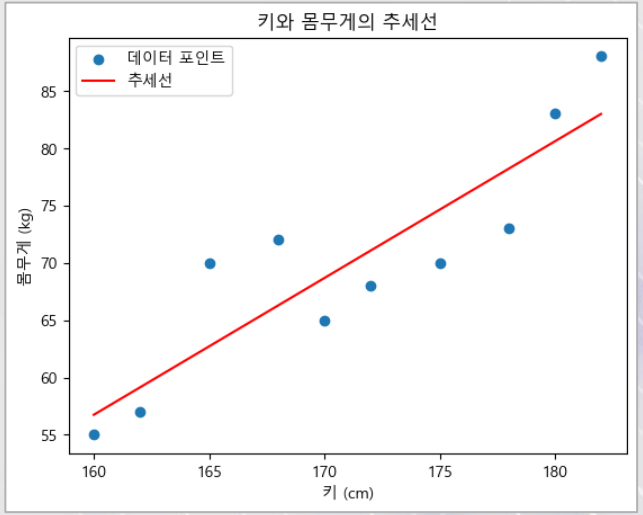
# Numpy로 추세선 그리기

## np.ployfilt(), np.ploy1d()

```python
import matplotlib.pyplot as plt
import numpy as np

# 예시 데이터
height = [160, 162, 165, 168, 170, 172, 175, 178, 180, 182]
weight = [55, 57, 70, 72, 65, 68, 70, 73, 83, 88]

# 추세선을 위한 다항식 회귀
degree = 1  # 1차 다항식 (직선)
coefficients = np.polyfit(height, weight, degree) # ax + b
polynomial = np.poly1d(coefficients) # coefficients[0]x + coefficients[1]

# 추세선을 위한 x값 생성
height_range = np.linspace(min(height), max(height), 100)

# 한글 처리
plt.rc('font', family='Malgun Gothic')
plt.rc('axes', unicode_minus=False)

# 그래프 그리기
plt.scatter(height, weight, label='데이터 포인트')
plt.plot(height_range, polynomial(height_range), color='red', label='추세선')
```

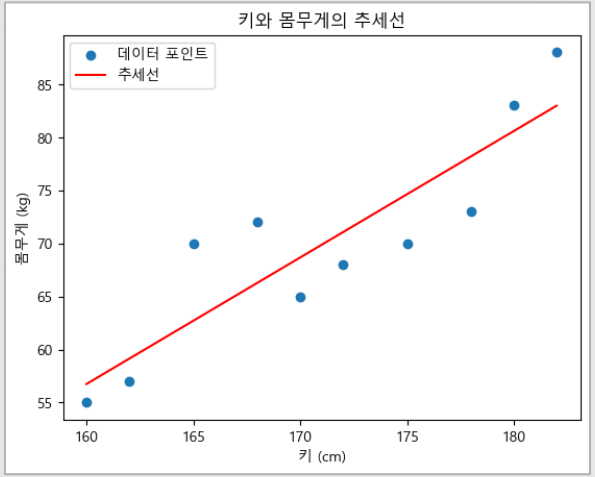
```
[    1.19239721 -134.03840186]
```

```
1.192 x - 134
```



# 키와 몸무게 관계 추세선

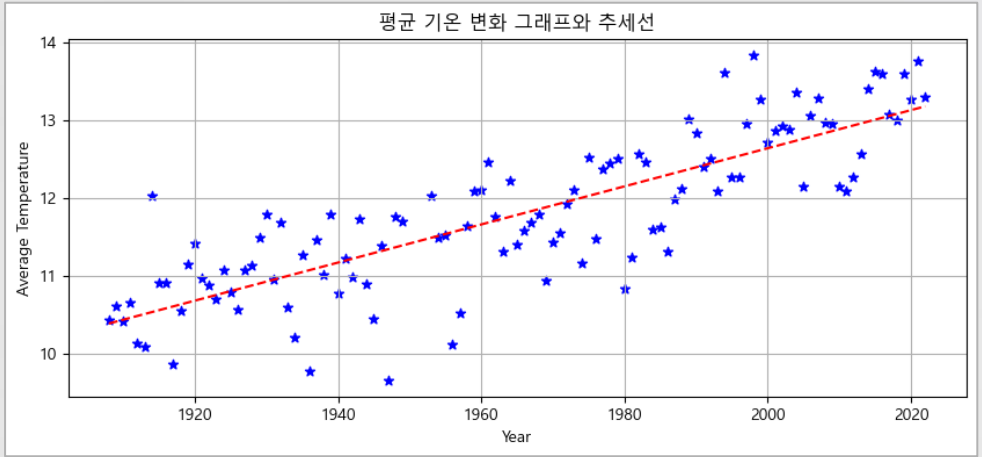
## 코드




```python
import matplotlib.pyplot as plt
import numpy as np

# 예시 데이터
height = [160, 162, 165, 168, 170, 172, 175, 178, 180, 182]
weight = [55, 57, 70, 72, 65, 68, 70, 73, 83, 88]

# 추세선을 위한 다항식 회귀
degree = 1  # 1차 다항식 (직선)
coefficients = np.polyfit(height, weight, degree) # ax + b
polynomial = np.poly1d(coefficients) # coefficients[0]x + coefficients[1]

# 추세선을 위한 x값 생성
height_range = np.linspace(min(height), max(height), 100)

# 한글 처리
plt.rc('font', family='Malgun Gothic')
plt.rc('axes', unicode_minus=False)

# 그래프 그리기
plt.scatter(height, weight, label='데이터 포인트')
plt.plot(height_range, polynomial(height_range), color='red', label='추세선')

# 그래프에 레이블 추가
plt.xlabel('키 (cm)')
plt.ylabel('몸무게 (kg)')
plt.title('키와 몸무게의 추세선')
plt.legend()

# 그래프 표시
plt.show()
```

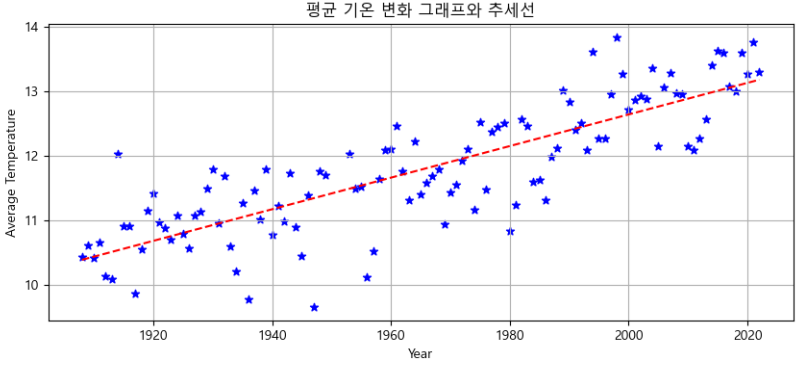
## 평균 기온 추세선 주 코드

```
plt.figure(figsize = (10,4))
plt.scatter(year, temp_mean, color='b', marker='*')

# 추세선 그리기
coeff = np.polyfit(year, temp_mean, 1)
poly = np.poly1d(coeff)
plt.plot(year, poly(year), '--r')
```



# 평균 기온 추세선 그리기

## 전체 코드




추세선 기울기: 0.024487114355201036

```python
import csv
import matplotlib.pyplot as plt

f = open('seoul.csv', encoding='cp949')
data = csv.reader(f)
next(data)

temp_day = []
temp_mean = []
year = []

for row in data :
    if int(row[0].split('-')[0]) > 1907:
        if row[2] != '':
            temp_day.append(float(row[2]))
            if row[0].split('-')[1] == '12' and row[0].split('-')[2] == '31' :
                year.append(int(row[0].split('-')[0]))
                temp_mean.append(sum(temp_day) / len(temp_day))
                temp_day = []

f.close()

plt.rcParams['font.family'] ='Malgun Gothic'
plt.rcParams['axes.unicode_minus'] = False

plt.figure(figsize = (10,4))
plt.scatter(year, temp_mean, color='b', marker='*')

# 추세선 그리기
coeff = np.polyfit(year, temp_mean, 1)
poly = np.poly1d(coeff)
plt.plot(year, poly(year), '--r')

plt.title('평균 기온 변화 그래프와 추세선')
plt.xlabel('Year')
plt.ylabel('Average Temperature')
plt.grid()
plt.show()
print(f'추세선 기울기: {coeff[0]}')
```

SUMMARY

# 학습정리

## 서울 일별 최고 기온 데이터 시각화

- 목록에 최고 기온 float(row[4]) 또는 float(row[-1])을 저장

## 특정 일자의 데이터 시각화

- 날짜 추출
  - ‘2025-05-16’.split(‘-’)[0], ‘2025-05-16’.split(‘-’)[1], ‘2025-05-16’.split(‘-’)[2]

## 서울 연도별 평균 기온 변화 시각화

- 365일의 평균기온합 / 365

## 추세선 그리기

- 추세선을 표현하는 기울기와 절편 구하기 a, b: np.ployfilt(x, y, 1)
- 1차원 다항식 연산식 생성 ax + b: np.ploy1d([a, b])