파이썬 데이터 분석과 머신러닝 숙명여자대학교창의융합연구소 초청강연회

강사: 김종훈 (제주대학교 초등컴퓨터교육전공 교수)
보조강사: 김동환 (동국대학교 컴퓨터·AI학부 학생)

· https://cafe.naver.com/scratchprogramming/14261
· 구글 로그인
· 구글 코랩: https://colab.research.google.com/

1. 자동차 연비, 배기량, CO2 배출량 분석

☑ 맷플롯립 한글 폰트 설정하기

```
!sudo apt-get install -y fonts-nanum
!sudo fc-cache -fv
!rm ~/.cache/matplotlib -rf
```

런타임 - 세션 다시 시작

```
import matplotlib.pyplot as plt

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

☑ 데이터 수집하기

공공데이터포털(https://www.data.go.kr/)에서 수집
https://cafe.naver.com/scratchprogramming/14261 게시판의 car20240731.csv 파일 다운로드
코랩 세션 저장소에 업로드

```
import pandas as pd

df = pd.read_csv('car20240731.csv')
display(df.head())
```

df['연료']

	업체명	모델명	연료	변속형식	차량형식
0	볼보	볼보 V60CCB5 AWD	휘발유	자동	하이브리드
1	볼보	볼보 XC60T8 AWD	전기+휘발유	자동	PHEV
2	벤츠	Mercedes-Benz GLC 220 d 4MATIC	경유	자동	하이브리드
3	벤츠	Mercedes-Benz E 450 4MATIC	휘발유	자동	하이브리드
4	랜드로버	더 뉴 레인지로버 P550e SWB	전기+휘발유	자동	PHEV

	업체명	모델명	연료	변속형식	차량형식	자동차 종류	도심주행연비	고속도로연비	CO2배출량(g_km)	1회충전주행거리(km)	배기량	등급	출시연도
0	볼보	볼보 V60CCB5 AWD	휘발유	자동	하이브리드	승용차	8.7	11.9	175.0	NaN	1969.0	4등급	2024
1	볼보	볼보 XC60T8 AWD	전기+휘발유	자동	PHEV	승용차	16.6	17.1	33.0	61.0	1969.0	PHEV	2024
2	벤츠	Mercedes-Benz GLC 220 d 4MATIC	경유	자동	하이브리드	승용차	13.8	15.5	130.0	NaN	1993.0	2등급	2024
3	벤츠	Mercedes-Benz E 450 4MATIC	휘발유	자동	하이브리드	승용차	9.0	13.1	162.0	NaN	2998.0	4등급	2024
4	랜드로버	더 뉴 레인지로버 P550e SWB	전기+휘발유	자동	PHEV	승용차	11.5	13.1	35.0	80.0	2996.0	PHEV	2024

· 데이터프레임(DataFrame)은 행과 열로 구성된 2차원 배열 구조로, 표 형태의 데이터를 저장하고 처리하는 데 적합하다. 여기서 열(column)은 '속성', 행(row)은 개별 데이터를 나타낸다.

· CSV 파일은 Comma-Separated Values의 줄임말로, 값들이 쉼표(,)로 구분된 텍스트 파일이다.

☑ 데이터 전처리하기

```
# 결측값(NaN)이 있는지 확인

print(pd.isna(df).sum())
print(len(df))
```

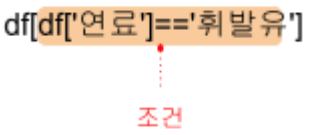
```
업체명              0
모델명              0
  ⋮
도심주행연비           1
고속도로연비           1
CO2배출량(g_km)     312
1회충전주행거리(km)    3095
배기량             311
  ⋮
3498
```

· NaN은 'Not a Number'의 줄임말로, 쉽게 말해 숫자가 아닌 값, 즉 비어 있는 값(결측값)을 뜻한다. 평균 계산 등 통계 처리할 때는 제거하거나 처리해주어야 한다.

```
# '배기량', '도심주행연비', 'CO2배출량(g_km)' 열에 결측값이 있는 행을 제거

df = df.dropna(subset=['배기량','도심주행연비','CO2배출량(g_km)'])
print(len(df))
```

```
3185
```

```
# df에서 '연료' 값이 '휘발유'인 행만 추출하여 df_gasoline으로 저장

df_gasoline = df[df['연료']=='휘발유']
display(df_gasoline.head())
print(len(df_gasoline))
```



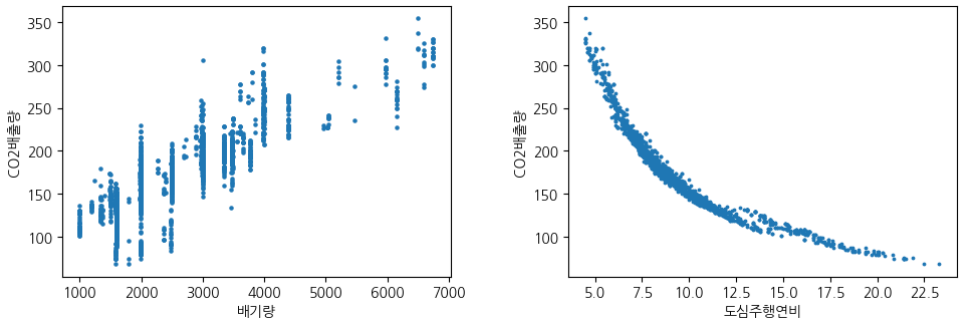
	업체명	모델명	연료	변속형식	차량형식	자동차 종류	도심주행연비	고속도로연비	CO2배출량(g_km)	1회충전주행거리(km)	배기량	등급	출시연도
0	볼보	볼보 V60CCB5 AWD	휘발유	자동	하이브리드	승용차	8.7	11.9	175.0	NaN	1969.0	4등급	2024
3	벤츠	Mercedes-Benz E 450 4MATIC	휘발유	자동	하이브리드	승용차	9.0	13.1	162.0	NaN	2998.0	4등급	2024
5	볼보	볼보 XC60B6 AWD	휘발유	자동	하이브리드	승용차	8.8	11.5	174.0	NaN	1969.0	4등급	2024
6	볼보	볼보 S60B5	휘발유	자동	하이브리드	승용차	10.5	14.3	143.0	NaN	1969.0	3등급	2024
7	벤츠	Mercedes-Maybach S 580	휘발유	자동	하이브리드	승용차	7.0	10.1	214.0	NaN	3982.0	5등급	2024

```
1996
```

☑ 데이터 분석하기

· 배기량과 CO2배출량, 도심주행연비와 CO2배출량 산점도 그리기

```
import matplotlib.pyplot as plt

plt.figure(figsize=(5,3.5))
plt.scatter(df_gasoline['배기량'], df_gasoline['CO2배출량(g_km)'], s=5)
plt.xlabel('배기량')
plt.ylabel('CO2배출량')
plt.show()

plt.figure(figsize=(5,3.5))
plt.scatter(df_gasoline['도심주행연비'], df_gasoline['CO2배출량(g_km)'], s=3)
plt.xlabel('도심주행연비')
plt.ylabel('CO2배출량')
plt.show()
```



· 산점도: 두 변수의 관계를 시각적으로 나타내기 위해 좌표 평면에 점을 찍어 만드는 그래프

```
import matplotlib.pyplot as plt
plt.scatter(x, y [옵션])
```

x축의 값이 'x'이고 y축의 값이 'y'인 좌표에 점을 찍어 산점도를 그린다.

<table>
<tr><th>옵션</th><th>설명</th></tr>
<tr><td>color=점 색상</td><td>점 색상을 설정한다.</td></tr>
<tr><td>s=점 크기</td><td>점 크기를 설정한다.</td></tr>
<tr><td>marker=점 모양</td><td>점 모양을 설정한다. 대표적인 marker 값은 다음과 같다.
<table>
<tr><th>market 값</th><th>설명</th><th>market 값</th><th>설명</th></tr>
<tr><td>'o'</td><td>원 모양(기본값)</td><td>'x'</td><td>x 모양</td></tr>
<tr><td>'^'</td><td>위방향 삼각형 모양</td><td>'*'</td><td>별 모양</td></tr>
<tr><td>'>'</td><td>아래방향 삼각형 모양</td><td>'h'</td><td>육각형 모양</td></tr>
<tr><td>'s'</td><td>사각형 모양</td><td>'D'</td><td>다이아몬드 모양</td></tr>
</table></td></tr>
<tr><td>alpha=투명도</td><td>투명도를 설정한다. 기본값은 1이다.</td></tr>
</table>

· 상관계수

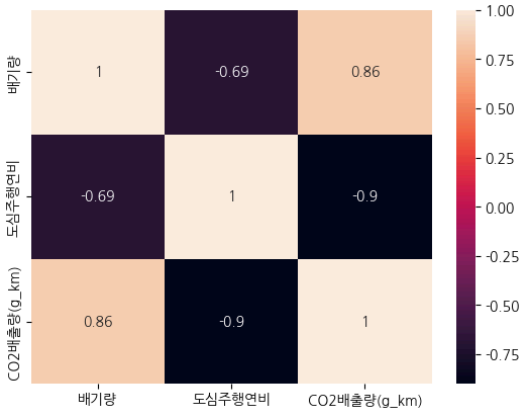
```
cor1 = df_gasoline['배기량'].corr(df_gasoline['CO2배출량(g_km)'])
print(cor1)
cor2 = df_gasoline['도심주행연비'].corr(df_gasoline['CO2배출량(g_km)'])
print(cor2)

corr = df_gasoline[['배기량','도심주행연비','CO2배출량(g_km)']].corr()
display(corr)
```

```
0.8568170602284881
-0.9001007202050113
```

	배기량	도심주행연비	CO2배출량(g_km)
배기량	1.000000	-0.692556	0.856817
도심주행연비	-0.692556	1.000000	-0.900101
CO2배출량(g_km)	0.856817	-0.900101	1.000000

· 상관계수: 두 변수 사이의 관련성을 수치로 나타낸 값

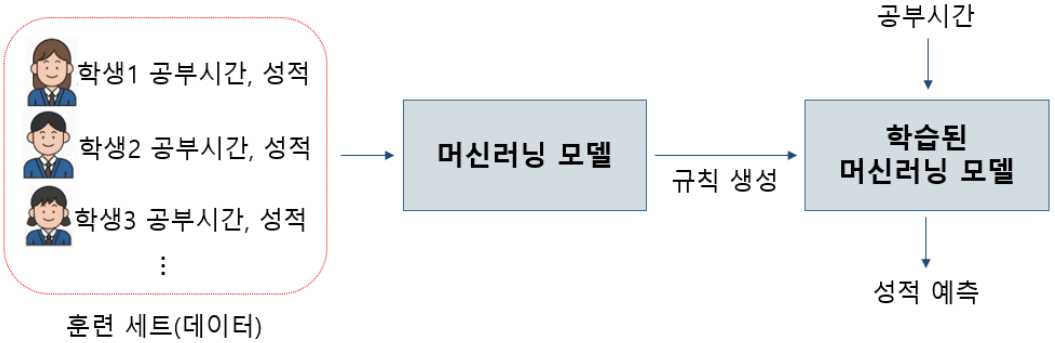
```
import seaborn as sns

sns.heatmap(corr, annot=True)
plt.show()
```



☑ 머신러닝(회귀 모델) 동작 과정

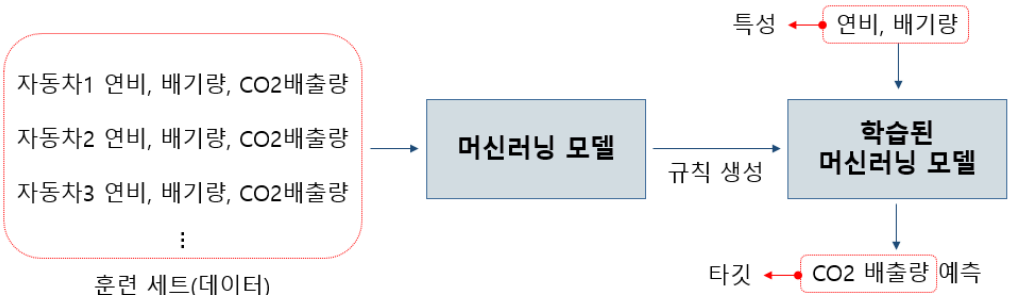
머신러닝(machine learning) : 컴퓨터가 데이터를 이용해 스스로 규칙을 익히고, 그 규칙을 바탕으로 새로운 상황을 예측하거나 판단하는 기술

· 공부시간으로 성적을 예측하는 머신러닝 모델

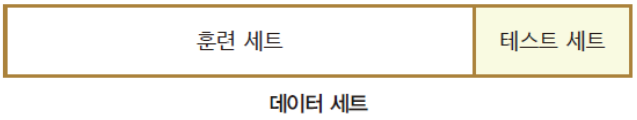


* 공부시간: 특성, 성적: 타깃

☑ 머신러닝 모델 생성하기

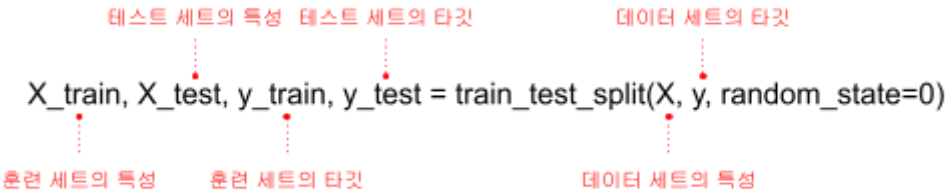


· 데이터 분할




```
from sklearn.model_selection import train_test_split

X = df_gasoline[['배기량', '도심주행연비']]   # 특성
y = df_gasoline['CO2배출량(g_km)']          # 타깃

X_train,X_test,y_train,y_test = train_test_split(X, y, random_state=0)
```

	특성	타깃
훈련 세트	X_train	y_train
테스트 세트	X_test	y_test



· 모델 생성

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

· LinearRegression()로 선형 회귀 모델 객체를 생성한다.
· model.fit(X_train, y_train)은 훈련 세트(데이터)를 이용해 모델을 학습시키는 과정으로 X_train은 특성(입력값), y_train은 타깃(정답)이다.

· 모델 평가

```
print(f'훈련 세트 정확도: {model.score(X_train, y_train):.3f}')
print(f'테스트 세트 정확도: {model.score(X_test, y_test):.3f}')
```

```
훈련 세트 정확도: 0.915
```

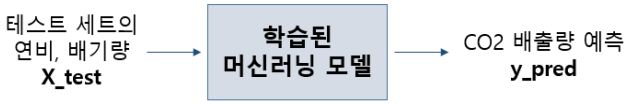
테스트 세트 정확도: 0.914

· 모델 예측

```
y_pred = model.predict(X_test)

print(y_test.values[:20])
print(y_pred[:20])
```

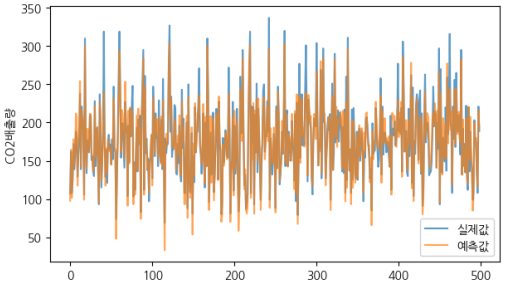
	특성	타깃
훈련 세트	X_train	y_train
테스트 세트	X_test	y_test

학습된 모델을 사용해 테스트 세트를 예측하는 과정으로, 테스트 세트의 특성인 X_test로 예측된 CO_2 배출량을 y_pred에 저장




```
[107. 162. 108. 123. 172. 139. 156. 188. 178. 129. 159. 177. 238. 139.  221. 202. 192. 106. 310. 177.]
[ 97.85141806 164.60040706 101.58988345 127.19989721 178.15440084  142.57531002 163.66579071
 211.76372726 207.09064552 117.47836138 172.54670275 193.61696812 254.30613882 141.77838646
 215.98602465  190.30441338 205.05666662  99.1338084  300.11220159 171.1427215 ]
```

```
plt.figure(figsize=(7,4))
plt.plot(y_test.values, alpha=0.7, label='실제값')
plt.plot(y_pred, alpha=0.7, label='예측값')
plt.ylabel('CO2배출량')
plt.legend()
plt.show()
```




```
import matplotlib.pyplot as plt
plt.plot([x], y, [옵션])
```

x축의 값이 'x'이고 y축의 값이 'y'인 선 그래프를 그린다.

옵션	설명
color=선 색상	선 그래프의 선 색상을 설정한다. 선 색상은 'blue'와 같은 색상 이름, '#CEFBC9'와 같은 색상 코드 등으로 지정할 수 있다.
marker=데이터 점 모양	데이터의 점 모양을 설정한다. 대표적인 marker 값은 다음과 같다.
linestyle=선 모양	선 그래프의 선 모양을 설정한다. 대표적인 linestyle 값은 다음과 같다.

'.'	점 모양	'x'	x 모양
'o'	원 모양	'*'	* 모양
'v'	아래방향 삼각형 모양	'+'	+ 모양
'^'	위방향 삼각형 모양	'h'	육각형 모양

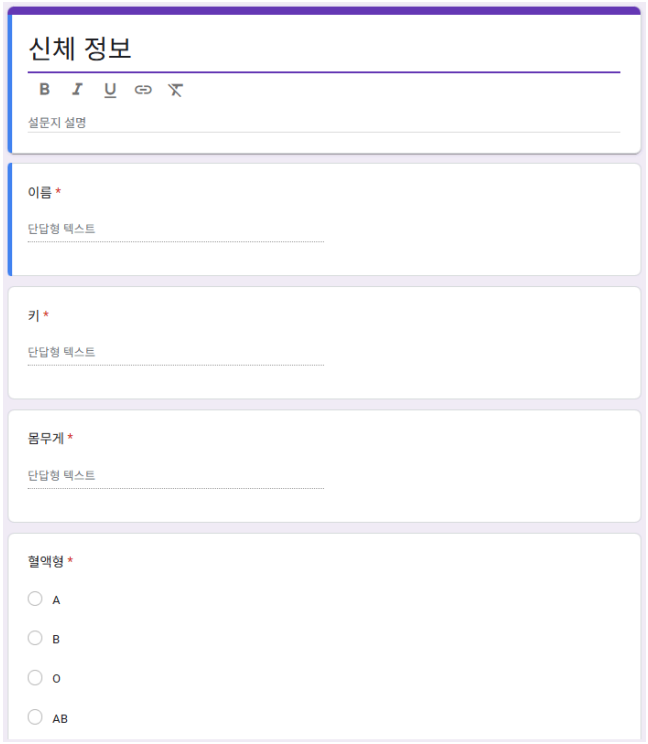
'-'	실선	'-.'	일점 쇄선

	'--'	파선	':'	점선

2. 신체 정보 설문 결과 분석하기

☑ 구글 설문

https://docs.google.com/forms/



- 설문 10개
- 스프레드시트 URL 복사

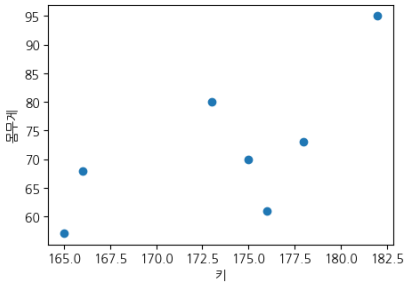
```
import pandas as pd
import gspread
from google.colab import auth
from google.auth import default

# 인증
auth.authenticate_user()
creds, _ = default()
gc = gspread.authorize(creds)

# 구글 시트 연결
url = '???'      # 복사한 스프레드시트 URL 붙이기
worksheet = gc.open_by_url(url).sheet1
rows = worksheet.get_all_values()

# 데이터프레임 생성 및 숫자형 변환
df = pd.DataFrame(rows[1:], columns=rows[0])
df['키'] = df['키'].astype(float)
```

	타임스탬프	이름	키	몸무게	혈액형
0	2025. 6. 26 오전 5:37:07	김민수	175.0	70.0	A
1	2025. 6. 26 오전 5:37:26	이서연	165.0	57.0	B
2	2025. 6. 26 오전 5:37:48	박지훈	178.0	73.0	A
3	2025. 6. 26 오전 5:38:15	최윤아	166.0	68.0	AB
4	2025. 6. 26 오전 5:38:40	정우진	173.0	80.0	O
5	2025. 6. 26 오전 5:39:03	강하늘	176.0	61.0	B
6	2025. 6. 26 오전 5:39:26	윤지호	182.0	95.0	A

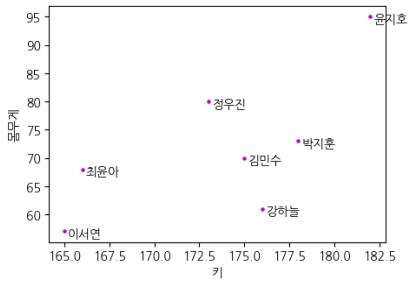
```
df['몸무게'] = df['몸무게'].astype(float)
display(df)
```

☑ 키와 몸무게 산점도 그리기

```
import matplotlib.pyplot as plt

plt.figure(figsize=(5, 3.5))
plt.scatter(df['키'], df['몸무게'])
plt.xlabel('키')
plt.ylabel('몸무게')
plt.show()
```




```
# 주석 표시하기
plt.figure(figsize=(5, 3.5))
plt.scatter(df['키'], df['몸무게'], color='m', s=5)    # 'm': 자홍색
for index, row in df.iterrows():
  plt.annotate(row['이름'],xy=(row['키'],row['몸무게']),xytext=(3,-5),textcoords='offset points')
plt.xlabel('키')
plt.ylabel('몸무게')
plt.show()
```



‘김민수’	175.0	70.0
row['이름']	row['키']	row['몸무게']

‘이서연’	165.0	57.0
row['이름']	row['키']	row['몸무게']

plt.annotate(주석, 기준 좌표, xytext=기준 좌표에서 떨어진 정도, textcoords=단위)

‘기준 좌표’로부터 ‘기준 좌표에서 떨어진 정도’만큼 떨어진 위치에 ‘주석’을 표시한다. xytext 단위는 주석 위치의 단위로, ‘단위’가 ‘offset points’면 포인트이고 ‘offset pixels’이면 픽셀이다.

☑ BMI 계산하기

$$BMI = \frac{\text{몸무게}}{\text{키}^2}$$

몸무게 단위는 kg, 키 단위는 m

```
df['BMI'] = df['몸무게'] / (df['키']/100)**2
display(df)
```

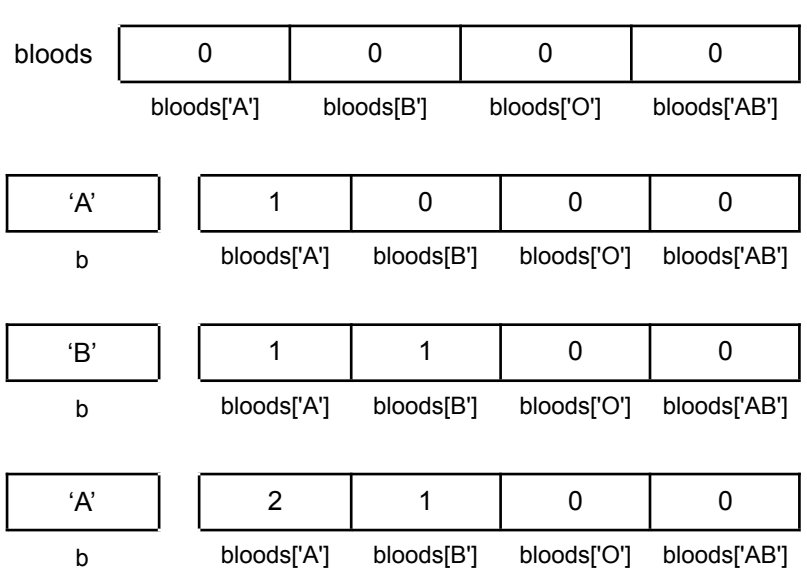
	타임스탬프	이름	키	몸무게	혈액형	BMI
0	2025. 6. 26 오전 5:37:07	김민수	175.0	70.0	A	22.857143
1	2025. 6. 26 오전 5:37:26	이서연	165.0	57.0	B	20.936639
2	2025. 6. 26 오전 5:37:48	박지훈	178.0	73.0	A	23.040020
3	2025. 6. 26 오전 5:38:15	최윤아	166.0	68.0	AB	24.677021
4	2025. 6. 26 오전 5:38:40	정우진	173.0	80.0	O	26.729927
5	2025. 6. 26 오전 5:39:03	강하늘	176.0	61.0	B	19.692665
6	2025. 6. 26 오전 5:39:26	윤지호	182.0	95.0	A	28.680111

☑ 혈액형 빈도 세기

```
bloods = {'A':0, 'B':0, 'O':0, 'AB':0}

for b in df['혈액형']:
  bloods[b] += 1

print(bloods)
```

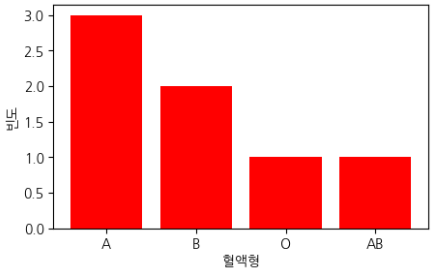
```
{'A': 3, 'B': 2, 'O': 1, 'AB': 1}
```

딕셔너리: 키(key)와 값(value)의 쌍으로 데이터를 저장하는 자료형



이름	키	몸무게	혈액형	BMI
김민수	175.0	70.0	A	22.857143
이서연	165.0	57.0	B	20.936639
박지훈	178.0	73.0	A	23.040020
최윤아	166.0	68.0	AB	24.677021

```
plt.figure(figsize=(5,3))
plt.bar(bloods.keys(), bloods.values(), color='red')
plt.xlabel('혈액형')
plt.ylabel('빈도')
plt.show()
```




```
import matplotlib.pyplot as plt
plt.bar(x, y, [옵션])
```

x축의 값이 'x'이고 막대의 높이가 'y'인 막대 그래프를 그린다.

옵션	설명
color=막대 색상	막대 색상을 설정한다.
edgecolor=막대 테두리 색상	막대 테두리 색상을 설정한다.
width=막대 너비	막대 너비를 설정한다. 기본값은 0.8이다.

☑ 머신러닝 모델 생성하기

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(df[['키']], df['몸무게'])      # 특성: 키, 타깃: 몸무게
```

데이터 크기가 작기 때문에 훈련 세트(훈련 데이터)와 테스트 세트(테스트 데이터)로 분할하지 않고, 전체 데이터를 사용하여 모델을 훈련한다.

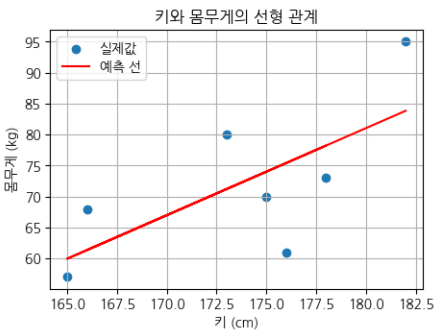
```
print(f"정확도: {model.score(df[['키']], df['몸무게']):.3f}")
```

정확도: 0.473

```
import numpy as np

print(model.predict([[175],[157]]))
```

[74.00870647 48.69900498]

```
import matplotlib.pyplot as plt

plt.figure(figsize=(5, 3.5))
plt.scatter(df['키'], df['몸무게'], label='실제값')
plt.plot(df['키'], model.predict(df[['키']]), color='red', label='예측 선')
plt.xlabel('키 (cm)')
plt.ylabel('몸무게 (kg)')
plt.legend()
plt.grid(True)
plt.show()
```



[챗GPT 프로젝트]

다음과 유사하게 질문하여 답변받은 주제 중 한 개를 선택하여 추가 질문하여 프로젝트를 완성하여라. 질문시 필요에 따라 대상(예: 중학생, 대학생), 분야(예: 환경, 건강 등)를 질문에 덧붙일 수 있다.

질문: LinearRegression()을 활용한 머신러닝 회귀 모델로 문제를 해결할 수 있는 융합형 탐구 주제 5가지를

제시해 주세요.

참고문헌

· 김종훈, 김동건, 『파이썬 마스터』, 한빛아카데미, 2025.