데이터 전처리

데이터 전처리 기법

- 데이터 실수화 (Data Vectorization) : 컴퓨터가 이해할 수 있는 값으로 데이터 실수화
- 데이터 정제 (Data Cleaning) : 불완전하거나 잡음이 섞인 데이터 제거
- 데이터 통합 (Data Integration) : 여러 개의 데이터 파일을 하나로 병합
- 데이터 축소 (Data Reduction) : 데이터 수를 줄이거나 차원을 축소
- 데이터 변환 (Data Transformation) : 데이터 정규화
- 데이터 균형 (Data Balancing) : 클래스 간 데이터 불균형

데이터 실수화, 정제, 통합, 축소

```python
w = []; h = []; t = []
with open("data/health.csv", "r") as file:
    lines = file.readlines()[1:]
    for line in lines:
        a, b, c = line.strip().split(",")
        h.append(float(a)) # 키
        w.append(float(b)) # 몸무게
        t.append(int(c))   # 어린이/청소년
data = [[x, y] for x, y in zip(h, w)]                # 리스트 생성 [키, 몸무게]
# data3 = [[x, y, z] for x, y, z in zip(h, w, t)]    # 리스트 생성 [키, 몸무게, 정답]
# ch_h = [x for x, y, z in data3 if z == 1]          # 어린이 키
# ch_w = [y for x, y, z in data3 if z == 1]          # 어린이 몸무게
# ad_h = [x for x, y, z in data3 if z == 0]          # 청소년 키
# ad_w = [y for x, y, z in data3 if z == 0]          # 청소년 몸무게
```

```python
from sklearn.neighbors import KNeighborsClassifier
neighbor = 3 # int(input("how many points?"))
kn = KNeighborsClassifier(n_neighbors = neighbor, p = 2)
kn.fit(data, t)
print("Eval:", kn.score(data, t))

# import random
# test_h = random.randrange(120, 160)
# test_w = random.randrange(20, 60)
test_h = 150; test_w = 29
print("Test:", test_h, test_w, "=>", kn.predict([[test_h, test_w]]))
print("Prob:", kn.predict_proba([[test_h, test_w]]))
```

```
Eval: 1.0
Test: 150 29 => [0]
Prob: [[0.66666667 0.33333333]]
```

실행결과

```
키,몸무게,눈,코,어린이여부
130.1,30.7,2,1,어린이
120.5,29.2,2,1,어린이
127.3,25.4,2,1,어린이
122.9,23.0,2,1,NA
126.0,25.8,2,1,어린이
```

```
키,몸무게,눈,코,어린이여부
152.8,49.9,2,1,청소년
155.9,46.2,2,1,청소년
158.5,60.0,2,1,청소년
156.6,62.2,2,1,청소년
150.1,49.4,2,1,청소년
```

전체 데이터

```
H,W,T
130.1,30.7,어린이
120.5,29.2,어린이
127.3,25.4,어린이
122.9,23.0,어린이
126.0,25.8,어린이
152.8,49.9,청소년
155.9,46.2,청소년
158.5,60.0,청소년
156.6,62.2,청소년
150.1,49.4,청소년
```

```
H,W,T
130.1,30.7,1
120.5,29.2,1
127.3,25.4,1
122.9,23.0,1
126.0,25.8,1
152.8,49.9,0
155.9,46.2,0
158.5,60.0,0
156.6,62.2,0
150.1,49.4,0
```

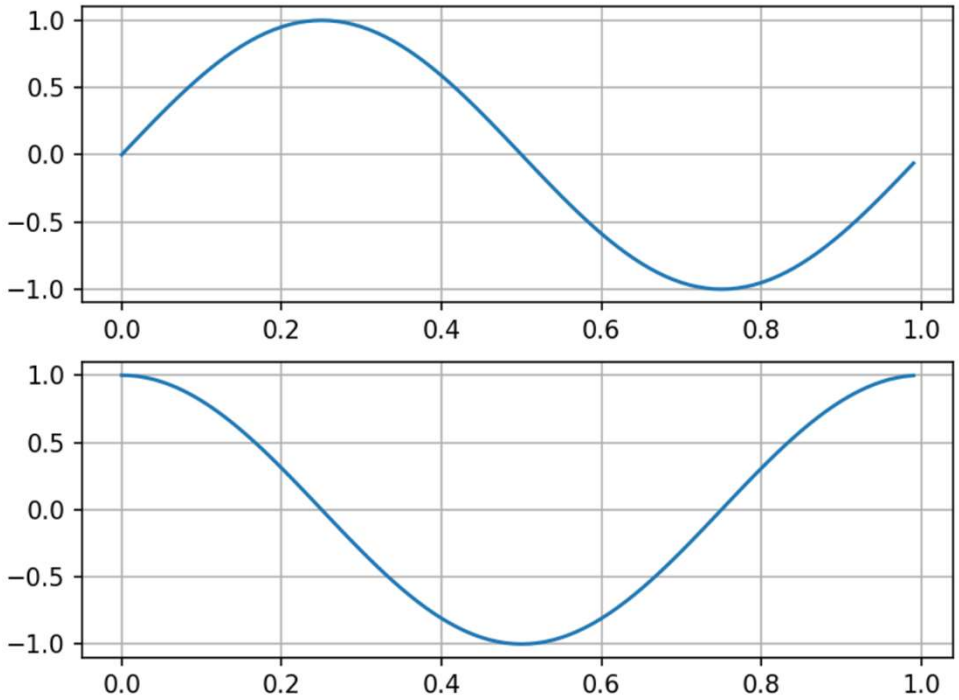
어린이

청소년

matplotlib.pyplot

```
import numpy as np
import matplotlib.pyplot as plt
t = np.arange(0, 100) * 0.01
s = np.sin(2 * np.pi * t)
c = np.cos(2 * np.pi * t)

plt.subplot(2, 1, 1); plt.plot(t, s); plt.grid()
plt.subplot(2, 1, 2); plt.plot(t, c); plt.grid()
plt.show()
```




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

x = np.arange(0, 5, 0.1)
y = np.sin(x)
plt.plot(x, y)
```

https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.html

데이터 시각화

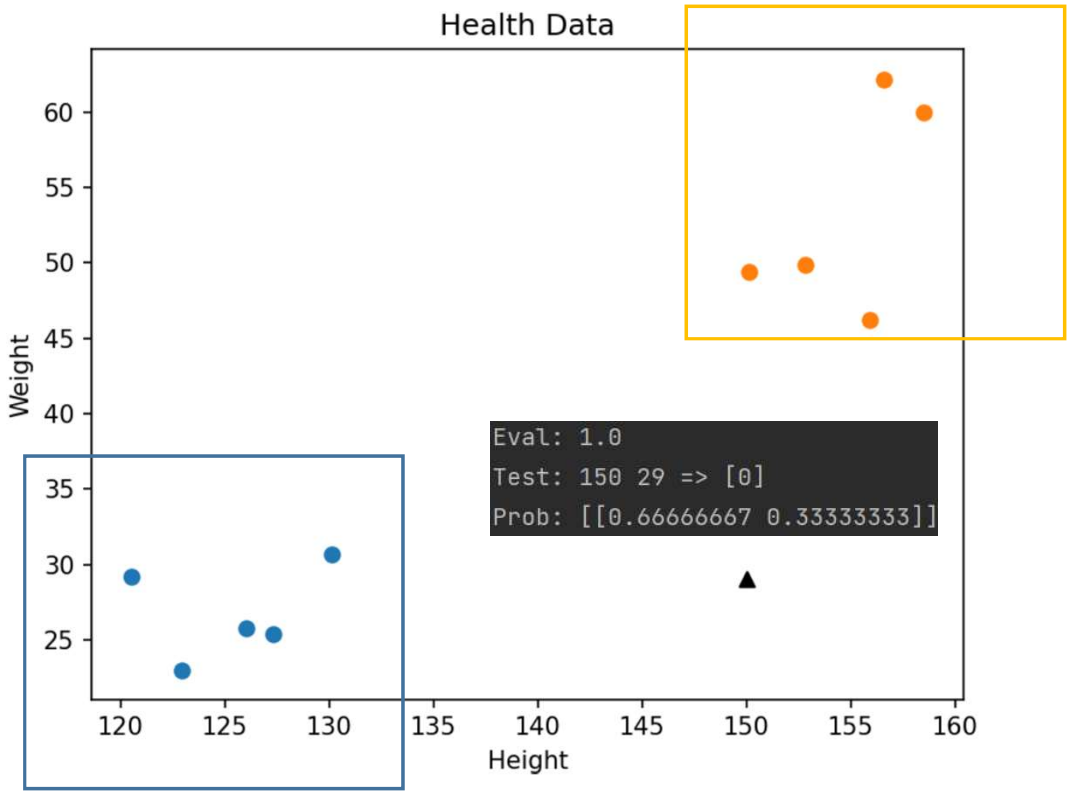
```
import matplotlib.pyplot as plt
plt.scatter(ch_h, ch_w)
plt.scatter(ad_h, ad_w)
plt.xlabel("Height")
plt.ylabel("Weight")
plt.title("Health Data")
test_h = 150; test_w = 29
plt.scatter(test_h, test_w, marker="^", c="black")
plt.show()
```

```
H,W,T
130.1,30.7,1
120.5,29.2,1
127.3,25.4,1
122.9,23.0,1
126.0,25.8,1
152.8,49.9,0
155.9,46.2,0
158.5,60.0,0
156.6,62.2,0
150.1,49.4,0
```

어린이

청소년

청소년



어린이

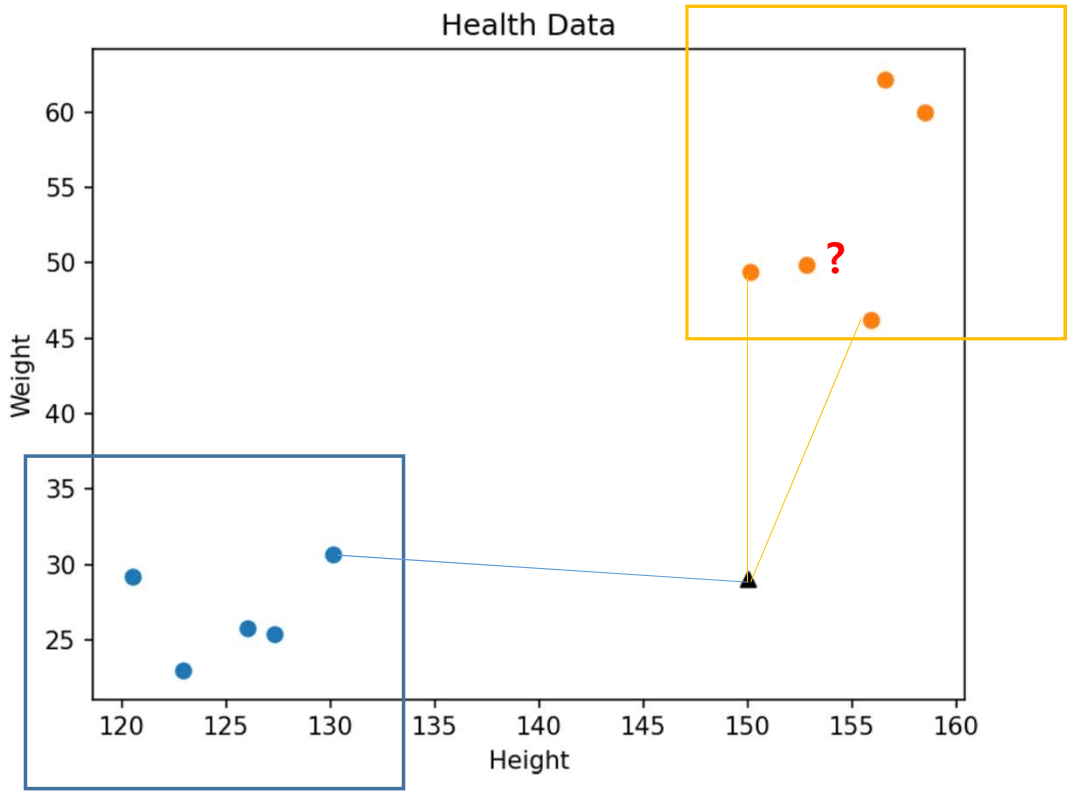
데이터 분석 (K=3)

```
test_h = 150; test_w = 29
print("Test:", test_h, test_w, "=>", kn.predict([[test_h, test_w]]))
print("Prob:", kn.predict_proba([[test_h, test_w]]))
dist, idx = kn.kneighbors([[test_h, test_w]], n_neighbors = neighbor)
print(dist, idx)
```

```
Test: 150 29 => [0]
Prob: [[0.66666667 0.33333333]]
[[18.18378398 19.97248107 20.4002451 ]] [[6 0 9]]
```

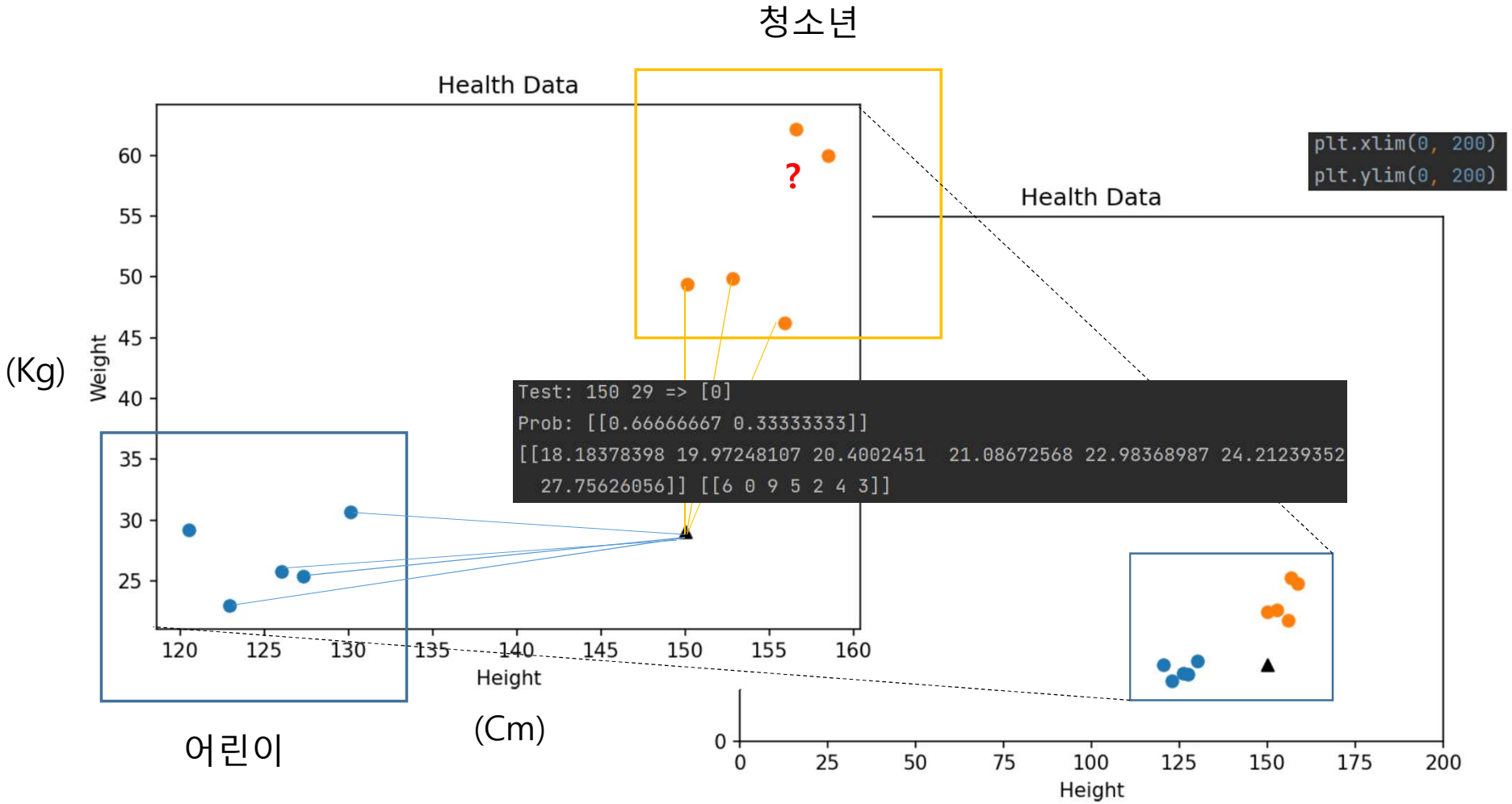
H, W, T	
130.1, 30.7, 1	어린이
120.5, 29.2, 1	
127.3, 25.4, 1	
122.9, 23.0, 1	
126.0, 25.8, 1	
152.8, 49.9, 0	어른
155.9, 46.2, 0	
158.5, 60.0, 0	
156.6, 62.2, 0	
150.1, 49.4, 0	

청소년



어린이

데이터 분석 (K=7)


청소년
Health Data
plt.xlim(0, 200)
plt.ylim(0, 200)
Health Data
?
Test: 150 29 => [0]
Prob: [[0.66666667 0.33333333]]
[[18.18378398 19.97248107 20.4002451 21.08672568 22.98368987 24.21239352
27.75626056]] [[6 0 9 5 2 4 3]]
Weight
(Kg)
Height
(Cm)
어린이

데이터 변환

- 데이터가 가진 특성 간 스케일 차이가 심하면 패턴을 찾는데 문제 발생
- 표준화 (Standardization) – 데이터가 표준정규분포의 속성을 가지도록 조정

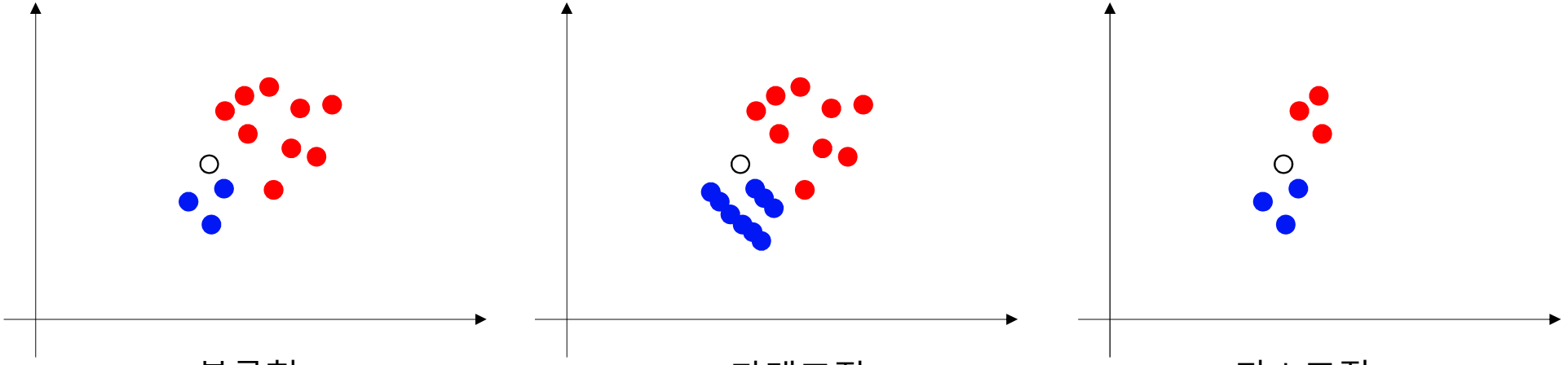
$$x_{std} = \frac{x - mean(x)}{sd(x)}$$

- 정규화 (Normalization) – 데이터의 값을 [0, 1]로 조정

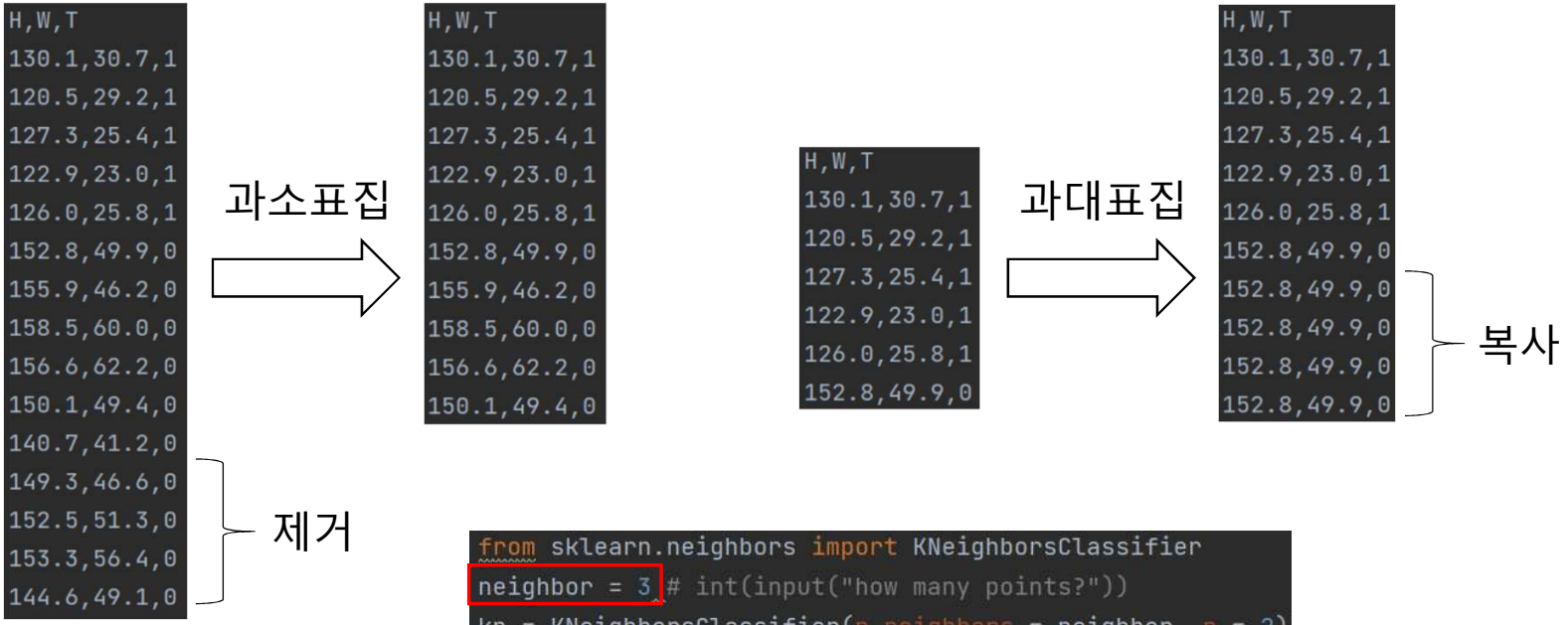
$$x_{nor} = \frac{x - min\,(x)}{\max(x) - min\,(x)}$$

데이터 불균형

- 분류 문제 해결 시, 특정 클래스의 관측치가 다른 클래스에 비해 매우 낮게 나타는 경우
- 과소표집 (Undersampling) – 다수 클래스의 표본을 임의로 데이터로부터 제거하는 것
- 과대표집 (Oversampling) – 소수 클래스의 표본을 복제하여 이를 데이터에 추가하는 것



과소표집, 과대표집




```python
from sklearn.neighbors import KNeighborsClassifier
neighbor = 3 # int(input("how many points?"))
kn = KNeighborsClassifier(n_neighbors = neighbor, p = 2)
kn.fit(data, t)
print("Eval:", kn.score(data, t))
```

참고자료

- 지능기전공학부 최유경 교수님 자료, https://github.com/sejongresearch/2021.MachineLearning
- 코랩(Colab), https://colab.research.google.com/
- 파이썬(Python), https://www.python.org/doc/
- 사이킷런(sckit-learn), https://scikit-learn.org/stable/index.html
- 판다스(pandas), https://pandas.pydata.org/
- 맷플롯립(matplotlib), https://matplotlib.org/
- 씨본(seaborn), https://seaborn.pydata.org/
- 캐글(Kaggle), https://www.kaggle.com/
- 넘파이(numpy), https://numpy.org/doc/stable/
- 스택오퍼플러우(stackoverflow), https://stackoverflow.com/