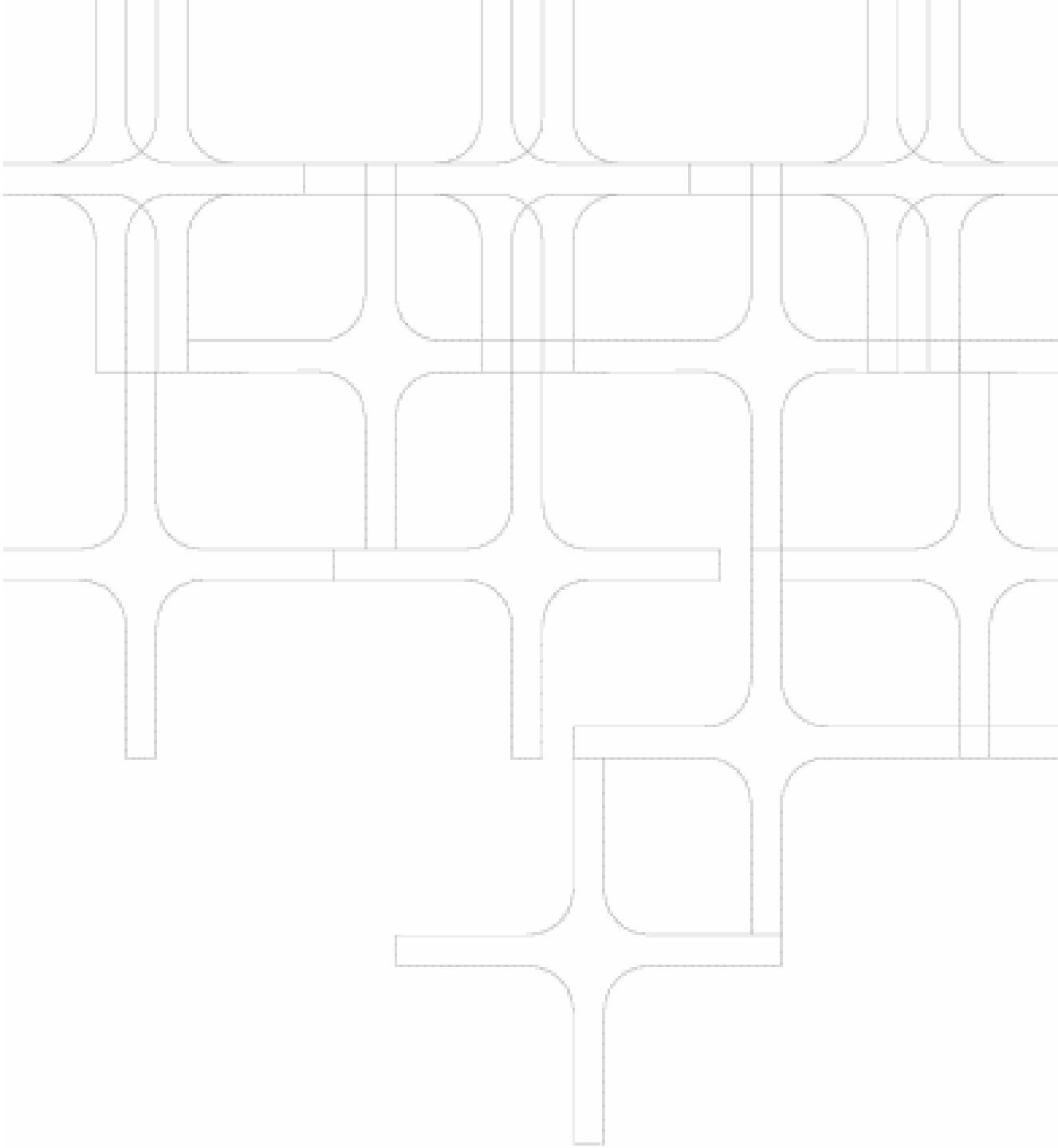
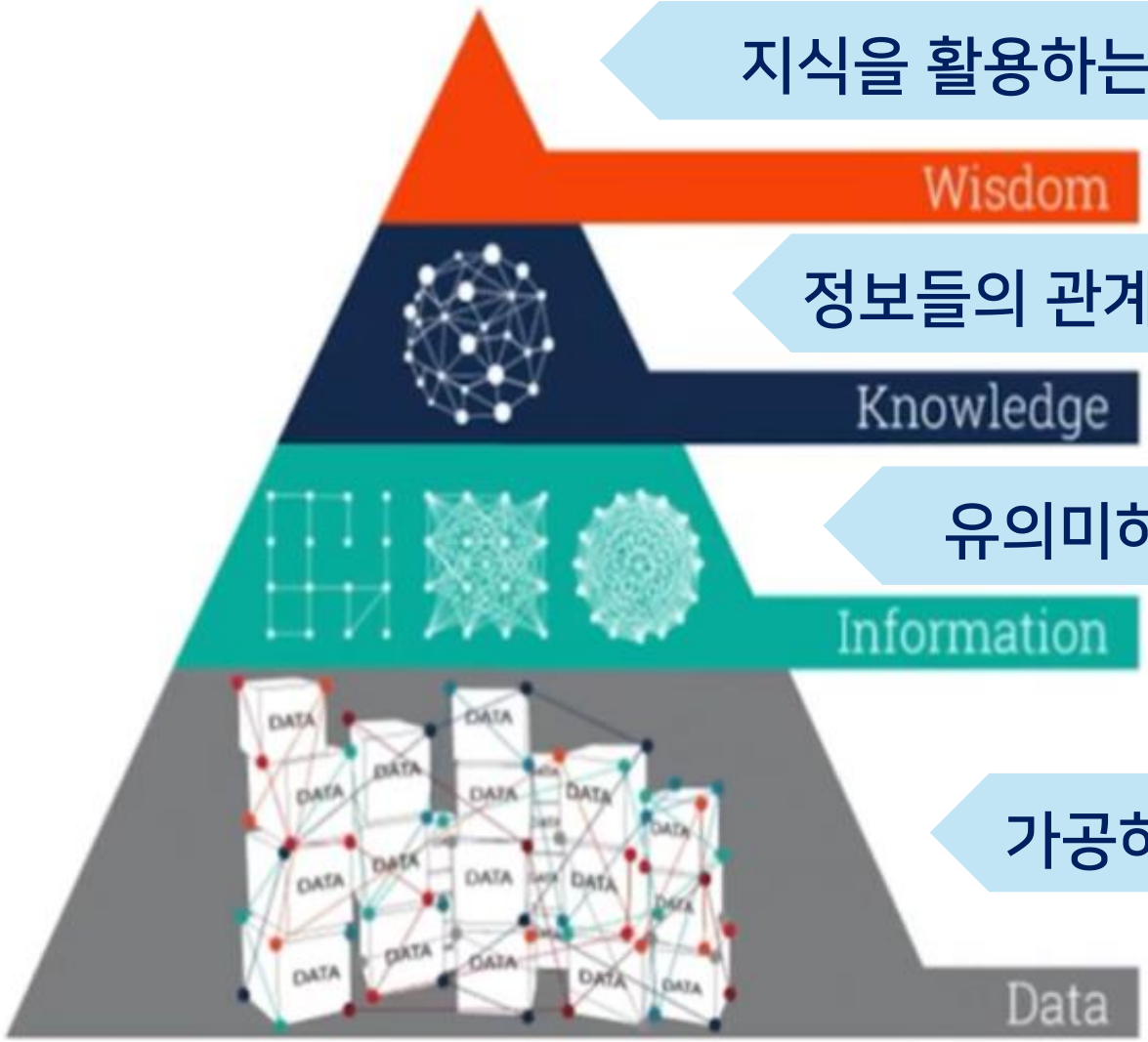
Chapter. 02

Data

- 데이터의 정의
- 데이터의 구성
- 데이터의 활용
- 데이터의 수집
- 데이터의 전처리





지식을 활용하는 창의적인 아이디어

정보들의 관계를 분석해 확인한 가치 있는 정보

유의미하게 가공된 2차 데이터

가공하기 전의 순수한 상태의 자료들

- 관찰이나 실험, 조사를 통해 수집되는 값 또는 특성
- 컴퓨터가 처리할 수 있는 문자, 숫자, 소리, 그림 따위의 형태로 된 자료
- 하나의 데이터는 여러 가지 속성들로 구성

직접 경험하지 않아도 데이터를 통해 경험을 얻을 수 있다.

https://www.sap.com/korea/products/technology-platform/what-is-big-data.html

다양한 유형의 데이터

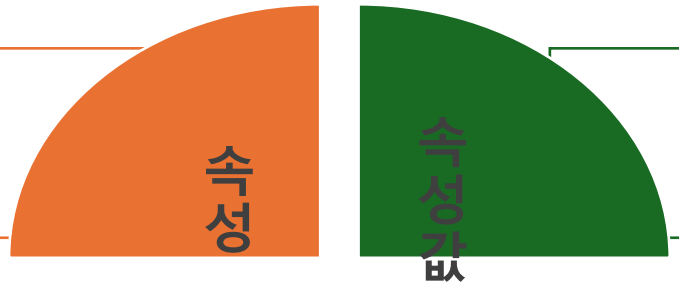
- ✓ 각 컬럼의 형식이 문자열, 숫자, 날짜 등으로 서로 다른 표
- ✓ 스프레드시트와 비슷한 데이터.
- ✓ 관계형 데이터베이스
- ✓ 탭이나 쉼표로 구분되는 텍스트 파일 형식으로 저장되는 대부분의 데이터를 포함
- ✓ 다차원 배열(행렬)
- ✓ SQL에서 기본키, 외래키 같은 키 컬럼에 의해 서로 연관되는 여러 가지 표
- ✓ 일정하거나 일정하지 않은 간격의 시계열

데이터의 속성

✓ 데이터는 다른 것과 구별할 수 있는 성질, 속성을 포함합니다.

학교 도서관 이용자 데이터는 어떤 속성으로 구성될까?

이름, 성별, 구분, 학년, 반번호,
도서관 이용 횟수, 대출 권 수



- 각 속성이 갖는 실제 값
- 문자형, 수치형 데이터

데이터의 속성

이름, 성별, 구분, 학년, 반번호,
도서관 이용 횟수, 대출 권 수

속성

속성값

- 각 속성이 갖는 실제 값
- 문자형, 수치형 데이터

문자형 데이터

도서관 이용자 데이터 속성 테이블

숫자형 데이터

이름	성별	구분	학년	학반 번호	이용 횟수	대출 권 수
김○○	남	학생	1	1305	5	5
박○○	남	학생	2	2103	7	2
이○○	여	학생	3	3512	3	4
임○○	여	교사	—	—	5	6

속성

속성값

[출처: 씨마스 인공지능 기초 교과서]

데이터의 속성

archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.data

```
0.00632  18.00   2.310  0  0.5380  6.5750  65.20  4
0.02731   0.00   7.070  0  0.4690  6.4210  78.90  4
0.02729   0.00   7.070  0  0.4690  7.1850  61.10  4
0.03237   0.00   2.180  0  0.4580  6.9980  45.80  6
0.06905   0.00   2.180  0  0.4580  7.1470  54.20  6
0.02985   0.00   2.180  0  0.4580  6.4300  58.70  6
0.08829  12.50   7.870  0  0.5240  6.0120  66.60  5
0.14455  12.50   7.870  0  0.5240  6.1720  96.10  5
0.21124  12.50   7.870  0  0.5240  5.6310 100.00  6
0.17004  12.50   7.870  0  0.5240  6.0040  85.90  6
0.22489  12.50   7.870  0  0.5240  6.3770  94.30  6
0.11747  12.50   7.870  0  0.5240  6.0090  82.90  6
0.09378  12.50   7.870  0  0.5240  5.8890  39.00  5
```

데이터 테이블, 샘플

속성, 필드, 피쳐, 열

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.9	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	48.9	4.9671	2	242.0	17.8	396.9	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.9	5.33	36.2
5	0.02985	0.0	2.18	0	0.458	6.43	58.7	6.0622	3	222.0	18.7	394.12	5.21	28.7
6	0.08829	12.5	2.18	0	0.524	6.012	66.6	5.5605	5	311.0	15.2	395.6	12.43	22.9
7	0.14455	12.5	7.87	0	0.524	6.172	96.1	5.9505	5	311.0	15.2	396.9	19.15	27.1
8	0.21124	12.5	7.87	0	0.524	5.631	100	6.0821	5	311.0	15.2	386.63	29.93	16.5
9	0.17004	12.5	7.87	0	0.524	6.004	85.9	6.5921	5	311.0	15.2	386.71	17.1	18.9
10	0.22489	12.5	7.87	0	0.524	6.377	94.3	6.3467	5	311.0	15.2	392.52	20.45	15.0

인스턴스, 튜플, 행

피쳐 벡터

데이터

Data의 종류

- **수치형 Numerical data :** 연속형과 이산형 데이터
- **문자형 Object data :** 문자로만, 문자와 숫자로 구성된 데이터
- **범주형 Categorical data :** 순서형과 명목형 데이터
- **불리언형 Boolean data :** 논리값인 참(True)와 거짓(False) 중 하나로 표현하는 데이터

수치형 Data

- **연속형 데이터(continuous data)** : 값이 끊어지지 않고 계속 연결되는 종류의 데이터로, 실수와 관련된 값. 평균, 분산 등 통계적 기법 적용 가능
- **이산형 데이터(discrete data)** : 분리해서 표현하는 데이터, 일종의 라벨. 텍스트 형태의 값도 숫자 형태로 바꾸어 사용

연속형 데이터	이산형 데이터
• 값이 끊어지지 않고 연결되어 표시 • 일반적인 실수나 정수 값	• 값이 연속적이지 않고 끊어서 표현되는 값들 • 라벨로 구분되는 값들 • 숫자이지만 그 값 들 간에 스케일이 없다
• 온도, 시험 점수의 평균, 자동차 속도 등	• 성별, 주소, 설문조사 척도 등

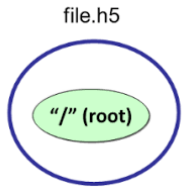
범주형 Data

- **명목형 데이터(continuous data)** : 순서가 없다
- **순서형 데이터(discrete data)** : 순서가 있다

명목형 데이터	순서형 데이터
• 순서를 매길 수 없음 • 비교, 최빈값이 있음	• 순서가 있지만 항목별 차이가 일정하지 않음 • 비교, 최빈값, 순서, 중앙값이 있음
• 긍정과 부정 • 개와 고양이	• 만족도 • 별점

데이터의 저장 형식

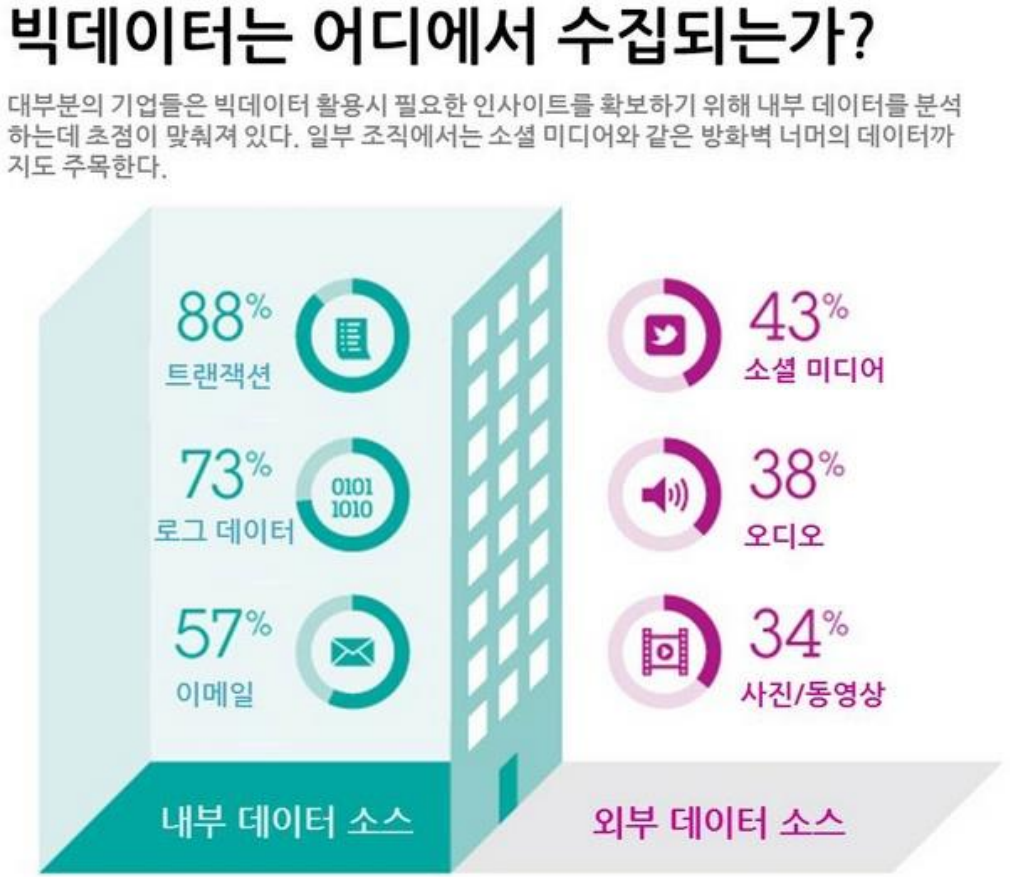
- **데이터의 형식** : 데이터 테이블 형태로 저장할 수 있는 형식
- **csv(comma separate value)** : 컬럼을 콤마로 분리해서 저장하는 데이터 파일 형식
- **xlsx :** 엑셀 파일 형식
- **Json, xml :** 트리형식으로 저장
- **pickle** : 파이썬에 특화된 데이터 저장 형식
- **H5** : 큰 데이터를 저장할 때 사용하는 이진 데이터 형식



제공되는 데이터

- 제공 데이터
- 제품 API
- 외부제공업체로부터 구매

- 직접 수집하는 방법
  - 파일 다운로드
  - 웹 크롤링/웹 스크래핑



https://coolenjoy.net/bbs/votes/5176?sst=wr_datetime&sod=asc&sop=and&page=248

웹 크롤러 Web Crawler

- www(world wide web)을 체계적, 자동화된 방법으로 탐색하고 정보를 자동적으로 수집하는 컴퓨터 프로그램

웹 크롤링(Web crawling)

- 자동적으로 화면에 있는 data를 가져옴
- 실시간 연동, 자동으로 업데이트를 하므로써 데이터의 최신 상태를 유지함.

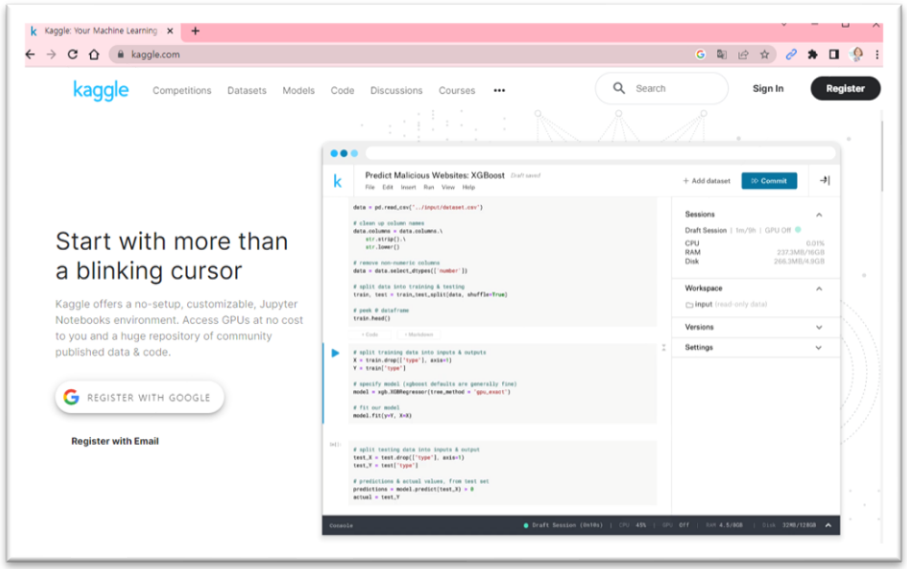
웹 스크래핑(Web scrapping)

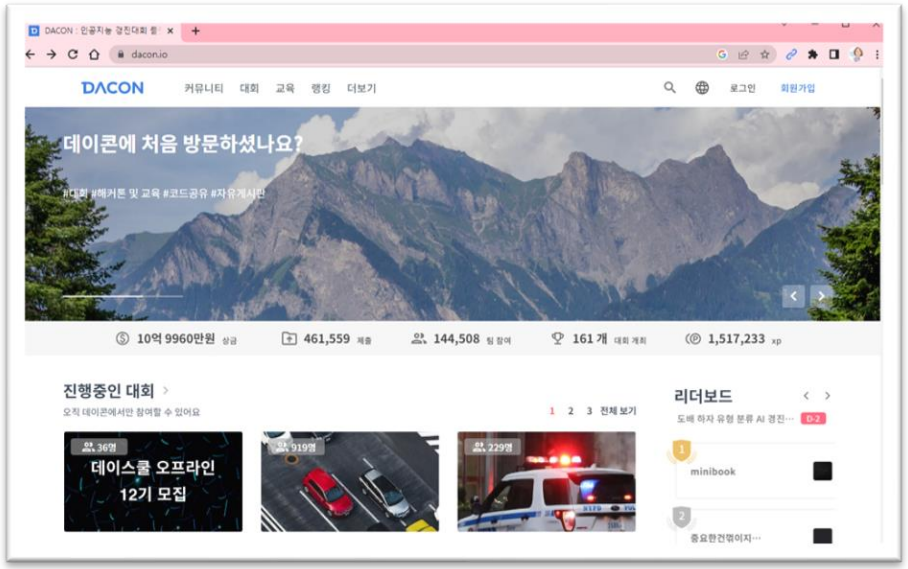
- 자동화 안됨
- Scrapping 하는 시점에서의 데이터만 갖고 오기
- 특정 요구 사항을 처리하는데 사용

데이터의 수집처

- **데이터 확보를 위한 사이트 : 캐글과 데이콘**

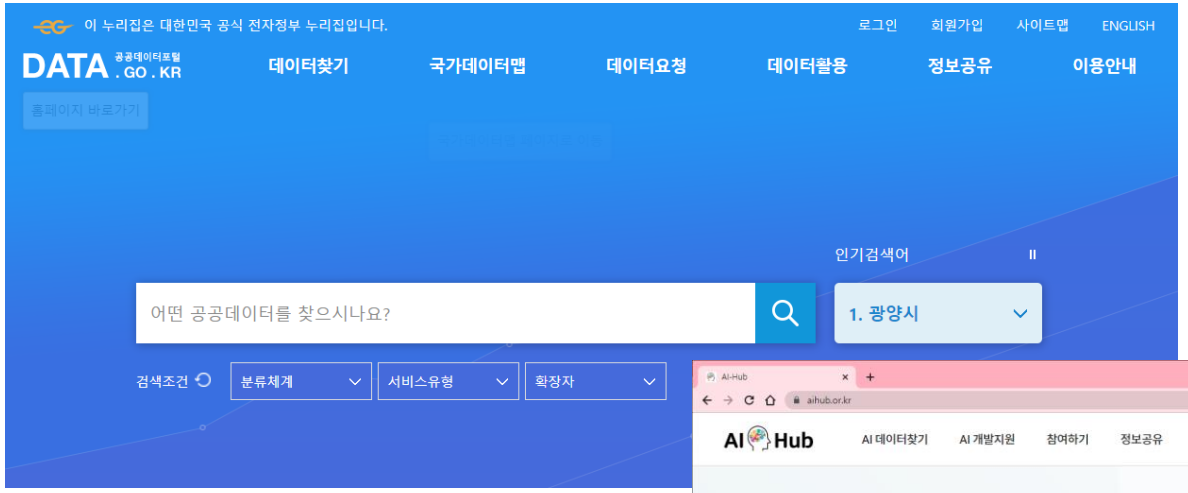
캐글은 2017년에 구글에 인수, 데이터 분석의 표준적인 프레임워크로 사용되고 있고, 데이콘은 국내 스타트업이 운영하는 데이터 대회 사이트이다

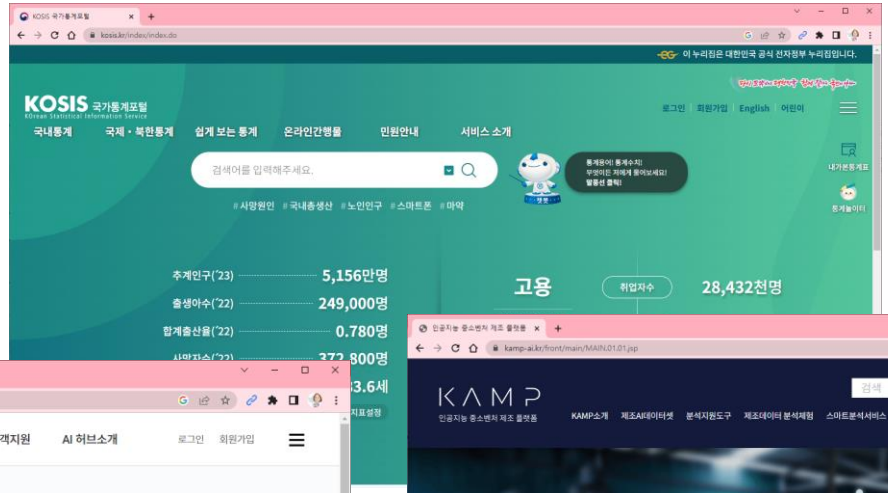


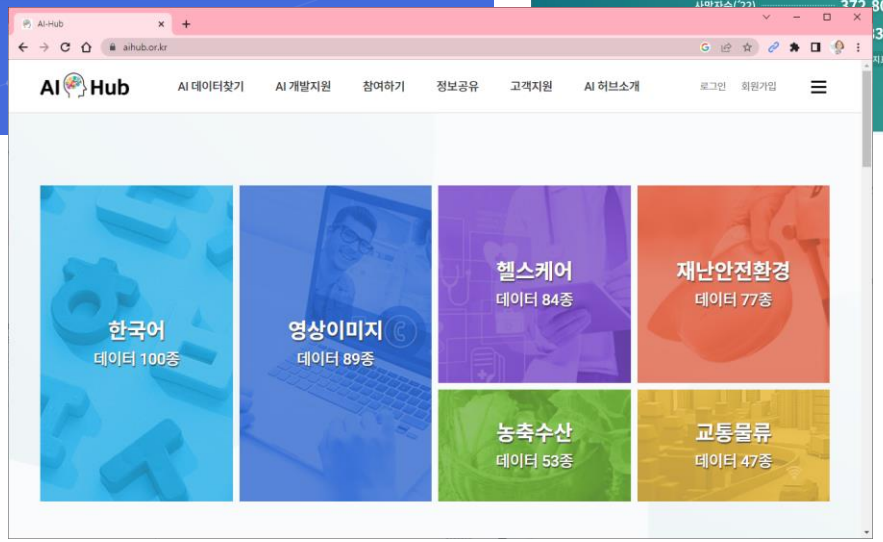


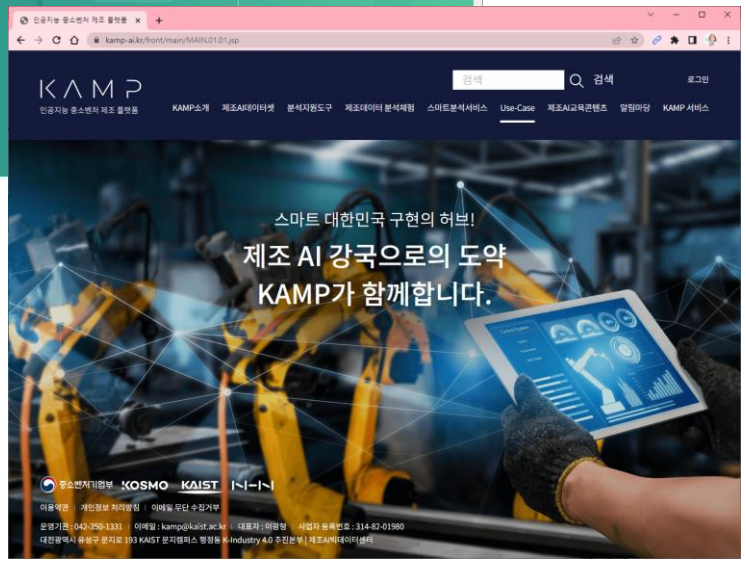
데이터의 수집 방법

- **공공데이터 확보를 위한 사이트 :** 공공데이터 포털, 국가통계포털 등









결측치 처리

- 데이터 수집 과정에서 값이 기록되지 않은 것.
- 넘파이 배열에서는 결측치를 np.nan으로 표현.
- 판다스 데이터프레임에서는 결측치를 NaN으로 표현.

결측치가 있는 데이터프레임

```
import pandas as pd
import numpy as np

df = pd.DataFrame({'A': [1, 2, np.nan, 4, 5],
                   'B': [6, 7, 8, np.nan, 10],
                   'C': [11, 12, 13, np.nan, np.nan]})
df
```

	A	B	C
0	1.0	6.0	11.0
1	2.0	7.0	12.0
2	NaN	8.0	13.0
3	4.0	NaN	NaN
4	5.0	10.0	NaN

결측치 처리

- 판다스의 isna( ) 함수로 데이터프레임에서 결측치가 어디에 있는지 빠르게 확인.
- 데이터가 있으면 False로 표시되고 결측치는 True로 표시됨.
- sum( ) 함수를 이용하여 열별 결측치 개수를 확인.
- dropna() 함수로 결측치 제거

결측치 위치 확인

```
pd.isna(df)
```

	A	B	C
0	False	False	False
1	False	False	False
2	True	False	False
3	False	True	True
4	False	False	True

열에 포함된 결측치 개수 확인

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

```
A 1
B 1
C 2
dtype: int64
```

결측치가 A열, B열, C열에 각각 1개, 1개, 2개 있음.

	A	B	C
0	1.0	6.0	11.0
1	2.0	7.0	12.0
2	NaN	8.0	13.0
3	4.0	NaN	NaN
4	5.0	10.0	NaN

행별로 모든 결측치 제거

```
df_drop_nan = df.dropna( )
df_drop_nan
```

	A	B	C
0	1.0	6.0	11.0
1	2.0	7.0	12.0

B와 C에 있는 결측치를 제거→ 다섯 행 중 2개 행이 남음.

결측치 처리

- 데이터 양이 많지 않을 경우 결측치를 제거하기보다 다른 값(숫자, 문자)으로 대체
- 평균 대체는 데이터 분포에 영향을 적게 주는 방법

숫자로 결측치 대체

```
df_0 = df['C'].fillna(0)
print(df_0)
```

```
0 11.0
1 12.0
2 13.0
3 0.0
4 0.0
Name: C, dtype: float64
```

fillna( ) 함수 인자로 결측치 대체.
C열에 있는 결측치를 모두 0으로 대체.

문자로 결측치 대체

```
df_missing = df['A'].fillna('missing')
df_missing
```

```
0 1.0
1 2.0
2 missing
3 4.0
4 5.0
Name: A, dtype: object
```

A열에 있는 결측치를 문자열 'missing'으로 대체.

평균으로 결측치 대체

```
# df.fillna(df.mean( ), inplace=True)
df_mean = df.fillna(df.mean( ))
print(df, '\n')
print(df_mean)
```

```
     A    B    C
0  1.0  6.0 11.0
1  2.0  7.0 12.0
2  NaN  8.0 13.0
3  4.0  NaN  NaN
4  5.0 10.0  NaN
```

```
     A    B    C
0  1.0 6.00 11.0
1  2.0 7.00 12.0
2  3.0 8.00 13.0
3  4.0 7.75 12.0
4  5.0 10.00 12.0
```

결측치 처리

- **주변 데이터로 결측치 대체**
- 각 열에 서로 다른 값을 할당하기

주변 데이터로 결측치 대체

```
print(df, '\n')
#결측치 바로 위의 값으로 대체하기
df_ffill = df.fillna(method='ffill')
print(df_ffill, '\n')
#결측치 바로 아래의 값으로 대체하기
df_bfill = df.fillna(method='bfill')
print(df_bfill)
```

df

```
    A   B   C
0 1.0 6.0 11.0
1 2.0 7.0 12.0
2 NaN 8.0 13.0
3 4.0 NaN NaN
4 5.0 10.0 NaN
```

df_ffill

```
    A   B   C
0 1.0 6.0 11.0
1 2.0 7.0 12.0
2 2.0 8.0 13.0
3 4.0 8.0 13.0
4 5.0 10.0 13.0
```

df_bfill

```
    A   B   C
0 1.0 6.0 11.0
1 2.0 7.0 12.0
2 4.0 8.0 13.0
3 4.0 10.0 NaN
4 5.0 10.0 NaN
```

df_ffill은 fillna( ) 함수의 method 인자 값을 'ffill'로 입력하여 결측치를 바로 위의 데이터로 대체.

df_bfill은 method 인자 값을 'bfill'로 입력하여 결측치를 바로 아래의 데이터로 대체.

결측치 처리

- 주변 데이터로 결측치 대체
- **각 열에 서로 다른 값을 할당하기**

각 열에 서로 다른 값 할당

```
# 'Name' 열의 결측치를 'Unknown'으로 대체
df['Name'].fillna('Unknown', inplace=True)

# 'Age' 열의 결측치를 나이의 평균값으로 대체
df['Age'].fillna(df['Age'].mean(), inplace=True)

# 'Salary' 열의 결측치를 급여의 중앙값으로 대체
df['Salary'].fillna(df['Salary'].median(), inplace=True)
```

df

	Name	Age	Salary
0	Anna	29.0	NaN
1	Ben	30.0	50000.0
2	Charlie	NaN	55000.0
3	NaN	22.0	60000.0

df_bfill

	Name	Age	Salary
0	Anna	29.0	55000.0
1	Ben	30.0	50000.0
2	Charlie	27.0	55000.0
3	Unknown	22.0	60000.0

- ✓ inplace= True 는 대체한 값을 데이터프레임에 반영하기 여부의 설정
- ✓ mean()은 평균값 계산, median() 중앙값 계산 함수

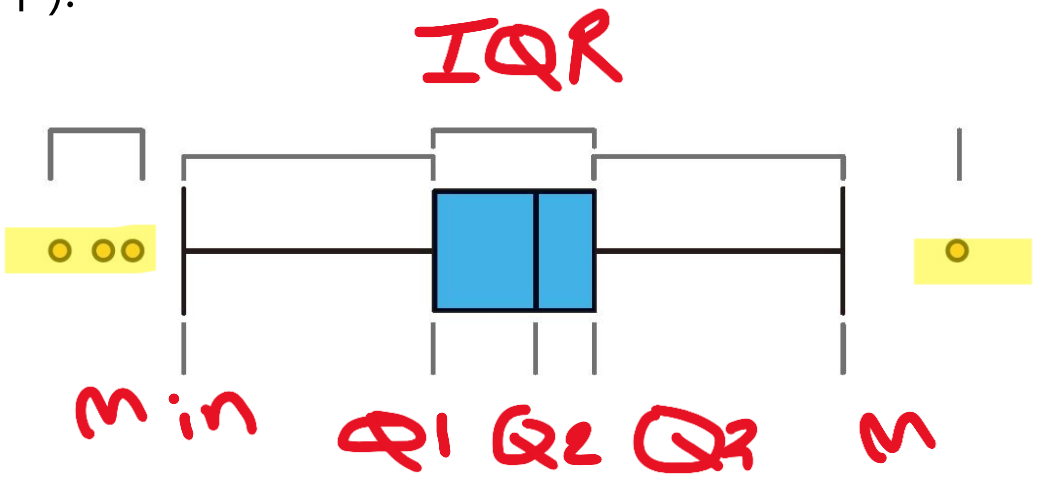
이상치 처리

- **이상치(Outlier)는 데이터셋에서 대부분의 데이터가 모인 범위를 크게 벗어난 값.**

	데이터 1	데이터 2
데이터	1,2,2,3,3,5	1,2,2,3,3,500
평균	2.86	85.17
중앙값	3	2.5
표준편차	1.35	203.23

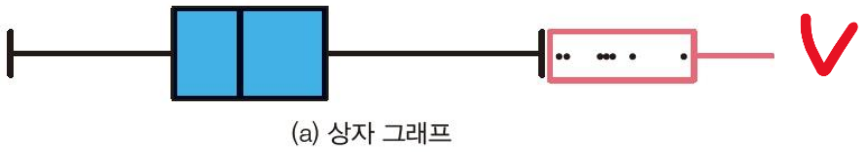
이상치 처리

- ✓ **IQR(Interquartile range)은 제1사분위수에서 제3사분위수까지의 거리.**
- ✓ IQR의 1.5배보다 멀리 떨어진 데이터를 이상치로 간주하여 제거.
  - Q1은 데이터의 첫번째 사분위수(25번째 백분위수).
  - Q2는 데이터의 두번째 사분위수(중앙값, 50번째 백분위수).
  - Q3은 데이터의 세번째 사분위수(75번째 백분위수).
- ✓ **최솟값보다 작거나 최댓값보다 큰 값이 이상치**
  - 최소값 = Q1 - 1.5 × IQR
  - 최대값 = Q3 + 1.5 × IQR로 정함.

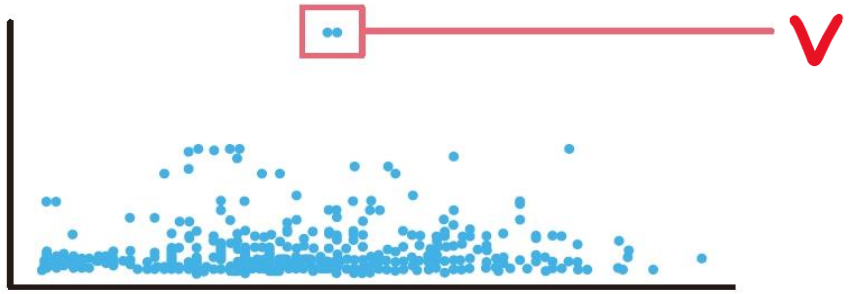


이상치 처리

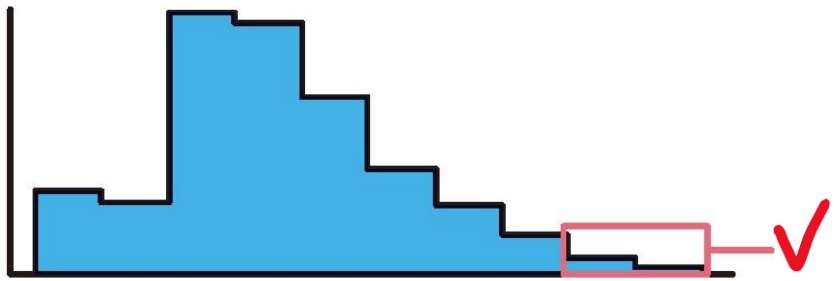
- 이상치를 확인할 때 데이터 시각화를 이용하는 방법



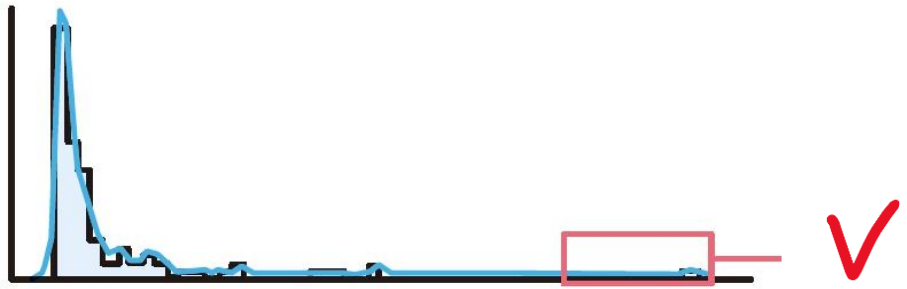
(a) 상자 그래프



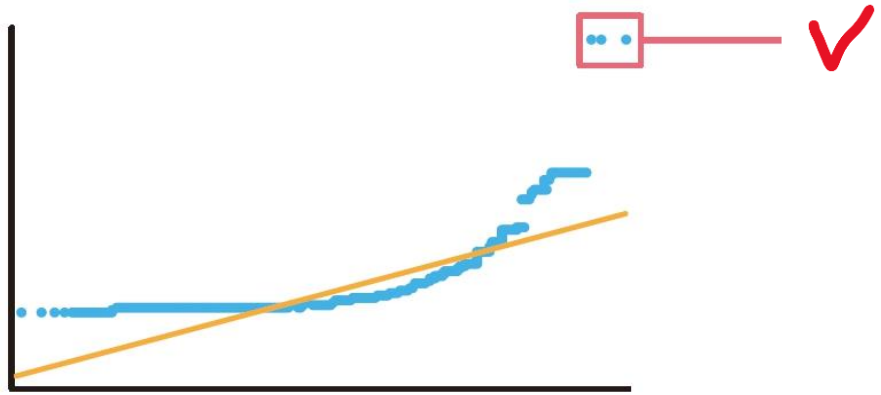
(b) 산점도



(c) 히스토그램



(d) 분포도



(e) QQ 플롯

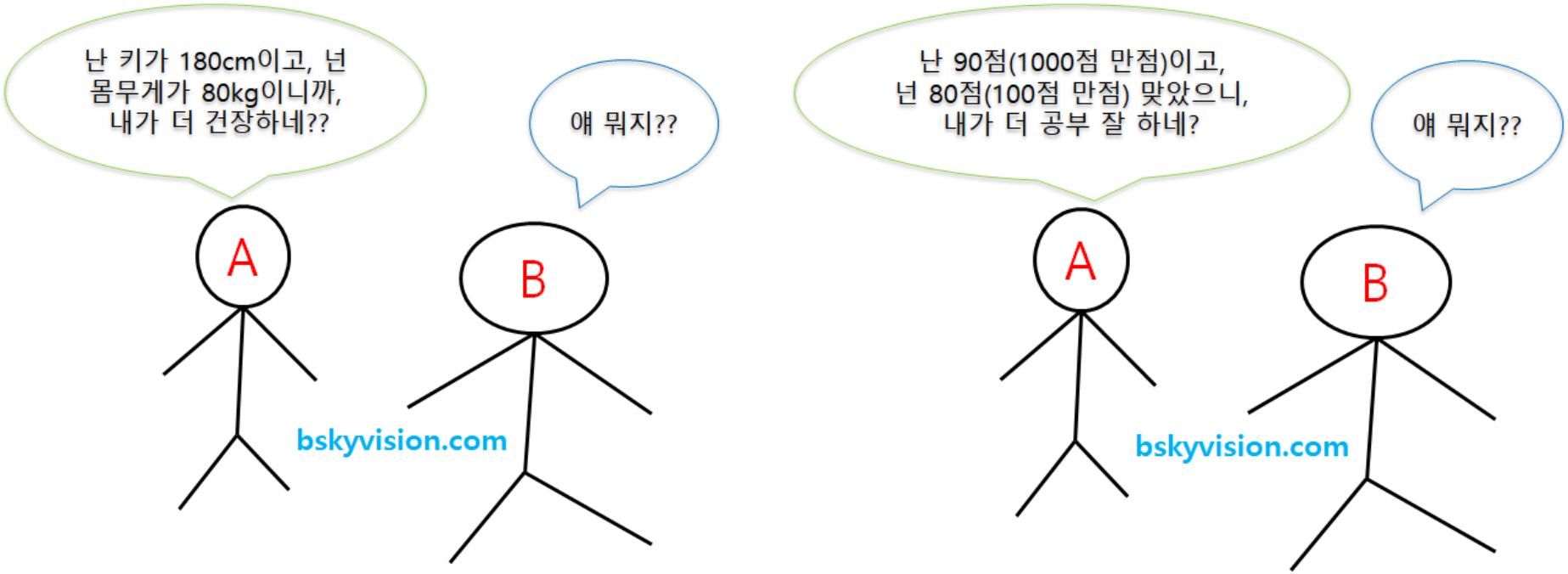
이상치 처리 방법

1. 이상치를 처리하지 않고 그대로 사용
2. 이상치를 포함하는 행을 삭제
3. fence를 벗어나는 값들을 fence의 값으로 치환

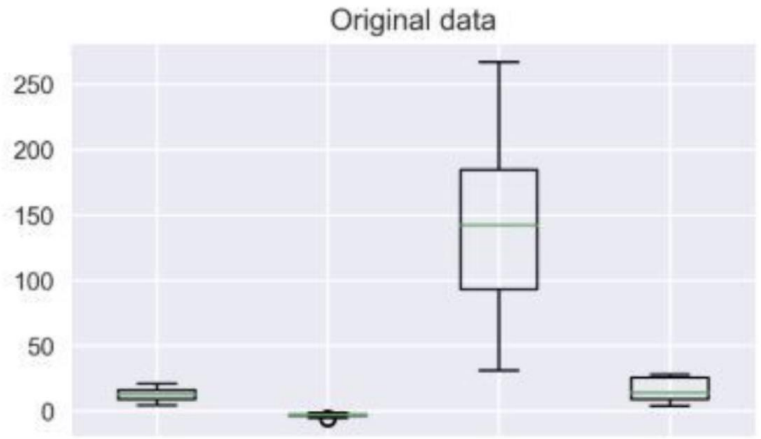
예) 나이 데이터의 이상치 중 1980이 있다 -> 2023 – 1980 값인 43으로 치환
나이 데이터의 이상치 중 223이 있다 -> 제거 혹은 upper fence 값으로 치환
나이 데이터의 이상치 중 101이 있다 -> 현실에서도 충분히 있을 수 있으므로 그대로 사용

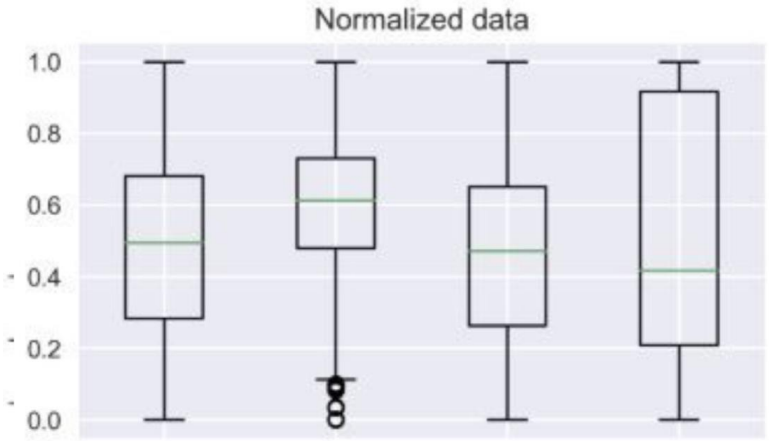
표준화와 정규화

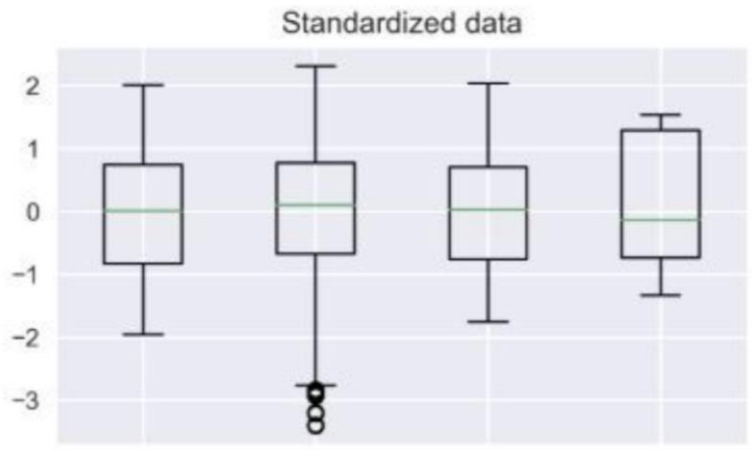
- **스케일링 기법 :** 각 특성의 값들을 일정한 범위 내로 조정하여 데이터 분포를 균일화
- 각 특성의 값들을 일정한 범위 내로 조정하여 데이터 분포를 균일화



표준화와 정규화


Original data
250
200
150
100
50
0


Normalized data
1.0
0.8
0.6
0.4
0.2
0.0


Standardized data
2
1
0
−1
−2
−3

표준화와 정규화

1) 표준화(Standardization)

데이터의 평균을 0으로, 표준 편차를 1로 변환해서 비교나 통계분석에 적합하게 만든다

normalization에 비해 이상치에 대한 영향을 덜 받음

$$Z\text{ 점수 } = \frac{(X - \text{평균})}{\text{표준편차}}$$

2) 정규화(Normalization)

데이터 중 max 값을 1, min 값을 0으로 하여 전체 데이터를 0~1로 변환

다양한 범위와 단위의 데이터를 서로 비교할 수 있음.

$$X' = \frac{(X - Xmin)}{Xmax - Xmin}$$

표준화와 정규화

1) 파이썬의 사이킷런의 StandardScaler의 fit_transform() 함수로 표준화(Standardization)

2) 역시 사이킷런의 MinMaxScaler의 fit_transform() 함수로 정규화(Normalization)

표준화와 정규화 값 비교

```
# 표준화를 위한 scaler 객체 생성
scaler = StandardScaler()
# 'Salary' 열 표준화
df['Salary_Standardized'] = scaler.fit_transform(df[['Salary']])
```

```
# 정규화를 위한 scaler 객체 생성
min_max_scaler = MinMaxScaler()
# 'Salary' 열 정규화
df['Salary_Normalized'] = min_max_scaler.fit_transform(df[['Salary']])
```

	Age	Salary	Salary_Standardized	Salary_Normalized
0	25	40000	-1.207020	0.000000
1	30	50000	-0.742781	0.166667
2	35	60000	-0.278543	0.333333
3	40	80000	0.649934	0.666667
4	45	100000	1.578410	1.000000

Chapter. 02

Data

- 데이터의 정의
- 데이터의 구성과 용어
- 데이터의 활용
- 데이터의 수집
- 데이터의 전처리