AI와 데이터 기초 II

데이터의 연산 및 데이터 정제

오늘 수업은

- numpy 이해와 자료구조
  - numpy의 정의
  - numpy를 사용하는 이유
  - numpy의 자료구조
  - 수열 생성과 그래프 표현
  - random 함수들
  - 기본 통계 함수들

- 데이터 정제
  - 데이터 정제의 정의
  - 결측치 확인 및 편집
  - 이상치 확인 및 편집
  - 중복값 확인 및 삭제

numpy 이해와 자료구조

numpy 란?

- 넘파이(numpy)
  - 수치 연산, 과학 연산을 위한 파이썬 외부 라이브러리
  - 복잡한 연산을 수행하는 데이터분석, 시각화, 머신러닝 등의 작업에 필수
  - 벡터, 행렬 등의 자료구조 및 연산 지원

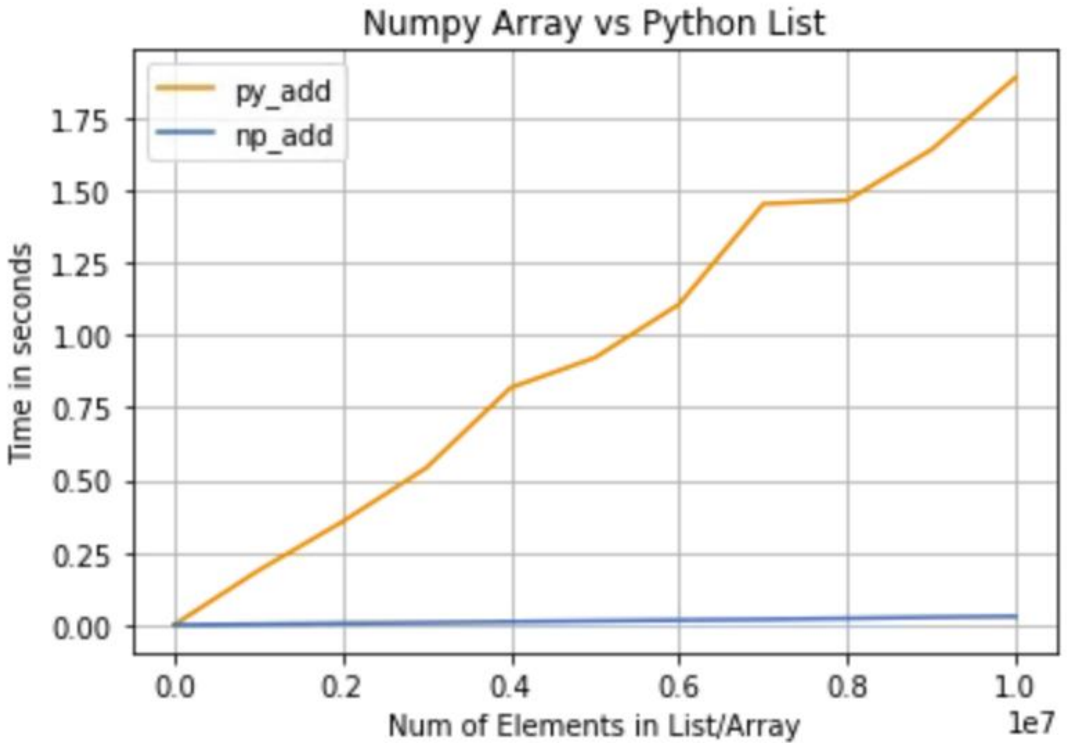
- 제공하는 기능들
  - 통계 함수들 : 최대, 최소, 평균, 중간값, 분산, 표준편차, n분위수
  - 수학 함수들 : 삼각함수, 로그함수 등
  - 벡터 및 행렬 연산 : 행렬의 곱, 역행렬, 전치행렬(array 라는 이름으로 제공)
  - 공학수학, 선형대수학 등

numpy 의 장점

- 데이터를 생성할 수 있다.
  - 특정 패턴의 수열, 랜덤 수, 특정 분포에 근거한 데이터, 수학 함수 데이터(삼각 함수 등)

- 많은 데이터를 쉽고 빠르게 처리할 수 있다.
  - 리스트로 하는 것보다 수행 속도가 훨씬 빠름
  - 코드도 훨씬 짧음

- 복잡한 연산을 수행할 수 있다.
  - 통계, 선형 대수(행렬 연산 등), 푸리에 연산 등



출처:Performance of Numpy Array vs Python List, Cory Gough

numpy 설치하기

❖설치

```
pip install numpy
```

❖주피터 노트북과 코랩은 이미 설치되어 있음

❖라이브러리 선언

```
import numpy as np
```

numpy가 왜 필요할까?

❖문제1

- 1부터 20까지 3씩 증가되는 수열(리스트)을 생성하시오.
- 리스트의 모든 항목 값에 10을 더하시오.
- 리스트의 모든 항목 값에 2를 곱하시오.

❖문제2

- 0.2부터 2.5까지 0.3씩 증가되는 수열을 생성하시오.
- 수열의 모든 항목 값에 10을 더하시오.
- 수열의 모든 항목 값에 2를 곱하시오.

numpy 왜 필요할까?(문제1)

- 1부터 20까지 3씩 증가되는 리스트를 생성하기
- 수열의 모든 값에 10을 더하기
- 수열의 모든 값에 2를 곱하기

```
1 x = list(range(1, 20, 3))
2 x
```

```
[1, 4, 7, 10, 13, 16, 19]
```

```
1 for i in range(len(x)):
2   x[i] = x[i] + 10
3 x
```

```
[11, 14, 17, 20, 23, 26, 29]
```

```
1 for i in range(len(x)):
2   x[i] = x[i] * 2
3 x
```

```
[22, 28, 34, 40, 46, 52, 58]
```

numpy 왜 필요할까? (문제2)

❖문제2

- 0.5부터 2.5까지 0.3씩 증가되는 수열(리스트)를 생성하기
- 수열의 모든 값에 10을 더하기
- 수열의 모든 값에 2를 곱하기

```
1 x = list(range(0.5, 2.5, 0.3))
2 x
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-9-7501f1d54c9c> in <cell line: 1>()
----> 1 x = list(range(0.5, 2.5, 0.3))
      2 x

TypeError: 'float' object cannot be interpreted as an integer
```

❖문제1로 문제2를 해결할때 문제점

- range() 함수로는 실수에 대한 수열을 생성 불가능
- 반복문을 사용해야 하는 번거로움
- 복잡한 소스코드

numpy 왜 필요할까? (문제2)

- 0.5부터 2.0까지 0.3씩 증가되는 Array 생성
- 수열의 모든 값에 10을 더하기
- 수열의 모든 값에 2를 곱하기

```
[1]  1 import numpy as np
     2
     3 x = np.arange(0.5, 2.0, 0.3)
     4 x
```

```
array([0.5, 0.8, 1.1, 1.4, 1.7])
```

```
[2]  1 x = x + 10
     2 x
```

```
array([10.5, 10.8, 11.1, 11.4, 11.7])
```

```
[3]  1 x = x * 2
     2 x
```

```
array([21. , 21.6, 22.2, 22.8, 23.4])
```

문제 : numpy 왜 필요할까? (점수 평균)

- 특강수업에 참여하고 있는 5명의 학생의 중간고사와 기말고사의 성적은 다음과 같다.
  - 중간고사 성적 : [90, 80, 70, 60, 50]
  - 기말고사 성적 : [80, 70, 60, 50, 40]

- 중간고사와 기말고사 성적을 이용하여 평균과 가산점의 결과를 계산하시오.
  - 중간고사와 기말고사의 각 점수에 따른 평균을 구하시오. [85, 75, 65, 55, 45]
  - 전체적으로 +7의 가산점을 적용하시오. [92, 82, 72, 62, 52]

numpy 사용하지 않는 경우와 사용하는 경우 비교

```
 1 중간 = [90, 80, 70, 60, 50]
 2 기말 = [80, 70, 60, 50, 40]
 3 최종 = []
 4
 5 n = len(중간)
 6 for i in range(n) :
 7     v = (중간[i] + 기말[i]) / 2
 8     최종.append( v )
 9
10 print('평균 = ', 최종)
11
12 for i in range(n) :
13     최종[i] = 최종[i] + 7
14
15 print( '가산점 적용후 = ', 최종 )
```

numpy 사용하지 않는 경우

```
 1 import numpy as np
 2
 3 중간 = [90, 80, 70, 60, 50]
 4 기말 = [80, 70, 60, 50, 40]
 5
 6 mid   = np.array(중간)
 7 final = np.array(기말)
 8 최종 = (mid + final)/2
 9 print('평균 = ', 최종)
10
11 최종 = 최종 +7
12 print( '가산점 적용후 = ', 최종 )
```

numpy 사용하는 경우

numpy의 자료구조(array)

❖1차원 행렬

- 리스트를 만든 후 Array로 변환하여 생성

```
1 listData = [1, 2, 3, 4, 5]
2 print(listData)
3 print(type(listData))
```

```
[1, 2, 3, 4, 5]
<class 'list'>
```

```
1 import numpy as np
2
3 arrData = np.array(listData)
4 print(arrData)
5 print(type(arrData))
```

```
[1 2 3 4 5]
<class 'numpy.ndarray'>
```

numpy의 자료구조(array)

❖2차원 행렬

- 2차원 리스트 생성 후 array로 변환

```
1 arrData2 = np.array([[1, 2, 3],
2                      [4, 5, 6],
3                      [10, 11, 12]])
4 arrData2
```

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [10, 11, 12]])
```

numpy를 이용한 데이터 생성 및 통계

수열 생성

- np.arange(a, b, c)
  - a : 시작값
  - b : 종료값
  - c : 증감값
  - 실수값 지원
  - list가 아니라 array 형태로 생성

❖ np.linspace(a, b, c)
  - a : 시작값
  - b : 종료값
  - c : 데이터 생성 개수(등급의 개수)
    - 100으로 하면 100개의 구간으로 나누는 수열 생성

range()와 np.arange()의 차이점

❖공통점

- (시작값, 종료값, 증감값) 동일

❖차이점

- range() : 시작값, 종료값, 증감값에 **실수 지원 안함**
- np.arange() : 시작값, 종료값, 증감값에 **실수 지원**

```
1 list(range(1, 10, 1))
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
1 list(range(1.5, 10.5, 2.1))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-23-7f0a3141493f> in <cell line: 1>()
----> 1 list(range(1.5, 10.5, 2.1))

TypeError: 'float' object cannot be interpreted as an integer
```

```
1 list(np.arange(1, 10, 1))
```

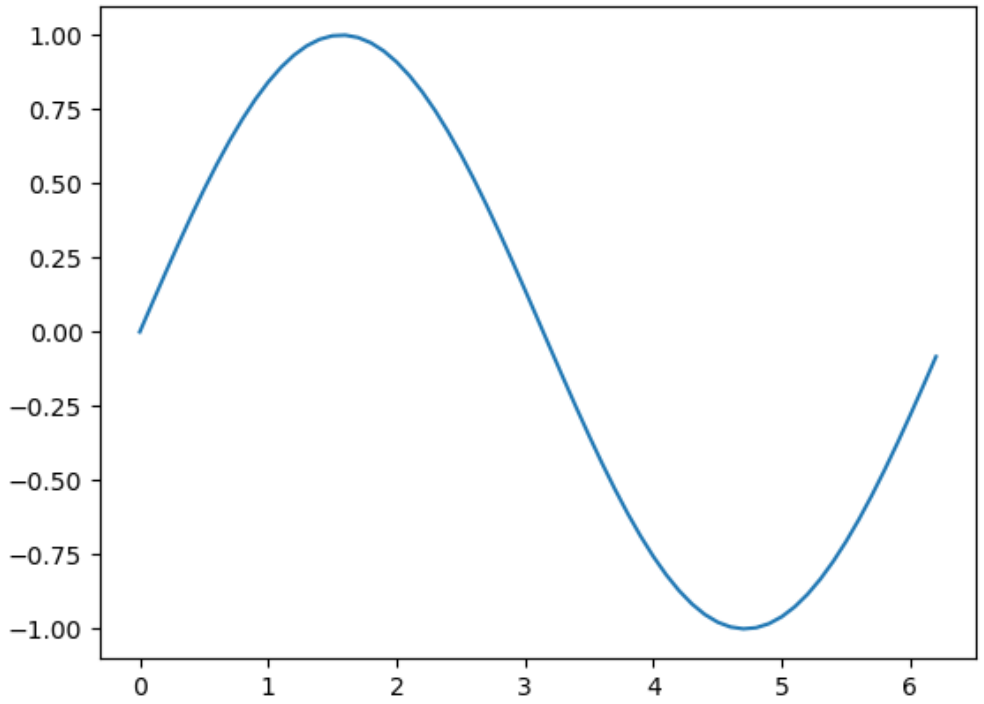
```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
1 list(np.arange(1.5, 10.5, 2.1))
```

```
[1.5, 3.6, 5.7, 7.800000000000001, 9.9]
```

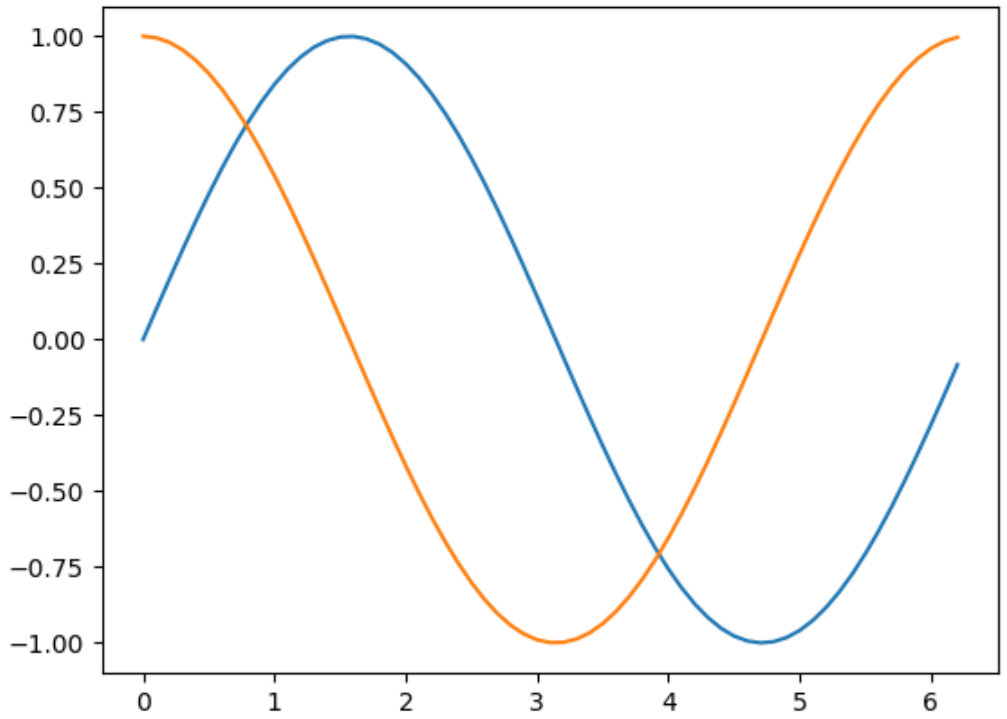
np.arange()를 이용한 Sin 함수 그리기

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

xlist = np.arange(0, 3.14*2, 0.1)
ylist = np.sin( xlist )

plt.plot(xlist, ylist)
plt.show()
```

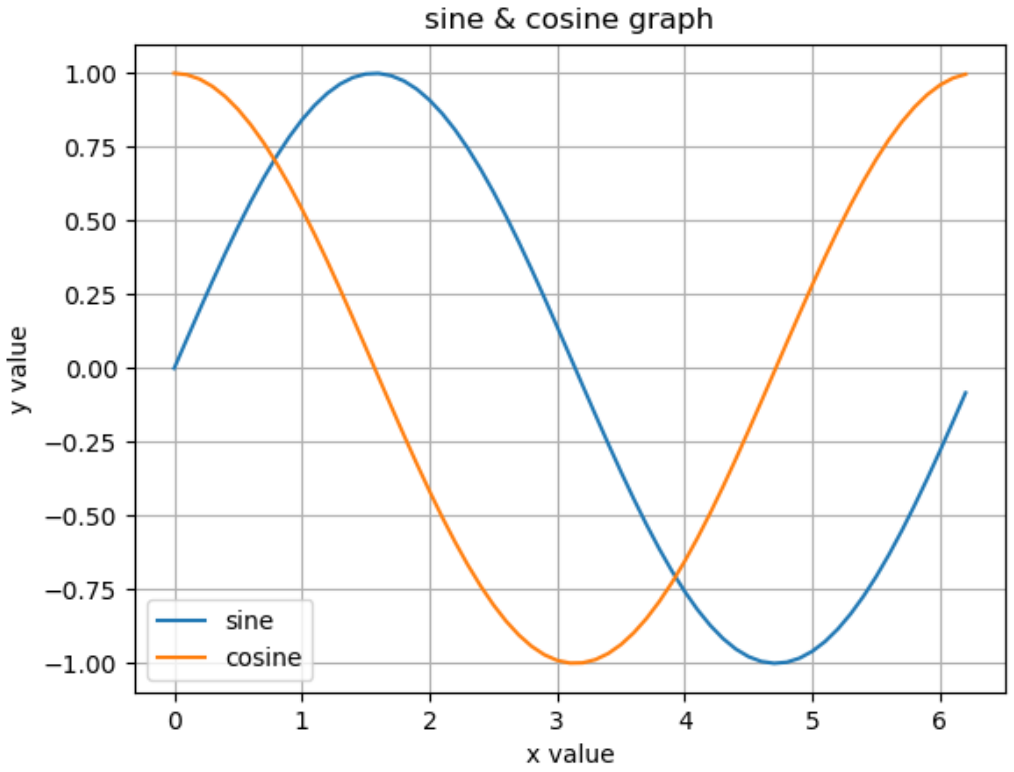


Sin, Cos 함수 그리기

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

x = np.arange(0, 3.14*2, 0.1)
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1)
plt.plot(x, y2)

plt.show()
```



Sin, Cos 자세히 그리기




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

x = np.arange(0, 3.14*2, 0.1)
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1, label='sine')
plt.plot(x, y2, label='cosine')

plt.title('sine & cosine graph')
plt.xlabel('x value')
plt.ylabel('y value')
plt.grid(True)
plt.legend()

plt.show()
```

np.linspace()

❖np.linspace(a, b, c)

- a : 시작값 #정수, 실수
- b : 종료값 #종료값을 포함한 정수, 실수
- c : 데이터 생성 개수(등급개수) #정수
  - 100으로 하면 100개의 구간으로 나누는 수열 생성

```
1 import numpy as np
2
3 a = np.linspace(1, 10, 10)
4 a
```

```
array([ 1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
```

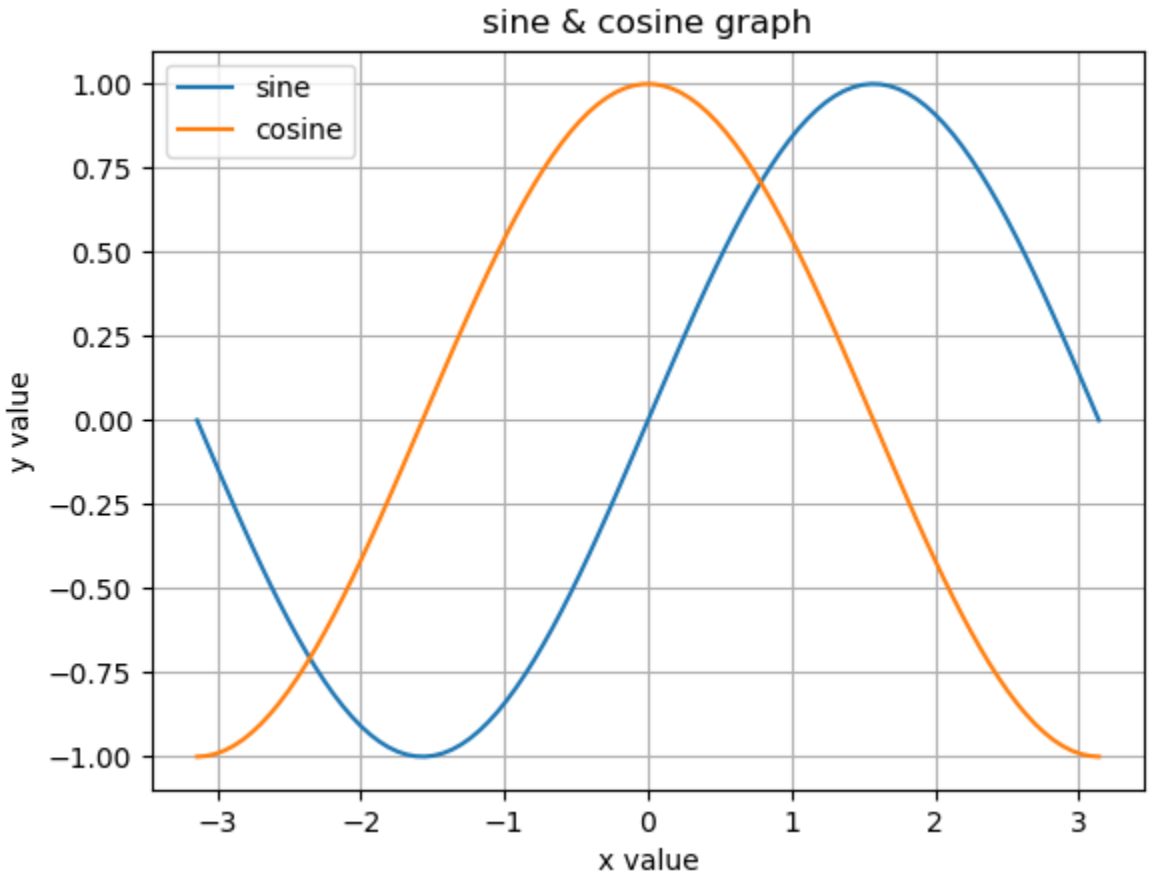
np.linspace()

❖0부터 25까지의 범위를 30등분하기

```
1 import numpy as np
2
3 a = np.linspace(0, 25, 30)
4 a
```

```
array([ 0.        ,  0.86206897,  1.72413793,  2.5862069 ,  3.44827586,
        4.31034483,  5.17241379,  6.03448276,  6.89655172,  7.75862069,
        8.62068966,  9.48275862, 10.34482759, 11.20689655, 12.06896552,
       12.93103448, 13.79310345, 14.65517241, 15.51724138, 16.37931034,
       17.24137931, 18.10344828, 18.96551724, 19.82758621, 20.68965517,
       21.55172414, 22.4137931 , 23.27586207, 24.13793103, 25.        ])
```

np.linspace() 활용




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

x = np.linspace(-np.pi, np.pi, 100)
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1, label='sine')
plt.plot(x, y2, label='cosine')

plt.title('sine & cosine graph')
plt.xlabel('x value')
plt.ylabel('y value')
plt.legend()
plt.grid(True)
plt.show()
```

random 함수들 : 난수 생성

- np.random.rand(a, b)
  - 0.0~1.0사이의 실수형 난수
  - (a , b) : 난수로 이루어진 2차원 array 생성
    # b 생략시 난수로 이루어진 1차원 array 생성
- np.random.randint(a, b, size=(x,y))
  - a부터 b까지의 정수형 난수 생성
  - size(x,y) : 2차원 array 로 생성
- np.random.normal(a, b, c)
  - 정규분포를 갖는 난수 생성
  - a : 모평균
  - b : 표준편차
  - c : 생성할 개수

random 함수들 : 난수 생성

- np.random.rand(a, b)
  - 0.0~1.0사이의 실수형 난수 생성
  - (a , b) : 난수로 이루어진 2차원 array 생성

```
# b 생략시 난수로 이루어진 1차원 array 생성. 즉 a 개수만큼 생성
```

```
1  np.random.rand(5)
```

```
array([0.04740552, 0.37827322, 0.58528326, 0.33715437, 0.87254352])
```

```
1  np.random.rand(2,3)
```

```
array([[0.78218873, 0.62574872, 0.2548894 ],
       [0.79800068, 0.97700598, 0.93269461]])
```

random 함수들 : 난수 생성

- 10~20사이의 난수를 생성하고자 할때

```
1 x = 10
2 y = 20
3 (y-x)*np.random.rand(5) + x
```

```
array([18.82239174, 12.36185349, 14.6245015 , 10.3006198 , 13.75933655])
```

random 함수들 : 난수 생성

- np.random.randint(a, b, size=(x,y))
  - a부터 b-1까지의 정수형 난수 생성
  - size(x,y) : 난수 생성 개수

```
# 2차원 array 로 생성
```

```
1 np.random.randint(2, 10, size=10)
```

```
array([9, 6, 7, 4, 4, 5, 5, 4, 5, 7])
```

```
1 np.random.randint(1, 11, size=(4,7))
```

```
array([[ 2,  1,  4,  6,  4,  7,  3],
       [ 5,  4,  4,  2,  9, 10,  8],
       [ 8,  5, 10,  4,  2,  4,  7],
       [ 4,  2,  7,  1,  1,  3,  5]])
```

random 함수들 : 난수 생성

- np.random.normal(a, b, c)
  - 정규분포를 갖는 임의의 난수 생성
  - a : 평균
  - b : 표준편차
  - c : 생성할 개수

```
1  np.random.normal(55, 5, 10)
```

```
array([46.51276787, 53.99194182, 53.777356  , 59.0822788 , 55.63037258,
       59.82335208, 61.96516865, 51.35143241, 56.63152502, 53.43816751])
```

기본 통계 함수들

- 5명의 학생들에 대한 성적을 입력받아 데이터를 생성하시오.
- 생성한 성적에 대하여 아래의 통계, 수학 계산을 수행하시오.
  - 합 (Sum)
  - 평균 (Average)
  - 분산 (Variation)
  - 표준편차 (Standard Deviation)

```
1번 성적입력 -> 1
2번 성적입력 -> 2
3번 성적입력 -> 3
4번 성적입력 -> 4
5번 성적입력 -> 5

입력된 전체점수
[1, 2, 3, 4, 5]

합계 :          15
평균 :        3.00
분산 :        2.00
편차 :        1.41
```

numpy 활용하지 않는 경우 : 1단계

```
scores = []
for i in range(5) :
    val = int(input('%d번 성적입력 -> ' % (i+1) ))
    scores.append( val )

print('\n입력된 전체점수')
print( scores, '\n')
```

```
# 합계, 평균 구하기
sumR = sum( scores )
aveR = sumR / len(scores)

print('합계 : %10d' % sumR)
print('평균 : %10.2f' % aveR)
```

numpy 활용하지 않는 경우 : 2단계

```
scores = []
for i in range(5) :
    val = int(input('%d번 성적입력 -> ' % (i+1) ))
    scores.append( val )

print('\n입력된 전체점수')
print( scores, '\n')
```

```
# 합계, 평균 구하기
sumR = sum( scores )
aveR = sumR / len(scores)

# 분산, 표준편차
r = 0
for i in range(5) :
    r += (scores[i] - aveR)**2
varR = r / len(scores)
stdR = varR**(1/2)

print('합계 : %10d' % sumR)
print('평균 : %10.2f' % aveR)
print('분산 : %10.2f' % varR)
print('편차 : %10.2f' % stdR)
```

numpy 활용하는 경우

```
import numpy as np

scores = []
for i in range(5) :
    val = int(input('%d번 성적입력 -> ' % (i+1) ))
    scores.append( val )

print('\n입력된 전체점수')
print( scores )
```

```
sumR = np.sum(scores)
aveR = np.mean(scores)
varR = np.var(scores)
stdR = np.std(scores)

print('합계 : %10d' % sumR)
print('평균 : %10.2f' % aveR)
print('분산 : %10.2f' % varR)
print('편차 : %10.2f' % stdR)
```

데이터 정제

- 결측치, 이상치, 중복값 관리

데이터 정제란?

- 오류 데이터
  - 데이터 수집 과정에서 누락되거나 범위에서 벗어나는 데이터
  - 데이터 분석 결과가 왜곡되어 신뢰할 수 없음

- 데이터 정제 : 잘못된 데이터를 찾아서 오류를 수정하는 것

- 데이터 정제가 필요한 데이터 종류
  - 결측치 : 특정 행 및 열에 누락된 데이터
  - 이상치 : 정상 범위에서 벗어난 데이터
  - 중복값 : 데이터 내에 동일 데이터가 존재

결측치

❖누락된 데이터, NaN(Not a Number), “?” , “-” 등으로 표시됨

❖결측치가 발생하는 경우
- 데이터가 수집되지 않은 경우
- 측정 장치의 고장 및 사고로 확보할 수 없을 때

❖결측치를 찾는 이유
- 함수 적용이 안되는 경우 발생
- 분석 결과가 왜곡됨
- 데이터 분석 결과를 신뢰할 수 없음

```
import pandas  as pd

score = pd.read_excel('/content/성적.xlsx')
score
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
2	NaN	NaN	NaN	NaN	NaN
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
7	S1393	남	2.0	25.0	NaN
8	S1399	남	NaN	2.0	17.0

결측치 찾기

❖변수명.info()

- DataFrame의 행/열 정보를 통해서 확인

```
score.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32 entries, 0 to 31
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      31 non-null     object
 1   성별      31 non-null     object
 2   출석      25 non-null     float64
 3   프로젝트     25 non-null     float64
 4   실험      27 non-null     float64
dtypes: float64(3), object(2)
memory usage: 1.4+ KB
```

ID: 1개
성별 : 1개
출석 : 7개
프로젝트 : 7개
실험 : 5개

결측치 찾기

❖변수명['열이름'].value_counts(dropna=False)

- 데이터의 빈도수를 활용하여 확인
- dropna=False : NaN인 데이터를 포함하여 데이터 빈도수 알려줌

```
score['출석'].value_counts(dropna=False)
```

```
2.0     9
NaN     7
19.0    5
18.0    3
16.0    3
17.0    2
12.0    1
9.0     1
15.0    1
Name: 출석, dtype: int64
```

```
score['프로젝트'].value_counts(dropna=False)
```

```
NaN     7
12.0    4
2.0     4
16.0    4
25.0    4
11.0    2
17.0    2
22.0    1
18.0    1
23.0    1
24.0    1
5.0     1
Name: 프로젝트, dtype: int64
```

결측치 찾기(df.isna())

❖변수명.isna()

- 테이블 전체에서 결측치 찾기
- 결측치 유무를 True/False 값으로 반환

```
score.isna()
```

	ID	성별	출석	프로젝트	실험
0	False	False	False	False	False
1	False	False	False	False	False
2	True	True	True	True	True
3	False	False	False	False	False
4	False	False	True	False	False
5	False	False	False	False	False
6	False	False	False	False	False
7	False	False	False	False	True

결측치 찾기(df.isna())

❖변수명['열이름'].isna()

- 특정열에서 결측치 찾기

```
score['프로젝트'].isna()
```

```
0    False
1    False
2     True
3    False
4    False
5    False
6    False
7    False
8    False
9    False
```

```
score['실험'].isna()
```

```
0    False
1    False
2     True
3    False
4    False
5    False
6    False
7     True
8    False
9    False
```

```
score[['출석','실험']].isna()
```

	출석	실험
0	False	False
1	False	False
2	True	True
3	False	False
4	True	False
5	False	False
6	False	False
7	False	True
8	True	False
9	False	False

결측치 찾기(df.isna())

❖ pd.isna(변수명).sum()

- 각 열의 결측치 개수 확인

```
score.isna().sum()
```

```
ID      1
성별      1
출석      7
프로젝트     7
실험      5
dtype: int64
```

결측값 제거하기(df.dropna())

```
import pandas  as pd

score2 = pd.read_excel('/content/성적2.xlsx')
score2
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
2	NaN	NaN	NaN	NaN	NaN
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
7	S1393	남	NaN	25.0	NaN
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0

```
score2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      14 non-null     object
 1   성별      14 non-null     object
 2   출석      11 non-null     float64
 3   프로젝트    12 non-null     float64
 4   실험      13 non-null     float64
dtypes: float64(3), object(2)
memory usage: 728.0+ bytes
```

결측값 제거하기(df.dropna())

❖ df.dropna(subset=['열이름1' , '열이름2'])

- subset=['열이름'] : 특정열에 NaN이 존재하는 행만 선택하여 삭제
- subset=['열이름1' , '열이름2']) :열이름1 또는 열이름2에 NaN이 있는 모든 행 삭제
  - 단점 : 분석에 필요한 데이터도 삭제될 수 있음

```
score2.dropna(subset=['출석'])
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
3	S1254	남	2.0	16.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0

```
score2.dropna(subset=['프로젝트','실험'])
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0

결측값 제거하기(df.dropna())

- df.dropna(axis=0, how= ‘any’ , thresh=개수, inplace=True)
  - 행과 열에 존재하는 결측치를 선택하여 삭제
  - axis=0 : 행과 열을 선택하여 삭제
    - axis=0 : 행 삭제
    - axis=1 : 열 삭제
  - how= ‘any’ : 결측치의 포함 정도에 따라 삭제
    - how= ‘any’ : 하나라도 포함하면 행/열 삭제
    - how= ‘all’ : 모두 포함하면 행/열 삭제
  - thresh=개수 : 유효한 데이터가 존재하는 ‘개수’ 이상만 남기고 삭제
  - inplace=True : 원본 데이터에 반영

결측값 제거하기(df.dropna())

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
2	NaN	NaN	NaN	NaN	NaN
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
7	S1393	남	NaN	25.0	NaN
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0

```
score2.dropna(axis=0)
```

```
score2.dropna(axis=0, how='any')
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
3	S1254	남	2.0	16.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0

결측값 제거하기(df.dropna())

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
2	NaN	NaN	NaN	NaN	NaN
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
7	S1393	남	NaN	25.0	NaN
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0

```
score2.dropna(axis=0, how='any')
```

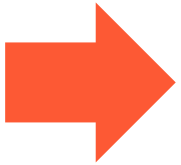
	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
3	S1254	남	2.0	16.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0

```
score2.dropna(axis=0, how='all')
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
7	S1393	남	NaN	25.0	NaN
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0

결측값 제거하기(df.dropna())

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
2	NaN	NaN	NaN	NaN	NaN
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
7	S1393	남	NaN	25.0	NaN
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0



```
score2.dropna(axis=0, thresh=4)
```

	ID	성별	출석	프로젝트	실험
0	S1233	남	2.0	12.0	25.0
1	S1244	남	17.0	2.0	22.0
3	S1254	남	2.0	16.0	25.0
4	S1256	3	NaN	11.0	25.0
5	S1384	남	12.0	22.0	12.0
6	S1391	여	18.0	25.0	8.0
8	S1399	남	NaN	2.0	17.0
9	S1411	여	19.0	12.0	17.0
10	S1411	여	19.0	12.0	17.0
11	S1414	여	18.0	NaN	23.0
12	S1421	남	9.0	12.0	1.0
13	S1424	여	19.0	2.0	15.0
14	S1428	남	19.0	NaN	2.0

결측치 치환하기(df.fillna())

- 데이트의 수가 적고 결측치가 많을 때 활용
- df.interpolate(): 앞/뒤 행의 중간값으로 치환

```
import pandas  as pd
score3 = pd.read_excel('/content/성적3.xlsx')
score3
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score3.interpolate()
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	7.0	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	12.5
7	S1399	1	10.5	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	12.0	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	9.0	2.0
14	S1433	1	19.0	16.0	5.0

결측치 치환하기(df.fillna())

❖ df.fillna(method= ‘ffill’) : 결측치가 있는 앞/뒤 행의 값으로 치환

- method= ‘ffill’ : 앞 행의 값으로 치환
- method= ‘bfill’ : 뒤 행의 값으로 치환

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score3.fillna(method='ffill')
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	2.0	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	8.0
7	S1399	1	2.0	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	12.0	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	2.0	2.0
14	S1433	1	19.0	16.0	5.0

```
score3.fillna(method='bfill')
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	12.0	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	17.0
7	S1399	1	19.0	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	12.0	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	16.0	2.0
14	S1433	1	19.0	16.0	5.0

결측치 치환하기(df.fillna())

- df.fillna(값, inplace=True)
  - Table에 있는 전체 NaN을 동일값으로 변경
  - 값 : 변환할 데이터, 데이터가 저장된 변수명, 열의 평균이나 중간값
    - df[‘math’].mean(): math 열의 평균값으로 대체
    - df.fillna({ ‘학과’ : ‘영어영문학과’ }) : 학과열의 NaN을 모두 ‘영어영문학과’로 변환
    - df.fillna({ ‘학과’ : ‘영어영문학과’ , ‘참여횟수’ :15})
      - 학과열의 NaN을 모두 ‘영어영문학과’로 변환하고 ‘참여횟수’를 15로 변환

결측치 치환하기(df.fillna())

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score3.fillna(50)
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	50.0	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	50.0
7	S1399	1	50.0	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	50.0	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	50.0	2.0
14	S1433	1	19.0	16.0	5.0

```
score3.fillna(score3['출석'].mean())
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.000000	12.000000	25.000000
1	S1244	1	17.000000	2.000000	22.000000
2	S1254	1	2.000000	16.000000	25.000000
3	S1256	3	13.461538	11.000000	25.000000
4	S1384	1	12.000000	22.000000	12.000000
5	S1391	2	18.000000	25.000000	8.000000
6	S1393	1	2.000000	25.000000	13.461538
7	S1399	1	13.461538	2.000000	17.000000
8	S1411	2	19.000000	12.000000	17.000000
9	S1411	2	19.000000	12.000000	17.000000
10	S1414	2	18.000000	13.461538	23.000000
11	S1421	3	9.000000	12.000000	1.000000
12	S1424	2	19.000000	2.000000	15.000000
13	S1428	1	19.000000	13.461538	2.000000
14	S1433	1	19.000000	16.000000	5.000000

결측치 치환하기(df.fillna())

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score3.fillna({'출석':10, '프로젝트':20})
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1256	3	10.0	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	10.0	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	20.0	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	20.0	2.0
14	S1433	1	19.0	16.0	5.0

이상치 확인 및 치환하기

이상치

- 정상 범위에서 벗어난 존재할 수 없는 값 또는 극단적인 값
- 예) 성별은 1(남)과 2(여) 값만을 갖는데 그외의 다른 데이터

이상치 확인하기

- df['열이름'].value_counts().sort_index()

이상치 데이터 치환하기

- df['열이름'] = df['열이름'].replace('찾는 데이터', '변환할 데이터')

이상치 확인 및 치환하기

```
score4['성별'].value_counts().sort_index()
```

```
1    8
2    5
3    2
Name: 성별, dtype: int64
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score4['성별'] = score4['성별'].replace(3, 2)
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	2	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	2	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

중복값 확인하기

❖ df.duplicated(['열이름'])

- '열이름' 을 기준으로 중복되는 행 검출

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score4.duplicated(['ID'])
```

```
0     False
1     False
2     False
3      True
4     False
5     False
6     False
7     False
8     False
9      True
10    False
11    False
12    False
13    False
14    False
dtype: bool
```

중복값 제거하기

- df.drop_duplicates(['열이름']) : 열이름을 기준으로 중복 데이터 행 삭제
- df.drop_duplicates(subset=['열이름' , '열이름'], keep= "last")
  - subset=['열이름' , '열이름']: 2개 열의 데이터가 일치하는 행 삭제
  - keep= " " : 데이터를 유지할 행 설정
    - 기본은 첫 번째 데이터를 유지
    - keep= "last" : 마지막 데이터 유지
    - keep=False : 모두 삭제

중복값 제거하기

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	3	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	3	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score4.drop_duplicates(['ID'])
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	2	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score4.drop_duplicates(subset=['ID','성별'])
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	2	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
8	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	2	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

중복값 제거하기

```
score4.drop_duplicates(subset=['ID','성별'], keep='last')
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	2	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
9	S1411	2	19.0	12.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	2	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0

```
score4.drop_duplicates(subset=['ID','성별'], keep=False)
```

	ID	성별	출석	프로젝트	실험
0	S1233	1	2.0	12.0	25.0
1	S1244	1	17.0	2.0	22.0
2	S1254	1	2.0	16.0	25.0
3	S1244	2	NaN	11.0	25.0
4	S1384	1	12.0	22.0	12.0
5	S1391	2	18.0	25.0	8.0
6	S1393	1	2.0	25.0	NaN
7	S1399	1	NaN	2.0	17.0
10	S1414	2	18.0	NaN	23.0
11	S1421	2	9.0	12.0	1.0
12	S1424	2	19.0	2.0	15.0
13	S1428	1	19.0	NaN	2.0
14	S1433	1	19.0	16.0	5.0