문제해결을 위한 데이터 분석 및 시각화

numpy이해와 numpy 활용 그래프 그리기

한성대학교 노은희 교수

"미래로 향하는 새로운 이정표"

오늘의 학습

학습내용

- numpy 이해
- numpy를 이용한 차트 그리기
- numpy 함수
- DataFrame을 numpy 배열 형식으로 변환

numpy 이해

넘파이(numPy) 기초: 배열 및 벡터 계산

파이썬 list와 numpy 배열

- numpy는 배열(array)을 다루는 도구
- 배열은 동일한 자료형을 가진 데이터를 연속적으로 저장
- numpy는 ndarray객체 제공
- ndarray는 n차원 배열 의미하고 동일한 자료형의 항목들만 저장
- 파이썬의 list는 동일하지 않는 자료형도 저장 가능
- ndarray는 배열과 배열간의 수학적인 연산 가능
- ndarray는 풍부한 함수 제공

numpy 배열 만들기

import numpy as np

파이썬 리스트

```
a1 = np.array([1,2,3,4,5])        # numpy배열을 만들 때 array()함수 사용
```

넘파이 배열

numpy 배열 만들기

numpy 배열 생성하기 => 1차원 배열

```
1  ### 시퀀스 데이터로부터 배열 생성
2  import numpy as np          # numpy 패키지 불러오기
3
4  a1 = np.array([1,2,3,4,5])       # numpy배열을 만들 때 array()함수 사용
5
6  print(a1.dtype)
7  print(a1)
```

```
int32
[1 2 3 4 5]
```

```
1  data1 = [1.0,2.0,3.0,4.0,5.0]
2  a2 = np.array(data1)
3
4  print(a2.dtype)
5  print(a2)
```

```
float64
[1. 2. 3. 4. 5.]
```

```
1  data2 = np.array(['1','2','3','4','5'])
2
3  print(data2)
4  print("data type=>",data2.dtype)     #U1은 numpy 데이터 형식의 유니코드이며 문자의 수는 1개라는 것을 의미
```

```
['1' '2' '3' '4' '5']
data type=> <U1
```

numpy 배열 만들기

numpy 배열 생성하기 ==> arange()

```
1  ##  arr = np.arange([start], stop[,step])
2  # start부터 시작해서 stop 전까지 step만큼 더해서 numpy 배열을 생성
3  b1 = np.arange(0,10,1)
4  b1
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
1  b2 = np.arange(0,10,2)
2  b2
```

```
array([0, 2, 4, 6, 8])
```

```
1  b3 = np.arange(5)
2  b3
```

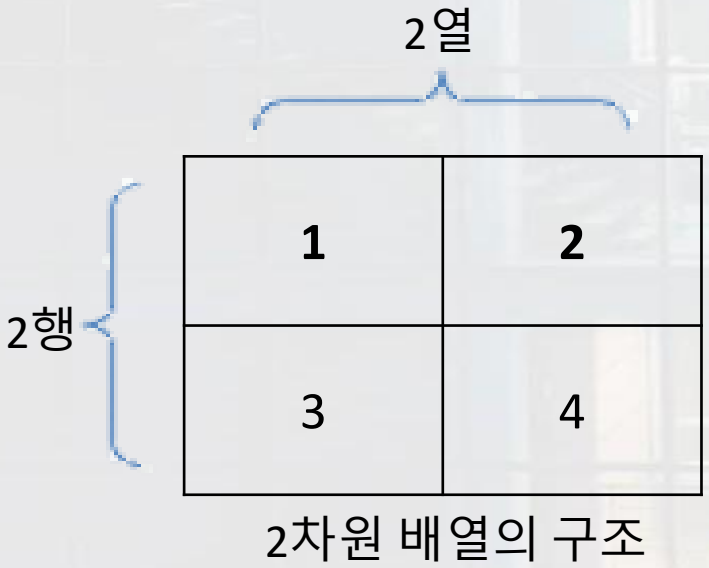
```
array([0, 1, 2, 3, 4])
```

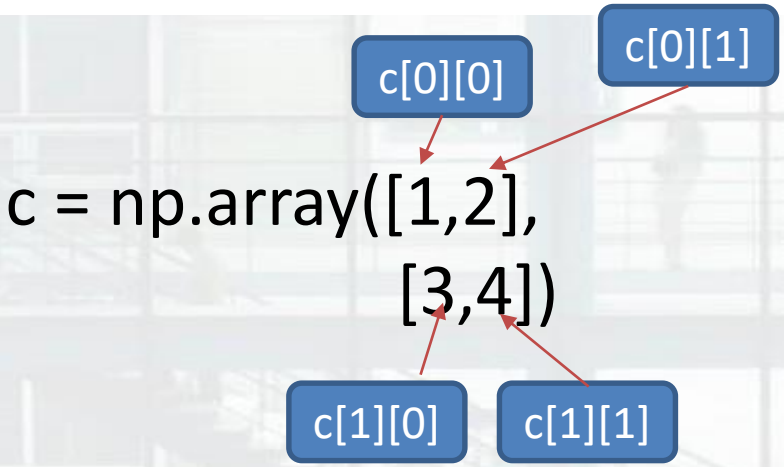
numpy 배열 만들기_다차원 배열

numpy 배열 생성하기 ==> 다차원 배열

```
1 import numpy as np
2 c = np.array([[1,2],[3,4]])    # array()함수는 리스트를 인수로 받아서 배열을 생성, 2차원 배열 생성
3 c
```

```
array([[1, 2],
       [3, 4]])
```





배열 생성하기

배열생성하기 ==> reshape(m,n)을 추가하면 m x n 형태의 2차원 배열로 변경

```
1  import numpy as np
2  arr1 = np.arange(20).reshape(4,5)  # reshape(m,n)을 추가하면 m x n 형태의 2차원 배열로 변경
3  arr1
```

```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19]])
```

```
1  arr2 = np.arange(30).reshape(3,10)
2  arr2
```

```
array([[ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14, 15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24, 25, 26, 27, 28, 29]])
```

```
1  arr3 = np.zeros((5,5))     # zeros()함수는 0으로 채워진 넘파이 배열 생성
2  arr3
```

```
array([[0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.]])
```

배열 생성하기

```
arr4 = np.ones((5,5))      # ones()함수는 1으로 채워진 넘파이 배열 생성
arr4
```

```
array([[1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.],
       [1., 1., 1., 1., 1.]])
```

```
# 넘파이 배열 차원 확인하기
print(arr1.shape)
print(arr2.shape)
```

```
(4, 5)
(3, 10)
```

배열의 사칙연산

```
1  ## 넘파이 배열 사칙 연산
2  arr5 = np.array([[1,2,3],
3                   [4,5,6]])
4
5  arr6 = np.array([[7,8,9],
6                   [10,11,12]])
7  print("-> arr5+arr6=")
8  print(arr5+arr6,"₩n")
9
10 print("->arr5-arr6=")
11 print(arr5-arr6,"₩n")
12
13 print("->arr5*arr6=")
14 print(arr5*arr6,"₩n")
15
16 print("->arr5/arr6=")
17 print(arr5/arr6,"₩n")
```

```
-> arr5+arr6=
[[ 8 10 12]
 [14 16 18]]

->arr5-arr6=
[[-6 -6 -6]
 [-6 -6 -6]]

->arr5*arr6=
[[ 7 16 27]
 [40 55 72]]

->arr5/arr6=
[[0.14285714 0.25       0.33333333]
 [0.4        0.45454545 0.5       ]]
```

array 인덱싱

numpy에서 사용되는 인덱싱은 python 인덱싱과 동일
python에서와 같이 0번째로 시작

numpy 배열 인덱싱

```
1  data = [100,50,80,88,70,79]
2  score = np.array(data)
3  score[0]
```

```
100
```

```
1  score[0:3]
```

```
array([100,  50,  80])
```

```
1  arr7 = np.array([[1,2],[3,4]])
2  arr7
```

```
array([[1, 2],
       [3, 4]])
```

```
1  arr7[0][1]
```

```
2
```

```
1  arr7[1][1]
```

```
4
```

numpy배열 이용 차트 그리기_하나의 차트에 여러 개의 그래프 그리기

```
1  # 하나의 차트에 여러개의 데이터 그리기
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  x = np.arange(1,11,1)
6  y = np.arange(10,20,1)
7  z = np.arange(20,30,1)
8
9  plt.plot(x,x,label='x')
10 plt.plot(x,y,label='y')
11 plt.plot(x,z,label='z')
12
13 plt.xlabel("X")
14 plt.ylabel("Y")
15
16 plt.legend(loc = 'upper left')
17 plt.title('My plot')
18 plt.show()
```

numpy배열 이용 차트 그리기_산점도 그래프

산점도 (Scatter plot)는 두 변수의 상관 관계를 직교 좌표계의 평면에 점으로 표현하는 그래프

array or list

```
plt.scatter(x,y)
```

색상과 크기 지정하기

크기와 색상지정

```
plt.scatter(x,y,s=size,c=colors)
```

투명도와 컬러맵 설정하기

투명도와 컬러맵 지정하기

```
plt.scatter(x,y,alpha=0.5, camp = 'jet')
```

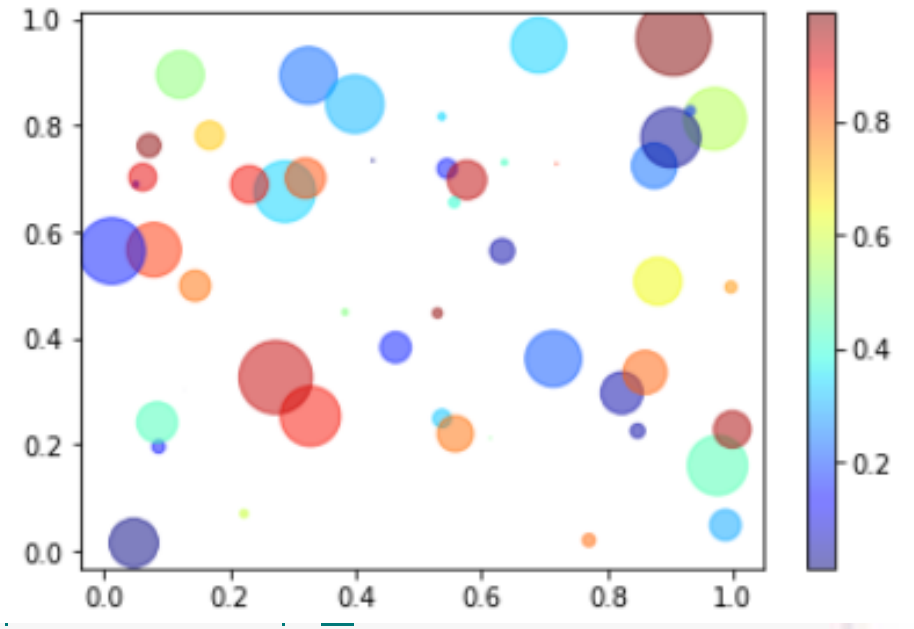
alpha 파라미터는 마커의 투명도를 지정, **0**에서 **1** 사이의 값을 입력
cmap 파라미터에 컬러맵에 해당하는 문자열을 지정

[실습하기]

numpy를 이용한 산점도 그래프 그리기

scatter()함수 사용

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  n = 50
5  x = np.random.rand(n)                    #50개의
6  y = np.random.rand(n)                    #50개의 랜덤 값 추출
7  size = (30 * np.random.rand(n))**2
8  colors = np.random.rand(n)
9
10 plt.scatter(x, y, s=size, c=colors, alpha=0.5, cmap='jet')
11 plt.colorbar()
12 plt.show()
```



random.rand() 함수는 주어진 형태의 난수 array를 생성
(0, 1) 범위에서 균일한 분포를 갖습니다.

numpy 배열 함수 사용

- .sum(): 모든 요소의 합
- .min(): 모든 요소 중 최소값
- .max(): 모든 요소 중 최대값
- .argmax(): 모든 요소 중 최대값의 인덱스
- .cumsum(): 모든 요소의 누적합

numpy배열 함수사용

```
1  a = np.arange(1,11,1).reshape(2,5)
2  print(a)
3
4  # 모든 요소의 합
5  print(a.sum())
6
7  # 모든 요소 중 최소값
8  print(a.min())
9
10 # 모든 요소 중 최대값
11 print(a.max())
12
13 # 모든 요소 중 최대값의 인덱스
14 print(a.argmax())
15
16 # 모든 요소의 누적합
17 print(a.cumsum())
18 # [  0   1   5  14  30  55  91 140]
```

```
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]]
55
1
10
9
[ 1  3  6 10 15 21 28 36 45 55]
```

DataFrame을 numpy 배열 형식으로 변환

.values 또는 .to_numpy() 를 사용해
DataFrame을 numpy 배열 형식으로 변환

DataFrame을 numpy로 변경

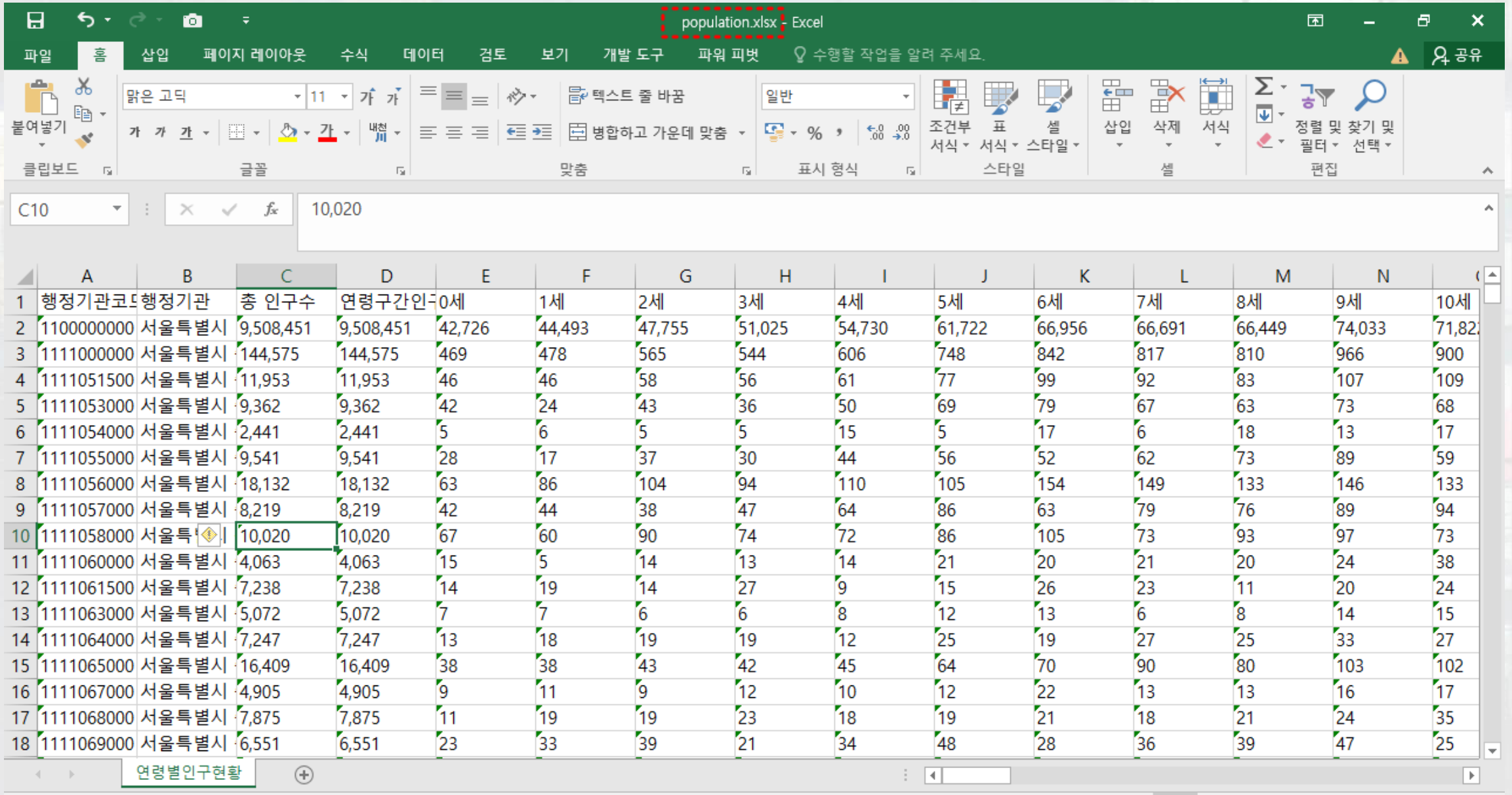
```
1  import pandas as pd
2
3  # DataFrame 생성
4  data = [['Park',21],['Kim',20],['Lee',22]]
5  df = pd.DataFrame(data, columns=['Name','Age'])
6  df
7
```

	Name	Age
0	Park	21
1	Kim	20
2	Lee	22

```
1  # .values 또는 .to_numpy() 를 사용해 numpy 배열로 변환
2  print(df.values)
3  print(df.to_numpy())
```

```
[['Park' 21]
 ['Kim' 20]
 ['Lee' 22]]
[['Park' 21]
 ['Kim' 20]
 ['Lee' 22]]
```

[실습하기]



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	
1	행정기관코드	행정기관	총 인구수	연령구간인구수	0세	1세	2세	3세	4세	5세	6세	7세	8세	9세	10세
2	1100000000	서울특별시	9,508,451	9,508,451	42,726	44,493	47,755	51,025	54,730	61,722	66,956	66,691	66,449	74,033	71,82
3	1111000000	서울특별시	144,575	144,575	469	478	565	544	606	748	842	817	810	966	900
4	1111051500	서울특별시	11,953	11,953	46	46	58	56	61	77	99	92	83	107	109
5	1111053000	서울특별시	9,362	9,362	42	24	43	36	50	69	79	67	63	73	68
6	1111054000	서울특별시	2,441	2,441	5	6	5	5	15	5	17	6	18	13	17
7	1111055000	서울특별시	9,541	9,541	28	17	37	30	44	56	52	62	73	89	59
8	1111056000	서울특별시	18,132	18,132	63	86	104	94	110	105	154	149	133	146	133
9	1111057000	서울특별시	8,219	8,219	42	44	38	47	64	86	63	79	76	89	94
10	1111058000	서울특별시	10,020	10,020	67	60	90	74	72	86	105	73	93	97	73
11	1111060000	서울특별시	4,063	4,063	15	5	14	13	14	21	20	21	20	24	38
12	1111061500	서울특별시	7,238	7,238	14	19	14	27	9	15	26	23	11	20	24
13	1111063000	서울특별시	5,072	5,072	7	7	6	6	8	12	13	6	8	14	15
14	1111064000	서울특별시	7,247	7,247	13	18	19	19	12	25	19	27	25	33	27
15	1111065000	서울특별시	16,409	16,409	38	38	43	42	45	64	70	90	80	103	102
16	1111067000	서울특별시	4,905	4,905	9	11	9	12	10	12	22	13	13	16	17
17	1111068000	서울특별시	7,875	7,875	11	19	19	23	18	19	21	18	21	24	35
18	1111069000	서울특별시	6,551	6,551	23	33	39	21	34	48	28	36	39	47	25

[실습하기]

```
1  # 엑셀파일 DataFrame 객체로 불러오기 ->원하는 지역 인구 검색
2  import numpy as np
3  import pandas as pd
4
5  df = pd.read_excel('population.xlsx', index_col = 1)
6  df
7  name = input('원하는 지역의 이름 입력=>')
8  a = df.index.str.contains(name)
9  df1 = df[a]
10 df1
11
```

원하는 지역의 이름 입력=>

원하는 지역의 이름 입력=>성북구 삼선동

행정기관	행정기관코드	총 인구수	연령구간인구수	0세	1세	2세	3세	4세	5세	6세	...	91세	92세	93세	94세	95세	96세	97세	98세	99세	100세 이상
서울특별시 성북구 삼선동	1129055500	22,879	22,879	69	62	64	83	78	106	123	...	26	14	13	14	7	5	3	0	4	10

1 rows × 104 columns

[실습하기]

```
1  # 검색한 DataFrame -> numpy로 배열로 변경하기
2  print(df1.values)
```

```
[[1129055500 '22,879' '22,879' '69' '62' '64' '83' '78' '106' '123' '123'
  '118' '150' '157' '155' '151' '170' '175' '187' '160' '201' '215' '235'
  '314' '369' '398' '436' '476' '434' '445' '457' '434' '420' '410' '324'
  '345' '300' '333' '313' '304' '257' '309' '319' '312' '312' '308' '291'
  '306' '328' '344' '374' '423' '410' '460' '452' '412' '405' '361' '332'
  '343' '313' '315' '311' '314' '373' '321' '301' '296' '272' '285' '253'
  '192' '211' '165' '163' '204' '168' '196' '199' '161' '160' '151' '176'
  '170' '128' '119' '122' '97' '80' '66' '56' '36' '38' '19' '26' '14'
  '13' '14' '7' '5' '3' '0' '4' '10']]
```

```
1  print(df1.to_numpy())
```

```
[[1129055500 '22,879' '22,879' '69' '62' '64' '83' '78' '106' '123' '123'
  '118' '150' '157' '155' '151' '170' '175' '187' '160' '201' '215' '235'
  '314' '369' '398' '436' '476' '434' '445' '457' '434' '420' '410' '324'
  '345' '300' '333' '313' '304' '257' '309' '319' '312' '312' '308' '291'
  '306' '328' '344' '374' '423' '410' '460' '452' '412' '405' '361' '332'
  '343' '313' '315' '311' '314' '373' '321' '301' '296' '272' '285' '253'
  '192' '211' '165' '163' '204' '168' '196' '199' '161' '160' '151' '176'
  '170' '128' '119' '122' '97' '80' '66' '56' '36' '38' '19' '26' '14'
  '13' '14' '7' '5' '3' '0' '4' '10']]
```

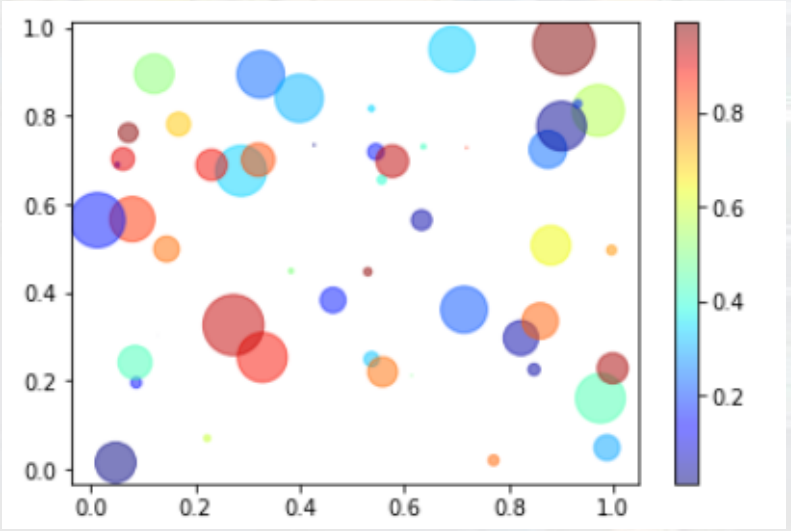
마무리

```
import numpy as np

a1 = np.array([1,2,3,4,5])      # numpy배열을 만들 때 array()함수 사용
```

파이썬 리스트

넘파이 배열



DataFrame을 numpy로 변경

```
1  import pandas as pd
2
3  # DataFrame 생성
4  data = [['Park',21],['Kim',20],['Lee',22]]
5  df = pd.DataFrame(data, columns=['Name','Age'])
6  df
7
```

	Name	Age
0	Park	21
1	Kim	20
2	Lee	22

```
1  # .values 또는 .to_numpy() 를 사용해 numpy 배열로 변환
2  print(df.values)
3  print(df.to_numpy())
```

```
[['Park' 21]
 ['Kim' 20]
 ['Lee' 22]]
[['Park' 21]
 ['Kim' 20]
 ['Lee' 22]]
```