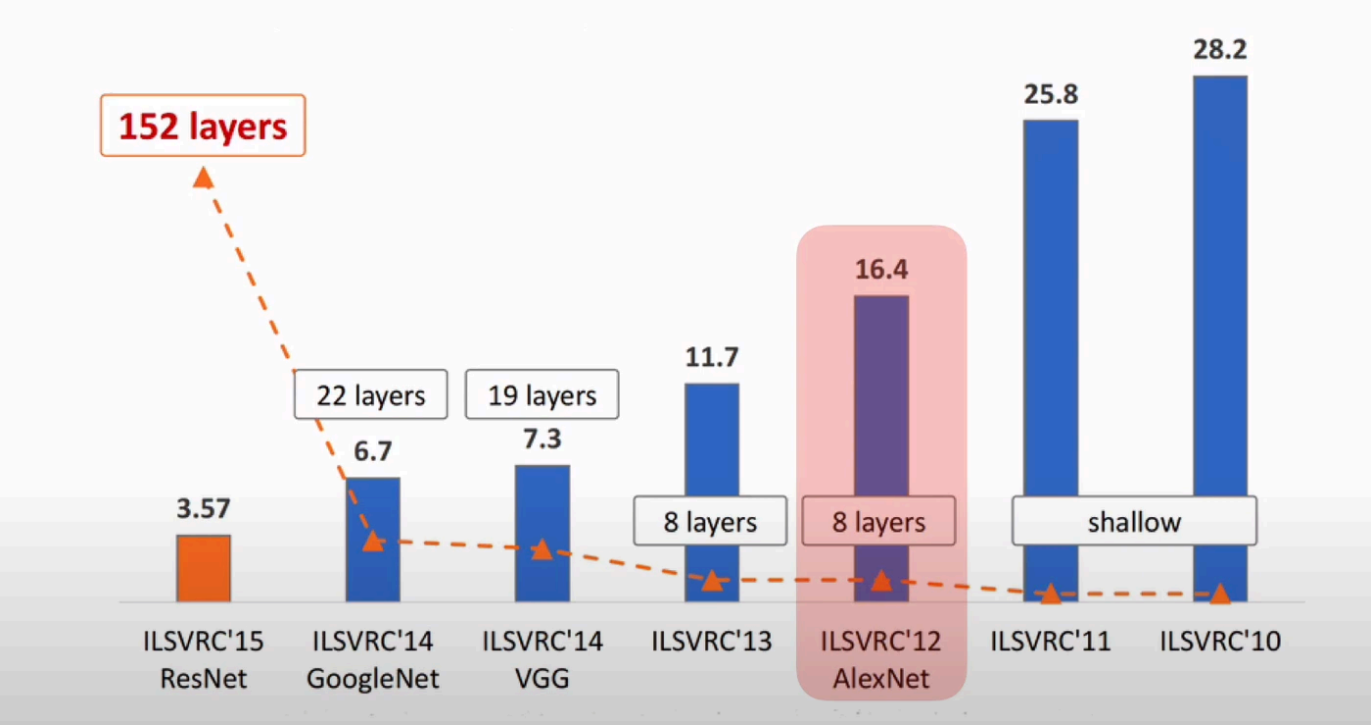
이미지 분류기

우리는 이제 표가 아닌 이미지 데이터를 사용해 보겠습니다. 요즘 딥러닝의 대표로 유명한 자율주행은 표 형태의 데이터를 이용하지 않습니다. 표가 아닌 형태의 데이터 학습은 어떻게 하는 걸까요.

오늘 우리는 가장 간단한 형태의 이미지 분류기를 만들어 보겠습니다. 표가 아닌 이미지를 가지고 학습하는 가장 간단한 형태의 딥러닝 모델입니다. 이미지는 우리 일상에 매우 흔하게 사용한 데이터죠. 카메라 기술이 발전하고 스마트폰으로 누구나 카메라를 사용할 수 있게 되면서 우린는 상당히 많은 데이터를 이밋 형태로 저장하고 사용합니다. 사진을 찍고 이미지를 검색하는 것은 매우 일상적인 일이 되었습니다. 인류에게는 꿈이 있었습니다. 컴퓨터에게 보는 법을 알려주는 것입니다. 사물을 보고 어떤 사물인지 이해하는 일을 컴퓨터에게도 알려주고 싶었습니다.

처음에는 사람보다 계산을 훨씬 잘하는 컴퓨터들이 두세살 아이도 할 수 있는 개와 고양이 구분, 그것을 알려주는 일이 그렇게 어렵지 않을 거라고 생각했어요. 금방 할 수 있을 줄 알았죠. AI의 선구자인 민스키 박사는 MIT 학부 신입생에게 방학과제로 컴퓨터 비전 과제를 줍니다. 무려 1966년 이었어요. 다시 말하지만 대학교 1학년에게 방학 과제로 준 것입니다. 당연히 그 일은 쉬운 일이 아니었고 컴퓨터 비전은 고사하고 이미지 분류하는 일조차 할 수 없었죠.



2012년 힌트 교숙 이미지 분류 딥러닝 모데 알렉사넷을 발표하기 전까지는 세상 누구도 하지 못하던 과제였습니다.

우리는 이번 수업에서 손글씨 숫자 이미지를 분류하는 문제를 중심으로 이미지 분류기를 만들어 볼 겁니다. 마지막 수업에는 10가지 종류의 사진 이미지를 분류하는 문제도 다뤄 보겠습니다. 각자가 가진 이미지를 가지고 학습하는 방법도 알려 드릴 겁니다. 이번 시간에도 가장 간단한 형태의 코드를 반복해서 경험하면서 해보겠습니다.

데이터와 차원

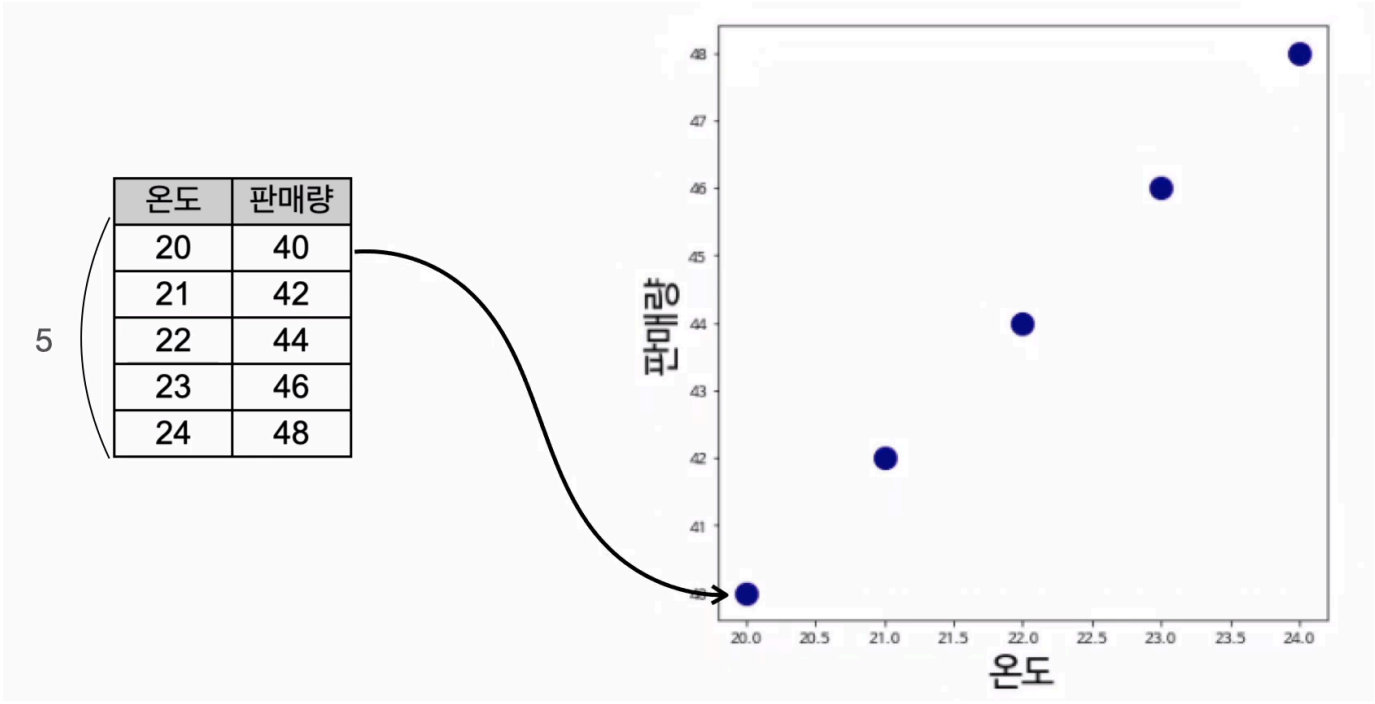
먼저 이미지를 학습 시키기 위해서는 이미지 데이터가 어떻게 생겼는지 이해하는게 필요합니다. 데이터 과학 분야가 어려운 이유중에 하나가 용어입니다. 그야말로 용어 지옥입니다. 특히 어려운 것은

1. 생소한 용어가 너무 많다.
2. 하나의 대상을 부르는 다양한 표현이 있다.
3. 하나의 표현이 몇가지 다른 의미로 해석되는 경우가 있다.

이렇게 세가지로 나눌수 있습니다. 우리가 배울 차원이라는 말도 이세번째에 속합니다. 이제부터 차원이라는 말에 두가지 의미를 알아보겠습니다.

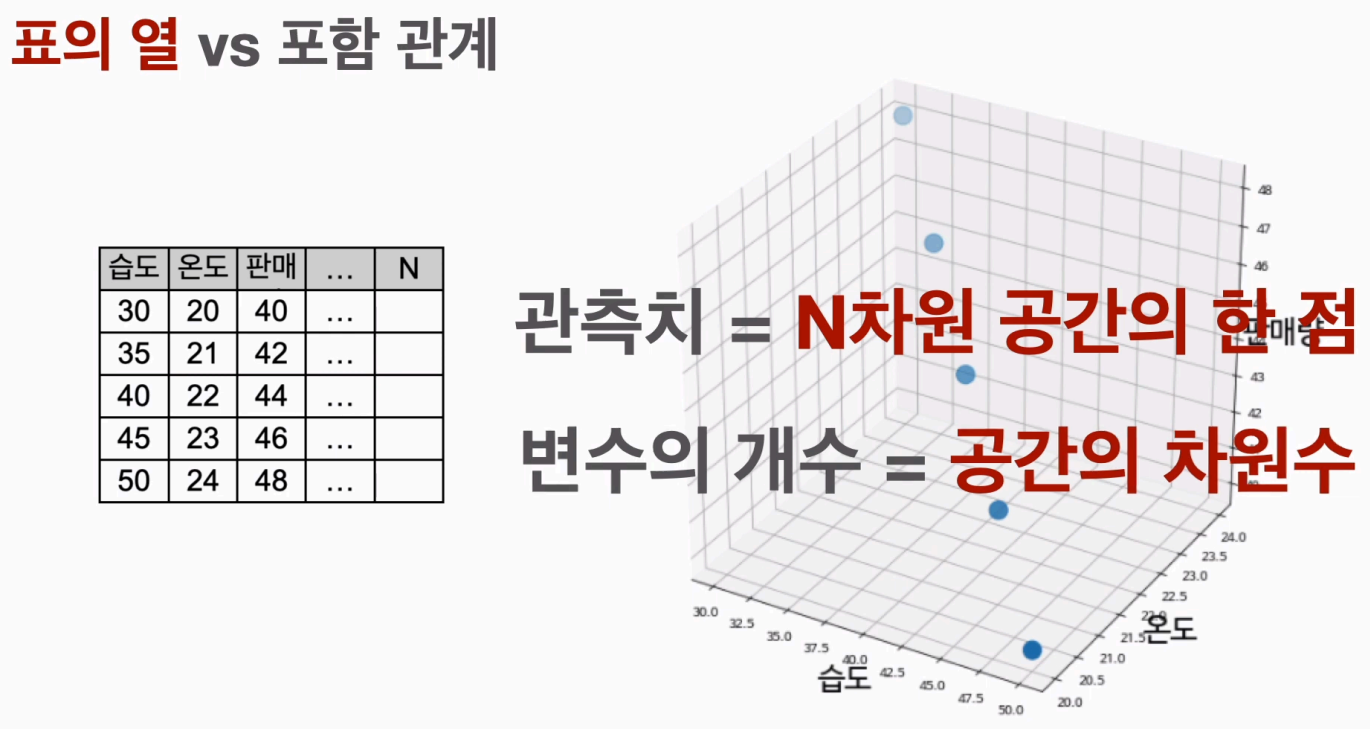
어떤때는 표에열컬럼의 관점에서 차원을 이야기 하고 어떤 때는 데이터 사이의 포함 관계를 바라보는 관점에서 차원을 말합니다.

우선 컬럼에 관점에서 차원을 사용하는 경우를 생각해 봅니다. 온도와 판매량 변수 2개로 이루어진 데이터가 있습니다. 관측치는 5개이고 각 데이터들을 2차원 좌표 평면 위에 나타낼 수 있는데 다섯 개의 점으로 표현할 수 있습니다. 각점은 1개의 관측치를 나타냅니다.



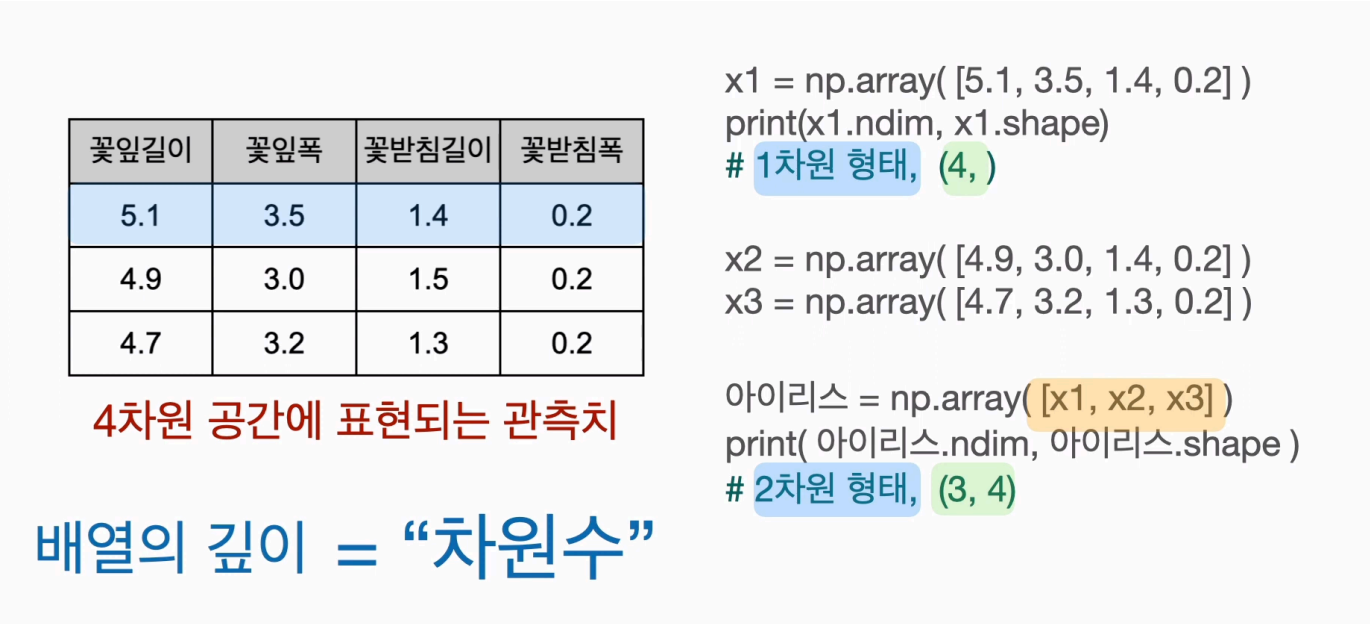
습도라는 변수를 추가하여 변수가 3개인 표를 구성한다면 데이터는 3차원 공간의 한점으로 표현할 수 있게 됩니다. 만약 컬럼의 숫자가 n개라면 각데이너틑 n차원 공간에 1점으로 표현 할 수 있습

니다. 즉 변수에 갯수는 데이터를 표현하는 공간에 차원수와 같습니다. 표에 데이터를 공간에 옮기게 되면 그때 우리는 모든 데이터를 기하학적인 관점으로 바라보고 해석하게 되는데요.



간단하게 설명하자면 데이터들간의 거릴 잴수 있게 되는 것이고 데이터들 사이에 가까운 정도를 숫자로 정밀하게 비교할 수 있게 된다는 의미입니다. 이렇듯 차원을 말할때 데이터를 공간 속에 표현한다는 맥락에서 사용하며 변수의 개수가 공간에 차원수가 되는 것입니다.

이번에는 데이터 포함 관계의 관점에서 차원이 사용되는 것을 살표 보겠습니다. 데이터 포함관계는 배열의 깊이와 관련이 있습니다. 아리리스의 데이터를 가져와서 보겠습니다.

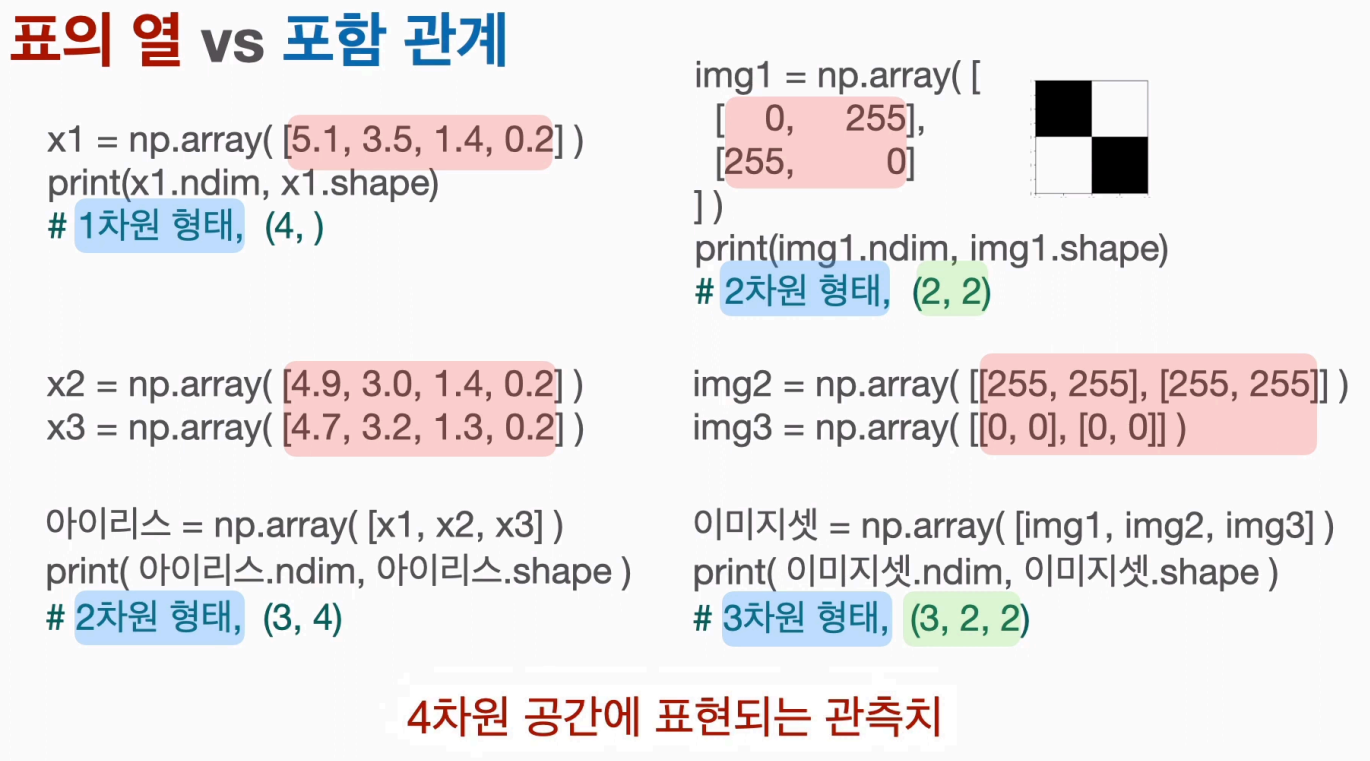


각 데이터는 4차원 공간에 표현된 데이터입니다. 방금 배운대로 열이 네개이므로 4차원 입니다. 관측치 하나를 코드로 표현해 보면 우측위와 같이 됩니다. 이렇게 하나의 배열에 값이 들어가 있는 모양을 1차원 이라고 합니다. x1의 쉐이프를 출력하면 숫자가 1개 들어가 있습니다. 그래서 1차원 입니다. 숫자가 4인 것은 네개의 요소를 지니고 있다는 뜻입니다. 다른 두개의 관측치도 각가 같은 코드로 작성해 볼 수 있습니다. 표는 이 관측치들의 모음으로 만들게 되는데요. 이것을 코드로 표현하

면 아래의 코드가 됩니다. 1차원의 배열 세계를 품고 있는 배열로 만들어졌습니다. 즉 배열의 깊이가 2가 되었고 깊이가 2라서 쉐이프를 출력하면 두개의 숫자가 들어가 있습니다. 이것을 2차원 형태라고 부릅니다. 이렇듯 차원을 말할때 데이터의 형태를 표현하는 맥락에서 사용하면 배열에 깊이는 차원수가 되어 보시는 것처럼 차원이라는 말은 맥락저긍로 다르게 사용됩니다.

변수에 숫자를 차원 수로 이해할 때도 있고 ㅜ배열에 깊이를 차원수라고 이해할 때도 있습니다.

이제부터 우리가 볼 이미지는 표에 관측치처럼 1차원 형태가 아니기 때문입니다. 이제 우리는 표를 벗어나 데이터를 이해할 준비가 되었습니다. 이미지 파일 하나가 데이터의 관점에서는 관측치의 하나라고 할 수 있습니다.



흑백의 이미지는 데이터 하나가 2차원 형태를 가집니다. 지금 샘플로 구성한 데이터는 2,2모양을 가지고 있는데요. 가로 2px, 세로 2px로 되어 있습니다. 출력해보면 그림에 나오는 것처럼 생긴 이미지 입니다. 이미지 데이터 자체는 뒤에 조금 더 자세히 살펴볼 것입니다. 지금은 데이터 모양 차원에 집중해 주세요. 왼쪽의 관측치를 2개 추가했듯이 이미지를 2개 더 축했습니다. 파이썬에서 이런 종류의 데이터를 선언할 때 줄바꿈이 의미가 없는데, 코드가 한 줄로 되어 있어서 img1을 만들 때 작성한 코드와 달라 보이지만 같은 형태의 데이터를 만드는 코드입니다. 좌측 하단에 코드를 보면 관측치를 모아 하나의 표를 구성했습니다. 마찬가지로 이미지들을 모아서 이미지셋을 구성했습니다. 이렇게 만들어진 이미지셋은 3차원 형태로 표현되어 있고 3, 2, 2 모양인 것을 알수 있습니다. 여기서 관측치들은 모두 가지고 있는 요소들의 개수가 4개입니다. 데이터 형태를 표현하는 관점에서 죄측의 x1은 1차원 형태이고 우측의 img1은 2차원 형태로 서로 다르지만 데이터 공간에 맥락에서는 x1과 img1 모두 동일하게 4차원 공간에 한점으로 표현할 수 있는 관측치인 것입니다.

데이터 형태의 맥락에서 한번 더 연습해 보겠습니다. 좌측 위에 표현된 d1은 1차원 형태의 데이터입니다. shape 를 추적하면 3이라는 숫자 하나만 있습니다. d2는 d1을 5개 가지고 있습니다. 2차원 형태입니다. shpae를 출력하면 5와 3으로 두개의 숫자를 가지고 있습니다.

```
d1 = np.array( [1, 2, 3] )
print( d1.ndim, d1.shape)
# 1차원 형태,  (3, )
```

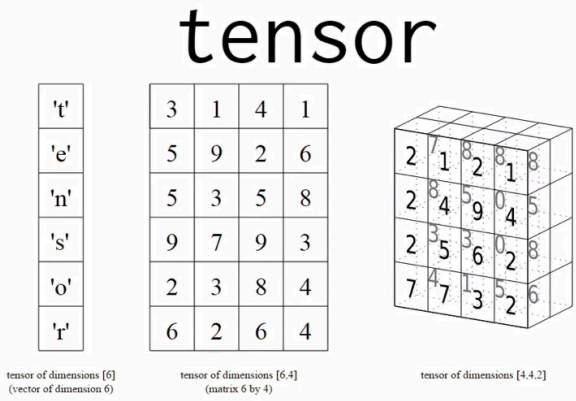
```
d2 = np.array( [d1, d1, d1, d1, d1] )
print( d2.ndim, d2.shape)
# 2차원 형태,  (5, 3)
```

```
d3 = np.array( [d2, d2, d2, d2] )
print( d3.ndim, d3.shape)
# 3차원 형태,  (4, 5, 3)
```

```
d4 = np.array( [d3, d3] )
print( d4.ndim, d4.shape)
# 4차원 형태,  (2, 4, 5, 3)
```

배열의 깊이 = 차원수

d3는 d2를 4개 가지고 잇습니다. shape는 4, 5, 3 이고 숫자의 개수가 3개이니까 3차원 형태입니다. d4는 d3를 두개 가지고 있습니다. shape는 2, 4, 5, 3이 되고 숫자의 갯수가 4개니까 4차원 형태입니다. 데이터 형태의 맥락에서 배열에 깊이가 차원수라는 것이 조금 더 이해가 되시나요? 이렇게 1차원 2차원 등 여러 차원 형태로 구성되어 있는 데이터의 모습을 **텐서**라고 합니다.



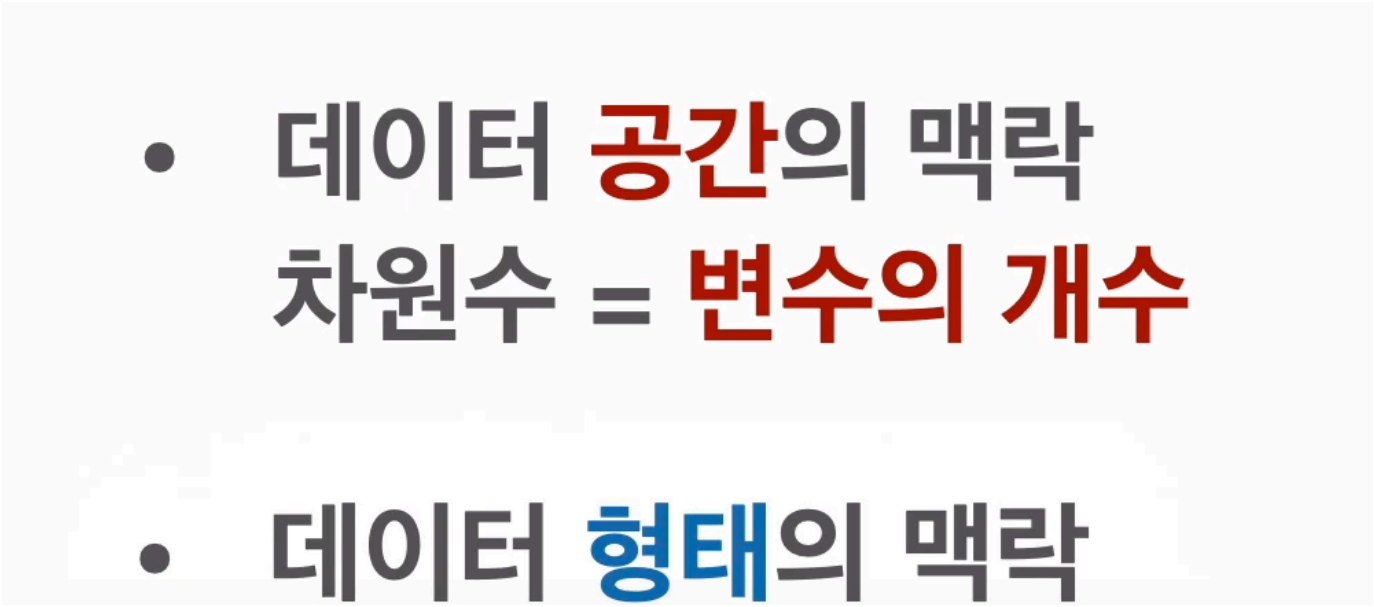

```
d2 = np.array( [d1, d1, d1, d1, d1] )
print( d2.ndim, d2.shape)
# 2차원 형태,  (5, 3)
```

```
d4 = np.array( [d3, d3] )
print( d4.ndim, d4.shape)
# 4차원 형태,  (2, 4, 5, 3)
```

배열의 깊이 = 차원수

우리가 지금 사용하는 라이브러리가 텐서플로어입니다. 텐서플로어는 이러한 텐서가 흘러가면서 모델을 학습한다는 것을 착안하여 이름을 붙인 것입니다.

정리하면 데이터 공간의 맥락에서는 변수의 개수가 차원의 수입니다. 데이터 형태의 맥락에서는 배열의 깊이가 차원의 수입니다.


• 데이터 공간의 맥락
차원수 = 변수의 개수
• 데이터 형태의 맥락