강원지역혁신플랫폼

기계학습

Machine Learning

차원 축소의 개념

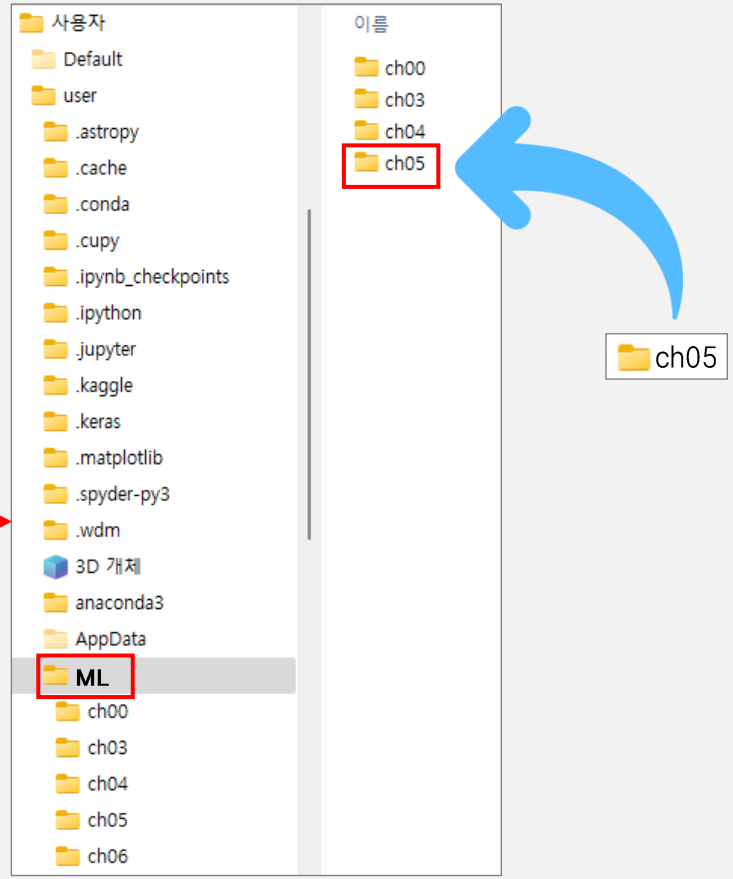
학습목표

- 차원 축소와 차원의 저주 개념을 이해하고 설명할 수 있습니다.

01 | 5주차 실습코드 복사하기

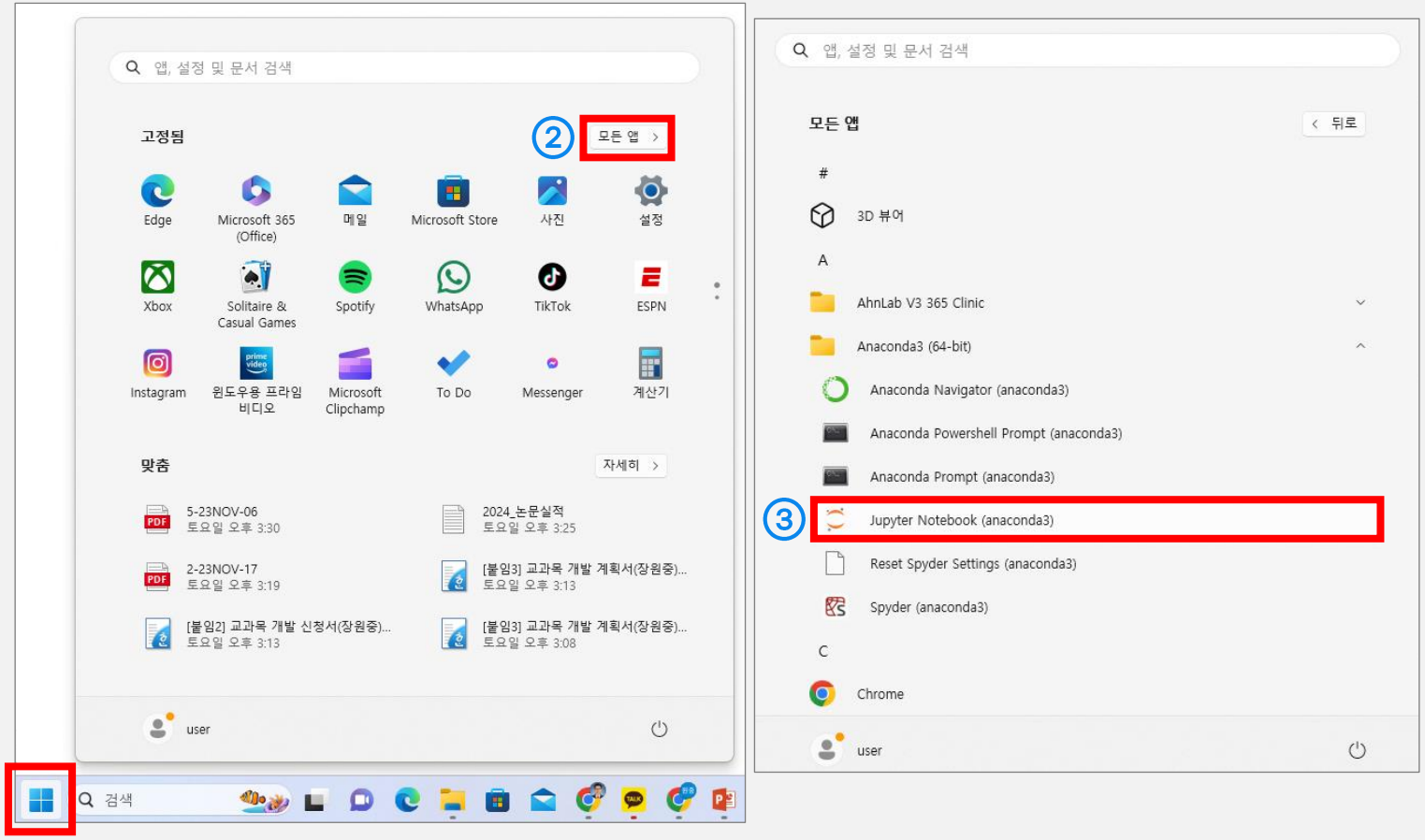
(권장) 아래와 같은 경로에 실행 소스가 존재하면 환경 구축 완료

- 5주차 실습코드 다운로드 → 압축해제 → ch05 폴더를 ML 하위 폴더로 복사
- c:₩Users₩user>ML> 컴퓨터이름 또는 사용자계정



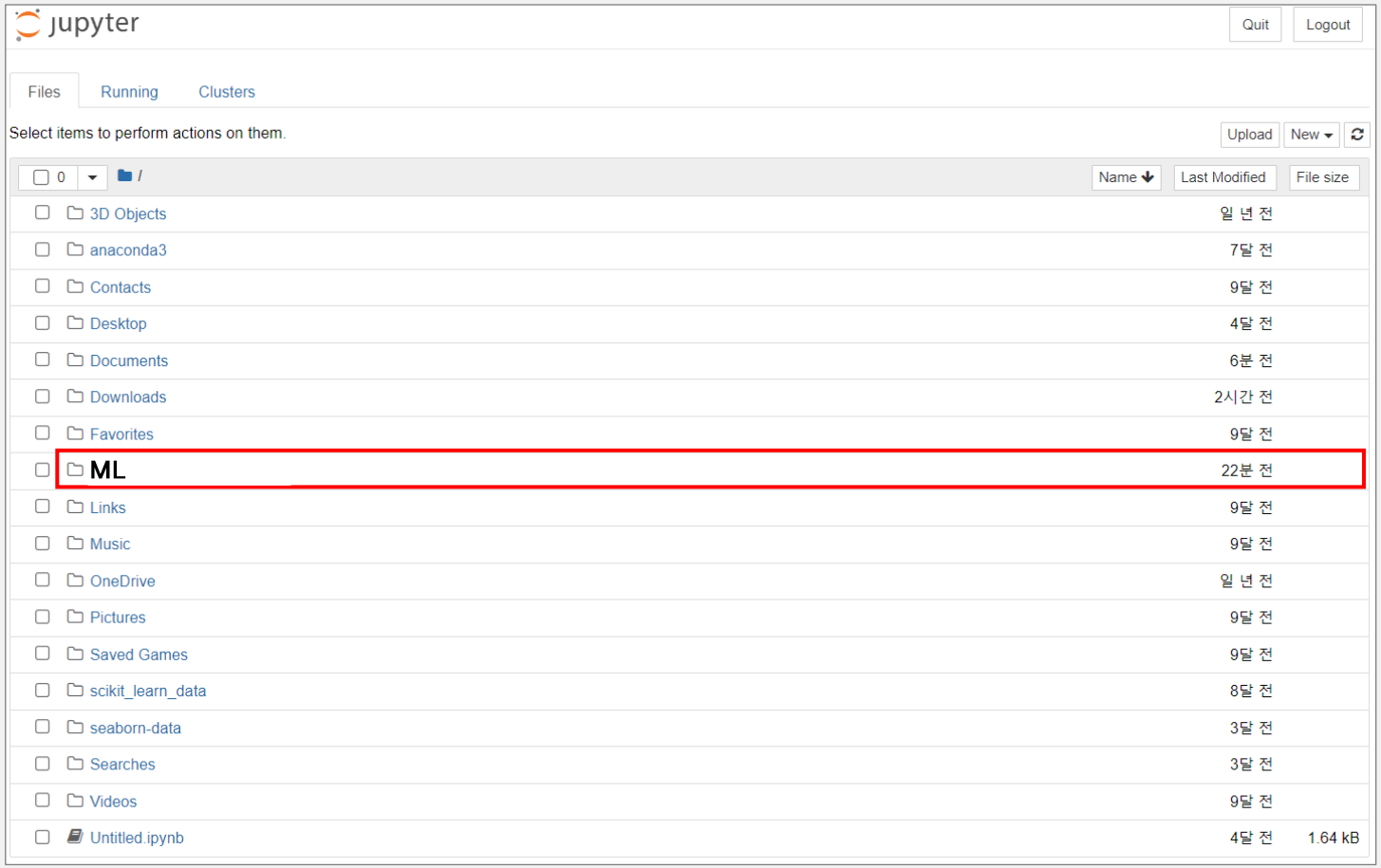
02 | Jupyter Notebook 실행하기

✦ ①시작 메뉴 클릭 > ②모든 앱 버튼 클릭 > ③Anaconda3(64-bit)
> “Jupyter Notebook (anaconda)” 메뉴 클릭하기



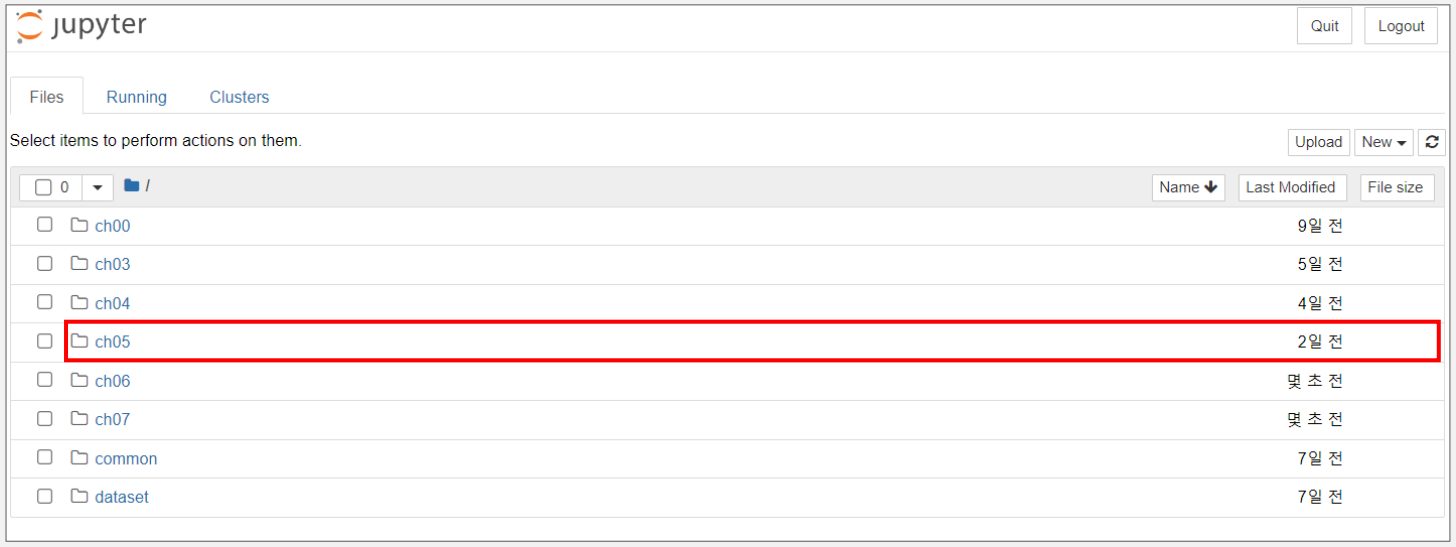
03 | ML 폴더

ML 폴더를 클릭하기



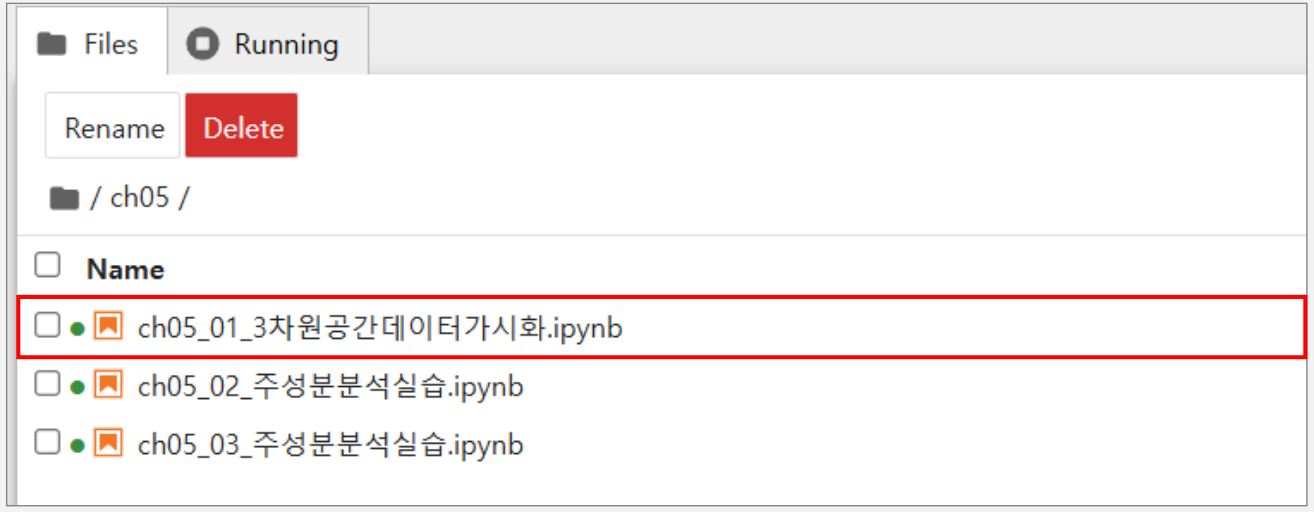
04 | ch05 폴더

ch05 폴더 클릭하기



05 | ch05_01_3차원공간데이터가시화.ipynb

✦ ch05_01_3차원공간데이터가시화.ipynb 파일 클릭하기



06 | 차원 축소의 개념

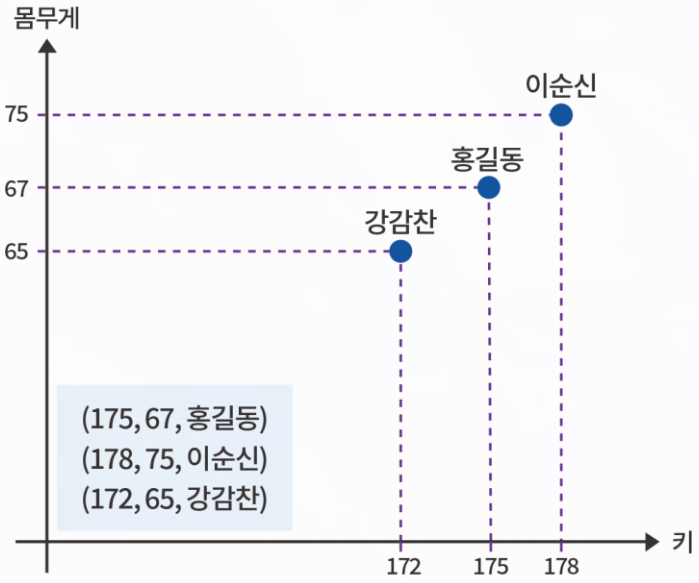
차원 축소의 개념

- 머신러닝 문제는 훈련 샘플 각각이 수천 심지어 수백만 개의 특성을 가지고 있음
  - 이런 많은 특성은 훈련을 느리게 할 뿐만 아니라, 좋은 솔루션을 찾기 어렵게 만듦
    - 이런 문제를 종종 차원의 저주(curse of dimensionality)라고 부름

- 차원(dimension)이라는 것은 어떤 공간에 존재하는 데이터들을 식별하는 데에 필요한 최소 수의 좌표값이라고 할 수 있음
  - 2차원 공간의 점들은 2개의 값을 가진 좌표로 표현함
  - 3차원 공간의 점들은 3개의 값을 가진 좌표로 표현함

06 | 차원 축소의 개념

- 사람의 정보를 (키, 몸무게, 이름)으로 표현하면, 데이터를 표현하는 값이 3개이므로 3차원 공간에서 다룬다고 할 수 있음
  - 일반적으로 각 차원은 크기를 비교할 수 있는 공간으로 가정함
    - 따라서 (키, 몸무게, 이름)으로 표현된 데이터는 (키, 몸무게) 2차원의 데이터 각각에 (이름)이 속성으로 부여된 것으로 볼 수 있음
      - 차원은 데이터를 다룰 때 각 데이터를 표현하는 특징(feature)의 수라고 할 수 있음

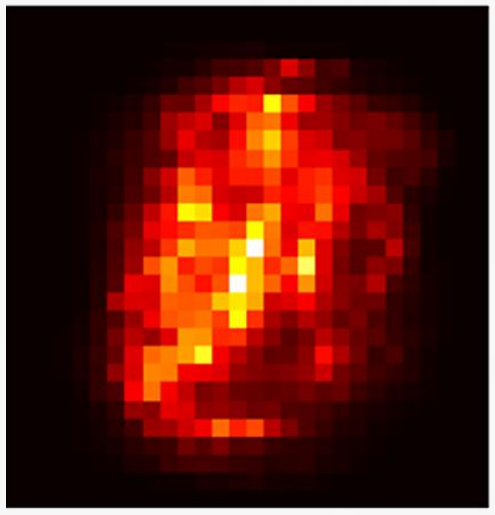


06 | 차원 축소의 개념

- 높은 차원의 데이터는 많은 양의 데이터를 가지고 있지만,
  이 데이터 가운데 상당한 부분은 중요하지 않은 내용일 가능성이 높음
  - 데이터를 분석하는 도구가 중요한 정보와 그렇지 않은 정보를 제대로 구별해내지 못 하면 올바른 분석을 하지 못 할 수도 있음
    - 높은 차원의 데이터를 다룰 때는 중요하지 않은 차원을 생략하고 중요한 차원만 남기는 차원 축소(dimensionality reduction)가 필요함
      - 차원을 축소하는 것은 필연적으로 정보의 손실을 가져올 수밖에 없음
      - 하지만, 데이터 분석의 효율을 높이기 위해 차원 축소는 필요함

06 | 차원 축소의 개념

- 예를 들어 아래 우측 그림과 같이 MNIST **이미지 경계**에 있는 **픽셀**은 **거의 항상 흰색**이므로 훈련 세트에서 이런 픽셀을 **완전히 제거**해도 **많은 정보를 잃지 않음**
  - 아래 **좌측 그림**을 보면 이런 픽셀들이 **분류 문제**에서 **크게 중요하지 않다**는 것을 알 수 있음

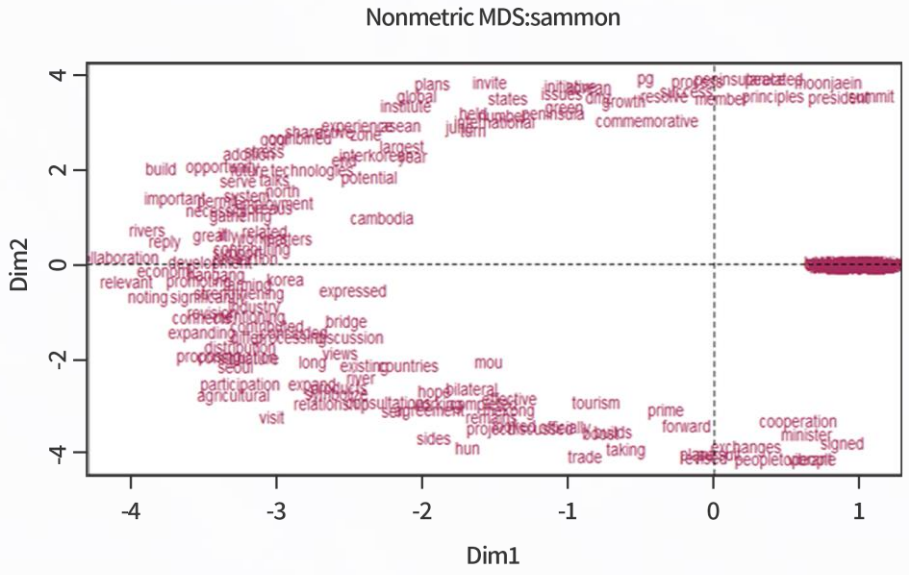


랜덤 포레스트 분류기에서 얻은 MNIST 픽셀 중요도

MNIST 이미지

06 | 차원 축소의 개념

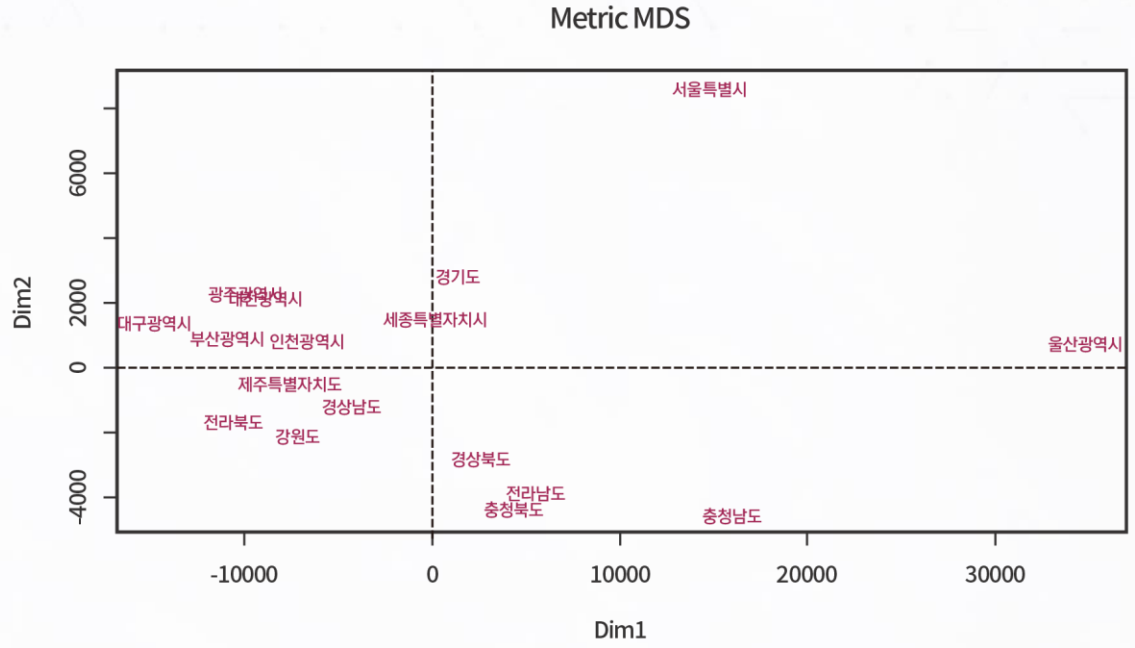
- 게다가 **인접한 두 픽셀**은 종종 **많이 연관**되어 있음
  - **두 픽셀**을 **하나의 픽셀**로 **합**치더라도(예를 들어 두 픽셀 강도를 평균 냄으로써) **잃는 정보**가 **많지 않을 것**임
  - 차원 축소는 **훈련 속도**를 **높이는 것**과 **데이터 시각화**(data visualization)에도 아주 **유용**함
    - 차원 수를 둘로(또는 셋으로)줄이면 **고차원 훈련 세트**를 하나의 **압축된 그래프**로 그릴 수 있고 **군집** 같은 **시각적인 패턴**을 감지해 **중요한 통찰**을 **얻는 경우**가 많음



100차원으로 학습 후 2차원으로 축소된 이미지

06 | 차원 축소의 개념

- 특히 최종 결과를 사용하는 **결정권자**에게 여러분의 판단을 **설명**하는데 **데이터 시각화**는 **필수적**임
  - 아래의 그림은 우리나라 17개 시도의 '**1인당 지역내총생산**', '**1인당 지역총소득**', '**1인당 개인소득**', '**1인당 민간소비**' 자료로 **차원 축소**한 결과임



4차원을 2차원으로 차원 축소

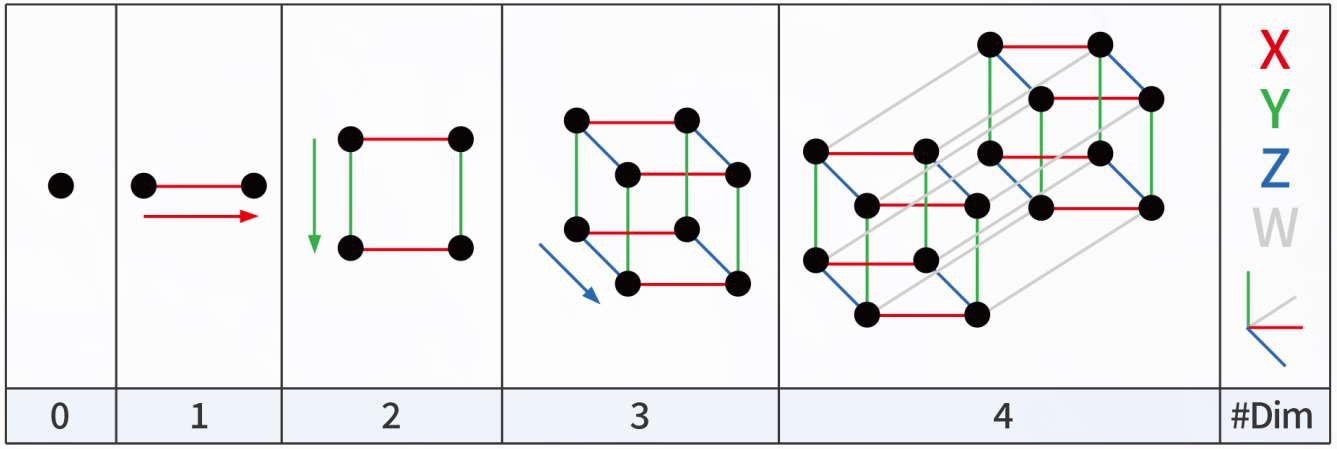
06 | 차원 축소의 개념

- 차원을 축소시키면 일부 정보가 유실됨
  - 그래서 훈련 속도가 빨라질 수는 있지만 시스템의 성능이 조금 나빠질 수 있음
  - 또한, 작업 파이프라인이 조금 더 복잡하게 되고 유지 관리가 어려워짐

  - 차원 축소를 고려하기 전에 훈련이 너무 느린지 먼저 원본 데이터로 시스템을 훈련해봐야 함
    - 어떤 경우에는 훈련 데이터의 차원을 축소시키면 잡음이나 불필요한 세부사항을 걸러내므로 성능을 높일 수 있음
    - 일반적으로는 훈련 속도만 빨라짐

차원의 저주

- 우리는 3차원 세계에서 살고 있어서 고차원 공간을 직관적으로 상상하기 어려움
  - 1,000차원의 공간에서 휘어져 있는 200차원의 타원체는 고사하고 기본적인 4차원 초입방체(hypercube)조차도 머릿속에 그리기 어려움



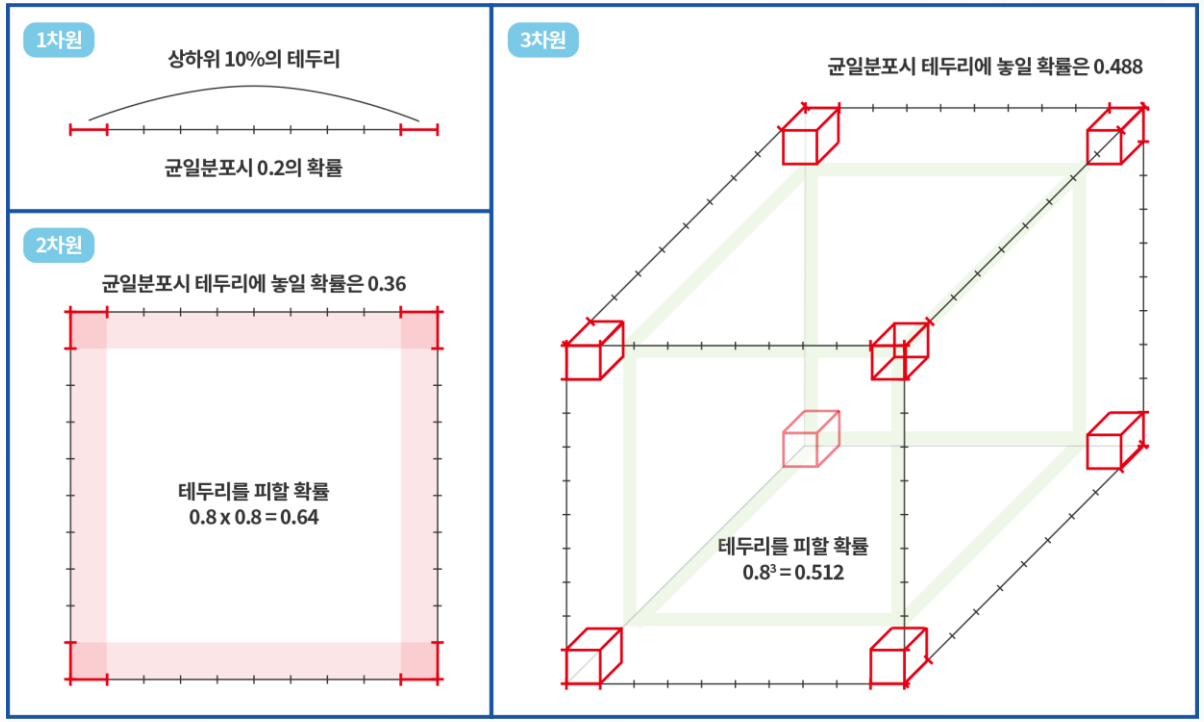
점, 선, 정사각형, 정육면체, 테서랙트(tesseract)
(0차원에서 4차원까지의 초입방체)

출처: 오렐리앙 제롱, 2018, 핸즈온 머신러닝

07 | 차원의 저주

차원의 저주

- 우리는 3차원 공간을 지각하는 데에 적합한 인식을 갖고 있어 고차원 공간은 우리의 직관에 반하는 특성이 많음
  - 차원이 높아지면 데이터들이 공간의 경계에 몰리게 됨
    - 아래의 그림은 데이터가 균일 분포로 1차원, 2차원, 3차원에 흩어져 있는 경우임



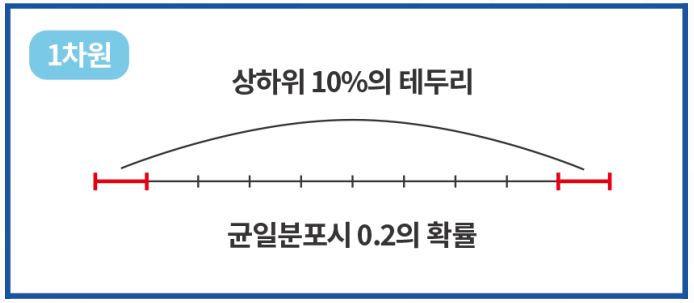
출처: 강영민 외, 2021, 머신러닝

07 | 차원의 저주

데이터가 균일 분포로 흩어져 있다고 가정할 때 1차원 공간에서 데이터 분포는 다음과 같음

- 1차원 공간에서는 하위 10%와 상위 10%의 범위에 놓일 확률은 0.2가 될 것임

테두리에 놓일 확률 = 0.1 + 0.1 = 0.2

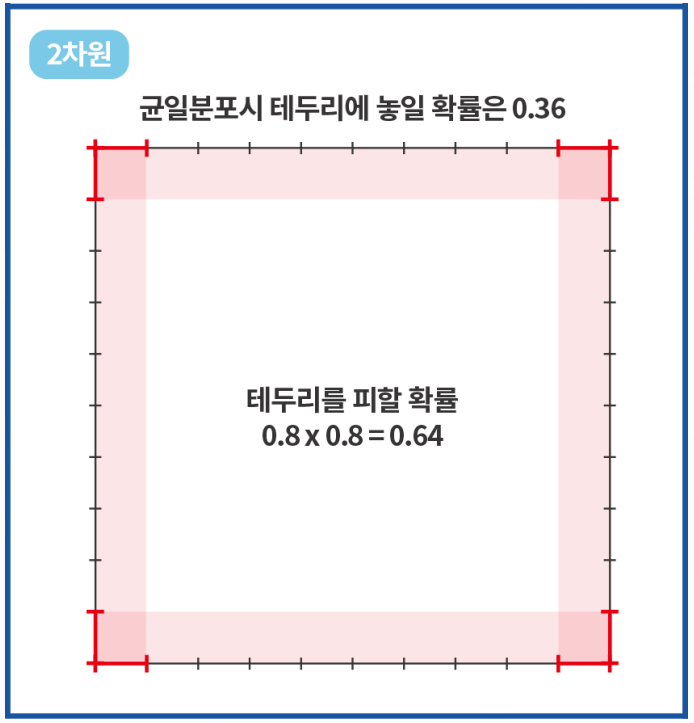


07 | 차원의 저주

- 데이터가 균일 분포로 흩어져 있다고 가정할 때 2차원 공간에서 데이터 분포는 다음과 같음
  - 2차원 공간에서는 하위 10%와 상위 10%를 벗어나 있는 데이터는 0.8의 제곱인 0.64의 확률

테두리를 피할 확률 = 0.8 x 0.8 = 0.64

테두리에 놓일 확률 = (1 – 테두리를 피할 확률) = 1 – 0.64 = 0.36



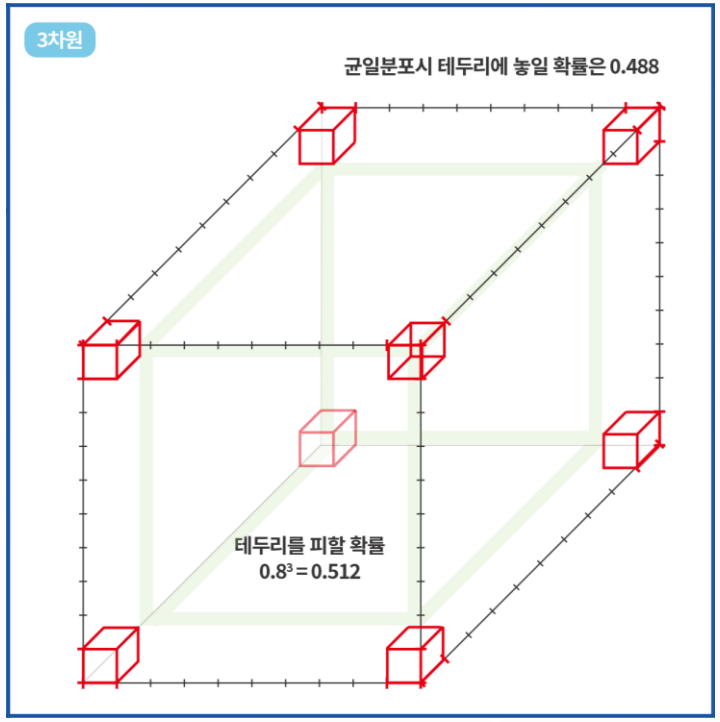
07 | 차원의 저주

데이터가 균일 분포로 흩어져 있다고 가정할 때 3차원 공간에서 데이터 분포는 다음과 같음

- 3차원 공간에서는 하위 10%와 상위 10%의 벗어나 있는 데이터는 0.8의 세 제곱인 0.512의 확률

테두리를 피할 확률 = 0.8 x 0.8 x 0.8 = 0.512

테두리에 놓일 확률 = (1 – 테두리를 피할 확률) = 1 – 0.512 = 0.488



07 | 차원의 저주

데이터가 균일 분포로 흩어져 있다고 가정할 때 10차원 공간에서 데이터 분포는 다음과 같음

- 10차원 공간에서는 하위 10%와 상위 10%의 벗어나 있는 데이터는 0.8의 열 제곱인 0.107의 확률

테두리를 피할 확률 = 0.8^{10} = 약 0.107

테두리에 놓일 확률 = (1 – 테두리를 피할 확률) = 1 – 0.107 = 0.893

차원 수	테두리에 놓일 확률
1차원	0.200
2차원	0.360
3차원	0.488
10차원	0.893

07 | 차원의 저주

- 각 차원의 단위 입방체 내의 두 점 사이의 거리를 생각해 보자.
  - n차원 공간에 있는 두 점의 각 차원별 차이를 d_i라고 하면 두 점의 거리는 다음과 같음

$$\sqrt{\sum_{i=1}^{n} {d_i}^2}$$

  - 두 점이 모든 차원에서 평균적인 차이 d_{avg} 만큼 떨어져 있다고 하면 두 점의 평균 거리는 다음과 같음
    - 이 값은 n이 커지면 큰 값이 됨

$$\sqrt{n \cdot {d_{avg}}^2}$$

07 | 차원의 저주

- 데이터를 다루기 위한 차원이 높아지면 대부분의 점들이 경계선에 놓이게 됨
  - 데이터들 사이의 거리가 멀어져서 공간 내 데이터의 밀도가 낮아지게 되어 데이터들 사이의 관계를 파악하는 것이 힘들어짐
    - 이것을 차원의 저주(curse of dimensionality)라 부름
    - 차원의 저주를 피하는 방법은 데이터를 표현하기 위한 특징의 수를 줄이는 것임
      - 즉 데이터를 더 낮은 차원에서 표현하면 됨
      - 이것을 차원 축소라고 함

07 | 차원의 저주

기계학습에서 **피쳐의 개수**가 **엄청나게 많이 늘어**나고 이에 따라 **데이터**를 **확보**해야 **어느 정도 성능**이 갖추어진 **기계학습 모델**을 만들 수 있음

- 이렇게 **차원**이 계속 **늘어난다**면 점점 우리가 **상상할 수 없는 범위**로 들어가게 됨
  - 예를 들어 **보스턴 집값** 예측(Boston House Price)데이터 셋에서 **피쳐의 개수**는 총 **13개**임

속성명	속성 설명
CRIM	자치시(town)별 1인당 범죄율
ZN	25,000 평방피트를 초과하는 거주지역의 비율
INDUS	비소매 상업 지역이 점유하고 있는 토지의 비율
CHAS	찰스강에 대한 더미 변수 (강의 경계에 위치한 경우는 1, 아니면 0)
NOX	10ppm당 농축 일산화질소
RM	주택 1가구당 평균 방의 개수
AGE	1940년 이전에 건축된 소유주택의 비율
DIS	5개의 보스턴 직업센터까지의 접근성 지수
RAD	방사형 도로까지의 접근성 지수
TAX	10,000달러 당 재산세율
PTRATIO	자치시(town)별 학생/교사 비율
B	1000(Bk-063)^2 여기서 Bk 는 자치시별 흑인의 비율을 말함
LSTAT	모집단의 하위계층의 비율 (%)
MEDV	본인 소유의 주택가격 (중앙값) (단위: $1,000)

07 | 차원의 저주

- 예를 들어 3개의 피쳐는 3차원의 공간에 표현할 수 있음
  - 하지만, 차원이 계속 늘어난다면 점점 우리가 상상할 수 없는 범위로 들어가게 됨

- 기계학습에서는 피쳐 개수가 증가하게 되면 데이터를 표현해야 하는 공간이 지속적으로 늘어나게 되고 이에 대한 처리가 어렵게 되는 것임
  - 이런 문제는 데이터의 분포나 모델을 추정하는 데에 어려움을 야기함
    - 따라서, 피쳐의 개수를 줄이기 위한 차원 축소, 차원 제거 등의 접근을 사용하게 됨

08 | 3차원 공간의 데이터 가시화

[실습] 3차원 공간의 데이터 가시화

- 3차원 공간에 u = $\left[\frac{1}{\sqrt{3}}\frac{1}{\sqrt{3}}\frac{1}{\sqrt{3}}\right]^T$ 축과 v = $\left[\frac{1}{\sqrt{2}}0-\frac{1}{\sqrt{2}}\right]^T$ 를 축으로 하는 2차원 부분 공간에 약간의 잡음을 더해 데이터를 생성함
  - 이 데이터를 3차원 공간에 가시화 함

08 | 3차원 공간의 데이터 가시화

다음은 데이터 형상이 (1000, 3)인 3차원 데이터를 생성하는 코드이다.

- 3차원 공간에 u = $\left[\frac{1}{\sqrt{3}}\frac{1}{\sqrt{3}}\frac{1}{\sqrt{3}}\right]^T$ 축과 v = $\left[\frac{1}{\sqrt{2}}0-\frac{1}{\sqrt{2}}\right]^T$ 를 축으로 하는 2차원 부분 공간에 약간의 잡음을 더해 데이터를 생성함
  - 실행 결과 아래와 같이 데이터 형상이 (1000, 3)인 3차원 데이터가 생성된 것을 볼 수 있음

```
u = np.array([1,1,1]) / np.sqrt(3)
v = np.array([1,0,-1]) / np.sqrt(2)

# 1000개의 데이터 셋 생성하기
n_data = 1000
X = []
for _ in range(n_data) :
   r_coeff = np.random.randn(2,)
   data = 2.0* r_coeff[0] * u + r_coeff[1] * v + 0.1 * np.random.rand(3,)
   X.append(data)

X = np.array(X)
print(X.shape)    # (1000, 3)
```

08 | 3차원 공간의 데이터 가시화

- 다음은 맷플롯립을 이용한 3차원 투영이 가능한 axes를 생성하고, 3차원 점들을 scatter3D() 메소드로 그리는 코드이다.
  - 아래의 3차원 공간의 데이터를 차원 축소하는 방법을 생각해 보자.
    - 실행 결과 아래와 같이 3차원 공간의 데이터가 가시화된 것을 볼 수 있음

```
# 가시화를 위한 맷플롯립 figure 만들기
fig = plt.figure(figsize = (10, 7))
ax = plt.axes(projection ="3d")

# 3차원 공간에 데이터 가시화
ax.scatter3D(X[:,0], X[:,1], X[:,2], color = "green")
plt.title("3D scatter plot of Dataset")
plt.show()
```