인공지능

딥러닝 기초_데이터다루기

강성관 (silicon1@hanmail.net)

딥러닝 기초

딥러닝과 데이터

머신러닝

딥러닝

빅데이터

좋은 결과가 나올까요 ?

얼마나 필요한 데이터를 가지고 있는가 중요함

딥러닝의 시작은 데이터를 분석하고 조사하는 것부터 시작해요

피마 인디언 데이터 분석하기

비만은 유전일까요 ? 후천적 요인 (식습관 등)일까요 ?

↓

모두가 원인이었다는
(피마 인디언의 사례 분석)

1950년대에는 피마 인디언 사람에게는 비만이 한 명도 없었는데 지금은 전체 부족 중 60%가 당뇨, 80%가 비만으로 고통 받고 있음.

이는 영양분을 체내에 저장하는 능력이 뛰어난 피마 인디언과 미국의 기름진 패스트푸드 문화가 만나서 발생한 것

피마 인디언 데이터 분석하기

- 피마 인디언을 대상으로 한 당뇨병 여부를 측정한 데이터셋

- 데이터셋 구성
  - 768명의 데이터
  - 8개의 정보와 1개의 클래스로 구성

	0	1	2	3	4	5	6	7	8
	pregnant	plasma	pressure	thickness	insulin	BMI	pedigree	age	class
	과거임신 회수	공복혈당	확장기 혈압	삼두근 피부 주름 두께	혈청 인슐린	체질량 지수	당뇨병 가족력	나이	당뇨 (1:당뇨)
1	6	148	72	35	0	33.6	0.627	50	1
2	1	85	66	29	0	26.6	0.351	31	0
3	8	183	64	0	0	23.3	0.672	32	1
...									
768	1	93	70	31	0	30.4	0.315	23	0

피마 인디언 데이터 가공하기

- 임신 횟수와 당뇨병 발병 확률

```
7  print(df[["pregnant", "class"]].groupby(["pregnant"], as_index=False)
       .mean().sort_values(by="pregnant", ascending=True))
```

- groupby(["pregnant"]: pregnant를 기준으로 새로운 그룹을 생성
- as_index=False : pregnant 옆에 새로운 인덱스를 만들어줌
- mean() : 평균 계산
- sort_values(by="pregnant", ascending=True) : pregnant를 기준으로 오름차순 정렬

피마 인디언 데이터를 시각적으로 표현하기

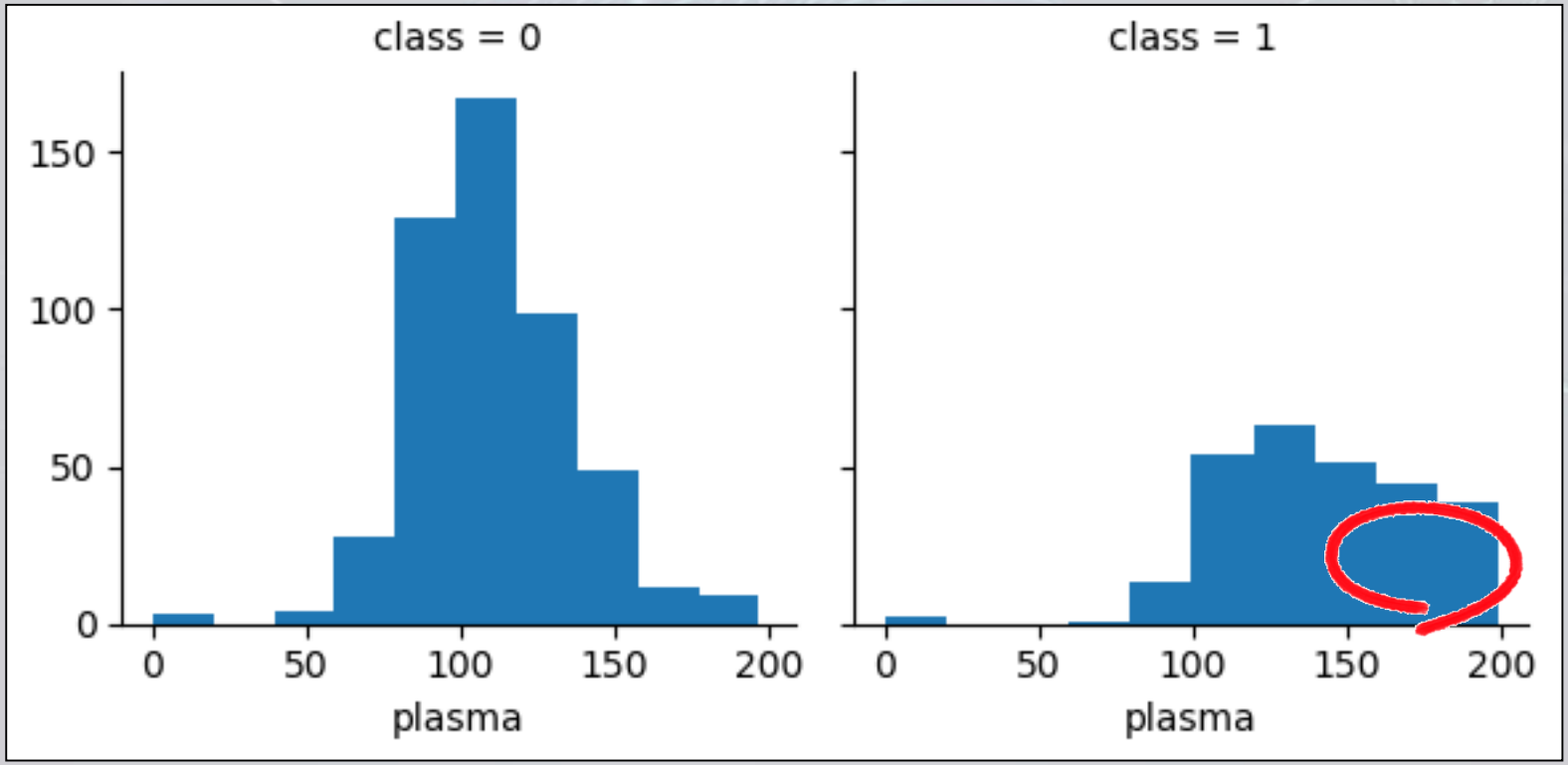
- 상관관계가 높을수록 (큰 수) 밝은 색으로 표시

피마 인디언 데이터를 시각적으로 표현하기

- plt.figure(figsize=(12,12)) : 그래프의 크기를 설정
- heatmap() : 두 항목씩 짝을 지은 뒤 각각 어떤 패턴으로 변화하는지 색상으로 출력하는 함수
  - 두 항목이 전혀 다른 패턴으로 변하면 0, 서로 비슷한 패턴으로 변하면 1에 가까운 값이 출력
  - df.corr() : 항목 간의 상관관계 분석
  - linewidths : 각 셀의 테두리 두께
  - vmax : 색상의 밝기값
  - cmap : 미리 정해진 matplotlib 색상의 설정값
  - linecolor : 각 셀의 테두리 색상
  - annot : 각 셀 값을 표시할지 결정 (True : 숫자)

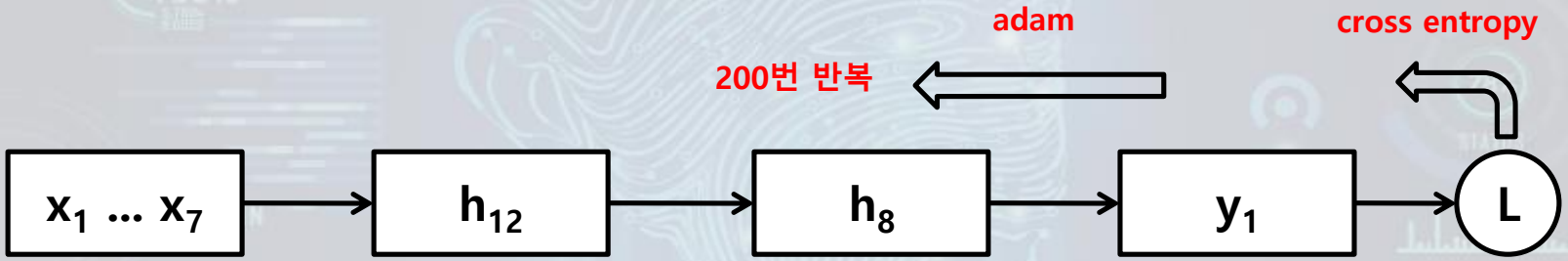
피마 인디언 데이터를 시각적으로 표현하기

- class가 1인 경우 (당뇨병 환자) plasma 항목의 수치가 150이상인 경우가 많음



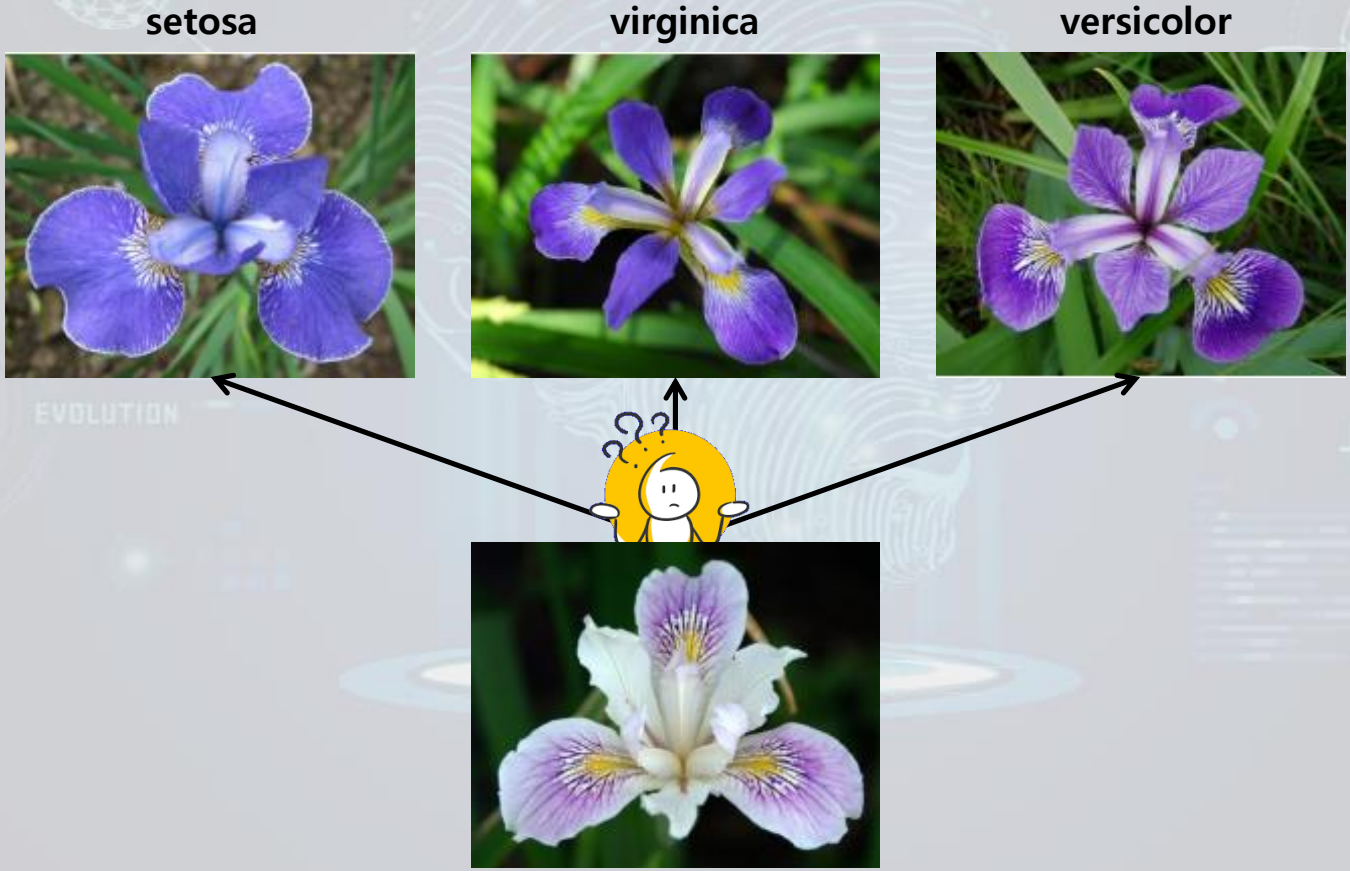
피마 인디언의 당뇨병 예측하기

- 은닉층 2개 사용
- 입력 8개, 출력 1개
- 한번에 처리할 데이터 수 10개, 200번 반복



다중 분류 문제

- 붓꽃 (Iris)의 품종 분류

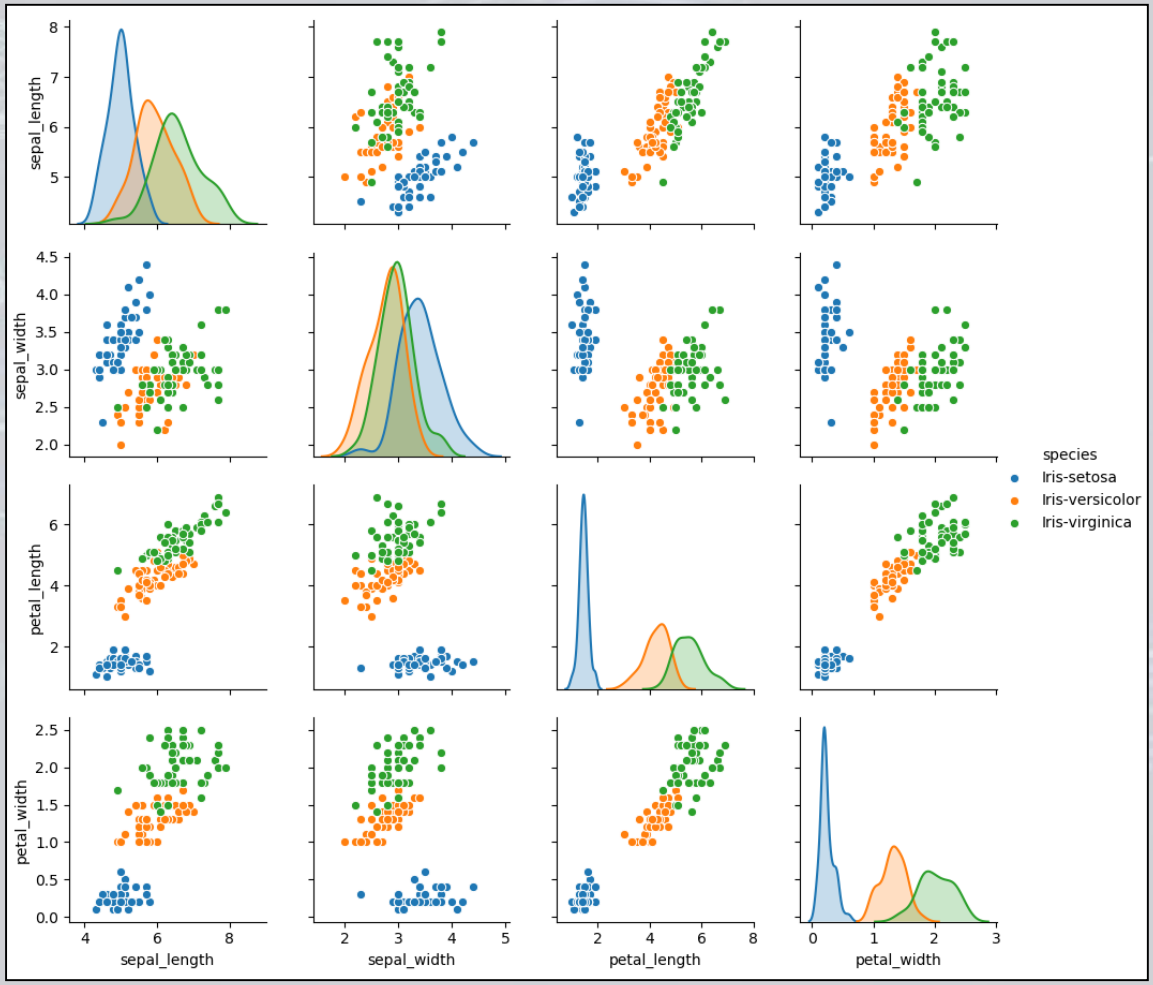


다중 분류 문제

- iris 데이터셋 구성
  - 150개의 데이터
  - 4개의 정보와 1개의 클래스(3개의 품종)로 구성

	0	1	2	3	4
	sepal_length	sepal_width	petal_length	petal_width	species
	꽃받침 길이	꽃받침 넓이	꽃잎 길이	꽃잎 넓이	품종
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
...	...	...	...	...	...
150	5.9	3.0	5.1	1.8	Iris-virginica

다중 분류 문제 – 시각적 표현



원-핫 인코딩 (one-hot encoding)

- sklearn : 문자열을 숫자로 바꿔주는 기능이 포함된 라이브러리
- np_utils : 원-핫 인코딩을 위한 라이브러리

- 원-핫 인코딩 : Y 값을 0과 1로만 된 형태로 바꿔 주는 것

원-핫 인코딩 (one-hot encoding)

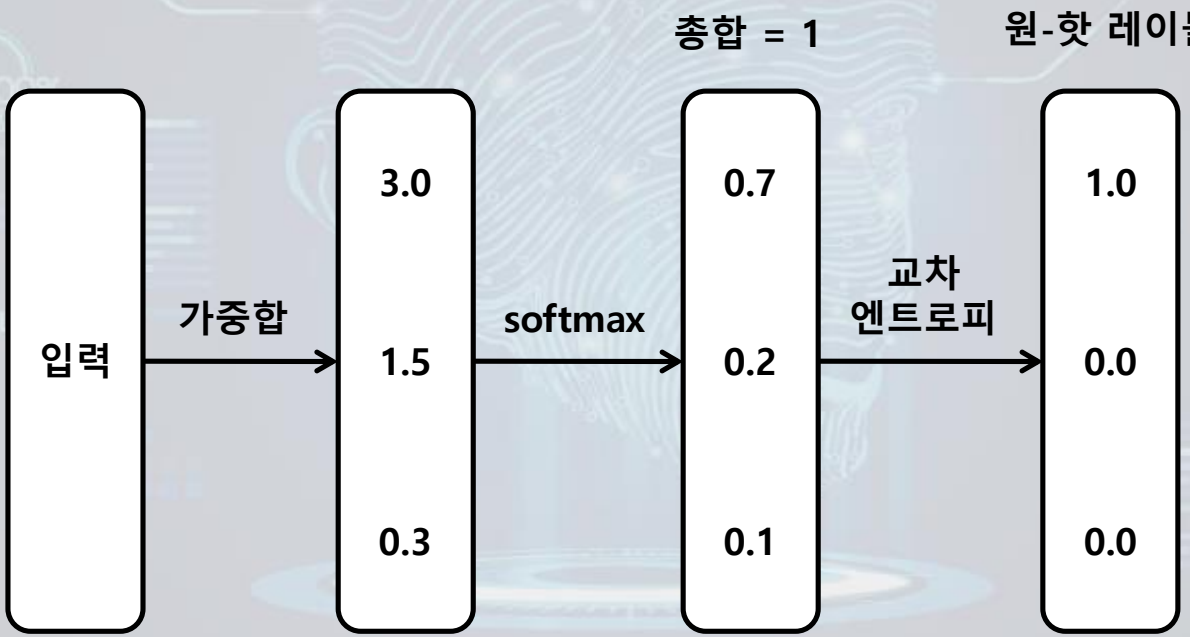
```
20  e = LabelEncoder()
21  e.fit(Y_obj)
22  Y = e.transform(Y_obj)
```

- [20] 라벨 인코더 객체를 생성 (라벨 값들을 0부터 n-1 범위의 값들로 인코딩하기 위한 객체)
- [21] 라벨을 종류별로 구분
- [22] 구분된 라벨에 대해 1부터 n-1 값으로 인코딩

라벨	A, B, B, C, A, B, C, A, B
fit()	A, B, C
transform()	0, 1, 1, 2, 0, 1, 2, 0, 1

다중 분류 문제 – softmax 활성화 함수

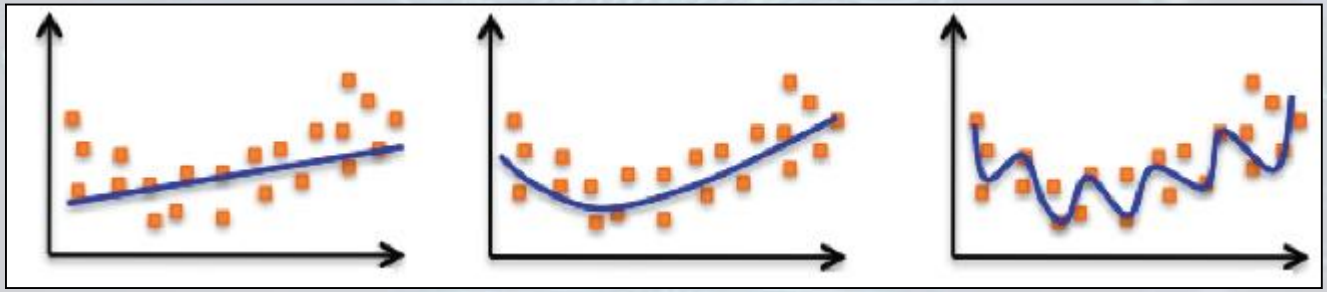
- softmax : 총합이 1인 형태로 확률에 따라 값이 정해짐



과적합 피하기

- 과적합 (Over fitting) : 너무 많은 속성을 사용하면 한 가지 모델에만 적용이 가능하고 새로운 모델에서는 잘 맞지 않게 됨.
- 과적합은 층이 너무 많거나 변수가 복잡해서 발생하기도 하고 테스트셋과 훈련셋이 중복될 때도 발생

- 광석과 일반 돌을 가져다 놓고 음파 탐지기를 쓴 후 그 결과를 데이터 정리하고 역전파를 사용하여 신경망이 얼마나 광석과 돌을 구분하는지 실험



Underfitting Generalization Overfitting

과적합 피하기

- 음파탐지기로 광석과 일반 돌을 측정한 데이터
- sonar 데이터셋 구성
  - 208개의 데이터
  - 60개의 정보 (컬럼 수)와 1개의 클래스로 구성

	0	1	...	59	60
1	0.0200	0.0371		0.0032	**R**
2	0.0453	0.0523		0.0044	**R**
3	0.0262	0.0582		0.0078	**R**
...					
208	0.0260	0.0363		0.0115	**M**

과적합 피하기

```
13  df = pd.read_csv('sonar.csv', header=None)
```

- [13] 변수명이 없는 데이터를 불러올 때 header=None으로 설정 (sonar.csv는 헤더가 없는 데이터)
- 이전 소스와 같이 names=[“이름1”, “이름2”, “이름3”] 형태로 이름을 설정할 수 있음
- 첫 번째 행이 칼럼 이름이라면 header=0로 설정

0.0200	0.0371
0.0453	0.0523
0.0262	0.0582
0.0260	0.0363

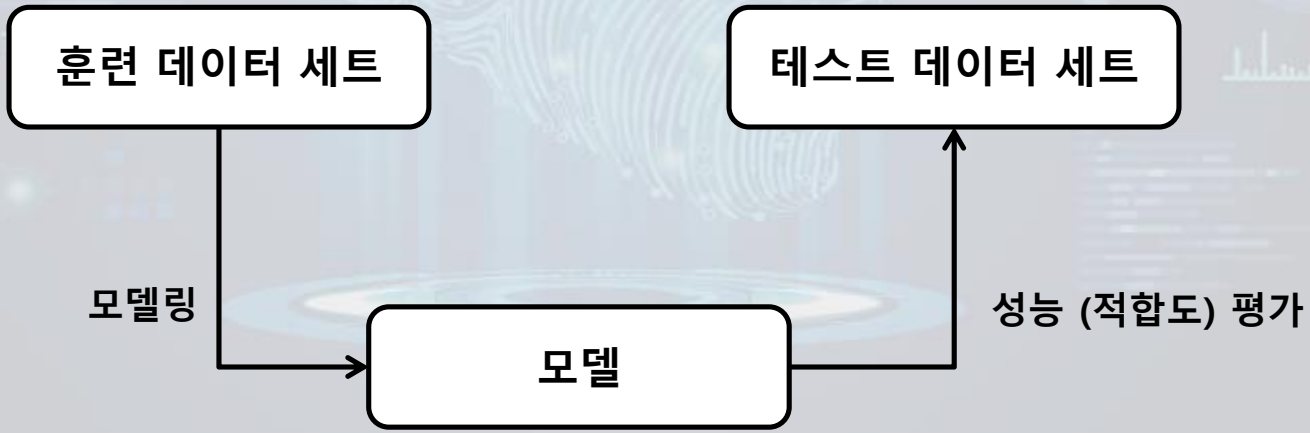
컬럼명이 없는 데이터

A	B
0.0200	0.0371
0.0453	0.0523
0.0262	0.0582
0.0260	0.0363

컬럼명이 있는 데이터

과적합 피하기 – 훈련셋과 테스트셋 나누기

- 과적합을 피하기 위해서는 훈련셋과 테스트셋을 완전히 구분하고 학습과 동시에 테스트를 병행
- 만약, 훈련셋이 총 100개의 샘플로 이루어져 있다면 70개는 훈련셋으로 30개는 테스트셋으로 사용
- 모델 : 70개의 훈련셋으로 학습한 결과로 도출



과적합 피하기 - 훈련셋과 테스트셋 나누기

- 훈련셋만 가지고 평가할 때는 층을 더하거나 에포크 값을 높여 실행횟수를 늘리면 정확도가 증가
- 훈련셋으로 학습하고 테스트셋으로 평가한다면 그대로 나타나지 않음 → 테스트셋의 결과가 낮다면 학습의 결과는 과적합이 일어났다고 할 수 있음. → 학습을 진행해도 테스트 결과가 더 이상 좋아지지 않는 지점에서 멈춰야 함.

이전까지는 훈련셋과 테스트셋을 구분하지 않았음

↓

데이터를 훈련셋과 테스트셋으로 나누고
훈련셋으로 학습하고
테스트셋으로 검증하도록 해보자.

과적합 피하기 - 훈련셋과 테스트셋 나누기

- 학습이 계속되면 학습 셋에서의 정확도는 올라가지만 테스트 셋에서는 과적합이 발생 → 학습을 진행해도 테스트 결과가 좋아지지 않는 지점에서 학습을 중지해야 함.