Undergraduate Research Internship in Affective AI LAB.

혼자 공부하는 머신러닝 & 딥러닝

1주차 : ch02. 데이터 다루기

컴퓨터공학과 12201682 공지윤

목차

기존 인공지능 모델의 문제와 해결 ①

In [7]:
```
from sklearn.neighbors import KNeighborsClassifier
```

In [4]:
```
fish_length = [25.4, 26.3, 26.5, 29.0, 29.0, 29.7, 29.7, 30.0, 30.0, 30.7, 31.0, 31.0,
               31.5, 32.0, 32.0, 32.0, 33.0, 33.0, 33.5, 33.5, 34.0, 34.0, 34.5, 35.0,
               35.0, 35.0, 35.0, 36.0, 36.0, 37.0, 38.5, 38.5, 39.5, 41.0, 41.0, 9.8,
               10.5, 10.6, 11.0, 11.2, 11.3, 11.8, 11.8, 12.0, 12.2, 12.4, 13.0, 14.3, 15.0]
fish_weight = [242.0, 290.0, 340.0, 363.0, 430.0, 450.0, 500.0, 390.0, 450.0, 500.0, 475.0, 500.0,
               500.0, 340.0, 600.0, 600.0, 700.0, 700.0, 610.0, 650.0, 575.0, 685.0, 620.0, 680.0,
               700.0, 725.0, 720.0, 714.0, 850.0, 1000.0, 920.0, 955.0, 925.0, 975.0, 950.0, 6.7,
               7.5, 7.0, 9.7, 9.8, 8.7, 10.0, 9.9, 9.8, 12.2, 13.4, 12.2, 19.7, 19.9]
```

In [5]:
```
fish_data = [[l, w] for l, w in zip(fish_length, fish_weight)]
fish_target = [1] * 35 + [0] * 14
```

In [8]:
```
kn = KNeighborsClassifier()
kn.fit(fish_data, fish_target)
```

Out[8]:
```
KNeighborsClassifier
KNeighborsClassifier()
```

In [10]:
```
kn.score(fish_data, fish_target)
```

Out[10]: 1.0

기존 인공지능 모델의 문제와 해결 ①

문제 : 훈련 데이터 그대로 테스트를 한다면 당연히 100%의 정확도를 가지게 되는 것 아닐까?

↓

해결 : 훈련에 사용하지 않은 데이터로 테스트를 해야 한다!

훈련 데이터	테스트 데이터
1번째 ~ 35번째	36번째 ~ 49번째

```
In [11]:
1 train_input = fish_data[:35]
2 train_target = fish_target[:35]
3 test_input = fish_data[35:]
4 test_target = fish_target[35:]

In [12]:
1 kn.fit(train_input, train_target)

Out[12]:
▾ KNeighborsClassifier
KNeighborsClassifier()

In [13]:
1 kn.score(test_input, test_target)

Out[13]: 0.0
```

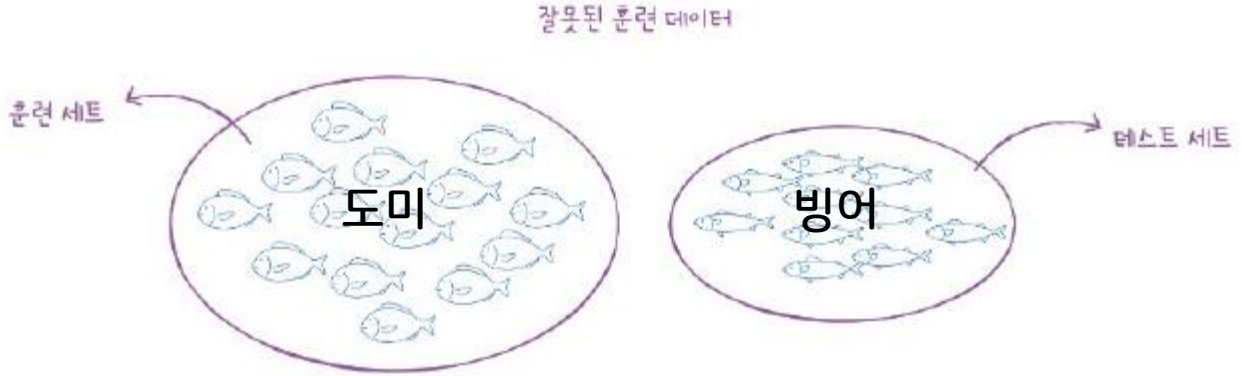
기존 인공지능 모델의 문제와 해결 ②

```
In [11]:
1 train_input = fish_data[:35]
2 train_target = fish_target[:35]
3 test_input = fish_data[35:]
4 test_target = fish_target[35:]

In [12]:
1 kn.fit(train_input, train_target)

Out[12]:
▾ KNeighborsClassifier
KNeighborsClassifier()

In [13]:
1 kn.score(test_input, test_target)

Out[13]:
0.0
```

기존 인공지능 모델의 문제와 해결 ②

문제 : 랜덤 하지 않은 데이터로 학습 시 0%의 정확도를 가진다!

A 길이의 B 무게인 물고기는 '도미' 인 것은 학습했지만, C 길이의 D 무게인 물고기가 '빙어' 인 것은 학습하지 못했다.



샘플링 편향 (Sampling Bias)

해결 : 학습 데이터와 테스트 데이터에 타깃이 랜덤하게 구성되도록 한다!

기존 인공지능 모델의 문제와 해결 ②

In [20]:
```
import numpy as np
input_arr = np.array(fish_data)
target_arr = np.array(fish_target)
```

In [22]:
```
np.random.seed(42)
index = np.arange(49)
np.random.shuffle(index)
print(index)
```

```
[13 45 47 44 17 27 26 25 31 19 12  4 34  8  3  6 40 41 46 15  9 16 24 33
 30  0 43 32  5 29 11 36  1 21  2 37 35 23 39 10 22 18 48 20  7 42 14 28
 38]
```

In [23]:
```
train_input = input_arr[index[:35]]
train_target = target_arr[index[:35]]
test_input = input_arr[index[35:]]
test_target = target_arr[index[35:]]
```

In [24]:
```
kn = kn.fit(train_input, train_target)
kn.score(test_input, test_target)
```

Out[24]: 1.0

In [25]:
```
kn.predict(test_input)
```

Out[25]: array([0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0])

In [26]:
```
test_target
```

Out[26]: array([0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0])

기존 인공지능 모델의 문제와 해결 ③

index로 학습 데이터와 테스트 데이터를 분리하지 않고, 간단하게 train_test_split()으로 나눌 수 있다.

In [27]:
```
1 import numpy as np
```

In [28]:
```
1 fish_data = np.column_stack((fish_length, fish_weight))
2 fish_target = np.concatenate((np.ones(35), np.zeros(14)))
```

In [29]:
```
1 from sklearn.model_selection import train_test_split
2 train_input, test_input, train_target, test_target = train_test_split(fish_data, fish_target, random_state=42)
```

In [47]:
```
1 np.unique(fish_target, return_counts=True)
```

Out[47]: (array([0., 1.]), array([14, 35], dtype=int64)) 원본 데이터 - 1 : 2.5

In [46]:
```
1 np.unique(test_target, return_counts=True)
```

Out[46]: (array([0., 1.]), array([3, 10], dtype=int64)) 테스트 데이터 - 1 : 3.3

기존 인공지능 모델의 문제와 해결 ③

문제 : 원본 데이터와 분리된 데이터의 샘플링 편향이 발생한다!

↓

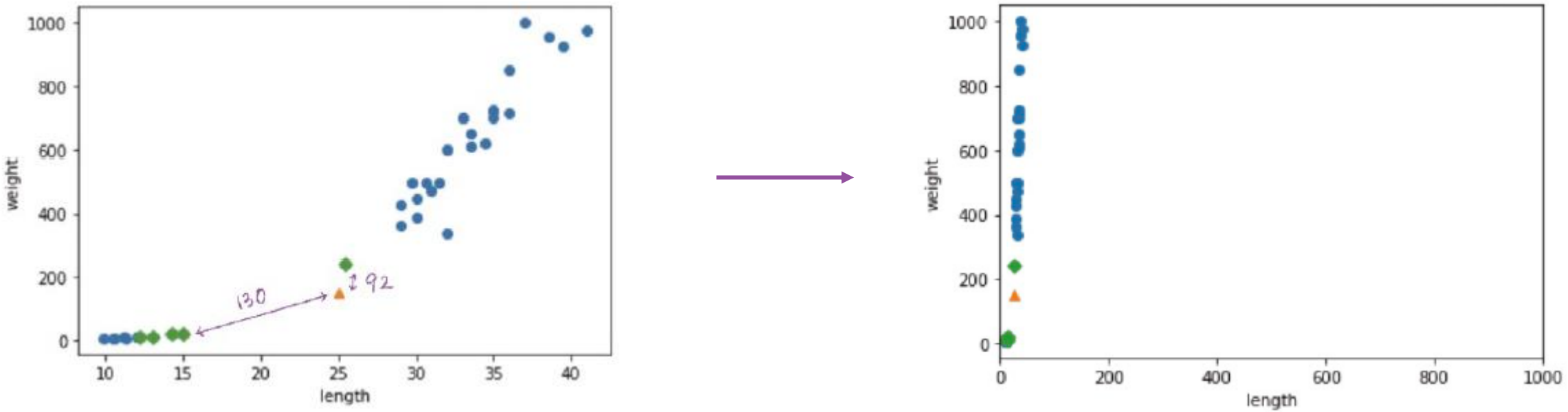
해결 : {stratify = 원본타깃데이터} 옵션을 준다!

```
In [48]: 1 train_input, test_input, train_target, test_target = train_test_split(fish_data, fish_target, stratify=fish_target, random_state=42)

In [49]: 1 np.unique(test_target, return_counts=True)

Out[49]: (array([0., 1.]), array([4, 9], dtype=int64))
```

기존 인공지능 모델의 문제와 해결 ④

문제 : 피처 간의 스케일이 달라서 분류의 정확도를 떨어뜨린다!

해결 : 피처들의 스케일을 똑같이 맞춰준다!




```
In [58]: 1 train_target[indexes]

Out[58]: array([[1., 0., 0., 0., 0.]])
```

기존 인공지능 모델의 문제와 해결 ④

중요한 점 : 학습 데이터의 평균과 표준편차로 테스트 데이터를 정규화해야 한다.

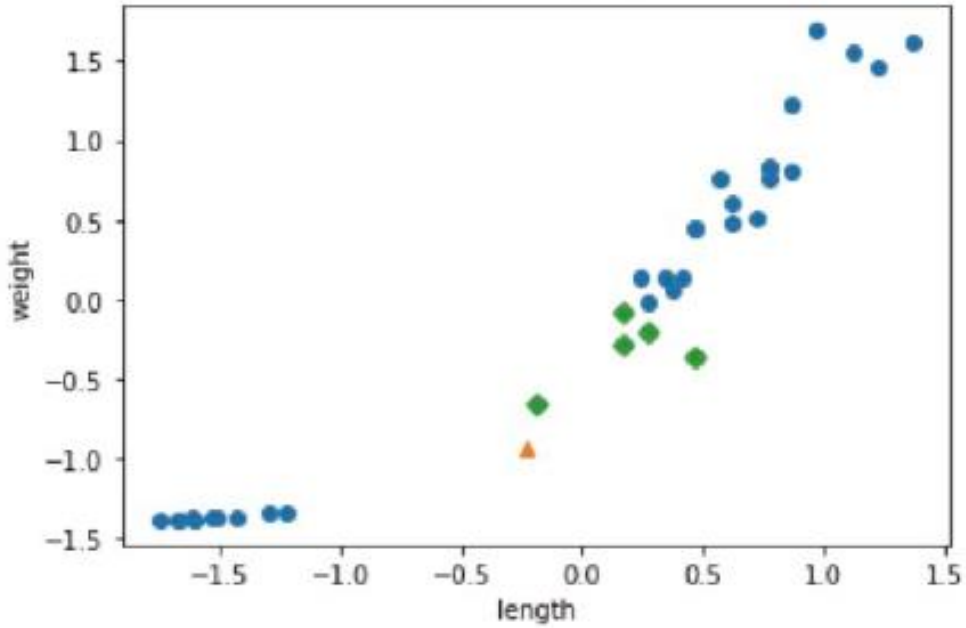
$$Z = \frac{X - m}{\sigma}$$ z 표준화

```
In [59]:
1 mean = np.mean(train_input, axis=0)
2 std = np.std(train_input, axis=0)
```

```
In [60]:
1 train_scaled = (train_input - mean) / std
2 test_scaled = (test_input - mean) / std
```

```
In [61]:
1 kn.fit(train_scaled, train_target)
2 kn.score(test_scaled, test_target)
```

```
Out[61]: 1.0
```




```
In [63]:
1 train_target[indexes]
```

```
Out[63]: array([[1., 1., 1., 1., 1.]])
```

Summary

1. 훈련에 사용하지 않은 데이터로 테스트를 해야 한다.

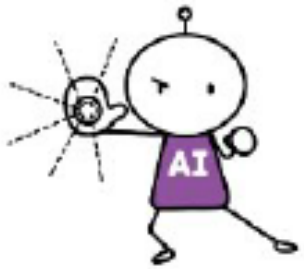
2. 학습 데이터와 테스트 데이터에 타깃이 랜덤하게 구성되도록 한다.

3. train_test_split에 {stratify = 원본타깃데이터} 옵션을 주어서 샘플링 편향을 최소화한다.

4. 표준화를 통해 피처들의 스케일을 똑같이 맞춰준다.
이때, 학습 데이터의 평균과 표준편차로 테스트 데이터를 표준화한다.

Undergraduate Research Internship in Affective AI LAB.

감사합니다 :>



컴퓨터공학과 12201682 공지윤