기계학습 2주차  
  
#01-1 인공지능과 머신러닝, 딥러닝  
  
**인공지능(AI)**은 사람처럼 학습하고 추론할 수 있는 지능을 가진 컴퓨터 시스템을 개발하는 기술이다.

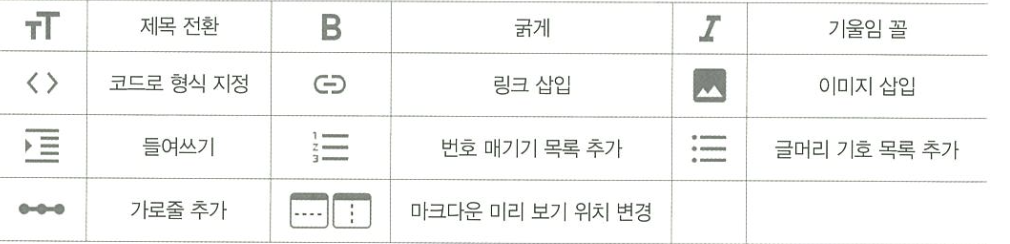
* **초기 역사**
  + 1943년 워런 매컬록과 월터 피츠가 뇌 뉴런 개념을 발표.
  + 1950년 앨런 튜링이 튜링 테스트를 제안.
  + 1956년 다트머스 AI 컨퍼런스에서 인공지능 연구가 본격화.
* **황금기와 한계**
  + 1957년 프랑크 로젠블라트가 퍼셉트론(초기 형태의 로지스틱 회귀) 발표.
  + 1959년 허블과 비셀이 시각 피질 뉴런 연구로 노벨상 수상.
  + 하지만 컴퓨터 성능의 한계로 인해 첫 번째 AI 겨울이 시작되었고, 이후 전문가 시스템이 등장하며 두 번째 AI 붐과 겨울이 반복됨.
* **현대와 대중화**
  + AI 연구는 여러 어려움을 극복하며 발전해 대중의 관심을 받기 시작.
  + 영화와 드라마에서는 지능적인 시스템이 자주 등장하지만, 실제 기술은 영화 속 AI와는 차이가 있음.

**머신러닝(Machine Learning)**은 규칙을 프로그래밍하지 않아도 자동으로 데이터에서 규칙을 학습하는 알고리즘을 연구하는 분야이며, 사이킷런이 대표적인 라이브러리이다.

**딥러닝(Deep Leaming)**은 인공 신경망이라고도 하며 , 텐서플로와 파이토치가 대표적인 라이브러리이다.  
  
  
#01-2 코랩과 주피터 노트북

* **코랩**은 구글 계정이 있으면 누구나 사용할 수 있는 웹 브라우저 기반의 파이썬 코드 실행 환경이다.
* **노트북**은 코랩의 프로그램 작성 단위이며 일반 프로그램 파일과 달리 대화식으로 프로그램을 만들 수 있기 때문에 데이터 분석이나 교육에 매우 적합합니다. 노트북에는 코드 , 코드의 실행 결과 , 문서를 모두 저장하여 보관할 수 있다.
* **구글 드라이브**는 구글이 제공하는 클라우드 파일 저장 서비스이다. 코랩에서 만든 노트북은 자동으로 구글 클라우드의 Colab Notebooks' 폴더에 저장되고 필요할 때 다시 코랩에서 열 수 있다.

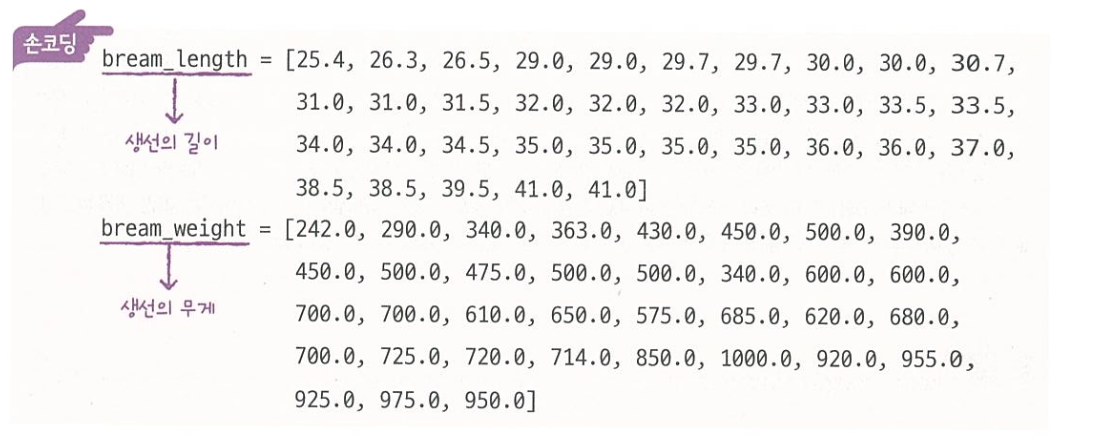
텍스트 셀 툴바



텍스트 셀에 사용할 수 있는 마크다운

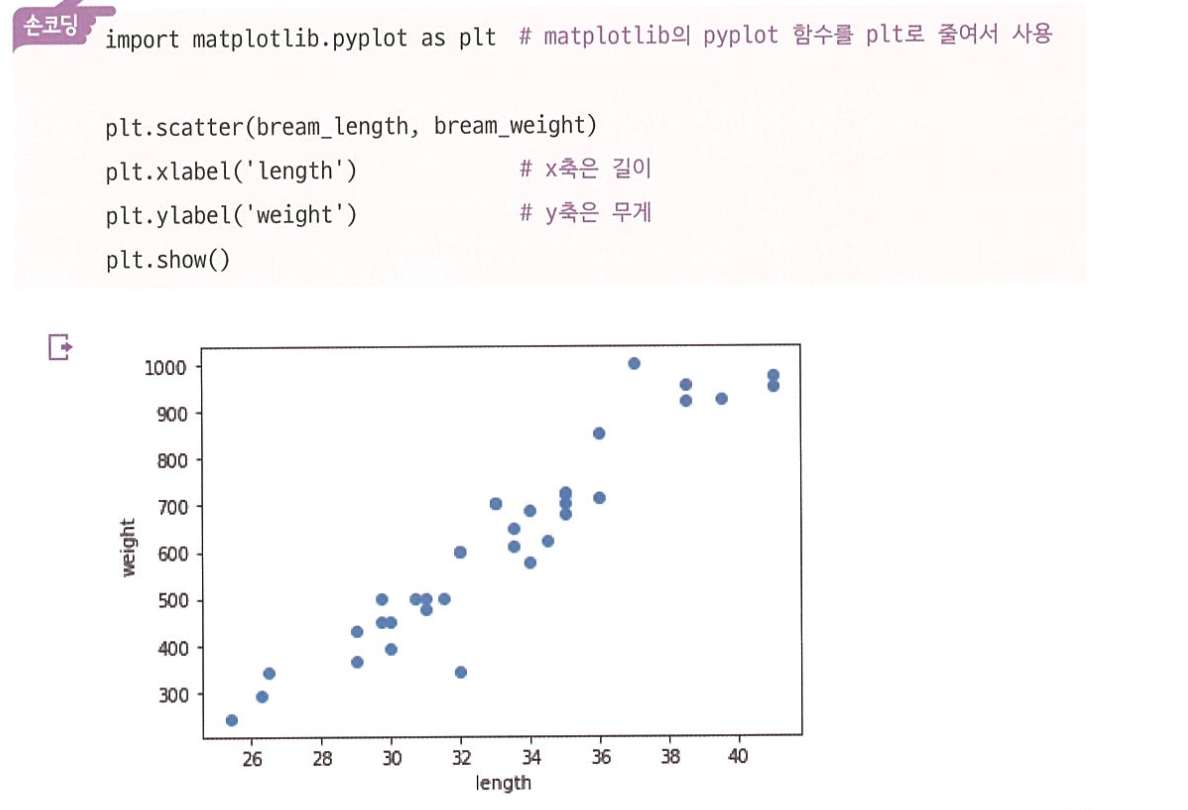
#01-3 마켓과 머신러닝

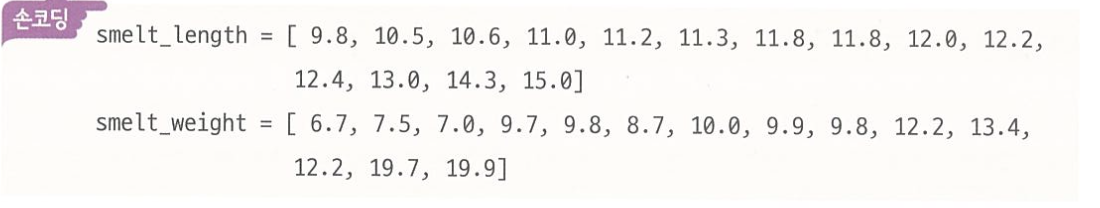
생선 분류 문제  
: 생선의 이름 확인 문제 해결을 위해

1. 도미 데이터 준비

(도미의 길이와 무게를 특성이라고 한다)

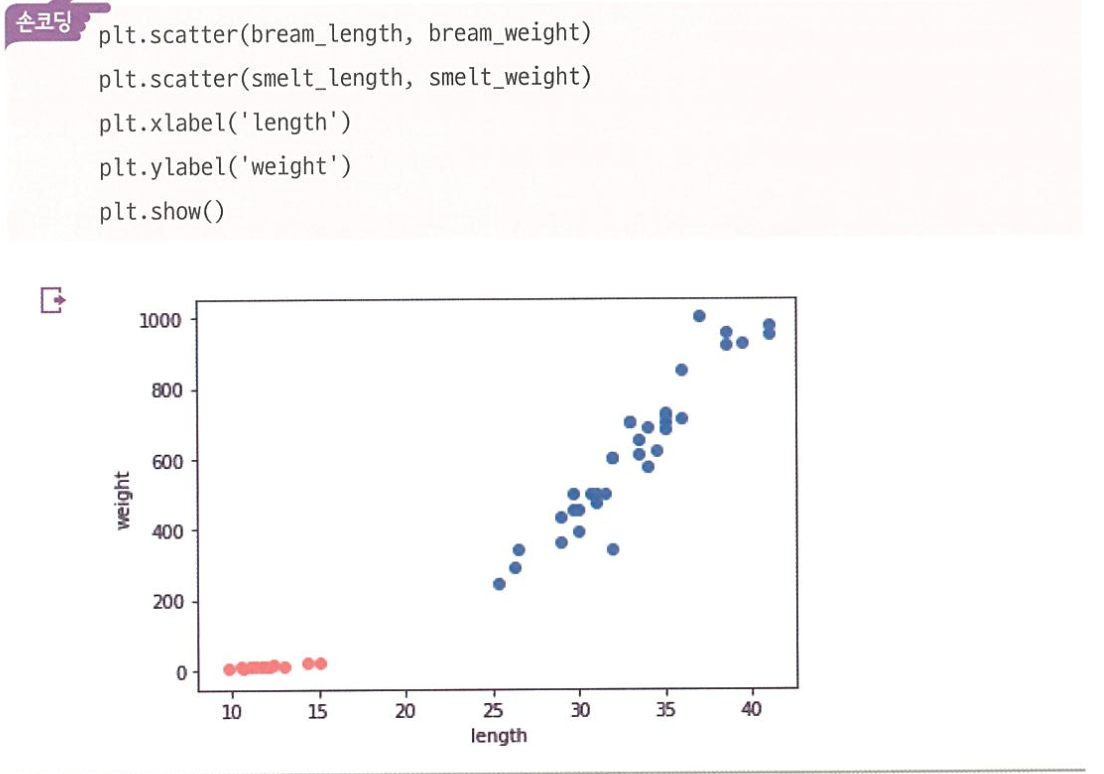
이러한 특성을 그래프에서 점으로 표시하여 산점도 그래프를 만든다.

(파이썬에서 과학계산용 그래프를 그리는 대표적인 패키지는 맷플롯립)이런 식으로 산점도 그래프가 일직선에 가까운 형태로 나타나는 경우를 **선형**적이라고 한다.

1. 방어 데이터 준비

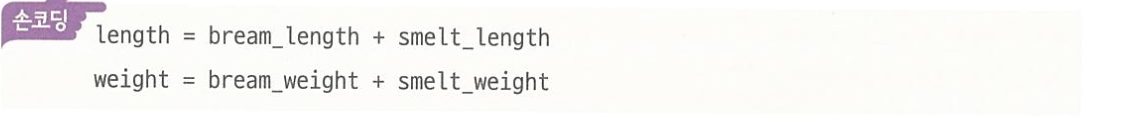
도미 데이터와 동일하게 방어의 특성을 그래프로 그린다

(차이를 확실히 볼 수 있도록 하나의 그래프에 도미와 방어의 데이터를 함께 그린다)



K- 최근접 이웃 알고리즘

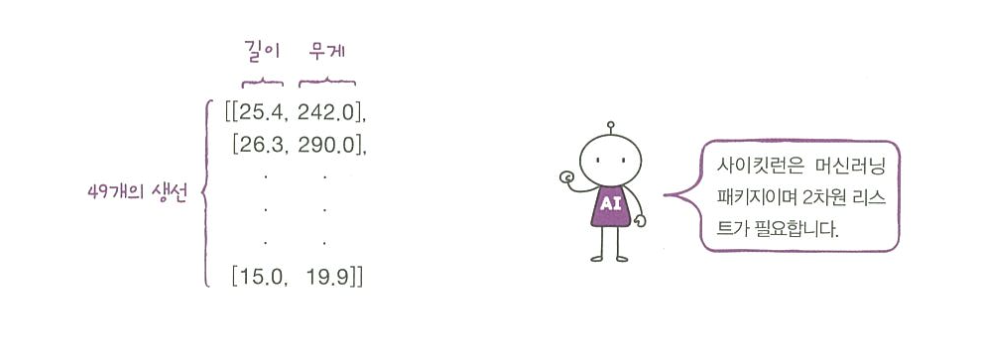
앞에서 준비했던 두 리스트를 더하여 아래와 같이 하나의 리스트로 만든다



하나로 합쳐진 ngth 와 weight 리스트는 다음과 같다.

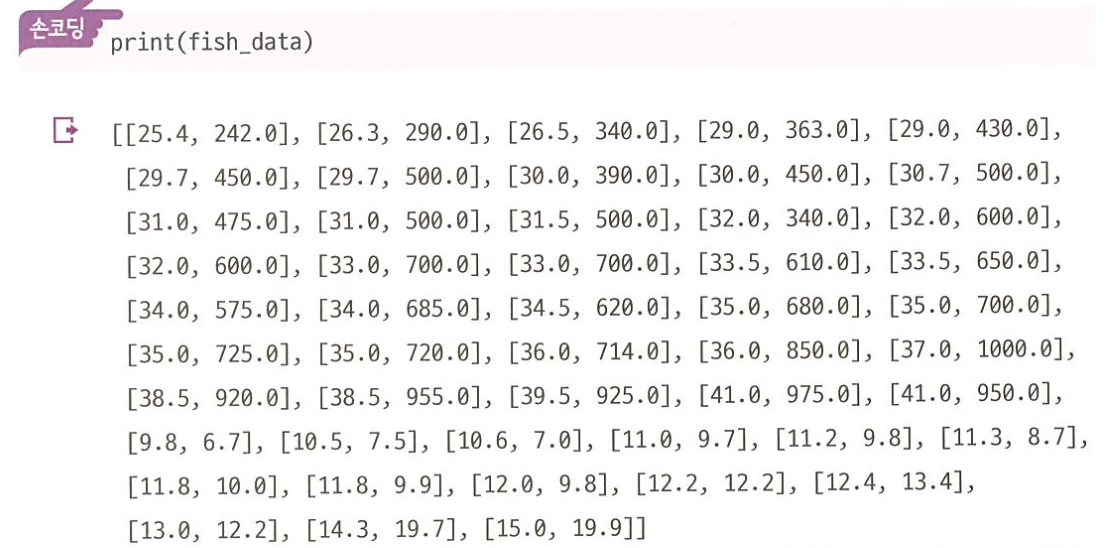


사이킷런 패키지를 사용하기 위해 세로 방향으로 늘어뜨린 2차원 리스트로 만든다.



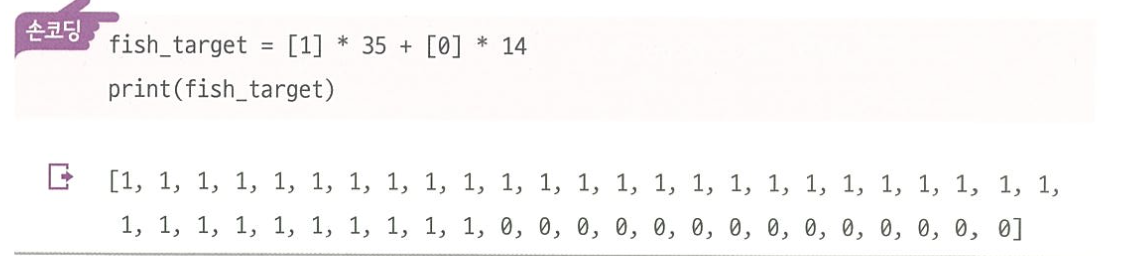
zip() 함수와 리스트 내포 구문을 사용해 length와 weight 리스트를 2 차원 리스트로 만든다.



출력하여 확인해보면 아래와 같은 값이 나온다  
  


첫번째 생선의 길이 25.4cm 와 무게 242.0g 이 하나의 리스트를 구성하고 이런 리스트가 모여 전체

리스트를 만들었다.. 이런 리스트를 2차원 리스트 혹은 리스트의 리스트라고 부른다.

이제 각각 어떠한 종류의 생선인지 답을 만들어야 한다. 컴퓨터 프로그램은 문자를 직접 이해하지 못하므로 도미를 숫자 0, 방어를 숫자 1로 표현하면 아래와 같은 출력값을 얻을 수 있다.  
  


이제 사이킷런 패키지에 k-최근접 이웃 알고리즘을 구현한 KNeighborsClassifier를 임포트 한다.  
  


임포트한 KNeighborsClassifier 클래스의 객체를 만든다.

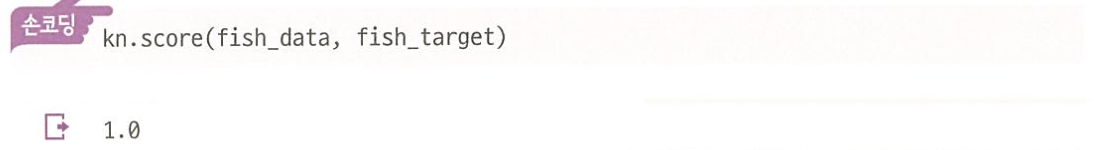


이 객체에 fish\_data 와 fish\_arget 을 전달하여 도미를 찾기 위한 기준을 학습시킨다. 이런 과정

을 머신러닝에서는 **훈련**이라고 부른다. 사이킷런에서는 fit() 모델에 데이터를 전달하여 메서드가 이런 역할을 한다.

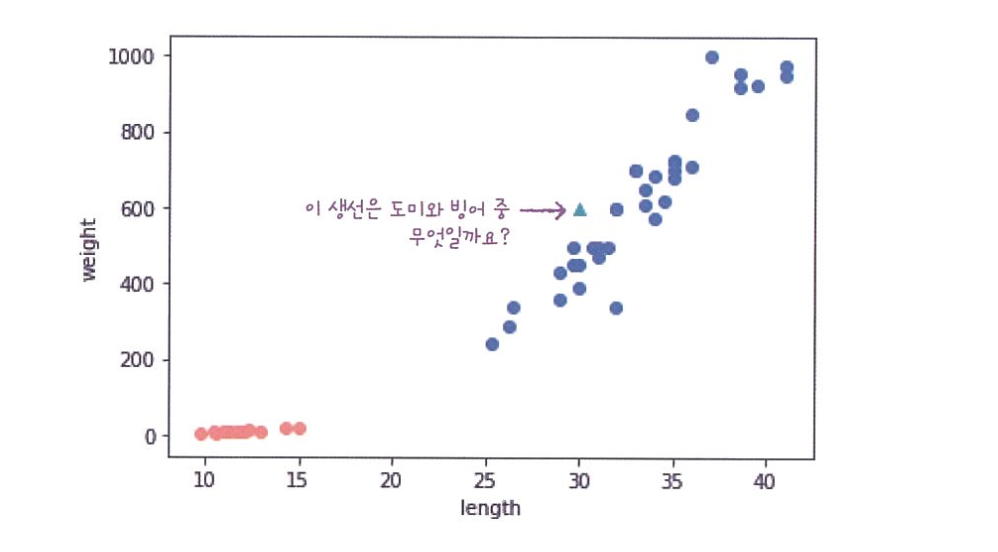


이제 객체 ( 또는 모델 ) kn이 얼마나 잘 훈련되었는지 평가를 한다. 메서드는 0 에서 1 사이의 값을 반환하며 1은 모든 데이터를 정확히 맞혔다는 것을 나타낸다. 예를 들어 0.5라면 절반을 맞혔다는 의미이다.



이처럼 1이 나오면 이 모델의 정확도가 100%라고 할 수 있다.

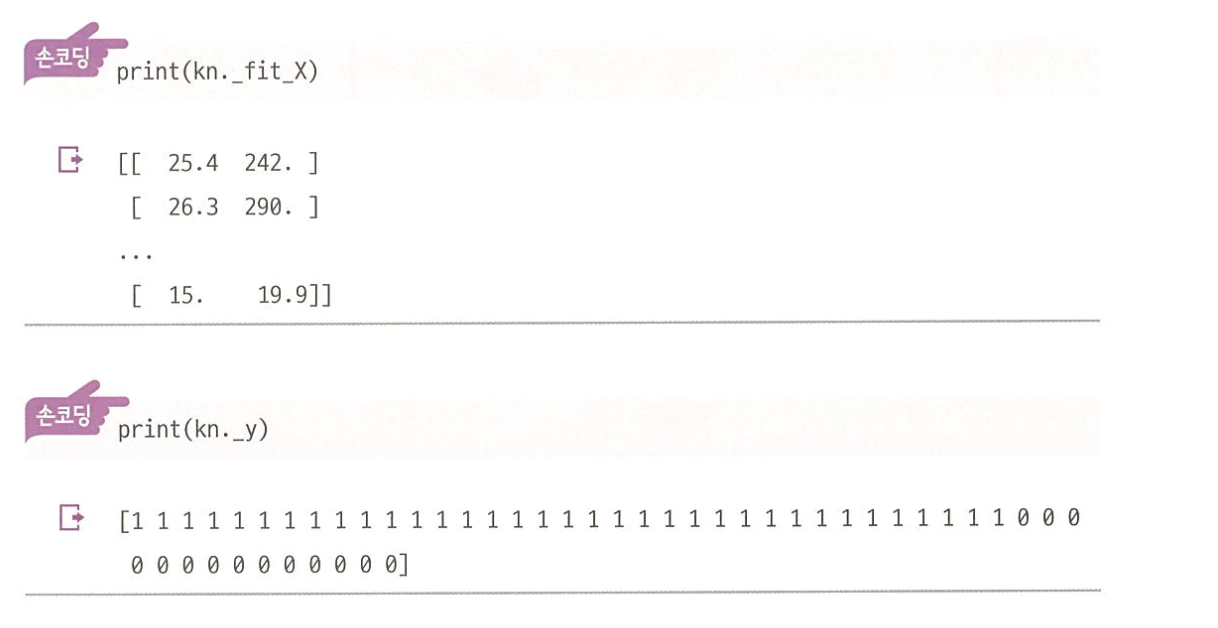
만약 새로운 데이터가 아래처럼 삼각형으로 되어있다면 이 삼각형이 도미인지 빙어인지 어떻게 알 수 있을까?



대부분의 사람들이 삼각형 주변에 다른 도미 데이터가 많기 때문에 삼각형 데이터는 도미라고 판단할 것이다. 위의 프로그램에서 사용된 k- 최근접 이웃 알고리즘도 마찬가지로 삼각형 주위에 도미 데이터가 많으므로 삼각형을 도미라고 판단할 것이다. 실제로 그런지 확인하기 위해 아래와 같은 코드를 실행해보면  
  


이러한 값이 출력된다. 앞서 도미는 1, 방어는 0으로 가정했었기에 array([1])이라는 출력값은 도미를 나타낸다는 걸 알 수 있다.

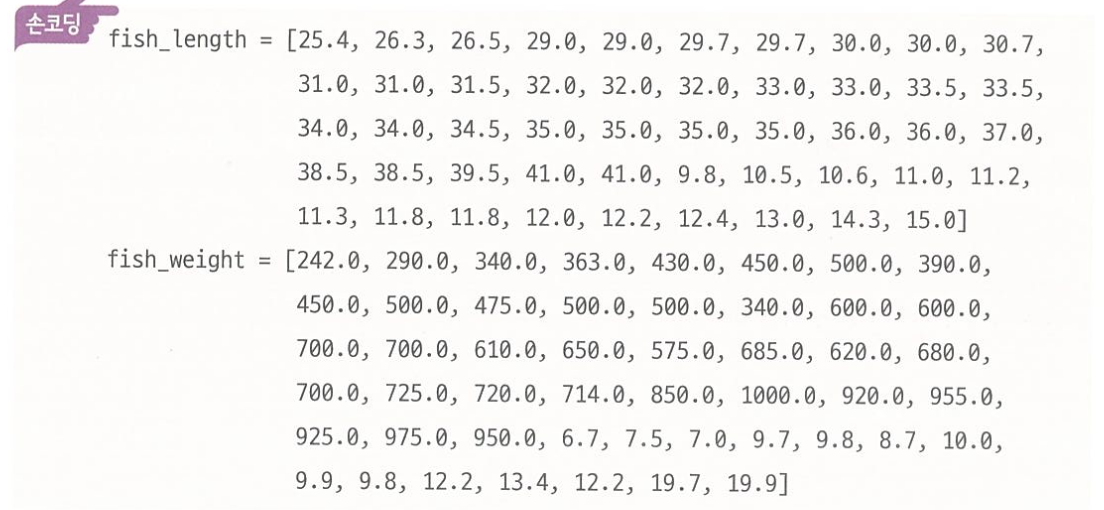
k - 최근접 이웃 알고리즘을 위해 준비해야 할 일은 데이터를 모두 가지고 있는 게 전부라는 장점이 있다. 하지만 이러한 특징은 메모리를 많이 차지하고 계산하는 데도 많은 시간이 소요된다는 단점도 존재한다. 사이킷런의 KNeighborsClassifier 클래스도 마찬가지이다.



#02-1 훈련 세트와 테스트 세트

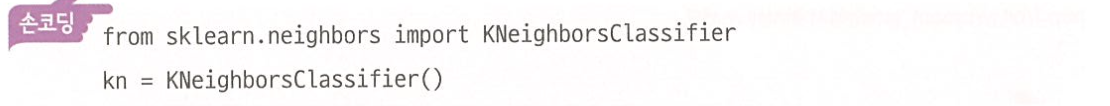
* **지도학습(Supervised Learning)**: 입력 데이터와 해당 정답(레이블)을 이용해 예측 모델을 학습하는 머신러닝 방법
* **비지도학습(Unsupervised Learning)**: 정답(레이블) 없이 입력 데이터 간의 패턴이나 구조를 발견하는 머신러닝 방법
* **훈련 세트(Training Set)**: 머신러닝 모델을 학습시키기 위해 사용되는 데이터셋
* **테스트 세트(Test Set)**: 학습된 모델의 성능을 평가하기 위해 사용되는 데이터셋

먼저 1장에서처럼 도미와 빙어의 데이터를 합쳐 하나의 파이썬 리스트로 준비한다.

이제 두 파이썬 리스트를 순회하면서 각 생선의 길이와 무게를 하나의 리스트로 담은 2차원 리스트를 만든다.



이때 하나의 생선 데이터를 **샘플**이라고 부른다. 도미와 빙어는 각각 35마리, 14마리가 있으므로 전체 데이터는 49개의 샘플이 있는 것이다. 사용하는 특성은 길이와 무게 2개이며 처음 35개를 훈련 세트로, 나머지 14개를 테스트 세트로 사용할 것이다.



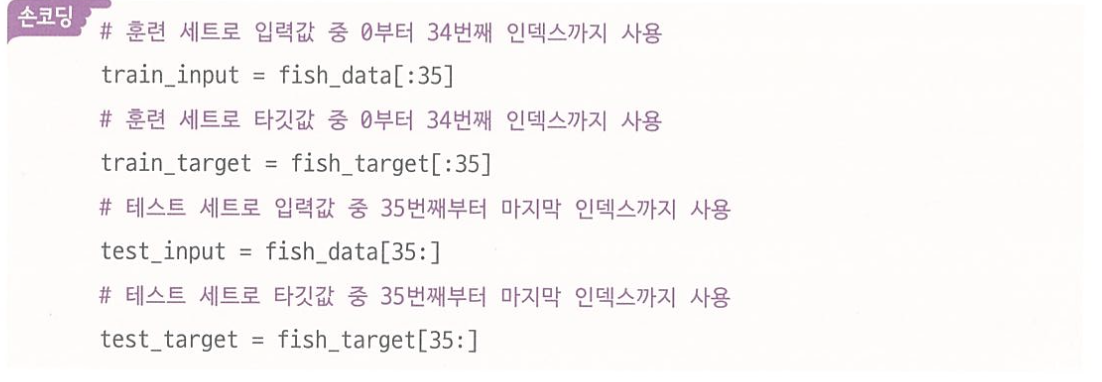
1장에서 했던 것과 동일하게 사이킷런의 KNeighborsClassifier 클래스를 임포트하고 모델 객체를 만든다.

그리고 전체 데이터 중 처음 35개 데이터를 선택한다.

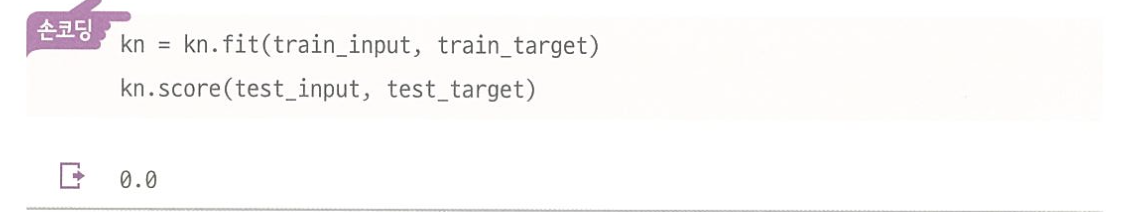
(인덱싱)

인덱싱


(슬라이싱)



데이터를 준비했으니 이제 훈련 세트로 fit() 메서드를 호출해 모델을 훈련하고 , 테스트 세트로 score() 메서드를 호출해 평가한다.



나쁜 모델이 아님은 확실하지만 정확도가 0이 나오는 것은 어떠한 문제가 있다는 것을 알 수 있다. 전체 49개의 데이터는 35개의 도미와 14개의 빙어가 순서대로 들어가있어 처음부터 35개의 샘플을 훈련 세트, 14개의 샘플을 테스트 세트로 두면 각각 훈련 세트에는 빙어 데이터가 없고, 테스트 세트에는 도미 데이터가 없게 된다. 이런식으로 샘플링이 한쪽으로 치우쳐있는 것을 샘플링 편향이라고 한다.  
  
혼공머신이 훈련 세트와 테스트 세트를 나누기 전에 데이터를 섞거나 골고루 샘플을 뽑아서 만들어야 하는데 이러한 작업을 간편하게 처리할 수 있는 넘파이 라이브러리가 있다.

* **넘파이** : 대규모 다차원 배열과 행렬 연산을 효율적으로 처리하기 위한 라이브러리

넘파이 라이브러리 임포트

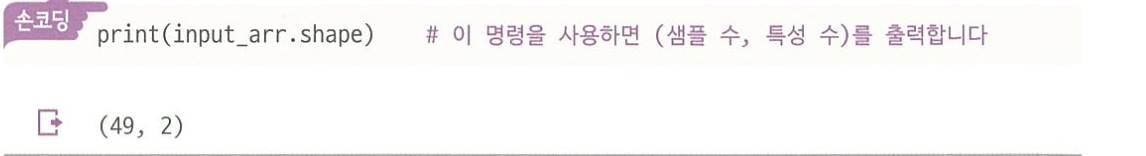


그리고 파이썬 리스트를 넘파이 배열로 바꾼다.  
  


nput\_alr 배열을 출력해 보면 아래와 같은 값이 나온다.

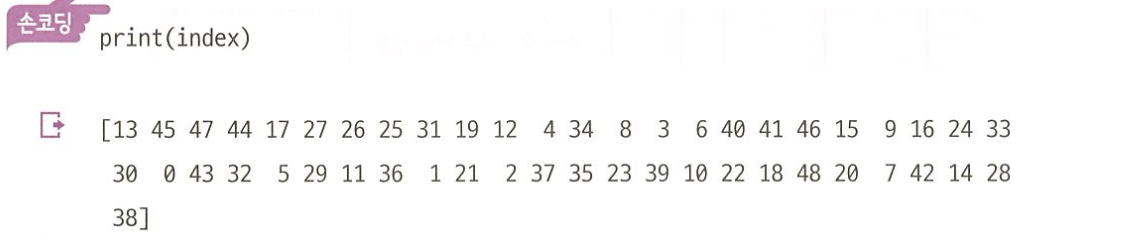


(샘플 수와 특성 수 확인)



넘파이 arange() 함수를 사용하면 0 에서부터 48 까지 1 씩 증가하는 인덱스를 간단히 만들 수 있다. 그다음 이 인덱스를 랜덤하게 섞는다.



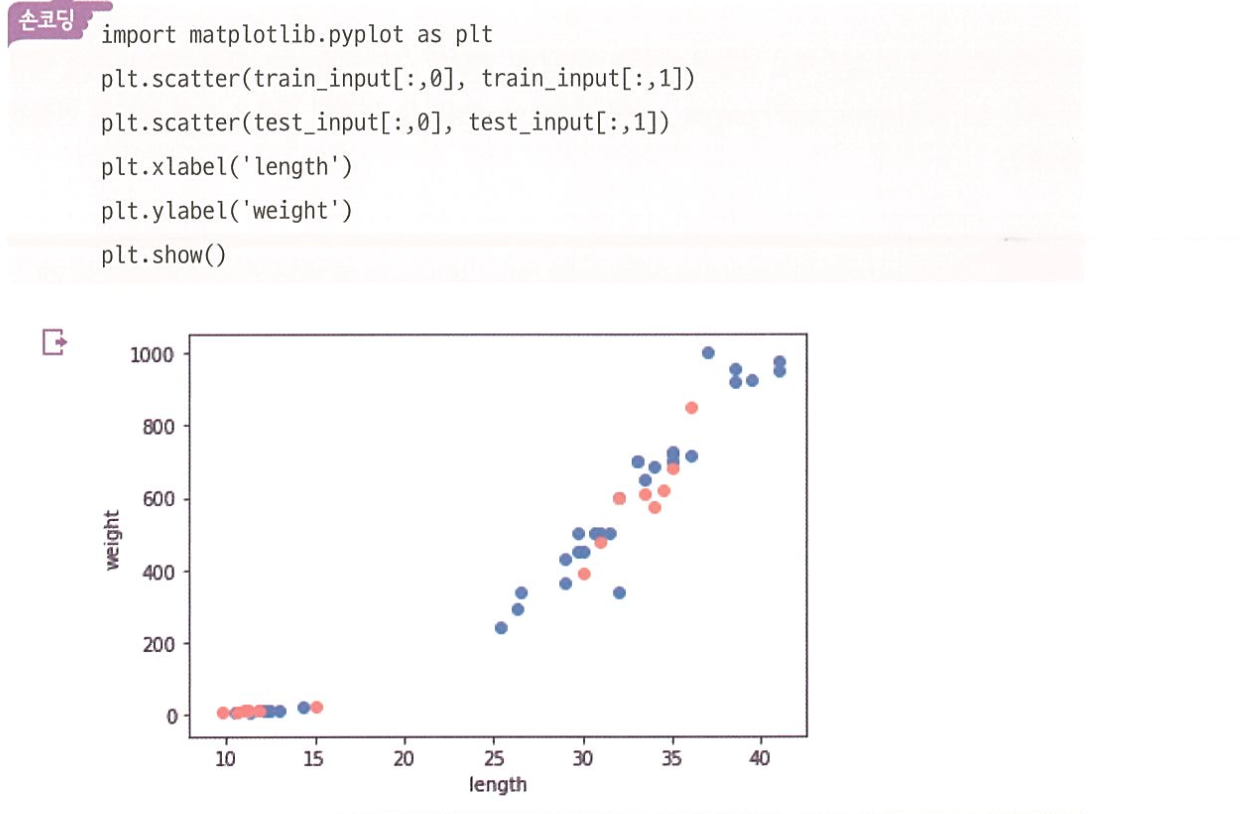
만들어진 인덱스를 출력해보면 아래와 같다.  
  


0부터 48까지 정수가 잘 섞인 것을 볼 수 있다. 이제 랜덤하게 섞인 인덱스를 사용해 전체 데이터를 훈련 세트와 테스트 세트로 나눈다.  
(배열 인덱싱)

비슷한 방식으로 리스트 대신 넙파이 배열을 인덱스로 전달할 수도 있다.  


이번에는 14개의 테스트 샘플을 만든다.  
  


이제 훈련 세트와 테스트 세트에 도미와 빙어가 잘 섞였는지 확인하기 위해 산점도를 그린다.



이제 k- 최근접 이웃 모델을 훈련시킨다.

fit() 메서드를 실행 할 때마다 KNeighborsClassifier 클래스의 객체는 이전에 학습한 모든 것을 잃어버린다. 이전 모델을 그대로 두고 싶다면 KNeighborsClassifier 클래스 객체를 새로 만들어야 한다.



인덱스를 섞어 만든 train\_input과 train\_target으로 모델을 훈련시켰으므로 다음은 test\_input

과 test\_arget으로 이 모델을 테스트 한다.



정확도가 100%임을 알 수 있다.