KNN의 적용

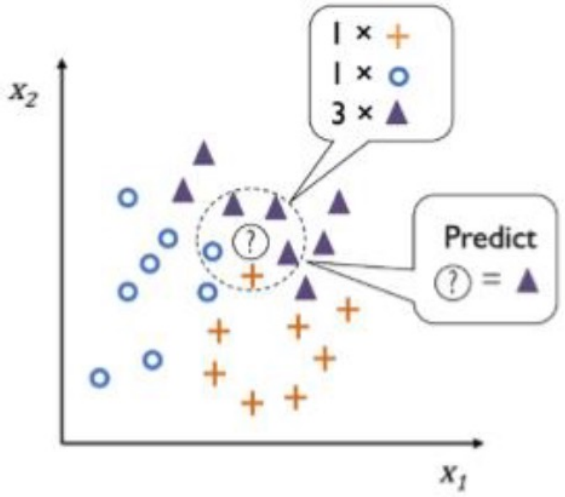
Regression

KNN 회귀 정의

- KNN 회귀(regression)도 KNN 분류(classification)과 동일
  - y 의 예측 치 계산만 다름

- K개 관측치 (x_i, y_i)에서 $\bar{y}$를 계산하여 적합 치로 사용

- 주어진 특성 변수 x에 대응하는 y의 예측 치
  - $\frac{1}{k}\sum_{i=1}^{k} y_i$ 단, y_i는 x에 가장 가까운 K 개의 학습 데이터 y



KNN 회귀

- **단순 회귀 란?**
  - 가까운 이웃들의 단순한 평균을 구하는 방식

- **가중 회귀(Weighted regression) 란?**
  - 각 이웃이 얼마나 가까이 있는지에 따라 가중 평균(weighted average)을 구해 거리가 가까울수록 데이터가 더 유사할 것이라고 보고 가중치를 부여하는 방식

- **예시)**

영화 X의 등급을 찾기 위해 3-NN 검색 결과		
영화 A	등급: 5.0	X까지의 거리: 3.2
영화 B	등급: 6.8	X까지의 거리: 11.5
영화 C	등급: 9.0	X까지의 거리: 1.1

  - **단순 평균: 6.93, 가중 평균: 7.9 ➔**

$$\frac{\frac{5.0}{3.2}+\frac{6.8}{11.5}+\frac{9.0}{1.1}}{\frac{1}{3.2}+\frac{1}{11.5}+\frac{1}{1.1}}=7.9$$

KNN 회귀 실습

- KNN 회귀를 이용한 영화 평점 예측
  - 평이 좋다” vs “평이 나쁘다” 레이블로 분류하는 게 아니라 실제 IMDb* 등급(별점)을 예측하는 것.

*IMDb 란? 인터넷영화데이터베이스

```
1
2 from sklearn.neighbors import KNeighborsRegressor
3
4 regressor = KNeighborsRegressor(n_neighbors = 3, weights = "distance")
5
6 training_points = [
7   [0.5, 0.2, 0.1],
8   [0.9, 0.7, 0.3],
9   [0.4, 0.5, 0.7]
10 ]
11
12 training_labels = [5.0, 6.8, 9.0]
13 regressor.fit(training_points, training_labels)
14
15
16 unknown_points = [
17   [0.2, 0.1, 0.7],
18   [0.4, 0.7, 0.6],
19   [0.5, 0.8, 0.1]
20 ]
21
22 guesses = regressor.predict(unknown_points)
23 guesses
```

“distance” ➔ 가중평균,
default = ”uniform”

```
array([7.28143288, 7.76451922, 6.8457845 ])
```