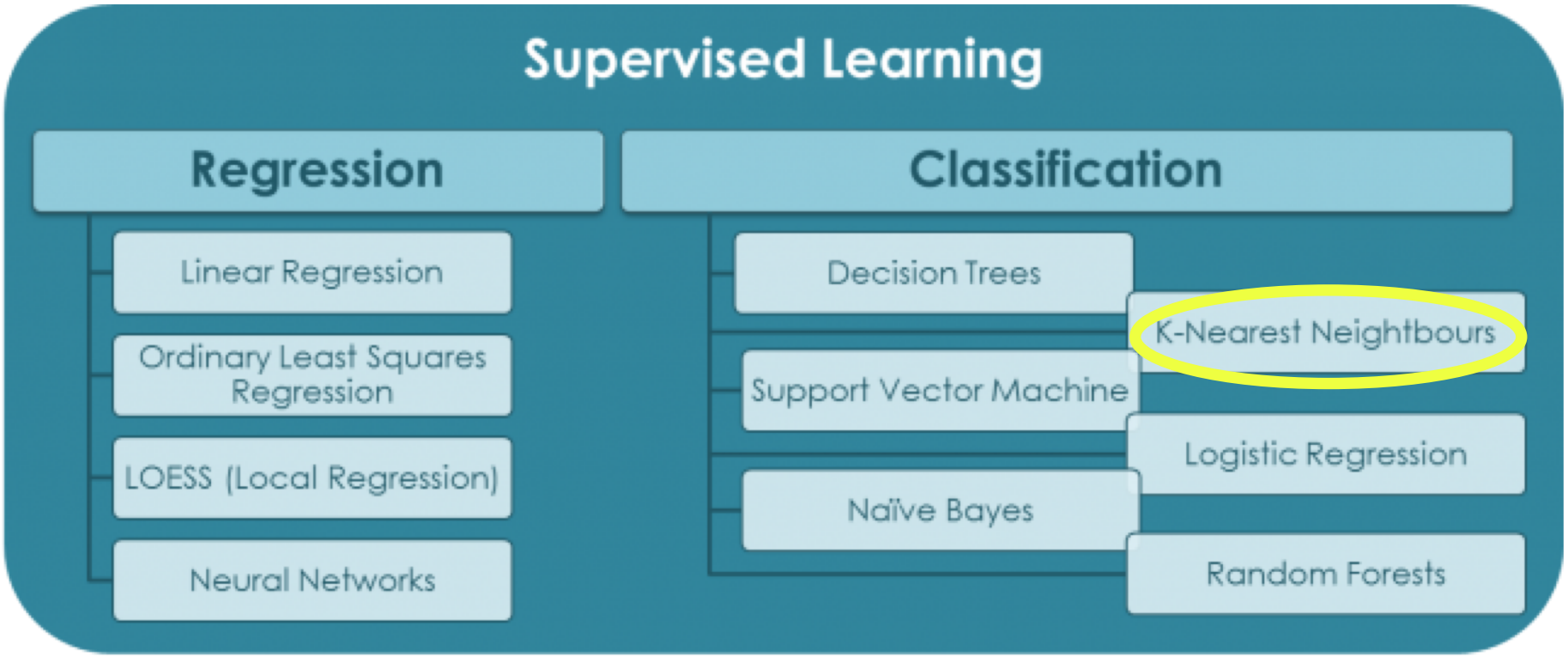
Ⅲ. 지도학습 (Supervised Learning)

- Classification

❶ KNN

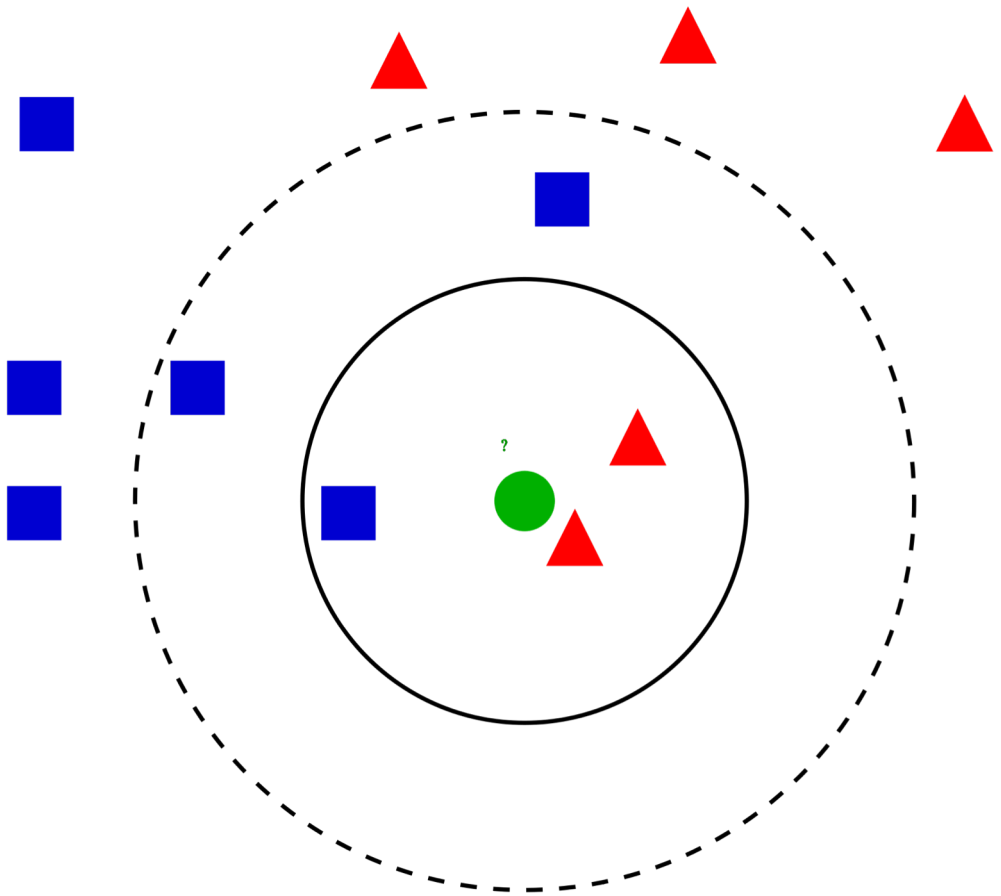

Supervised Learning
Regression
Linear Regression
Ordinary Least Squares Regression
LOESS (Local Regression)
Neural Networks
Classification
Decision Trees
K-Nearest Neightbours
Support Vector Machine
Logistic Regression
Naïve Bayes
Random Forests

KNN이란?

- **K-Nearest Neighbors algorithm (KNN)**은 구현하기 어렵지 않으면서도 비교적 나쁘지 않은 성능을 내는 Classification Algorithm 중 하나
- KNN은 가까운 데이터는 같은 label일 가능성이 크다고 가정하고 새로운 데이터가 들어오면, 그 데이터와 가장 가까운 k개의 데이터를 training set에서 뽑는다.
- 뽑은 k개의 데이터들의 label을 관측하고 그 중 가장 많은 label을 새로운 데이터의 label로 assign하는 알고리즘이다 (이런 방식을 majority voting이라고 한다).
- 이때 '가까움' 은 Euclidean distance로 측정해도 되고, 다른 metric이나 measure를 사용해도 된다.
- 이때 distance 혹은 similarity를 측정하기 위해서 반드시 metric을 사용해야하는 것은 아니다. 즉, metric의 세 가지 성질을 만족하지 않는 measure일지라도 두 데이터가 얼마나 '비슷하냐' 를 measure할 수 있는 measure라면 KNN에 적용할 수 있다

KNN이란?

- 아래 그림은 KNN이 어떻게 동작하는지 알 수 있는 간단한 예시이다. 아래 그림을 보면 k를 3으로 골랐을 때 초록색 데이터의 label은 빨강이 되고, k를 5로 골랐을 때는 파란색이 됨을 알 수 있다.



KNN (K-nearest Neighbor)

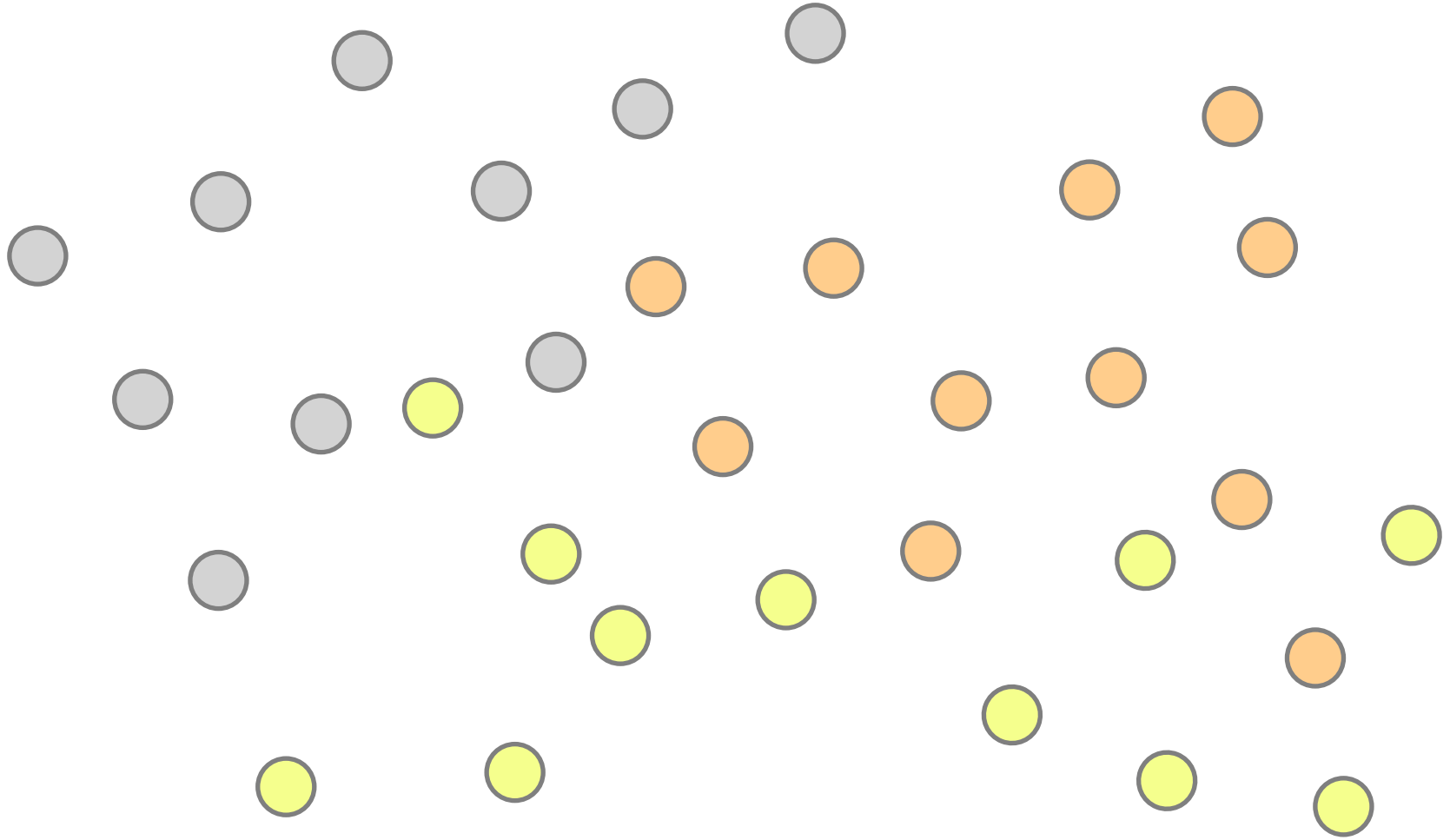
Nearest Neighbor ?
가장 가까운 이웃 ?

특정 데이터의 범주를 알지 못하는 경우, 해당 데이터가 가진
특성을 바탕으로 어떤 class 혹은 어떤 segment에 속할지
판단하는 방법

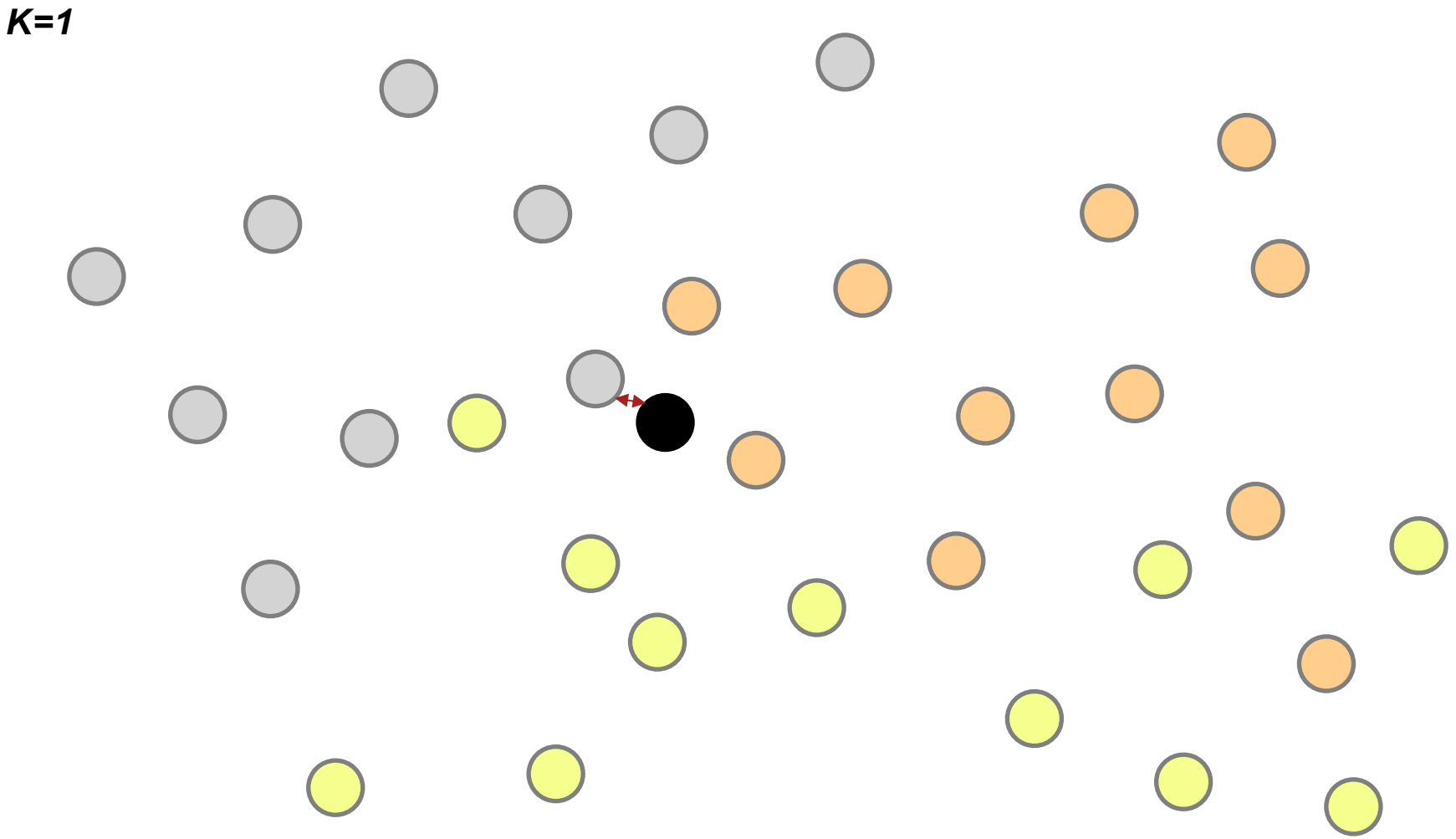
가령, 클러스터 분석 후 New data 추가 시 해당 데이터가
어느 클러스터에 속하는지 예측

여기서 k 는 가장 가까운 이웃의 수를 몇명으로 할 것인가임

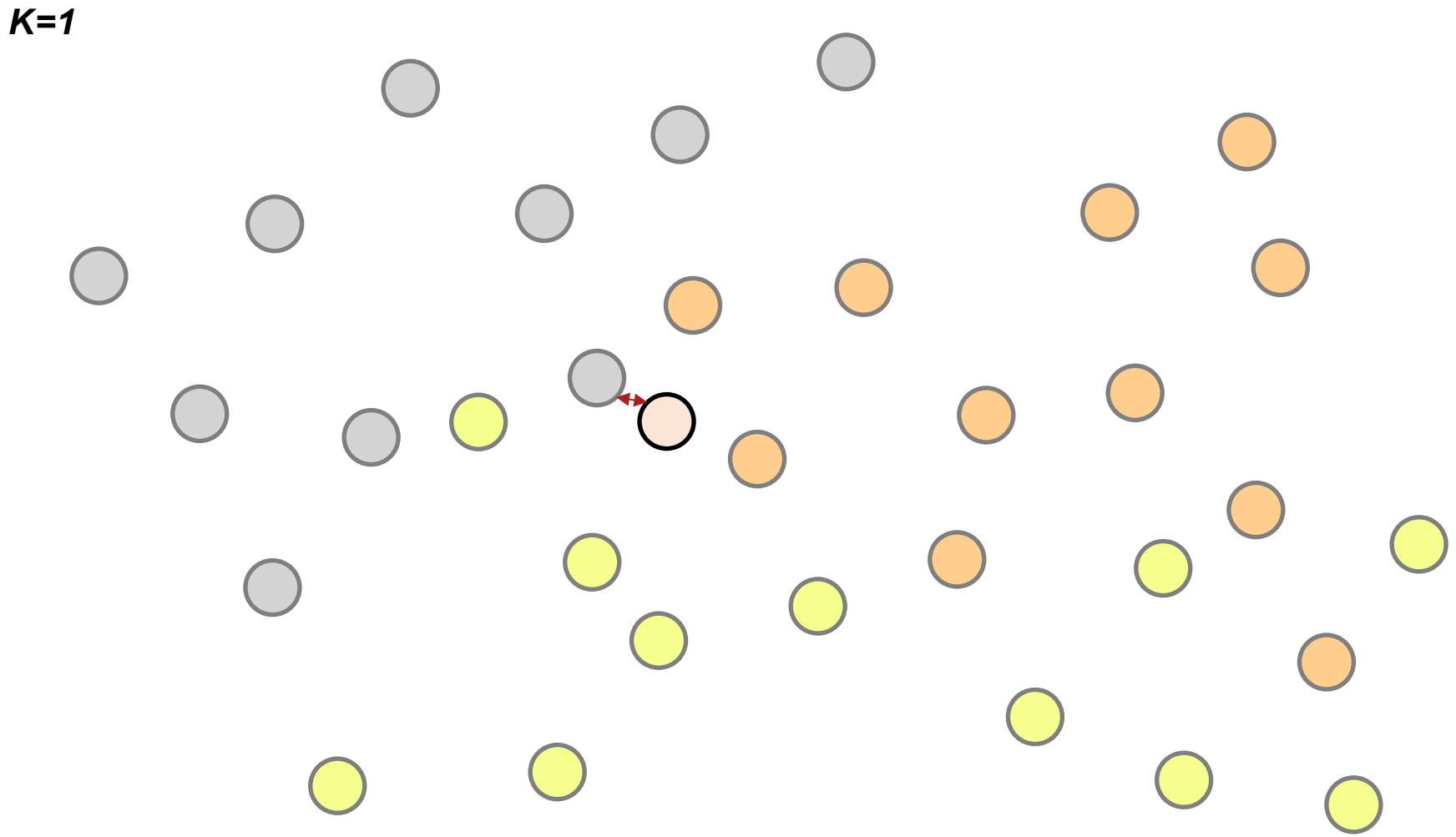
KNN (K-nearest Neighbor)



KNN (K-nearest Neighbor)

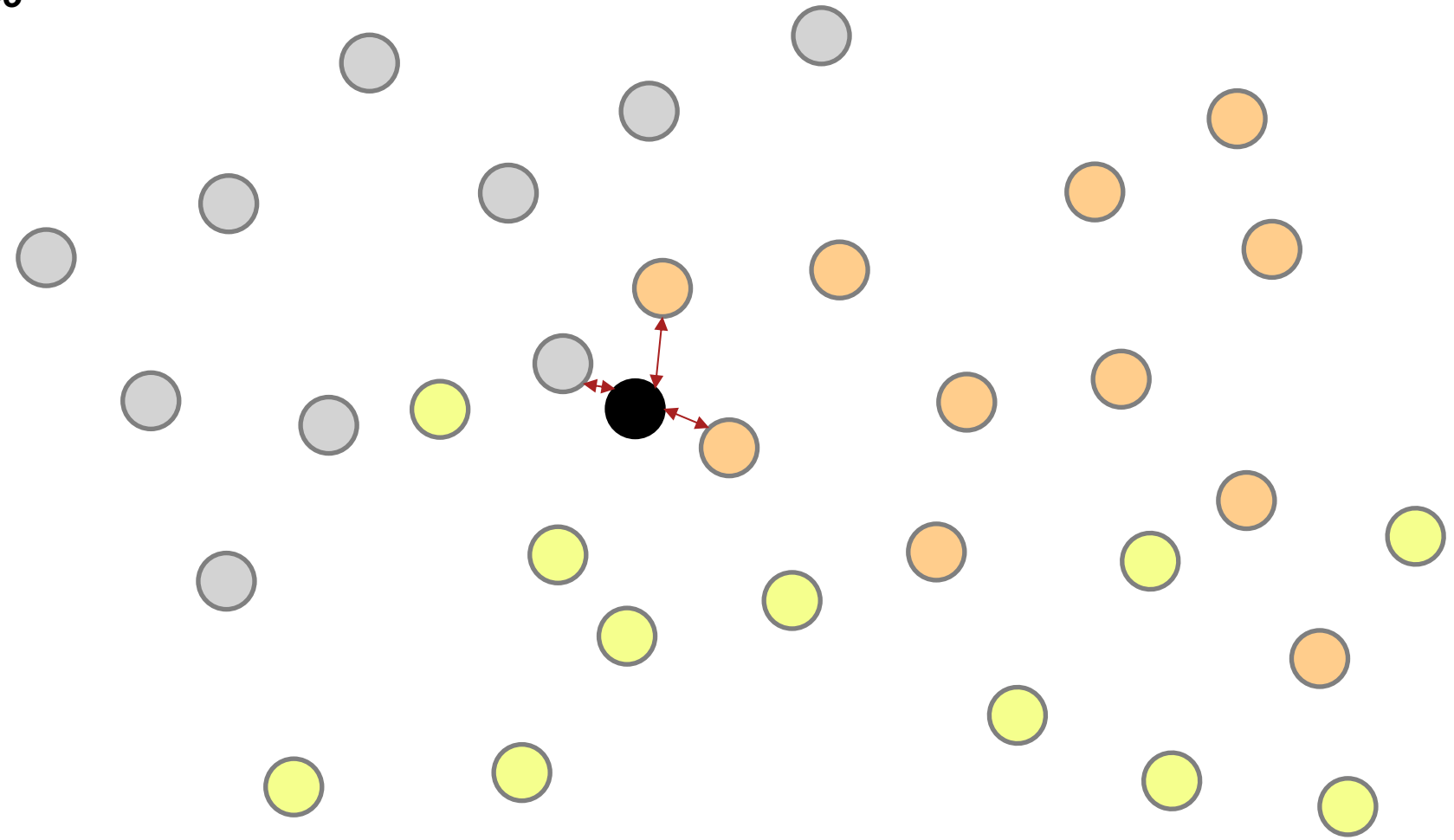


KNN (K-nearest Neighbor)



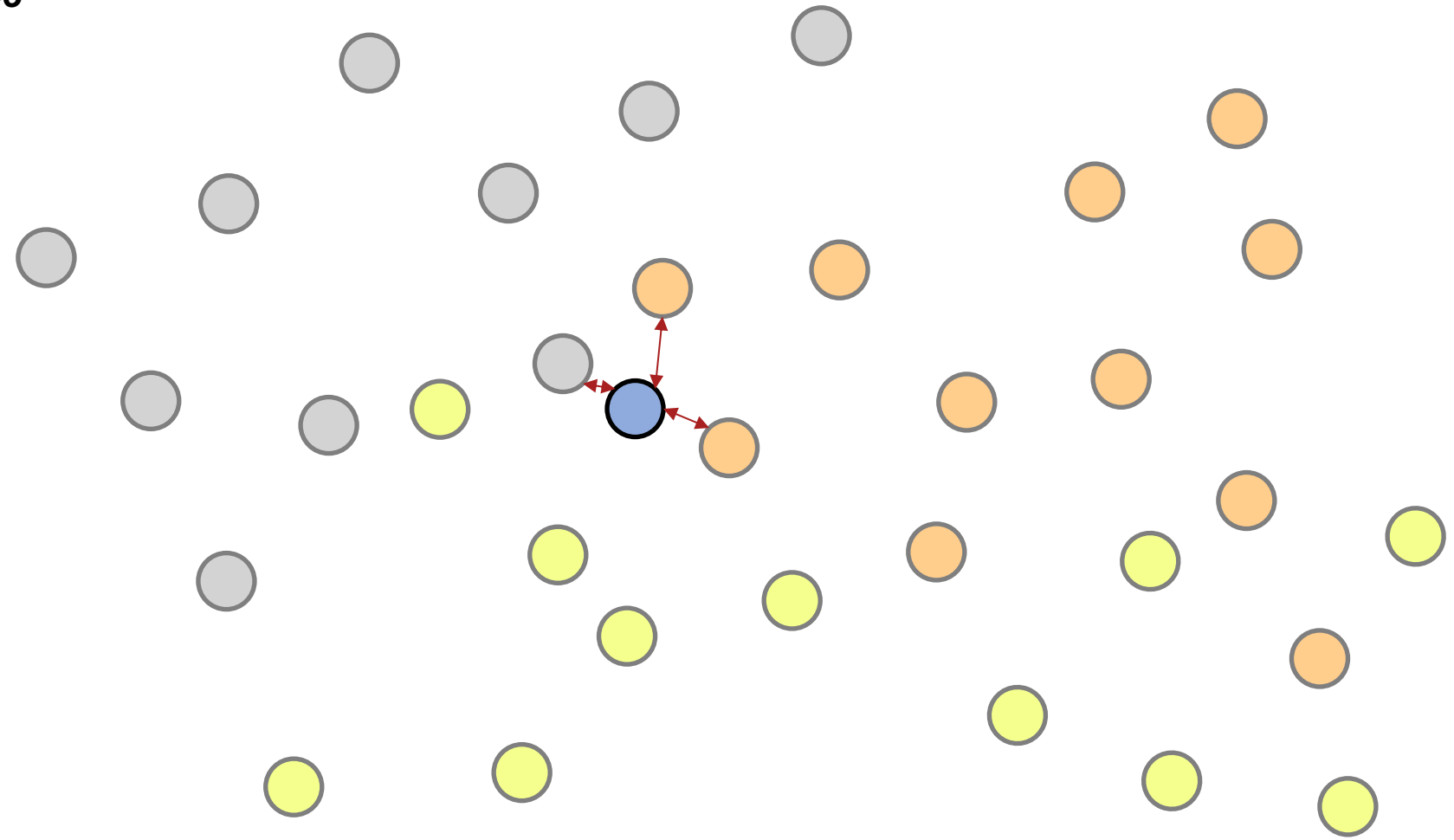
KNN (K-nearest Neighbor)

K=3



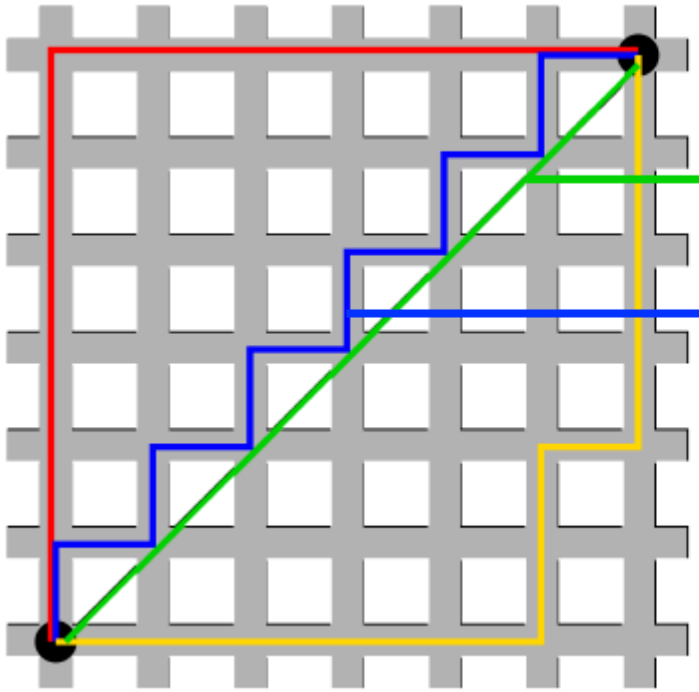
KNN (K-nearest Neighbor)

K=3



어떻게 근접한 이웃을 찾아갈까?

Distance



Euclidean Distance

Manhattan Distance

⋮

Cosine Similarity
Pearson Correlation

K는 몇으로 해야 하는가?

k **Rule of thumbs!**

K는 범주 개수의 배수로 두지 않는 것이 좋음.
이유는 동점이 나올 수 있기 때문!

KNN의 특징

- ✓ Lazy Learning
- ✓ 빠른 학습능력
- ✓ 느린 예측과정
- ✓ 명목형 데이터 & 결측 데이터에 대한 추가 처리 요구
- ✓ 거리를 기반으로 하므로 특정 Feature의 Scale이 크면, 불필요하게 큰 영향력을 가질 수 있음 -> Nomalizing 필요

KNN의 장점

- 구현하기에도 매우 간단
  - 새로 들어온 점과 나머지 점들간의 distance를 측정한 후 sorting하기만 하면 된다.
- 성능도 보통 크게 나쁘지 않은 값을 보임
  - 때문에 간단하게 개발할 필요가 있는 경우에 많이 사용하게 된다.
- 대부분의 머신러닝 툴박스들은 KNN의 다양한 variation까지 built-in function으로 지원
  - Matlab의 knnclassify가 대표적인 예.

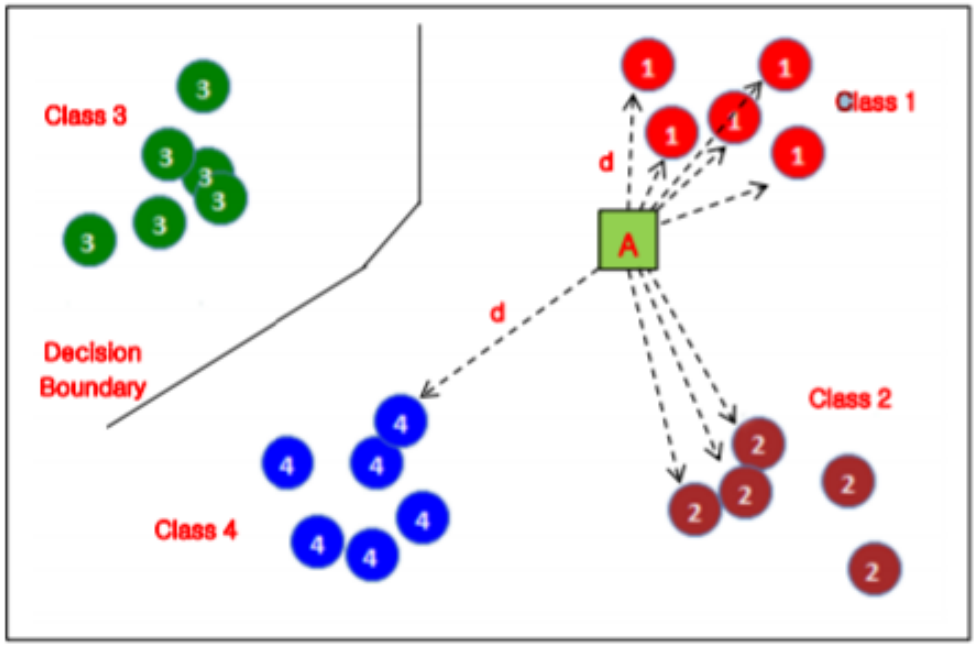
KNN의 장점

- 또한 KNN은 유용한 성질들이 많이 있다.
  - 예를 들어 만약 데이터의 개수가 거의 무한하게 있다면, KNN classifier의 error는 bayes error rate의 두 배로 bound가 된다는 특성이 있다.
  - 즉, 데이터가 엄청나게 많다면, KNN은 상당히 좋은 error bound를 가지게 된다는 것이다.
  - 즉, 단순한 휴리스틱 알고리즘이 아니라 엄밀하게 수학적으로 우수한 알고리즘임을 증명할 수 있는 알고리즘이라는 뜻이다.

- 또한 distance를 마음대로 바꿀 수 있다
  - 때문에 KNN은 변형하기에도 간단한 편이므로 데이터에 대한 가정을 모델에 반영하여 변형하기에 간편하다는 장점이 있다.

KNN 알고리즘 (근접이웃 알고리즘)

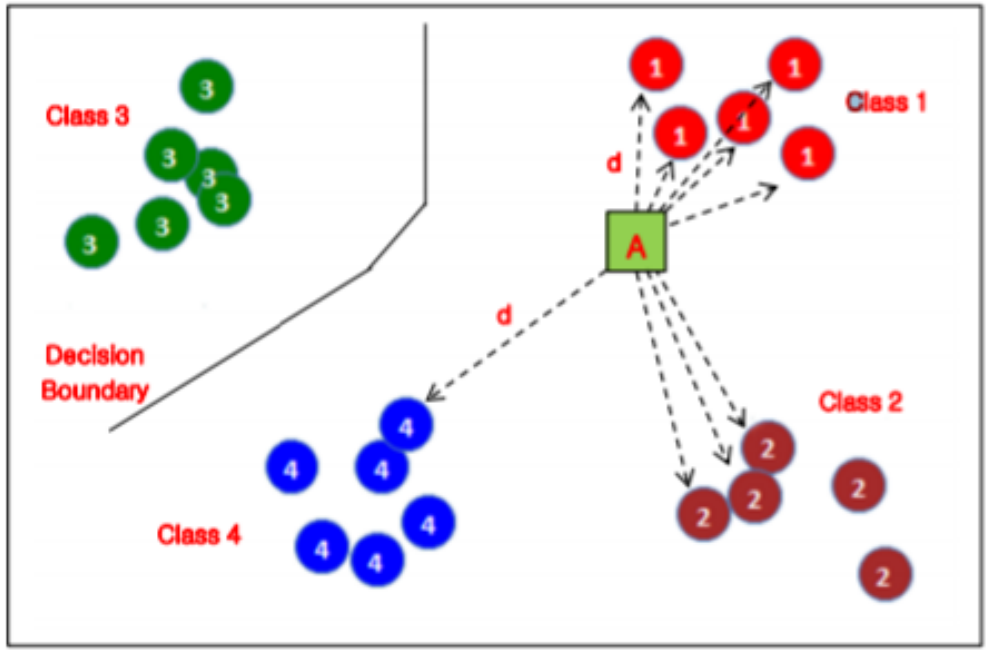
- 아래 그림과 같이 4개의 클래스로 구분된 데이터가 있을 때 점 A는 어느 클래스에 속한다고 평가할 수 있을까?



- 간단한 [illegible]스에 속한다고 할 수 있음
- 육안으로 보았을 때 점 A는 클래스 1에 속한다고 볼 수 있음.

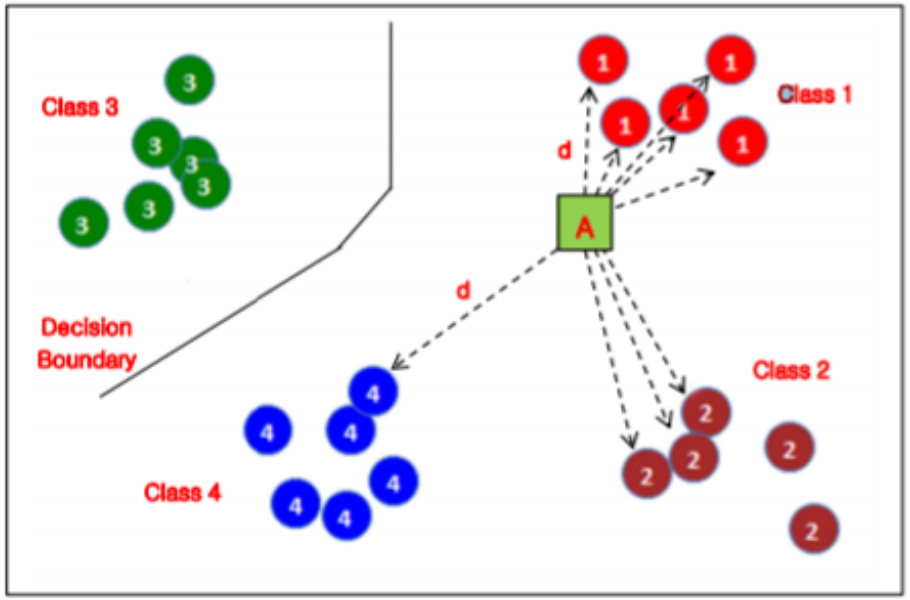
-> 다차원 공간 (다변량)인 경우는 육안으로 확인이 불가함.

KNN 알고리즘 (근접이웃 알고리즘)



- **KNN(K-nearest neighbor : 최 인접 이웃) 알고리즘**은 점 A로부터 가까운 거리에 있는 K 개의 점을 선택한 후, K 개의 점들이 가장 많이 속한 클래스를 찾아 점 A가 그 클래스에 속한다고 평가하는 알고리즘임. 예를 들어 A에서 가까운 9개의 점을 선택하였더니, 그 중 5개는 클래스 1에 속하고, 3개는 클래스 2에 속하고, 1개는 클래스 4에 속한다면 점 A는 클래스 2에 속한다고 평가함.

KNN 알고리즘



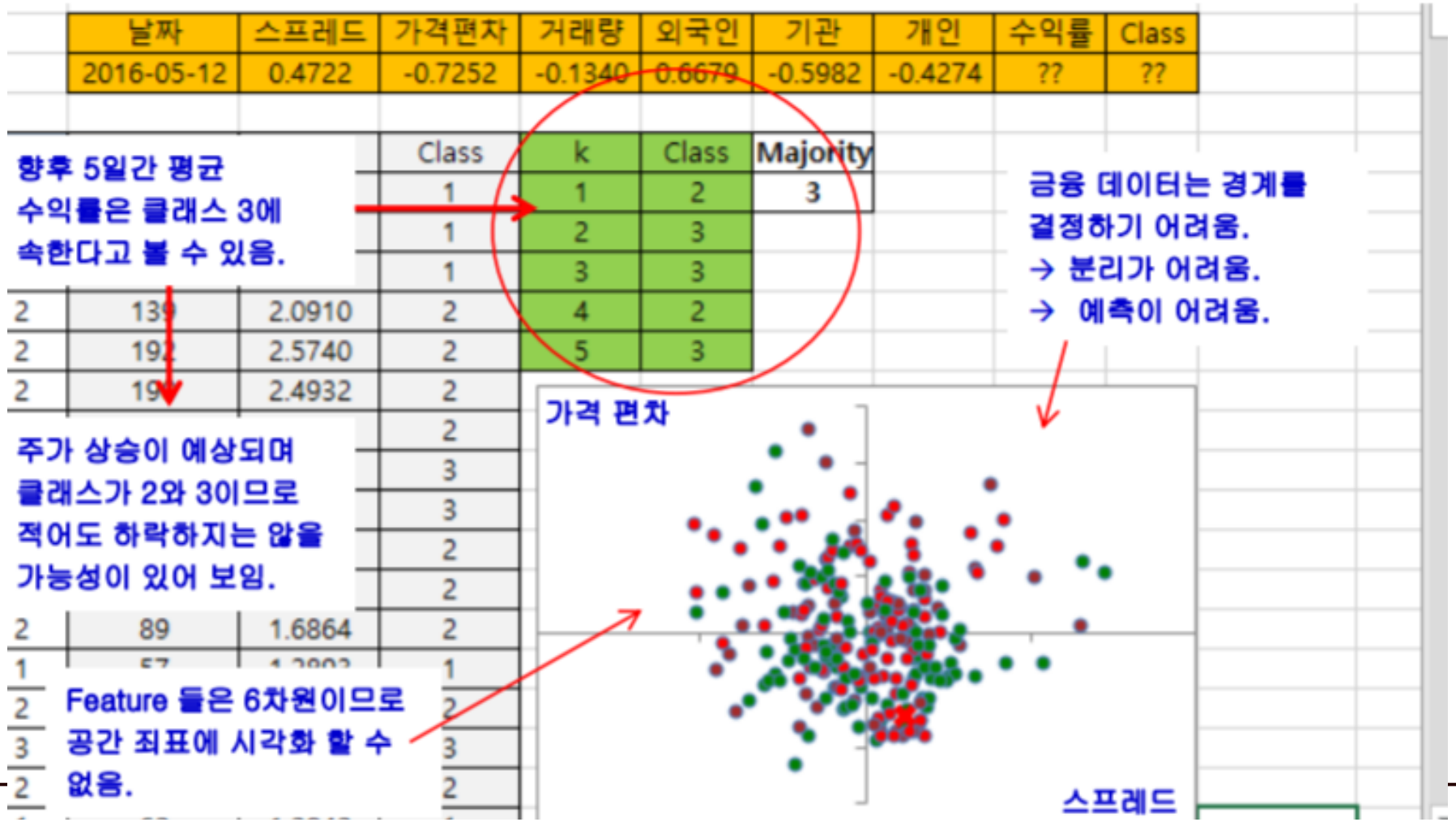
1) 점 A와 모든 훈련 데이터 사이의 거리를 측정함.
   – 거리는 평면상의 2점 사이의 거리로 측정 (Euclidean distance)
2) 측정한 거리가 작은 순으로 K 개의 점을 찾음. (ex : K = 9)
3) K개의 점들이 속한 클래스를 찾음.
4) K개의 점들이 속한 클래스가 가장 많은 것을 찾음. (다수결 원칙) 클래스 집합 = {1,1,1,1,1,2,2,2,4}라면 클래스 1이 가장 많음.
5) 점 A를 클래스 1로 분류함.
6) 시험 데이터를 이용하여 분류가 잘 되었는지 평가함.
7) K 값과 훈련 데이터, 시험 데이터를 바꾸어 가면서 가장 정확하게 분류하는 K 값을 찾음. – **Cross Validation**

KNN 알고리즘 예시 : 주가의 방향 예측

- 미래 수익률을 예측하기 위해 당일 데이터의 수익률을 5일 후 평균 수익률로 대체함. 당일의 Feature가 어떤 상태일 때 5일 후 평균 수익률은 얼마인지를 표현한 것임.
- 수익률 -0.2% 이하이면 클래스 1 (하락), -0.2% ~ +0.2% 이면 클래스 2 (보합), +0.2% 이상이면 3 (상승) 으로 분류함.
- 시험 데이터 (2016.5.12)의 Feature를 이용하여 미래의 수익률이 어느 클래스에 해당하는지 평가. -> 클래스 3 으로 분류됨.

KNN 알고리즘 예시 : 주가의 방향 예측

- **결과 :** 5일 후 평균 수익률이 +0.2% 이상 상승할 확률 = 3/5 = 60%, -0.2% ~ +0.2% 일 확률 = 2/5 = 40%, =0.2% 이하로 하락할 확률 = 0%



KNN 알고리즘 – 차원의 저주 (Curse of Dimensionality)

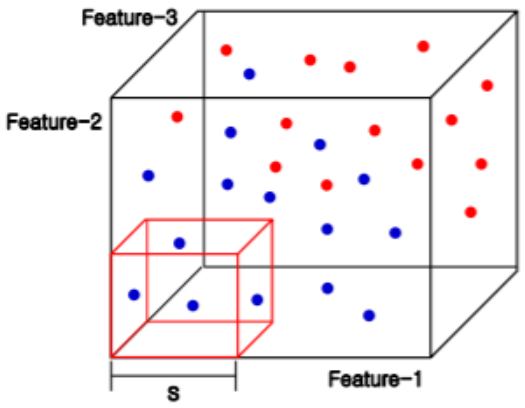
- KNN 분류기는 훈련 데이터가 충분히 많으면 좋은 성능을 나타내지만, **차원이 증가하면 (Feature 개수가 많아지면) 성능이 저하됨. : 차원의 저주**

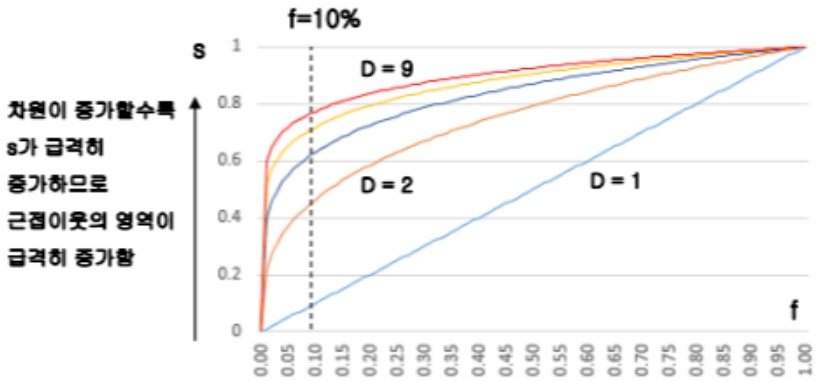
- 차원이 증가할수록 고차원 공간을 채울 데이터가 많이 필요하고, 멀리 떨어진 훈련 데이터를 참조해야 하기 때문에 "근접이웃"에 한정하기 어려움.

KNN 알고리즘 – 차원의 저주 (Curse of Dimensionality)

- 예를 들어 Feature 개수가 3개인 경우 (3차원 공간, D=3인 정육면체), 테스트 데이터 주변의 훈련 데이터의 개수가 전체 데이터의 10%를 유지하려면 한 변의 길이가 s =(0.1)^(1.3)=0.464인 정육면체가 필요함.

- D=10인 경우 s=(0.1)^(1/10)=0.8로 증가하므로, 근접이웃의 영역이 급격히 증가함. <- 근접이웃이 아닌 멀리있는 데이터를 참조해야함.





$$s(f) = f^{\frac{1}{D}}$$

$$s_2(0.1) = 0.1^{\frac{1}{2}} = 0.32$$

$$s_3(0.1) = 0.1^{\frac{1}{3}} = 0.46$$

$$s_{10}(0.1) = 0.1^{\frac{1}{10}} = 0.80$$

Weighted KNN

- **Weighted KNN** : 거리에 대해서 유사성 가중치를 부과한 KNN