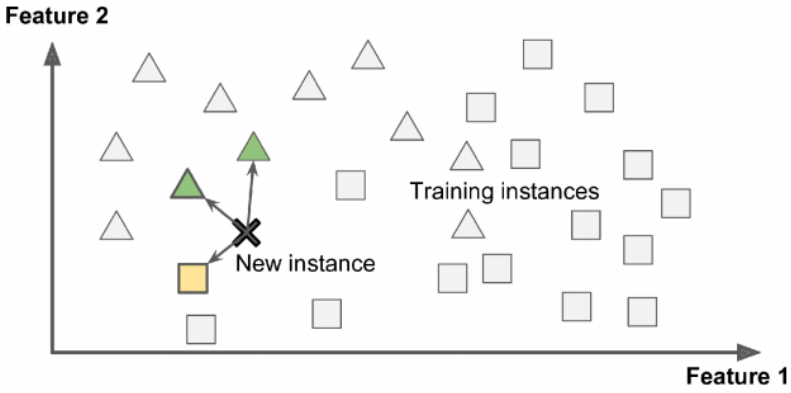
Clustering

Instance-based Learning

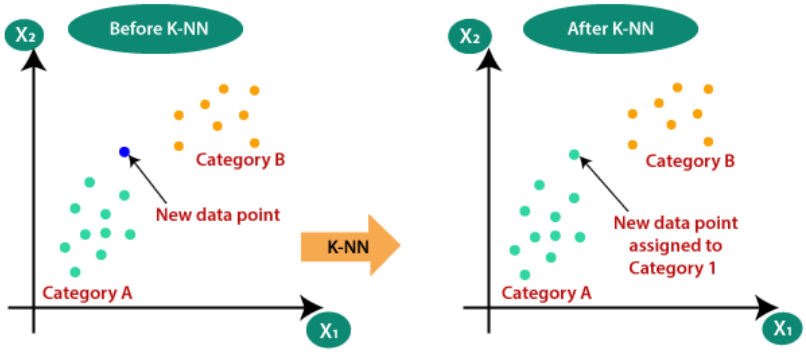


: 학습된 데이터를 기반으로, 새로운 데이터에 대해서는 유사도를 측정하여 가장 유사도가 높은 기존 데이터의 클래스로 분류함

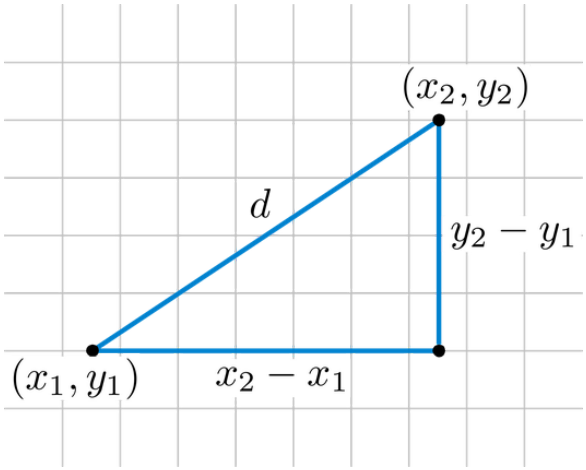
- 유사도 측정 방식은 데이터에 따라 다르게 진행 (ex. 텍스트 데이터는 공통으로 포함된 단어의 수를 기반으로 분류)
- 모델 기반 학습은 이와 다르게 데이터셋에 적합한 모델로 학습한 후에 새로운 데이터에 대한 분류를 예측하는 방법임

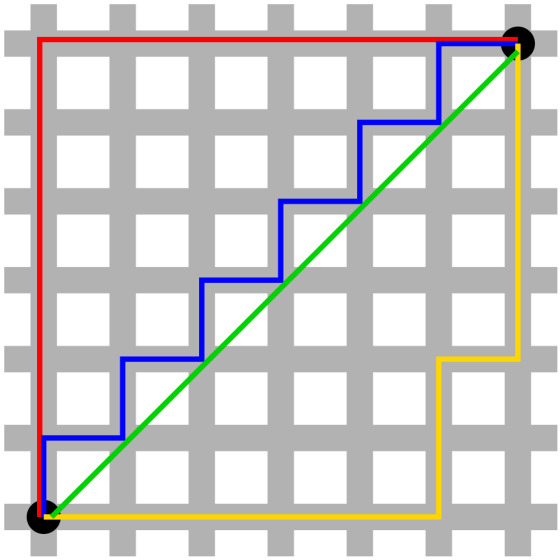
KNN (K-Nearest Neighbor) Algorithm

: 대표적인 인스턴스 기반 방법. non-parametric 지도학습의 일종으로, K개의 가까운 이웃의 속성에 따라 분류하여 레이블링을 하는 알고리즘



- classification, regression
- “가깝다”는 것에 대한 기준: 거리가 더 짧은 이웃이 가까운 이웃으로 취급. 유클리드 거리 (L2), 맨해튼 거리 (L1), 해밍거리 등





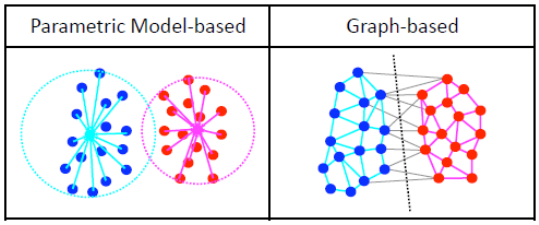
💡 주의점: 거리기반 모델이기 때문에 변수 값의 범위를 재조정 해주어야 함.
1) 최소-최대 정규화: 변수의 범위를 0%~100%까지 나타냄
2) z-점수 표준화: 변수의 범위를 정규분포화하여 평균을 0, 표준편차가 1이 되도록 함

- 구현 예시

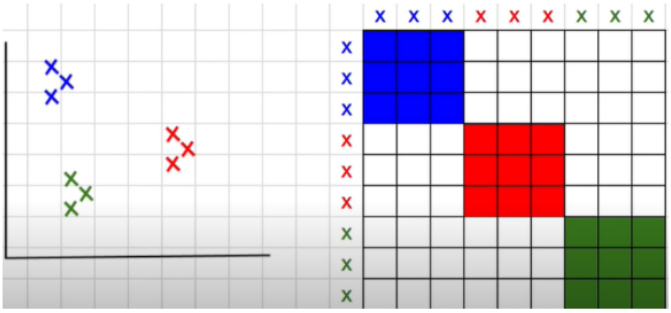
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split

iris = load_iris()

knn = KNeighborsClassifier(n_neighbors = 3)
X_train, X_test, Y_train, Y_test = train_test_split(iris['data'], iris['target'], test_size=0.25, stratify = iris['target'])
knn.fit(X_train, Y_train)
y_pred = knn.predict(X_test)
```

Spectral Clustering

: 그래프 기반의 군집화 방법. 각 데이터 사이에 선을 긋고 데이터의 유사도에 따라 가중치를 부여하고, 이를 기준으로 두 그룹으로 분류함



1) 주어진 데이터를 유사도 행렬 (affinity matrix)로 변환



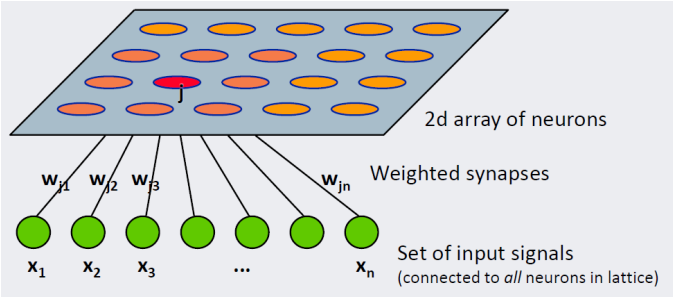
2) 유사도 행렬에서 w (데이터 점들 사이의 선/가중치) 값이 최대로 나오도록 하는 고유 벡터 구함

3) 고유벡터값들을 바탕으로 minimum out 혹은 normalized out 방법으로 구현

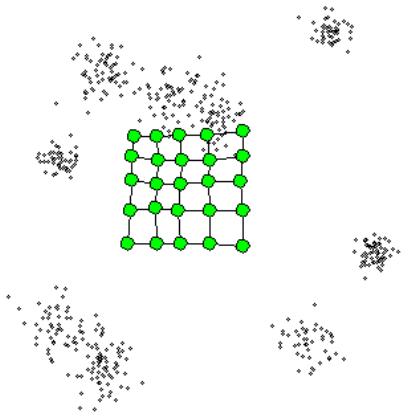
- minimum out: 데이터들 사이의 유사도 값들 중 가장 작은 값으로 각각의 클러스터로 나는 방법. 하지만 클러스터를 분류하는 가중치보다 더 작은 값이 같은 클러스터에 속하면 분류가 잘못 됨

SOM (Self-Organizing MAP; 자기조직화지도)

: 저차원 (2차원~3차원) 격자에 고차원 데이터가 대응하도록 인공신경망과 유사한 방식의 학습을 통해 군집을 도출해내는 기법. 고차원에서의 유사도를 최대한 보존하도록 학습



- 초록 노드 = 고차원 입력벡터의 각 요소. 저차원 격자 하나에 여러 입력벡터들이 속하면 서로 가까운 유사도를 가짐
- 학습 과정 (검정색 원데이터 공간=고차원)



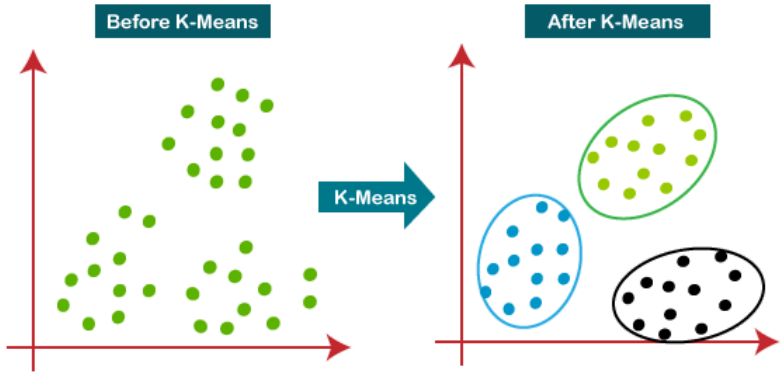
- 랜덤으로 초기화 후 검정색 점을 하나씩 추가하며 연두색 점 (격자벡터)의 위치를 검정 점의 위치와 비슷해지도록 업데이트함
- 검정색 점과 가까운 노드 (winning node)가 가장 많이 업데이트됨

Pational Clustering

: 전체 데이터의 영역을 특정 기준에 의해 동시에 구분하여 사전에 정의된 개수의 군집 가운데 속하게 하는 군집화 방법. 대표적: k-means 군집화

K-means Clustering

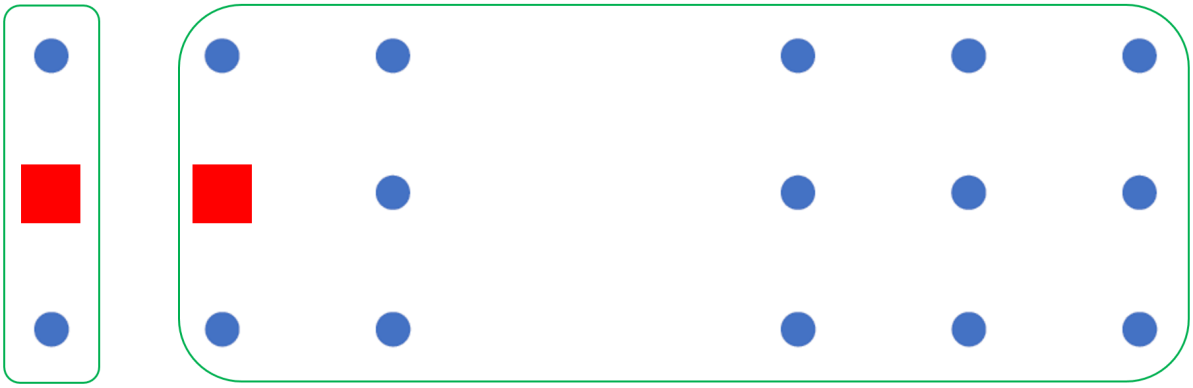
: 각 군집은 하나의 중심 (centroid)를 가지고, 가까운 중심에 할당된 개체들이 모여 하나의 군집을 형성. 사전에 군집 수 k를 정함



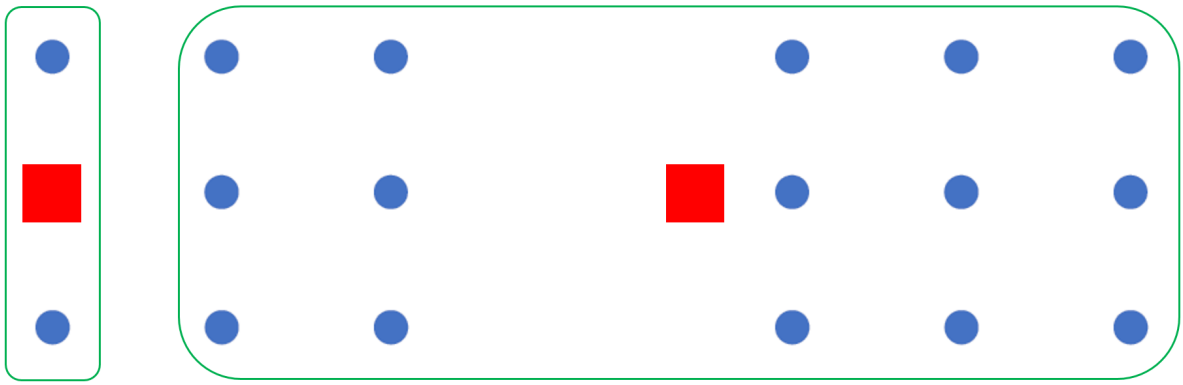
- 각 군집 centroid 위치 & 각 개체가 어떤 군집에 속해야 하는지를 동시에 찾아야 하기 때문에 EM (Expectation Maximization) 알고리즘을 적용
- 과정

중심 초기화

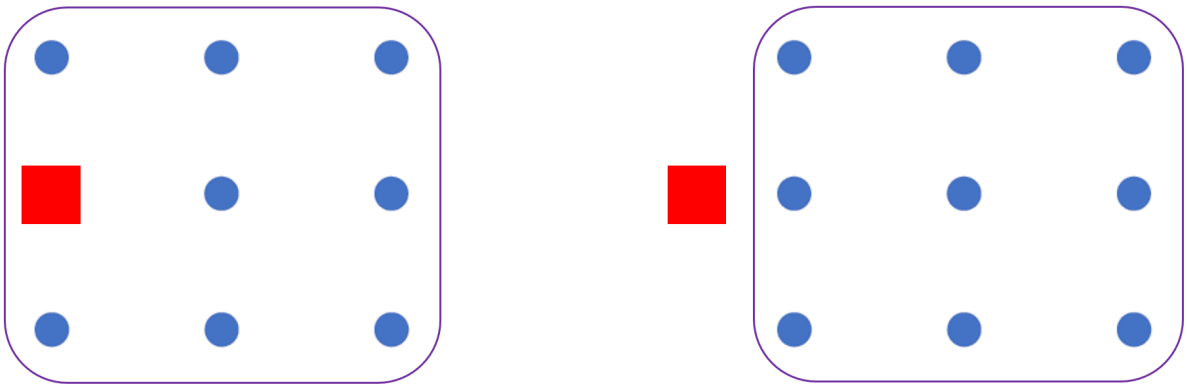
가까운 중심에 군집을 할당 (Expectation 스텝)



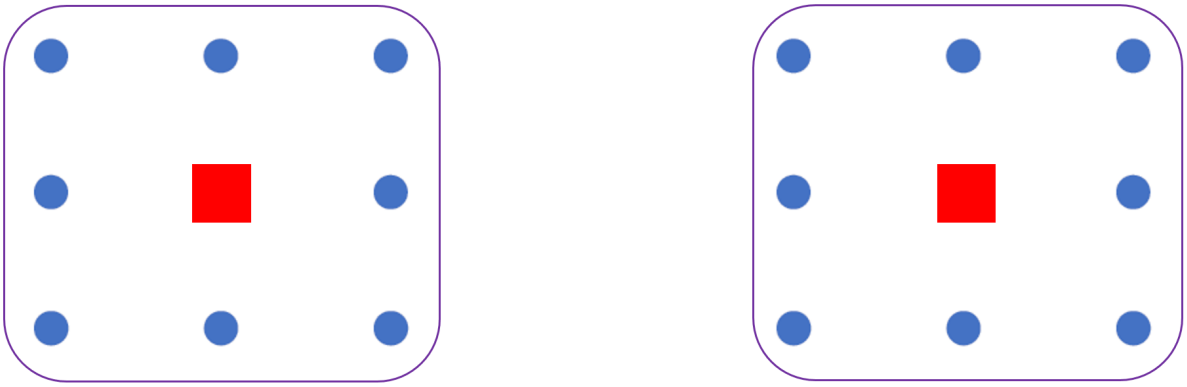
중심을 군집 경계에 맞게 업데이트 (Maximization 스텝)



가장 가까운 중심으로 다시 군집 할당 (Expectation 스텝)



수렴할 or 반복수 채우면 학습 종료

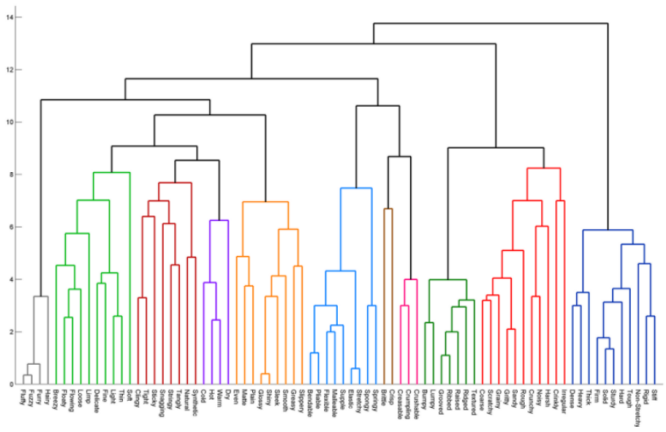


- 단점: 초기값 위치, 군집의 크기/밀도, 데이터의 분포가 특이한 경우 제대로 군집이 이루어지지 않을 수 있음

Hierarchical Clustering

: 계층 트리 모형을 이용해 개체들을 순차적&계층적으로 유사한 그룹과 통합하여 군집화 수행하는 알고리즘

덴도그램 생성 후 적절한 수준에서 트리 자름



- 모든 개체들 간 거리와 유사도가 계산되어 있어야 한다

참고

https://medium.com/@sanidhyaagrawal08/what-is-instance-based-learning-a9b06079e836

https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

https://velog.io/@jhlee508/머신러닝-KNNK-Nearest-Neighbor-알고리즘

https://techblog-history-younghunjo1.tistory.com/93

https://ratsgo.github.io/machine learning/2017/05/01/SOM/

https://ratsgo.github.io/machine learning/2017/04/19/KC/