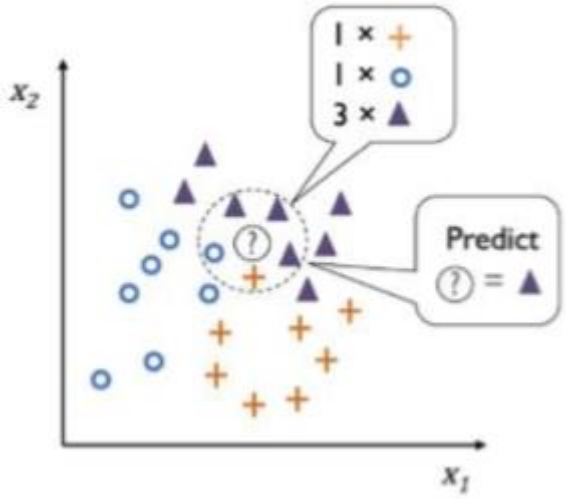
분류 (Classification) : 미리 정의된, 가능성 있는 여러 class label 중 하나를 예측 하는 것. 두 개로 분류하는 건 Binary classification, 셋 이상은 Multiclass classification. (ex : 얼굴 인식, MNIST 등)

회귀 (Regression) : 연속적인 숫자 또는 부동소수점수(실수)를 예측하는 것. (ex : 주식 가격 예측)

KNN 분류 : 주변 k개의 자료의 class 중 가장 많은 class로 분류하는 방식.



새로운 자료?를 가장 가까운 자료 5개 (k=5)를 투표하여 가장 많은 class로 할당.

Training-data 자체가 모형일 뿐 어떤 추정 방법도 모형도 없음 : 데이터의 분포를 표현하기 위한 Weight나 bias 같은 parameter를 추정하지 않음.

매우 간단한 방법이지만 performance는 떨어지지 않는다.

게으른 학습(lazy learner) 또는 사례중심학습(instance-based learning) : Linear classification 처럼 (훈련 데이터에서 판별 함수 (discriminative function)을 학습하는 대신) 모델을 학습시켜 파라미터를 구하는 대신 훈련 데이터 셋을 그냥 메모리에 저장한다.

데이터의 차원이 증가하면 차원의 저주 (curse of dimensionality) 문제가 발생함 즉 차원이 증가할 수록 성능 저하가 심함 : 데이터의 차원이 증가하면 (feature가 많아 지면) 동일한 데이터를 설명하더라도 빈 공간이 증가하여 저장 공간과 처리 시간이 불필요하게 증가된다. 또한 데이터가 feature에 비해 밀도가 sparse 해지면서 성능이 저하된다.

i번째 관측치와 j번째 관측치의 거리로 Minkowski 거리를 이용.

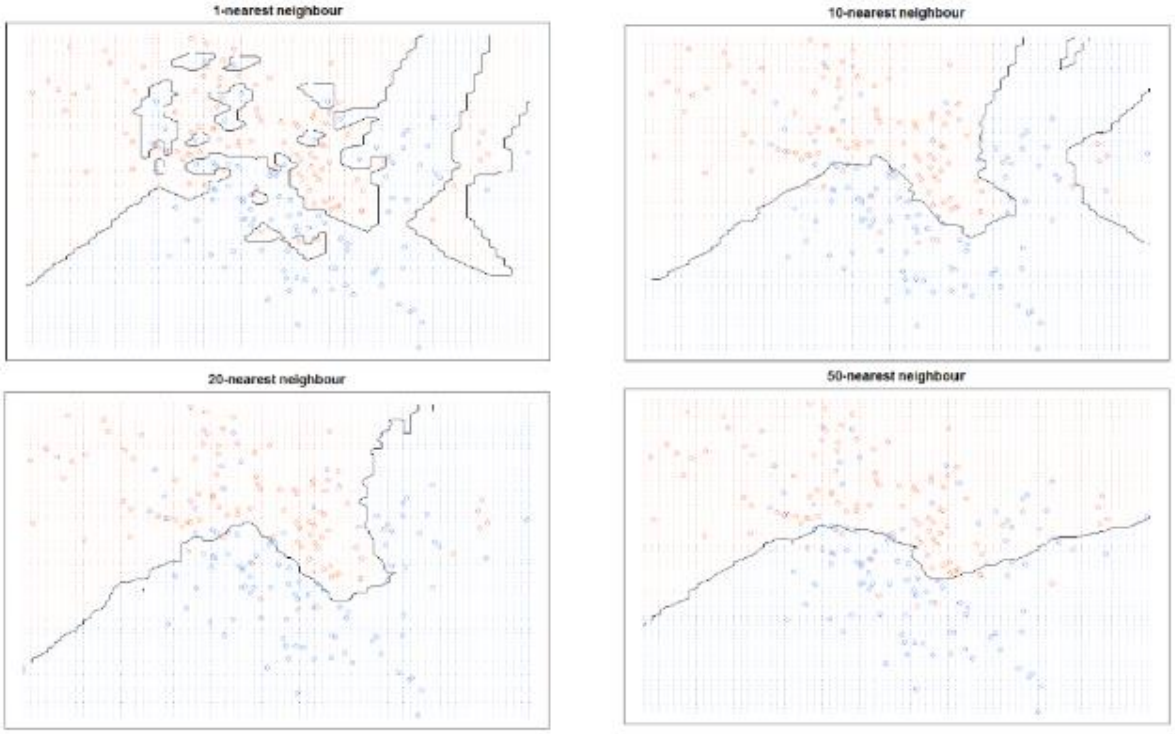
$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt[p]{\sum_{k=1}^{d} |x_{ik} - x_{jk}|^p}$$

P=1 : 맨하탄 거리(그리드) P=2 : 유클리디언 거리(피타고라스)

KNN의 Hyperparameter : 탐색할 이웃 수(k)와 거리 측정 방법(p)

- K가 작을 경우 데이터의 지역적 특성을 지나치게 반영하여 과적합(overfitting) 발생

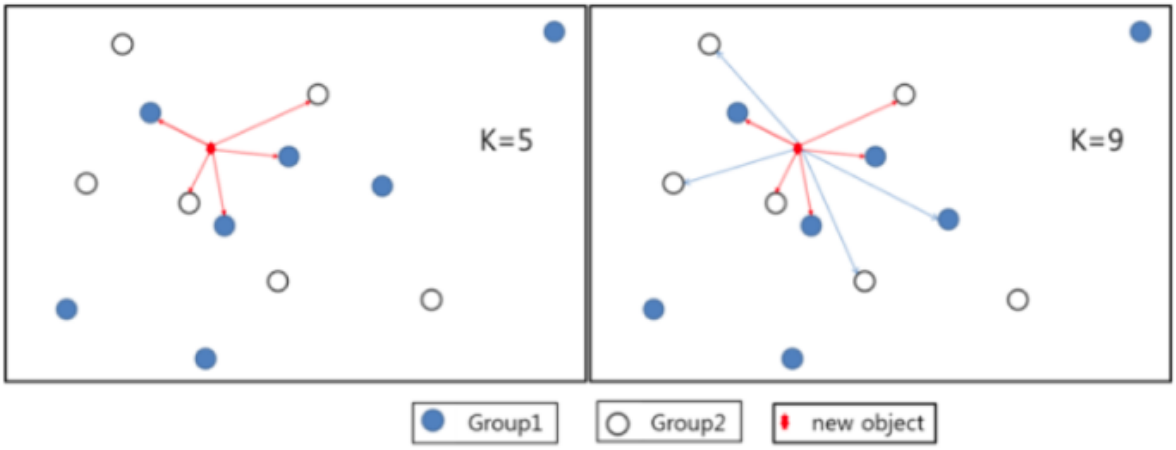
  반대로 매우 클 경우 모델이 과하게 정규화(underfitting) 발생



학습을 통해 데이터의 경향성을 파악하는 것이 중요하다.

K가 가지는 의미 : 새로운 자료에 대해 근접치 K의 개수에 따라 Group이 달리 분류됨.

- 다수결 방식(Majority voting): 이웃 범주 가운데 빈도 기준 제일 많은 범주로 새 데이터의 범주를 예측하는 것.
- 가중합 방식(Weighted voting): 가까운 이웃의 정보에 좀 더 가중치를 부여.



다수결 방식의 경우 K=5일 때 Group1, K=9일 때 Group2로 분류하지만

가중합 방식의 경우 둘 다 Group1으로 분류할 것이다.

KNN의 장단점 요약

- 장점 :
  1. 학습데이터 내에 끼어 있는 노이즈의 영향을 크게 받지 않는다.
  2. 학습데이터 수가 많다면 꽤 효과적인 알고리즘이다.
  3. 마할라노비스 거리(Mahalanobis distance: 평균과의 거리가 표준편차의 몇 배인지를 나타내는 값. 즉, 어떤 값이 얼마나 일어나기 힘든 값인지 또는 얼마나 이상한 값인지를 수치화 하는 한 방법)와 같이 데이터의 분산을 고려할 경우 매우 강건(robust)한 방법론
- 단점 :
  1. 최적 이웃의 수(k)와 어떤 거리 척도(distance metric)가 분석에 적합한 지 불분명해서 데이터 각각의 특성에 맞게 연구자가 임의로 선정해야 함. Best K는 데이터마다 다르기 때문에 다 해보는 탐욕적인 방식(Grid Search)로 탐색.
  2. 새로운 관측치와 각각의 학습 데이터 사이의 거리를 전부 측정해야 하므로 계산 시간이 오래 걸리는 한계.
  3. KNN의 계산 복잡성을 줄이려는 시도들 : Locality Sensitive Hashing, Network based Indexer, Optimized product quantization 등

기계 학습의 일반적인 실습 순서

1. 데이터셋 불러오기 (seaborn 라이브러리사용, 유명한 데이터 셋 대부분 지원)
2. 데이터셋 카테고리의 실수화 (setosa, versicolor, virginica -> '0', '1', '2')
3. 데이터 분할 (학습 데이터와 테스트 데이터)
4. (옵션) 입력데이터의 표준화
5. 모형 추정 혹은 사례 중심 학습 (Training -> Y^)
6. 결과 분석 (Confusion matrix로 확인)

Iris 데이터셋

데이터명 : IRIS (아이리스, 붗꽃 데이터)

레코드 수 : 150개 (N이 150)

필드 수 : 5개 (dimension(feature)가 5개)

데이터설명 : 아이리스(붓꽃)에 대한 데이터. 꽃잎의 각 부분의 너비와 길이 등을 측정한 150개의 레코드로 구성되어 있음. (150, 5)

필드의 이해 : 총 6개의 필드로 구성되어 있으나 caseno는 순서를 표시하므로 분석에서 제외. 2번째부터 5번째의 4개의 필드는 입력 변수(X)로 사용되고, 맨 뒤의 Species 속성이 목표(종속) 변수(Y)로 사용된다.

	caseno	SepalLength	SepalWidth	PetalLength	PetalWidth	Species
1	1	5.1	3.5	1.4	.2	setosa
2	2	4.9	3.0	1.4	.2	setosa
3	3	4.7	3.2	1.3	.2	setosa

IRIS 실습

KNN 회귀 : KNN 분류와 동일하나 y의 예측 치 계산만 다르다. K개의 주변 y를 다 더하고 평균을 구한다.

단순 회귀 : 가까운 이웃들의 단순한 평균을 구하는 방식

가중 회귀(Weighted regression) : 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 ➔** $\dfrac{\frac{5.0}{3.2}+\frac{6.8}{11.5}+\frac{9.0}{1.1}}{\frac{1}{3.2}+\frac{1}{11.5}+\frac{1}{1.1}} = 7.9$