7장 k - 최근접이웃 알고리즘

개요

새로운 레코드를 분류하거나 예측할 때, 학습 데이터 내 에서 찾아낸 유사한 레코드들을 필요로 함.

이러한 '이웃들'(neighbors)로 부터 새로운 레코드의 분류결과나 예측결과를 끌어내기 위해서는 투표, 평균과 이용

어떻게 유사도가 결정되는지, 어떻게 이웃들의 개수가 결정되는지, 어떻게 분류, 예측이 계산되는지를 알아봄.

데이터로부터 자동적으로 결과를 유도해 내고, 그 결과를 매우 잘 활용하는 기법 중 하나.

1. k - 최근접이웃 분류기 (범주형 결과)

학습데이터 셋으로 부터 분류를 하고자 하는 새로운 레코드와 유사한 k개의 레코드를 식별하는 것부터 시작.

그 후, 이런 유사 데이터를 활용하여 특정 클래스로 분류 → 즉, 이웃레코드들 중 상당수가 속한 클래스로 배정

↳ 새로운 레코드에 대한 예측변수들의 값을 $x_1, x_2, \cdots, x_p$ 라고 할 때, 예측변수의 공간 상에서 유사/근접한 레코드

즉, $x_1, x_2, \cdots, x_p$ 에 근접한 값을 갖는 레코드들을 학습데이터에서 찾음 → 이 이웃레코드가 속한 클래스를 참조하여 클래스 할당

- 이웃 결정하기

k-최근접이웃알고리즘 - 소속클래스(Y)와 예측변수들 $X_1, X_2, \cdots, X_p$ 간의 관계에 대한 가정을 만들지 않는 분류 기법

↳ 비모수적 방법으로, 미리 가정된 함수 형태에 대한 모수 추정을 하지 않는 대신 레코드들간의 유사성에서 정보를 얻음

※ 중요이유! 어떻게 예측변수의 값들에 기반해서 레코드 간의 거리를 측정하는가?

↳ 가장 보편적인 방법은 유클리드 거리 (Euclidean distance) ⇒ $\sqrt{(x_1-u_1)^2+(x_2-u_2)^2+\cdots+(x_p-u_p)^2}$ (각각의 레코드값)

거리에 대해서 많은 계산을 요함으로 계산 비용이 낮은 유클리드 거리 가장 많이 활용.

다양한 예측변수들의 척도를 균등하게 하기 위해 예측변수들을 표준화. → 학습데이터의 평균과 표준편차 이용, 새로운 레코드는 계산에서 제외, 검증데이터는 사용 X

- 분류규칙

거리계산 이후 어떻게 클래스를 배분할 것인가?

↳ 가장 단순한 경우 $k=1$ → 유사한 이웃 하나를 찾아서 그 이웃이 속한 클래스로 분류

↳ 학습데이터의 수가 많을 때 유용 1-최근접 알고리즘의 경우 각 클래스 별 확률 밀도함수를 정확히 알고있을 때의 오차를 2배 이상 넘지 않음.

$k>1$ 인 경우

① 분류될 레코드와 가장 가까운 k개의 이웃 찾음

② 그 중 다수가 속하는 클래스로 분류

- k 값 선택

k>1을 선택하면 k값이 커질수록 학습데이터의 노이즈로 인한 과적합의 위험을 줄여주는 평활효과를 얻는 장점이 있음.

↳ k가 너무 작으면 데이터의 노이즈를 적합할 위험.

↳ k가 너무 크면 데이터의 지역적 구조를 파악할 수 있는 능력을 놓칠 수 있다. (→ 알고리즘의 주된 장점)

극단적인 경우는 $k=n$ → 나이브규칙 (과적합과 이를 배제하는 것 사이의 균형 필요)

↳ 데이터의 성격에 따라 결정

데이터가 복잡하고 불규칙한 구조를 가질수록 k의 최적값은 더 작아짐

↳ 일반적으로 1~20

각 클래스별 동일 수가 나오는 것을 막기위해 홀수 사용

⇒ 가장 분류성능이 좋은 k을 선택 (여러 k값을 비교하는 과정 필요)

- 컷오프값 설정

k개 중에서 몇 개나 같은 클래스에 있어야 그 클래스로 분류할 것인가?

↳ 정확도나 오분류 비용을 절충하기 위해 다른 값 사용. (분류행렬이 영향을 받음)

2. k-최근접 이웃 예측기 (수치형 결과)

연속형 변수의 예측에도 쉽게 확장

첫 번째로 거리 계산 이후 이웃을 정하는 것은 동일

클래스를 결정하기 위해서는 이웃들의 평균 결과값 사용 (→ 가중평균값을 이용하기도 함 (거리가 멀수록 가중치 감소))

3. 알고리즘의 장점과 단점

장점: 모수에 대한 가정이 거의 없다
단순한 방법

단점: 학습세트로 부터 모수를 추정하는데 걸리는 시간은 없다고 해도, 데이터가 클 경우 이웃을 찾는데 걸리는 시간이 많이 걸릴 수 있음 → 해결책으로 ① 주성분분석과 같이 차원을 축소함으로서 거리계산 시간 단축
② 이웃을 빨리 찾기위해 검색트리와 같은 정교한 데이터 구조 사용
↳ 정확한 이웃을 찾는다기보다 근접하다고 볼 수 있는 이웃을 찾음

학습세트에 필요한 레코드의 수는 예측변수 p가 증가함에 따라 기하급수적으로 증가

↳ 데이터가 같이 증가하지 않으면 거리에 대한 기댓값이 급격하게 증가 (차원을 줄이는 이유)

나태 학습방법 (Lazy learner)

↳ 많은 학습시간이 소요되는 계산들이 예측단계에 집중 ⇒ 많은 수의 레코드 실시간 예측은 힘듦