fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

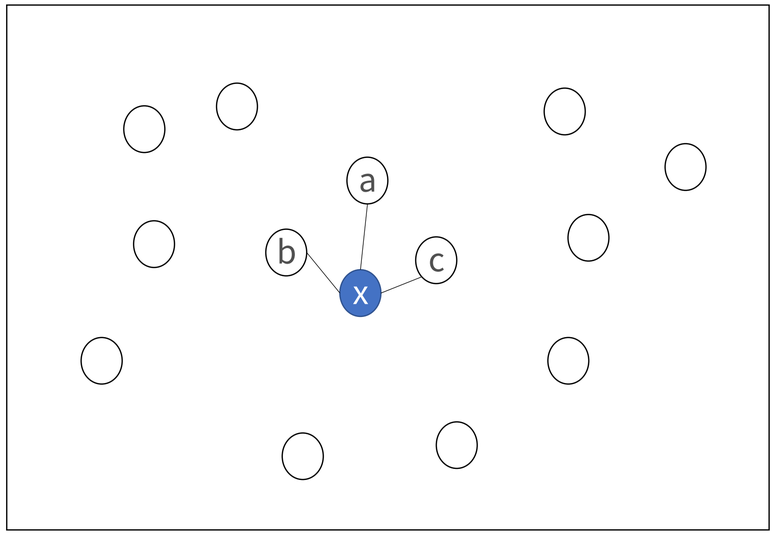
2. k-최근접 이웃

1 작동 과정 및 모델의 장단점

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

작동 과정

k-최근접 이웃은 한 샘플의 k개 이웃의 라벨을 바탕으로 해당 샘플의 라벨을 예측하는 모델입니다.



- 이웃: 한 샘플과 가장 가까운 샘플 집합
- x의 이웃: {a, b, c}
- (분류) a, b, c의 라벨의 최빈값으로 x의 라벨을 예측
- (회귀) a, b, c의 라벨의 평균값으로 x의 라벨을 예측

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

장단점

k-최근접 이웃은 어느 문제에도 적용하기 적합한 모델입니다.

장점

- 두 샘플 간 거리 혹은 유사도만 정의할 수 있으면 데이터 구조 등과 관계없이 사용할 수 있음
- 이웃을 바탕으로 예측을 수행하므로 다중 분류 및 회귀를 손쉽게 구현할 수 있음
- 라벨이 둘 이상인 다중 라벨 분류 및 회귀도 구현할 수 있음
- 이론적으로 학습이 필요 없음
- 학습 데이터가 축적되면 자연스레 모델이 업데이트되므로 언제 어떻게 모델을 업데이트할지 고민할 필요가 없음

단점

- 다른 모델에 비해 라벨을 예측하는 시간이 매우 길고 이론적인 학습이 없으므로 학습 데이터를 계속 갖고 있어야 함
  (이러한 이유로 사이킷런에서는 k-최근접 이웃을 공간을 기준으로 라벨을 학습하고 예측하도록 설계함)
- 필요에 따라 거리 척도와 유사도 척도를 새로 설계해야 함

fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

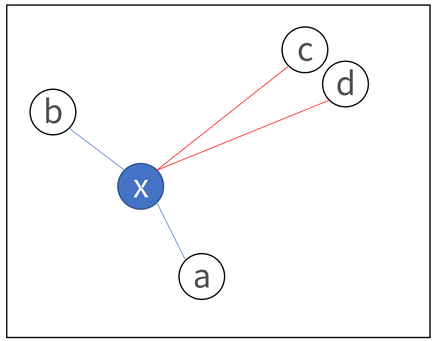
2. k-최근접 이웃

2 주요 하이퍼 파라미터

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

거리 및 유사도 척도
:개요

k-최근접 이웃 모델은 이웃을 바탕으로 라벨을 예측하므로 각 샘플을 이웃이라 판단하는 기준이 성능에 크게 영향을 끼칩니다.



- 유클리디안 거리를 사용했을 때의 x의 이웃: {a, b}
- 코사인 유사도를 사용했을 때의 x의 이웃: {c, d}

거리와 유사도는 반비례합니다.

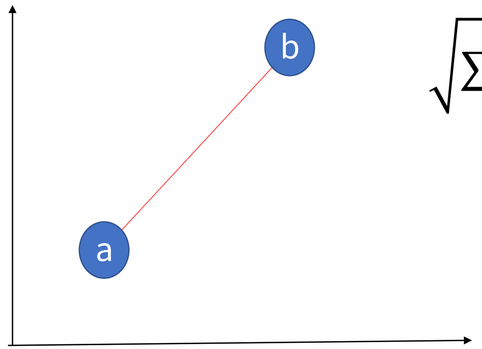
$$\text{거리} \propto 1/\text{유사도}$$

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

거리 및 유사도 척도
(1) 유클리디안 거리

유클리디안 거리 척도는 가장 널리 사용되는 거리 척도입니다.

두 샘플 $\boldsymbol{a} = (a_1, a_2, \cdots, a_m)$와 $\boldsymbol{b} = (b_1, b_2, \cdots, b_m)$에 대해, 유클리디안 거리는 다음과 같이 정의됨



$$\sqrt{\sum_{j=1}^{m}\left(a_j - b_j\right)^2}$$

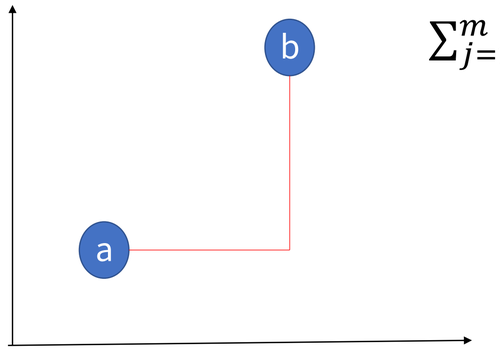
- 두 샘플 간 직선거리를 측정하므로 연속형 특징으로 구성된 데이터에 적합
- 특징 간 스케일 차이에 크게 영향을 받음

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

거리 및 유사도 척도
(2) 맨하탄 거리

맨하탄 거리 척도는 모두 정수형인 특징으로 구성된 데이터에 적합합니다.

두 샘플 $\boldsymbol{a} = (a_1, a_2, \cdots, a_m)$와 $\boldsymbol{b} = (b_1, b_2, \cdots, b_m)$에 대해, 맨하탄 거리는 다음과 같이 정의됨



$$\sum_{j=1}^{m} |a_j - b_j|$$

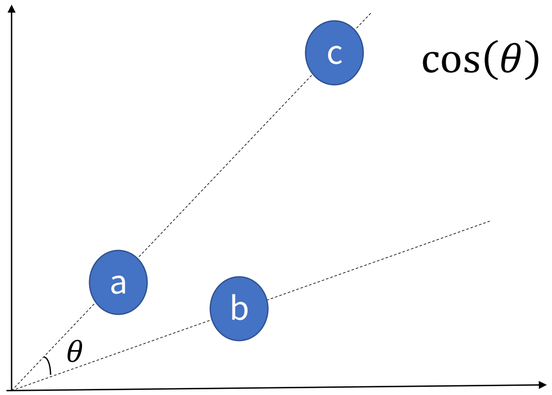
- 수직 혹은 수평 거리로만 구성됨
- 리커트 척도(Likert scale)를 사용한 설문 조사 데이터와 같이 모두 정수형인 특징으로 구성된 데이터에 적합

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

거리 및 유사도 척도
(3) 코사인 유사도

코사인 유사도는 두 샘플의 방향성이 얼마나 유사한지를 측정합니다.

두 샘플 $\boldsymbol{a} = (a_1, a_2, \cdots, a_m)$와 $\boldsymbol{b} = (b_1, b_2, \cdots, b_m)$에 대해, 코사인 유사도는 다음과 같이 정의됨



$$\cos(\theta) = \frac{\sum_{j=1}^{m} a_j \times b_j}{\sqrt{\sum_{j=1}^{m} (a_j)^2} \times \sqrt{\sum_{j=1}^{m} (b_j)^2}}$$

- 두 샘플 간 사이각만을 기준으로 두 샘플의 유사도를 측정하므로, 특징의 스케일과 직선거리에 영향을 받지 않음
- 예를 들어, a와 c는 코사인 유사도가 높고 유클리디안 거리는 먼 반면, a와 b는 코사인 유사도가 낮고 유클리디안 거리가 짧음

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

거리 및 유사도 척도
(4) 매칭 유사도

매칭 유사도는 전체 요소 가운데 일치하는 비율로 두 이진 벡터 간 유사도를 측정합니다

두 샘플 $\boldsymbol{a} = (a_1, a_2, \cdots, a_m)$와 $\boldsymbol{b} = (b_1, b_2, \cdots, b_m)$에 대해, 매칭 유사도는 다음과 같이 정의됨

$$\frac{1}{m}\sum_{j=1}^{m} I(a_j == b_j)$$

- $I(a_j == b_j)$: $a_j == b_j$이면 1을, 그렇지 않으면 0을 가짐
- 매우 직관적인 유사도 척도로 정확도와 그 개념이 유사함

(예시)

	1	2	3	4	5
a_j	0	1	1	0	1
b_j	0	0	1	1	1

매칭 유사도 = 3/5

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

거리 및 유사도 척도
(5) 자카드 유사도

자카드 유사도는 희소(sparse)한 데이터에 적합한 유사도 척도입니다.

두 샘플 $\boldsymbol{a} = (a_1, a_2, \cdots, a_m)$와 $\boldsymbol{b} = (b_1, b_2, \cdots, b_m)$에 대해, 자카드 유사도는 다음과 같이 정의됨

$$\frac{\sum_{j=1}^{m} I(a_j + b_j = 2)}{\sum_{j=1}^{m} I(a_j + b_j \geq 1)}$$

- 분모: 둘 중 하나라도 1을 갖는 요소의 개수
- 분자: 둘 다 1을 갖는 요소의 개수

자카드 유사도는 대부분 0을 갖는 희소(sparse)한 데이터에 적합합니다.

(예시) 대형 인터넷 쇼핑몰에서 판매하는 각 상품을 구매했는지를 바탕으로 고객을 특징화한 데이터

고객	상품 1	상품 2	상품 3	상품 4	…	상품 100,000
A	0	0	1	1	0	0
B	1	0	1	0	0	0

- 고객 A와 B의 매칭 유사도: 0.99998
- 두 고객 모두 구매한 상품은 하나(상품3)밖에 없는데도 유사도가 이렇게 높은 것은 논리적으로 어색함
- 같은 상품을 구매하지 않은 것이 유사하다고 보긴 어려움
- 따라서 고객 A와 B 중 한 명이라도 구매한 상품인 1, 3, 4에 대해서만 유사도를 계산하는 것이 적절

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

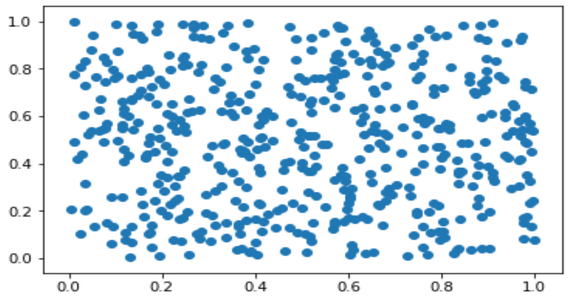
거리 및 유사도 척도
:요약

척도	계산식	사용하기 적합한 데이터 및 상황
유클리디안 거리	$\sqrt{\sum_{j=1}^{m}(a_j - b_j)^2}$	• 모든 특징이 연속형인 데이터 • 다른 거리 척도를 사용하기 애매한 상황
맨하탄 거리	$\sum_{j=1}^{m}\lvert a_j - b_j\rvert$	• 모든 특징이 정수형인 데이터
코사인 유사도	$\frac{\sum_{j=1}^{m} a_j \times b_j}{\sqrt{\sum_{j=1}^{m}(a_j)^2} \times \sqrt{\sum_{j=1}^{m}(b_j)^2}}$	• 방향 유사도를 측정해야 하는 상황 (예: 고객 유사도 측정 등)
매칭 유사도	$\frac{\sum_{j=1}^{m} C(a_j = b_j)}{m}$	• 희소하지 않은 이진 데이터
자카드 유사도	$\frac{\sum_{j=1}^{m} C(a_j + b_j = 2)}{\sum_{j=1}^{m} C(a_j + b_j \geq 1)}$	• 희소한 이진 데이터

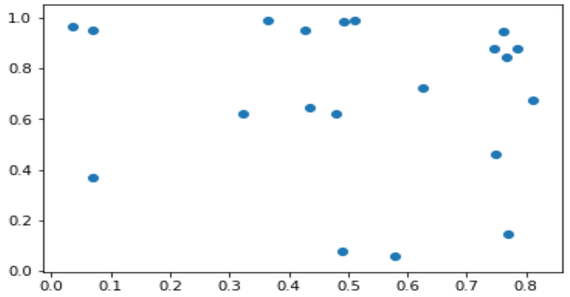
Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

이웃 수

최적의 k는 데이터 공간 크기 대비 샘플 수인 데이터 밀도에 비례합니다.



고밀도 데이터



저밀도 데이터

- 데이터 특성에 따른 최적의 k를 찾는 방법을 개발하고자 한 연구가 다수 있었으나, 모두 만족할만한 성과를 얻지 못함
- 데이터 특성에 따라 최적의 k를 찾으려면 가능한 모든 k를 비교해야 함
- 최적의 k는 데이터 공간 크기 대비 샘플 수인 데이터 밀도에 비례한다고 알려짐
- k가 클수록 한 샘플의 라벨을 예측하는 근거가 풍부하다고 할 수 있기 때문에 밀도가 높은 데이터에서는 k를 크게 설정해야 함
- 밀도가 낮은 데이터에서는 이웃까지 절대적 거리가 멀기 때문에 k를 작게 설정하는 것이 좋음
- 데이터 밀도가 매우 낮은 시계열 분류 문제를 해결하는데 k 최근접 이웃을 사용한다면 이웃 수를 1로 설정함

fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

2. k-최근접 이웃

3 스케일링과 특징 공학

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

스케일링 필요성 예제

k-최근접 이웃은 스케일링이 반드시 필요합니다.

	나이	소득(원)	연체 여부
고객 A	45	5천만	연체함
고객 B	30	1억	연체하지 않음
고객 C	30	6천만	?

k가 1이고 거리 척도가 맨하탄 거리인 k-최근접 이웃으로 연체 여부를 예측하는 예제

- 고객 C와 A의 거리: |30 – 45| + |6천만 - 5천만| = 1천만 15
- 고객 C와 B의 거리: |30-30| + |6천만 – 1억| = 4천만
- 고객 C의 이웃은 A이므로 연체할 것이라 예측함
- 나이의 스케일이 소득의 스케일보다 작아서, 소득보다 거리에 매우 적은 영향을 끼침을 알 수 있음

k-최근접 이웃은 스케일이 큰 특징에 크게 영향을 받으므로 스케일링을 반드시 해야 합니다.

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

스케일링 필요성 예제 (계속)

앞선 예제에서 스케일링을 하고 나니 결과가 달라짐을 알 수 있습니다.

	나이	소득(원)	연체 여부
고객 A	1	0	연체함
고객 B	0	1	연체하지 않음
고객 C	0	0.2	?

- 최소-최대 스케일링을 수행함
- 고객 C와 A의 거리: |1-0| + |0-0.2| = 1.2
- 고객 C와 B의 거리: |0-0| + |1-0.2| = 0.8
- 고객 C의 이웃은 B이므로 연체하지 않을 것이라 예측함
- 이번에는 나이도 소득과 비슷한 수준으로 이웃을 판단하는 데 기여함

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

특징 공학의 필요성

스케일링한다고 반드시 k-최근접 이웃 모델의 성능이 좋아진다고 보장할 수 없습니다.

	나이	소득(원)	연체 여부
고객 A	1	0	연체함
고객 B	0	1	연체하지 않음
고객 C	0	0.2	?

- 스케일링을 함으로써 나이도 예측에 활용하게 됨
- 만약 나이가 연체 여부와 무관한 특징이라면, 오히려 성능 저하로 이어질 수 있음

특징 공간이 잘 정의됐을 때만 스케일링이 유효하므로
k-최근접 이웃을 사용할 땐 스케일링뿐만 아니라 특징 공학도 같이 고려해야 합니다.

fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

2. k-최근접 이웃

4 사이킷런 실습

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

예제 데이터 불러오기

k-최근접 이웃 모델을 학습할 예제 데이터를 불러옵니다.

예제 데이터 불러오기

```
1  import pandas as pd
2  from sklearn.model_selection import train_test_split
3  df = pd.read_csv("../../data/classification/sonar.csv")
4  X = df.drop('y', axis = 1)
5  y = df['y']
6  X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 2022)
```

데이터 모양 확인

```
1  print(X_train.shape)
```

(156, 60)

- 샘플이 156개인데 비해 특징이 매우 많은 편임
- 즉, 데이터 밀도가 작다고 볼 수 있어, 이웃 수를 적게 잡는 것이 적절할 것으로 보임

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

모델 학습 및 평가

k-최근접 이웃 모델은 사이킷런의 neighbors 모듈로 구현할 수 있으며, 이 모듈에는 KNeighborsClassifier와 KNeighborsRegressor라는 클래스가 있습니다.

주요 인자

인자	설명	기본 값
n_neighbors	이웃 수	5
metric	거리 척도 ("euclidean", "manhattan", "matching", "jaccard")	“minkowski” [1]

k-최근접 이웃 모델 학습 예제

```
1  from sklearn.neighbors import KNeighborsClassifier as KNN
2  model = KNN(n_neighbors = 3).fit(X_train, y_train)
```

[1] 거리 척도의 기본값은 맨하탄 거리와 유클리디안 거리 등을 일반화한 거리 척도인 민코위스키 거리입니다. 그러나 인자 p의 기본값이 2라서 기본값이 유클리디안 거리라고 간주해도 무방합니다.

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

모델 학습 및 평가 (계속)

F1 점수를 사용해 모델을 평가하는 함수를 작성하고 활용하겠습니다.

모델 평가 예제

```
1  from sklearn.metrics import f1_score
2  def evaluate(model, X_test, y_test):
3      y_pred = model.predict(X_test)
4      score = f1_score(y_test, y_pred)
5      return score
6
7  score = evaluate(model, X_test, y_test)
8  print(score)
```

```
0.7906976744186046
```

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

스케일링 효과 확인

최소-최대 스케일링을 했을 때의 성능을 확인해보겠습니다.

스케일링 효과 확인

```
1  from sklearn.preprocessing import MinMaxScaler
2  scaler = MinMaxScaler().fit(X_train)
3  Z_train = scaler.transform(X_train)
4  Z_test = scaler.transform(X_test)
5
6  model2 = KNN(n_neighbors = 3).fit(Z_train, y_train)
7  score = evaluate(model2, Z_test, y_test)
8  print(score)
```

```
0.9047619047619048
```

- 스케일링하니 F1 점수가 0.79에서 0.90으로 0.11이나 증가했음
- 이는 스케일이 작으나 실제로 라벨을 예측하는데 효과적인 특징이 라벨 예측에 기여했기 때문이라 판단됨
- 단, 데이터에 따라서 스케일링 후 성능이 저하될 가능성도 있음에 주의해야 함