K-Nearest Neighbors (KNN) Algoritması

Atil Samancioglu

1 Giriş

K-En Yakın Komşu (KNN) algoritması, makine öğrenmesinde hem sınıflandırma hem de regresyon problemleri için kullanılan basit ama etkili bir algoritmadır. Özellikle anlaması ve uygulaması kolay olduğu için öğretici amaçlarla sıkça tercih edilir.

2 KNN ile Sınıflandırma

2.1 Temel Fikir

KNN algoritmasının sınıflandırma amacıyla kullanımı aşağıdaki adımlarla özetlenebilir:

1. Bir k değeri (komşu sayısı) belirlenir.
2. Yeni gelen test verisi için eğitim veri setindeki tüm örneklerle mesafe hesaplanır.
3. Mesafesi en yakın k örnek seçilir.
4. Bu k komşunun sınıf etiketleri sayılarak en sık görülen sınıf etiketi tahmin edilir.

2.2 Örnek

İki boyutlu bir uzayda, sınıfı 0 ve 1 olan veriler verilmiş olsun. Yeni bir test noktasının konumu verildiğinde, mesafesi en yakın 5 komşuya ($k = 5$) bakılır. Bu 5 komşunun çoğunluğu sınıf 1 ise, test noktasının sınıfı da 1 olarak atanır.

3 Mesafe Ölçümleri

KNN algoritması, test noktasına olan mesafeyi ölçerek en yakın komşuları bulur. En yaygın kullanılan iki mesafe ölçüsü şunlardır:

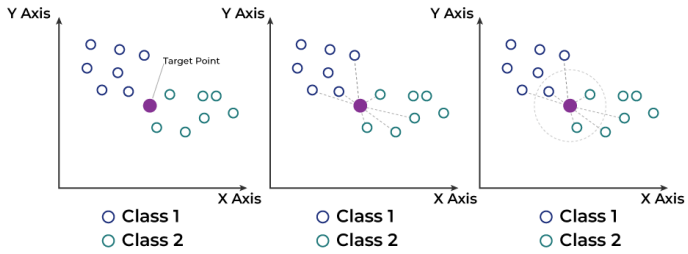

Figure 1: KNN Algoritması Geeks for Geeks görsel

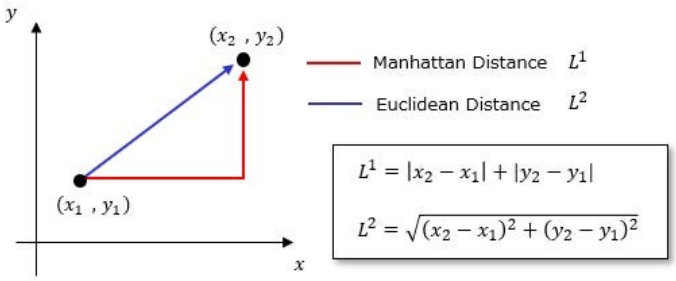

Figure 2: Manhattan vs Euclidean

3.1 Öklidyen Mesafesi (Euclidean Distance)

İki nokta arasındaki mesafe:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

n boyutlu uzayda genelleştirilmiş hali:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

3.2 Manhattan Mesafesi

$$d = |x_2 - x_1| + |y_2 - y_1|$$

Özellikle şehir bloklarında veya yalnızca yatay/dikey hareketin mümkün olduğu alanlarda tercih edilir.

4 Uygulamalı Örnek: KNN ile Sınıflandırma

Aşağıda, KNN algoritmasının manuel hesaplamayla nasıl çalıştığını gösteren küçük bir sınıflandırma örneği verilmiştir.

Veri Seti

İki özelliğe sahip 6 gözlem noktamız olduğunu düşünelim: Boy (*height*) ve Kilo (*weight*). Ayrıca her bireyin sporcu (*athlete*) olup olmadığı bilgisi (etiket) verilmiştir:

ID	Boy (cm)	Kilo (kg)	Athlete (Label)
1	170	65	Evet
2	165	72	Hayır
3	180	80	Evet
4	175	85	Hayır
5	160	60	Hayır
6	172	70	Evet

Şimdi yeni bir birey için tahmin yapalım:

$$\text{Yeni Kişi: Boy} = 168\,\text{cm}, \quad \text{Kilo} = 66\,\text{kg}$$

1. Adım: Mesafelerin Hesaplanması (Öklidyen)

$$\text{Mesafe} = \sqrt{(x_{\text{boy}} - x'_{\text{boy}})^2 + (x_{\text{kilo}} - x'_{\text{kilo}})^2}$$

Gözlem	Mesafe (Yeni kişiye)
1	$\sqrt{(170-168)^2 + (65-66)^2} = \sqrt{4+1} = \sqrt{5} \approx 2.24$
2	$\sqrt{(165-168)^2 + (72-66)^2} = \sqrt{9+36} = \sqrt{45} \approx 6.70$
3	$\sqrt{(180-168)^2 + (80-66)^2} = \sqrt{144+196} = \sqrt{340} \approx 18.44$
4	$\sqrt{(175-168)^2 + (85-66)^2} = \sqrt{49+361} = \sqrt{410} \approx 20.25$
5	$\sqrt{(160-168)^2 + (60-66)^2} = \sqrt{64+36} = \sqrt{100} = 10.00$
6	$\sqrt{(172-168)^2 + (70-66)^2} = \sqrt{16+16} = \sqrt{32} \approx 5.66$

2. Adım: En Yakın 3 Komşunun Belirlenmesi ($k = 3$)

En küçük 3 mesafe:

- Gözlem 1 (2.24) → Etiket: Evet
- Gözlem 6 (5.66) → Etiket: Evet
- Gözlem 2 (6.70) → Etiket: Hayır

3. Adım: Sınıf Etiketinin Belirlenmesi

3 komşudan 2 tanesi "Evet", 1 tanesi "Hayır" olduğuna göre:

Tahmin: Evet (Athlete)

Buradaki komşulardaki sınıfların sayılmasına oylama (voting) de denilebilir.

5 KNN ile Regresyon

KNN yalnızca sınıflandırma değil, regresyon problemlerinde de kullanılabilir. Sınıflandırma yönteminde en yakın komşular bulunup en çok komşunun hangi sınıfta olduğunu bulurken, regresyon yönteminde en yakın komşuların ortalaması alınarak tahmin edilen değer bulunur.

5.1 Temel Yöntem

Test verisinin k komşusu belirlendikten sonra, bu k komşunun hedef değişken değerlerinin ortalaması alınır:

$$\hat{y} = \frac{1}{k} \sum_{i=1}^{k} y_i$$

Not: Eğer veri setinde uç değerler varsa, ortalama yerine medyan kullanımı daha iyi sonuç verebilir.

6 KNN Optimizasyonu: KD-Tree ve Ball Tree

KNN algoritmasında test verisi için tüm eğitim verileriyle mesafe hesaplanması gerekir. Bu işlem büyük veri setlerinde zaman açısından oldukça maliyetlidir ($O(n)$ zaman karmaşıklığı). Bu sorunu azaltmak için iki veri yapısı kullanılabilir:

6.1 KD-Tree (k-dimensional tree)

KD-Tree, verileri k boyutlu bir uzayda bölerek ikili bir ağaç (binary tree) yapısında saklar.

Nasıl Çalışır?

1. İlk olarak tüm veriler bir düzlemde sıralanır.
2. İlk bölme, ilk özelliğin (örneğin *boy*) medyanına göre yapılır.
3. İkinci düzlem, ikinci özelliğin (örneğin *kilo*) medyanına göre yapılır.
4. Bu işlem sırayla (x, y, z, ...) eksenleri arasında geçiş yaparak tekrarlanır.

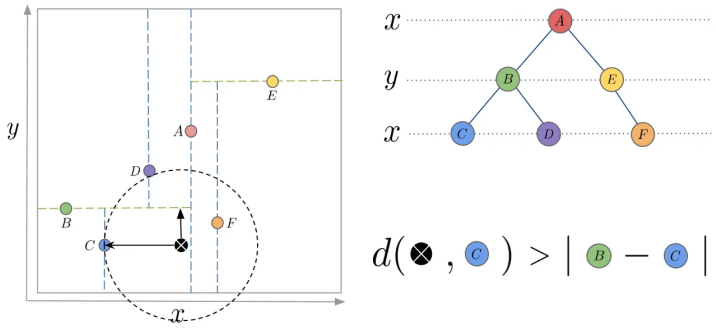

Figure 3: KD Tree Bhargamikumvar Görsel

5. Ağaç oluşturulduktan sonra, sorgu noktası sadece ilgili bölgelerdeki alt kümelerle karşılaştırılır.

Avantaj: Her sorguda yalnızca belirli bölgelerde arama yapıldığı için arama süresi önemli ölçüde azalır.

6.2 Ball Tree

Ball Tree, verileri küresel gruplar (toplar) halinde hiyerarşik olarak organize eder.

Nasıl Çalışır?

1. Benzer noktalar kümelere (ball) ayrılır.
2. Her küme kendi merkez noktasına ve yarıçapına sahiptir.
3. Yeni bir sorgu geldiğinde, sadece yakın kümeler (merkezleri sorguya yakın olan) kontrol edilir.
4. Uzakta kalan kümeler tamamen atlanabilir.

Avantaj: Özellikle yüksek boyutlu verilerde, Ball Tree yapılandırması KD-Tree'ye göre daha etkili olabilir.

Sonuç

KD-Tree ve Ball Tree gibi veri yapıları, KNN algoritmasını daha ölçeklenebilir hale getirir. Özellikle büyük veri kümelerinde sorgu süresini düşürür ve algoritmayı gerçek zamanlı kullanım için uygun hale getirir.

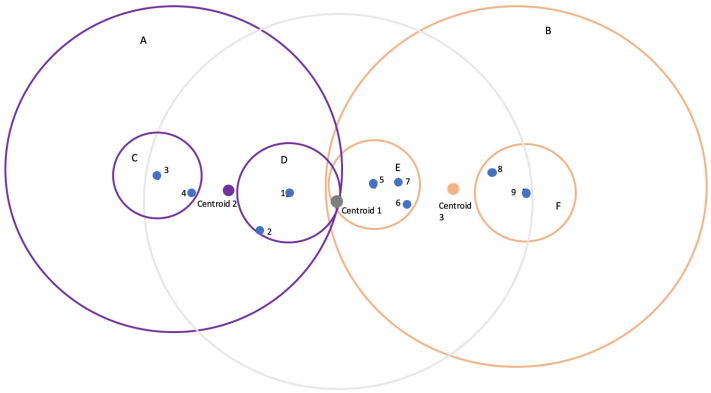

Figure 4: Ball Tree Bhargamikumvar Görsel

7 KNN için Uygunluk

KNN algoritması aşağıdaki durumlarda iyi çalışır:

- Veri boyutu çok yüksek değilse
- Özellikler benzer ölçekteyse (özellik mühendisliği ve ölçekleme yapılmışsa)
- Sınıflar arasındaki ayrım netse

8 Sonuç

KNN, basit yapısına rağmen etkili bir algoritmadır. Özellikle sınıflandırma problemlerinde görselleştirilebilir olması sayesinde öğretici bir model olarak sıkça tercih edilir. Mesafe ölçümüne dayandığı için veri ölçekleme ve öz nitelik mühendisliği kritik öneme sahiptir.