K-En Yakın Komşu

K-En Yakın Komşu (k-Nearest Neighbors, k-NN) algoritması, temel olarak bir sınıflandırma ve regresyon yöntemi olarak kullanılan basit ancak etkili bir makine öğrenimi tekniğidir. Veriye dayalı bir öğrenme yöntemi olan k-NN, yeni veri noktalarının etiketlenmesi veya değerlerinin tahmin edilmesi için kullanılır. İşte k-NN algoritmasının detayları:

k-NN Algoritmasının Çalışma Prensibi

1. **Veriye Dayalı Öğrenme:** k-NN algoritması, veri setindeki örneklerin etrafındaki benzer örnekleri kullanarak tahmin yapar. Veri seti, özellikler ve hedef değişken (etiket) olmak üzere iki bileşenden oluşur.
2. **Temel İlke:** Bir veri noktasının sınıflandırılması veya bir hedef değişkenin tahmin edilmesi, bu veri noktasına en yakın k sayıda komşusunun etiketlerine veya değerlerine dayanır. Buradaki k, komşuluk boyutunu ve bu komşuların sayısını belirler.
3. **Komşuluk Ölçümü:** k-NN'de genellikle Euclidean mesafe, Manhattan mesafe veya diğer benzer ölçümler kullanılarak komşuluk hesaplanır. Veri noktaları, özellik uzayında bu mesafelere göre birbirlerine ne kadar yakın olduklarına göre sıralanır.
4. **Sınıflandırma:** Sınıflandırma için, bir veri noktasının etiketi genellikle k en yakın komşusunun sınıflarının çoğunluk oyu ile belirlenir. Yani, en yakın k komşunun etiketlerinin hangi sınıfta daha fazla olduğuna göre veri noktası o sınıfa atanır.
5. **Regresyon:** Regresyon için, bir veri noktasının hedef değişken değeri genellikle k en yakın komşusunun ortalama değeri veya ağırlıklı ortalaması kullanılarak tahmin edilir. Ağırlıklı ortalamada, komşuların mesafelerine göre ağırlıklar atanabilir.

Parametreler ve Değişkenler

- **k Değeri:** k-NN algoritmasında en önemli parametre k'dır. Bu değer, komşuluk boyutunu ve algoritmanın performansını belirler. Küçük k değerleri, modele gürültüye duyarlılık ekleyebilirken, büyük k değerleri daha pürüzsüz bir karar sınırı sınırlamasına yol açabilir.
- **Uzaklık Ölçümü:** Komşuluk hesaplaması için kullanılan uzaklık metriği seçimi önemlidir. Özellikle veri setinin özellikleri ve dağılımı göz önünde bulundurularak Euclidean mesafe veya Manhattan mesafesi gibi uygun bir metrik seçilmelidir.

Avantajlar

- **Basitlik ve Etkinlik:** k-NN, uygulaması kolay ve anlaşılması basit bir algoritmadır. Ayrıca, eğitim süresi yoktur, çünkü model eğitimi sırasında veri seti tamamen yüklenir ve eğitilir.
- **Non-parametrik Yapı:** k-NN, modelin veriye uyum yeteneğini artırarak, veri setindeki karmaşıklıkları ve örüntüleri daha iyi yakalayabilir.

Dezavantajlar

- **Yüksek Hesaplama Maliyeti:** Büyük veri setleri veya çok sayıda özellik içeren veri setleri için hesaplama maliyeti yüksek olabilir. Çünkü her tahmin için tüm veri noktalarının mesafesi hesaplanmalıdır.

- **Boyutluluk Sorunu:** Özellik uzayındaki yüksek boyutluluk, k-NN'nin performansını etkileyebilir. Çünkü yüksek boyutlarda uzaklık hesaplama ve komşuluk belirleme zorlaşabilir.

Uygulama Alanları

- **Tıbbi Teşhisler:** Örneğin, bir hastanın teşhisini benzer hastaların verilerine dayanarak yapabilir.
- **Müşteri Segmentasyonu:** Benzer tüketici profilleri oluşturarak pazarlama stratejileri geliştirebilir.
- **Tavsiye Sistemleri:** Benzer kullanıcıların tercihlerine dayalı olarak ürün veya içerik önerileri yapabilir.

k-NN algoritması, bu esnekliği ve basitliği nedeniyle geniş bir kullanım alanına sahiptir, ancak veri setinin büyüklüğü, boyutluluğu ve veri dağılımı gibi faktörler model performansını etkileyebilir. Bu nedenle, kullanmadan önce veri setinin özelliklerini ve algoritmanın parametrelerini dikkatlice değerlendirmek önemlidir.

Örnek

K-En Yakın Komşu (KNN) algoritması, bir sınıflandırma veya regresyon problemi için kullanılan basit ve etkili bir makine öğrenmesi algoritmasıdır. İşte KNN algoritmasını kullanarak bir sınıflandırma örneği için Python kodu:

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

# Örnek veri seti oluşturma
X, y = make_blobs(n_samples=100, centers=3, random_state=42)

# Veri setini eğitim ve test kümelerine ayırma
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# KNN modelini oluşturma ve eğitme
k = 3  # K değeri (komşu sayısı)
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train, y_train)

# Test veri seti üzerinde tahmin yapma
y_pred = knn.predict(X_test)

# Doğruluk skorunu hesaplama
accuracy = accuracy_score(y_test, y_pred)
print(f"Doğruluk: {accuracy:.2f}")
```

```
# Sınıfları ve karar sınırlarını görselleştirme
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1),
                     np.arange(y_min, y_max, 0.1))
Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, alpha=0.8)
plt.scatter(X[:, 0], X[:, 1], c=y, s=30, edgecolors='k')
plt.title(f'K-En Yakın Komşu (K={k})')
plt.xlabel('Özellik 1')
plt.ylabel('Özellik 2')
plt.show()
```

Açıklamalar

1. **Kütüphanelerin İmport Edilmesi:**
   - `numpy` ve `matplotlib.pyplot` matematik işlemleri ve veri görselleştirme için kullanılır.
   - `sklearn.datasets` örnek veri seti oluşturmak için kullanılır.
   - `sklearn.model_selection.train_test_split` veri setini eğitim ve test kümelerine ayırmak için kullanılır.
   - `sklearn.neighbors.KNeighborsClassifier` KNN modelini oluşturmak için kullanılır.
   - `sklearn.metrics.accuracy_score` doğruluk skoru hesaplamak için kullanılır.
2. **Örnek Veri Seti Oluşturma:**
   - `make_blobs` fonksiyonu belirli bir dağılıma sahip örnek veri setleri oluşturur.
3. **Veri Setinin Eğitim ve Test Kümelerine Ayrılması:**
   - `train_test_split` fonksiyonu veri setini eğitim ve test kümelerine ayırır.
4. **KNN Modelinin Oluşturulması ve Eğitilmesi:**
   - `KNeighborsClassifier` sınıfı, KNN modelini oluşturur ve `fit` fonksiyonu ile eğitir.
5. **Tahmin ve Doğruluk Skorunun Hesaplanması:**
   - `predict` fonksiyonu test veri seti üzerinde tahmin yapar.
   - `accuracy_score` fonksiyonu tahmin doğruluğunu hesaplar.
6. **Görselleştirme:**
   - Karar sınırlarını (`contourf` fonksiyonu) ve veri noktalarını (`scatter` fonksiyonu) görselleştirir.

Bu kod örneği, KNN algoritmasının nasıl kullanıldığını ve sınıflandırma problemleri için nasıl uygulandığını göstermektedir. Veri seti ve K değeri (`k`) gibi parametreler değiştirilerek farklı senaryolar üzerinde denemeler yapabilirsiniz.