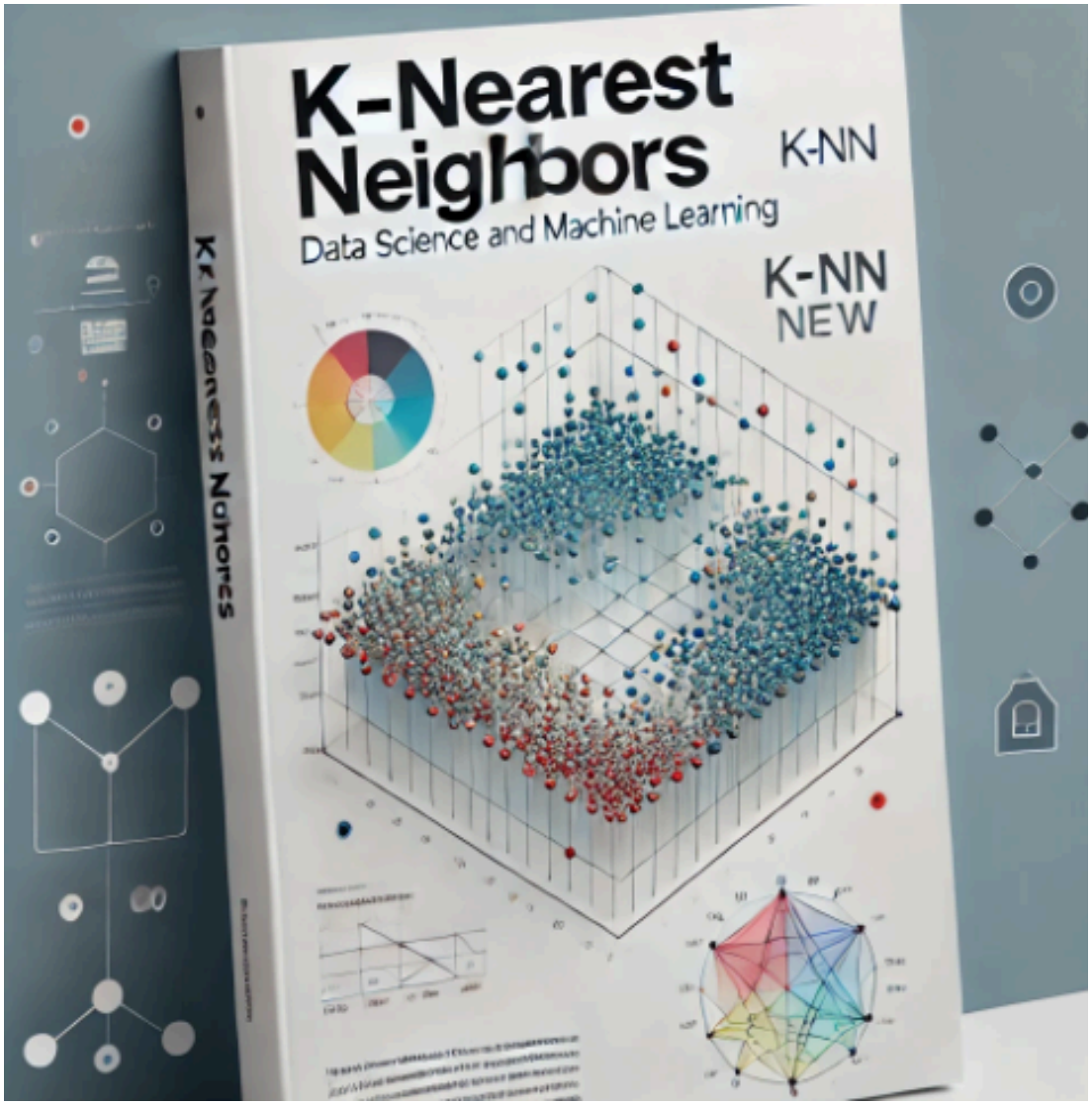
K-Nearest Neighbor

Materi pembelajaran tentang algoritma learning



Muchamad Kurniawan

#informatika_ITATS

PENDAHULUAN

K-Nearest Neighbors (K-NN) adalah algoritma klasifikasi yang sangat populer dalam pembelajaran mesin, khususnya dalam kategori supervised learning. Algoritma ini pertama kali dikembangkan oleh Evelyn Fix dan Joseph Hodges pada tahun 1951. K-NN bekerja dengan cara mengklasifikasikan suatu data berdasarkan kedekatan jarak dengan data lain dalam kelompok tertentu.

Prinsip Dasar dan Konsep

Prinsip dasar K-NN adalah memanfaatkan jarak untuk menentukan kelompok atau kelas suatu data. Algoritma ini menghitung jarak antara data baru yang akan diklasifikasikan dengan data yang ada pada dataset, kemudian memilih sejumlah tetangga terdekat yang ditentukan oleh parameter kkk. Berdasarkan mayoritas kelas dari kkk tetangga tersebut, data baru diklasifikasikan.

Filosofi

K-NN berlandaskan filosofi bahwa data yang memiliki kesamaan karakteristik akan berada dalam kelompok yang sama atau memiliki kedekatan jarak. Oleh karena itu, semakin dekat suatu titik data dengan titik lain yang telah diketahui kelasnya, semakin besar kemungkinan keduanya berada dalam kelas yang sama.

FORMULA DAN PROSEDUR

Formula Jarak

Euclidean Distance

Dua data, misalnya data A dan data B dengan koordinat (x_1, y_1) dan (x_2, y_2) dalam ruang dua dimensi, memiliki jarak Euclidean sebagai berikut:

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

Prosedur Algoritma K-NN

1. Tentukan nilai k (jumlah tetangga terdekat).
2. Hitung jarak antara data baru dengan seluruh data yang ada dalam dataset.
3. Urutkan data berdasarkan jarak terdekat.
4. Pilih k data terdekat.
5. Tentukan kelas berdasarkan mayoritas kelas dari k data tersebut.

Jarak Manhattan

Menghitung jarak dalam ruang grid atau "kotak-kotak" seperti jalan-jalan di kota. Jarak Manhattan untuk dua titik A dan B adalah:

$$d(A, B) = \sum_{i=1}^{n} |x_i - y_i|$$

Jarak Minkowski

Generalisasi dari jarak Euclidean dan Manhattan. Jika $p = 2$, hasilnya adalah jarak Euclidean; jika $p = 1$, hasilnya adalah jarak Manhattan.

$$d(A, B) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

Jarak Chebyshev

Mengukur jarak maksimum di sepanjang satu dimensi koordinat. Sering digunakan dalam perhitungan untuk papan catur atau dalam kasus jarak tak terbatas.

$$d(A, B) = \max(|x_i - y_i|)$$

Jarak Cosine

Digunakan untuk mengukur kesamaan antara dua vektor dalam ruang berdimensi tinggi. Dihitung berdasarkan sudut antara dua vektor, bukan panjang jarak absolut.

$$d(A, B) = 1 - \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \cdot \sqrt{\sum_{i=1}^{n} y_i^2}}$$

Jarak Mahalanobis

Cocok untuk data dengan distribusi variabel berbeda. Jarak ini mempertimbangkan korelasi antara variabel.

$$d(A, B) = \sqrt{(X - Y)^T S^{-1} (X - Y)}$$

Di mana S adalah matriks kovarians dari data.

Prosedur

Prosedur Algoritma K-NN

1. Tentukan nilai *k* (jumlah tetangga terdekat).
2. Hitung jarak antara data baru dengan seluruh data yang ada dalam dataset.
3. Urutkan data berdasarkan jarak terdekat.
4. Pilih *k* data terdekat.
5. Tentukan kelas berdasarkan mayoritas kelas dari *k* data tersebut.

Contoh Penyelesaian Manual

Misalkan terdapat lima data dengan koordinat sebagai berikut, dan kita ingin menentukan kelas dari titik baru X dengan koordinat (3,3) serta k=3:

Data	Koordinat (X, Y)	Kelas
A	(1, 2)	Biru
B	(4, 2)	Merah
C	(1, 3)	Biru
D	(2, 5)	Biru
E	(5, 3)	Merah

Hitung jarak antara titik X dengan masing-masing data.

Urutkan berdasarkan jarak terdekat.

Pilih tiga data dengan jarak terdekat (misalnya, A, C, D).

Tentukan kelas mayoritas dari ketiga data tersebut. Jika mayoritasnya adalah "Biru," maka titik X diklasifikasikan sebagai "Biru."

1. **Hitung Jarak Euclidean dari Titik X ke Setiap Titik Dataset**

   Rumus Euclidean Distance:

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

   - **Jarak ke Titik A (1, 2):**

$$d(X,A) = \sqrt{(3-1)^2 + (3-2)^2} = \sqrt{2^2 + 1^2} = \sqrt{4+1} = \sqrt{5} \approx 2.24$$

   - **Jarak ke Titik B (4, 2):**

$$d(X,B) = \sqrt{(3-4)^2 + (3-2)^2} = \sqrt{(-1)^2 + 1^2} = \sqrt{1+1} = \sqrt{2} \approx 1.41$$

   - **Jarak ke Titik C (1, 3):**

$$d(X,C) = \sqrt{(3-1)^2 + (3-3)^2} = \sqrt{2^2 + 0^2} = \sqrt{4} = 2.00$$

   - **Jarak ke Titik D (2, 5):**

$$d(X,D) = \sqrt{(3-2)^2 + (3-5)^2} = \sqrt{1^2 + (-2)^2} = \sqrt{1+4} = \sqrt{5} \approx 2.24$$

   - **Jarak ke Titik E (5, 3):**

$$d(X,E) = \sqrt{(3-5)^2 + (3-3)^2} = \sqrt{(-2)^2 + 0^2} = \sqrt{4} = 2.00$$

2. **Urutkan Jarak**

   Berikut daftar jarak dari titik X ke setiap titik dataset:

Data	Jarak ke X	Kelas
B	1.41	Merah
C	2.00	Biru
E	2.00	Merah
A	2.24	Biru
D	2.24	Biru

3. **Pilih $k = 3$ Titik Terdekat**

   Ambil tiga titik dengan jarak terdekat:

Data	Jarak ke X	Kelas
B	1.41	Merah
C	2.00	Biru
E	2.00	Merah

4. **Tentukan Kelas Berdasarkan Mayoritas**

   Dari tiga titik terdekat, kelasnya adalah:

   - Merah: 2 (B, E)
   - Biru: 1 (C)

   Mayoritas adalah Merah. Maka, titik X diklasifikasikan sebagai **Merah**.

Implementasi Program K-NN dengan Python

Berikut adalah implementasi sederhana K-NN dalam Python:

```
import numpy as np
from collections import Counter

# Fungsi untuk menghitung jarak Euclidean
def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))

# Kelas untuk algoritma KNN
class KNN:
    def __init__(self, k=3):
        self.k = k
```

```
    def fit(self, X_train, y_train):
        self.X_train = X_train
        self.y_train = y_train

    def predict(self, X_test):
        predictions = [self._predict(x) for x in X_test]
        return np.array(predictions)

    def _predict(self, x):
        distances = [euclidean_distance(x, x_train) for x_train in self.X_train]
        k_indices = np.argsort(distances)[:self.k]
        k_nearest_labels = [self.y_train[i] for i in k_indices]
        most_common = Counter(k_nearest_labels).most_common(1)
        return most_common[0][0]

# Data contoh
X_train = np.array([[1, 2], [4, 2], [1, 3], [2, 5], [5, 3]])
y_train = np.array(['Biru', 'Merah', 'Biru', 'Biru', 'Merah'])
X_test = np.array([[3, 3]])

# Menjalankan algoritma KNN
knn = KNN(k=3)
knn.fit(X_train, y_train)
predictions = knn.predict(X_test)

print("Kelas prediksi untuk data uji:", predictions[0])
```

Topik Skripsi Populer Menggunakan K-NN

Metode K-NN banyak digunakan dalam penelitian data mining, pengolahan data, dan pembelajaran mesin. Berikut adalah beberapa topik skripsi yang relevan dan populer untuk K-NN:

1. **Klasifikasi Sentimen dalam Teks**
   Menyelesaikan masalah klasifikasi teks, seperti sentimen positif, negatif, atau netral pada ulasan produk, komentar media sosial, dan artikel berita.
2. **Deteksi Penyakit Berdasarkan Data Medis**
   Penggunaan K-NN untuk mendiagnosis penyakit berdasarkan parameter kesehatan, seperti prediksi diabetes atau penyakit jantung berdasarkan data pasien.
3. **Sistem Rekomendasi Produk atau Konten**

Menggunakan K-NN untuk memberikan rekomendasi produk atau konten kepada pengguna berdasarkan preferensi atau riwayat belanja pengguna lain yang serupa.

4. **Pengklasifikasian Gambar dalam Computer Vision**
   Menggunakan K-NN untuk mengenali objek dalam gambar atau video, seperti mengenali wajah atau jenis objek dalam sistem pengawasan otomatis.
5. **Prediksi Harga Properti atau Saham**
   Menggunakan K-NN untuk memprediksi harga rumah atau harga saham berdasarkan karakteristik lokasi atau data historis.
6. **Klasifikasi Jenis Tanaman atau Hama dalam Pertanian**
   Menerapkan K-NN untuk mengidentifikasi jenis tanaman atau hama berdasarkan fitur seperti warna, ukuran, dan tekstur pada citra atau foto.
7. **Analisis Penipuan Kartu Kredit**
   Mendeteksi transaksi yang mencurigakan berdasarkan pola transaksi sebelumnya, berguna dalam meningkatkan keamanan finansial.
8. **Prediksi Kelulusan Mahasiswa Berdasarkan Data Akademik**
   Menganalisis data mahasiswa seperti nilai, tingkat absensi, dan kegiatan ekstrakurikuler untuk memprediksi tingkat kelulusan atau kesuksesan akademik.
9. **Pengenalan Suara**
   K-NN dapat digunakan untuk mengenali pola suara untuk penerapan dalam sistem seperti kontrol suara atau deteksi kebisingan.
10. **Analisis Data Geospasial**
    Menerapkan K-NN pada data geospasial untuk identifikasi titik lokasi, seperti pengelompokan tempat wisata atau analisis pola penyebaran penyakit dalam epidemiologi.

Kesimpulan

K-Nearest Neighbors (K-NN) adalah algoritma yang sederhana namun sangat efektif untuk klasifikasi. Dengan menggunakan jarak sebagai penentu kemiripan antar data, K-NN dapat digunakan dalam berbagai bidang, seperti pengenalan pola dan klasifikasi gambar. Namun, K-NN juga memiliki kelemahan dalam hal efisiensi jika jumlah data sangat besar, serta rentan terhadap outlier dan pemilihan nilai kkk yang kurang tepat. Implementasi dalam Python juga cukup sederhana, dan dengan pemahaman mendalam tentang konsep K-NN, kita bisa mengaplikasikannya dalam berbagai kasus klasifikasi.