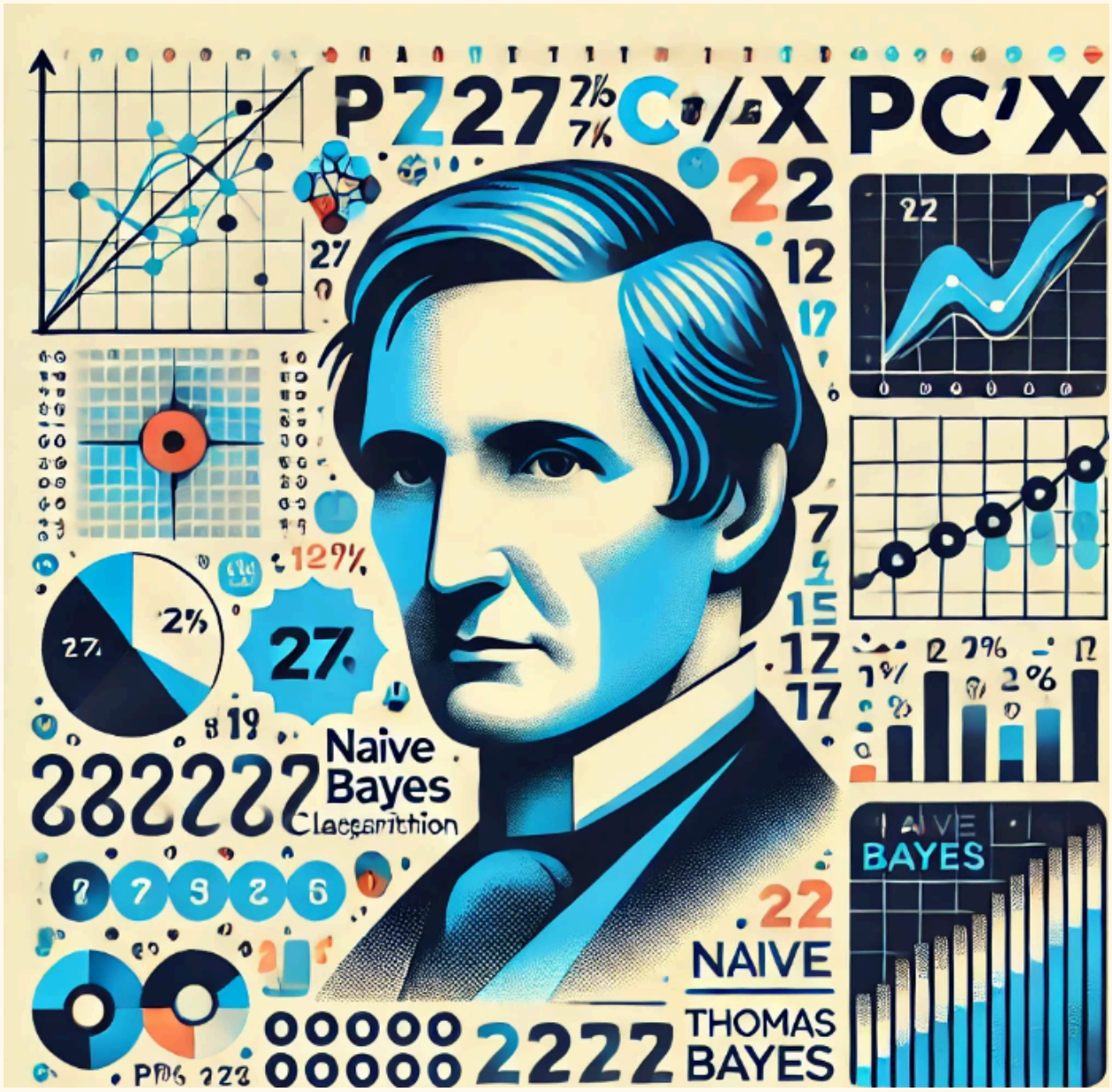
Naive Bayes Classification

By Muchamad Kurniawan



Algoritma Naive Bayes adalah salah satu algoritma pembelajaran mesin paling mendasar dan banyak digunakan, terutama dalam klasifikasi teks dan analisis sentimen. Artikel ini membahas sejarah, konsep dasar, formula, pseudocode, serta contoh penerapan algoritma Naive Bayes, baik secara manual maupun menggunakan program Python.

1. Sejarah dan Filosofi Algoritma Naive Bayes

Algoritma Naive Bayes didasarkan pada Teorema Bayes, yang diperkenalkan oleh seorang matematikawan Inggris, **Thomas Bayes**, pada abad ke-18. Teorema Bayes awalnya dikembangkan untuk memperkirakan probabilitas dari suatu peristiwa yang terjadi berdasarkan informasi atau bukti yang ada. Pada akhir abad ke-20, konsep ini diadaptasi ke dalam pembelajaran mesin oleh para ilmuwan komputer untuk membuat algoritma yang mampu "belajar" dari data dan melakukan klasifikasi.

Filosofi di balik Naive Bayes:

- Naive Bayes didasarkan pada konsep sederhana: semua fitur dianggap independen satu sama lain. Ini berarti bahwa kemunculan satu fitur tidak bergantung pada kemunculan fitur lainnya. Meskipun asumsi independensi ini sering kali tidak realistis, dalam banyak kasus, algoritma ini tetap memberikan performa yang baik.
- Filosofi probabilistik yang mendasari Naive Bayes memungkinkan algoritma ini untuk menimbang kemungkinan berdasarkan informasi yang ada, sehingga menghasilkan klasifikasi yang akurat meski dengan data terbatas.

2. **Konsep dan Prinsip Dasar Naive Bayes**

Naive Bayes menggunakan probabilitas untuk melakukan prediksi. Ia menghitung peluang bahwa suatu data masuk ke dalam kategori tertentu berdasarkan fitur-fitur yang dimilikinya. Dalam klasifikasi, Naive Bayes mencoba memprediksi kelas dari suatu data baru berdasarkan probabilitas kelas dari data pelatihan.

Jika kita memiliki dua kelas C_1 dan C_2 serta data X, Naive Bayes menghitung probabilitas dari setiap kelas yang ada:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

Di mana:

- $P(C|X)$ adalah probabilitas suatu kelas C diberikan data X.
- $P(X|C)$ adalah probabilitas data X muncul jika kelasnya C.
- $P(C)$ adalah probabilitas a priori dari kelas C.
- $P(X)$ adalah probabilitas total dari data X.

3. Formula Naive Bayes

Untuk dataset dengan beberapa fitur, kita asumsikan bahwa setiap fitur x_i dalam $X = \{x_1, x_2, \ldots, x_n\}$ adalah independen. Maka, kita dapat menulis:

$$P(C|X) \propto P(C) \prod_{i=1}^{n} P(x_i|C)$$

Naive Bayes memilih kelas C dengan probabilitas tertinggi berdasarkan produk dari peluang tiap fitur x_i dalam data.

Jenis-jenis Naive Bayes:

1. Multinomial Naive Bayes: Digunakan untuk data yang berbentuk frekuensi atau jumlah kejadian.
2. Bernoulli Naive Bayes: Digunakan untuk data biner atau Boolean.
3. Gaussian Naive Bayes: Digunakan untuk data kontinu yang mengikuti distribusi normal.

4. Pseudocode Naive Bayes Classification

Berikut adalah pseudocode untuk algoritma Naive Bayes:

1. **Inisialisasi**: Tentukan kelas dan fitur yang tersedia dalam dataset.
2. **Hitung Prior**: Hitung probabilitas prior dari setiap kelas.

3. **Hitung Likelihood**: Hitung probabilitas setiap fitur xix_ixi untuk setiap kelas.
4. **Prediksi**:
    - Untuk data baru, hitung P(C∣X)P(C|X)P(C∣X) untuk setiap kelas.
    - Pilih kelas dengan probabilitas terbesar.

Pseudocode:

```
1. Tentukan kelas C = {C1, C2, ..., Ck}
2. Hitung prior P(Ck) untuk setiap kelas Ck
3. Untuk setiap fitur x dalam X:
   - Hitung P(x | Ck) untuk setiap kelas
4. Untuk data baru X:
   - Untuk setiap kelas Ck:
     - Hitung likelihood P(Ck | X) = P(Ck) * P(x1 | Ck) * P(x2 | Ck) * ... * P(xn | Ck)
   - Pilih kelas dengan probabilitas tertinggi
```

5. Contoh Penerapan dengan Dataset Dummy

Misalkan kita memiliki dataset sederhana berikut dengan dua fitur: **Hujan** dan **Temperatur**, dan dua kelas: **Keluar** dan **Tidak Keluar**.

Hujan	Temperatur	Keluar
Ya	Dingin	Tidak
Tidak	Hangat	Keluar
Ya	Hangat	Keluar
Tidak	Dingin	Tidak
Ya	Hangat	Keluar
Tidak	Hangat	Keluar

Langkah 1: Hitung Prior untuk Setiap Kelas

$$P(Keluar) = \frac{4}{6} = 0.67, \quad P(Tidak\ Keluar) = \frac{2}{6} = 0.33$$

Langkah 2: Hitung Likelihood untuk Setiap Fitur Berdasarkan Kelas

1. **Probabilitas Hujan**
   - $P(Hujan = Ya|Keluar) = \frac{2}{4} = 0.5$
   - $P(Hujan = Ya|Tidak\ Keluar) = \frac{1}{2} = 0.5$
2. **Probabilitas Temperatur**
   - $P(Temperatur = Hangat|Keluar) = \frac{3}{4} = 0.75$
   - $P(Temperatur = Hangat|Tidak\ Keluar) = \frac{1}{2} = 0.5$

Contoh Prediksi untuk Data Baru: (Hujan=Ya, Temperatur=Hangat)

- Untuk kelas **Keluar**:

$$P(Keluar|X) = P(Keluar) \times P(Hujan = Ya|Keluar) \times P(Temperatur = Hangat|Keluar)$$
$$= 0.67 \times 0.5 \times 0.75 = 0.25125$$

- Untuk kelas **Tidak Keluar**:

$$P(Tidak\ Keluar|X) = P(Tidak\ Keluar) \times P(Hujan = Ya|Tidak\ Keluar) \times P(Temperatur = Hangat|Tidak\ Keluar)$$
$$= 0.33 \times 0.5 \times 0.5 = 0.0825$$

Kesimpulan: Karena $P(Keluar|X) > P(Tidak\ Keluar|X)$, kita memprediksi kelas **Keluar**.

6. **Implementasi Naive Bayes dengan Program Python**

Berikut adalah implementasi sederhana Naive Bayes dengan Python menggunakan *Scikit-Learn*.

```
# Import library yang diperlukan
from sklearn.naive_bayes import MultinomialNB
from sklearn.preprocessing import LabelEncoder
import numpy as np

# Dataset Dummy
```

```
data = [
    ['Ya', 'Dingin', 'Tidak'],
    ['Tidak', 'Hangat', 'Keluar'],
    ['Ya', 'Hangat', 'Keluar'],
    ['Tidak', 'Dingin', 'Tidak'],
    ['Ya', 'Hangat', 'Keluar'],
    ['Tidak', 'Hangat', 'Keluar']
]

# Preprocess data dengan mengonversi nilai string ke angka
le = LabelEncoder()
X = np.array([le.fit_transform([row[0], row[1]]) for row in data])
y = le.fit_transform([row[2] for row in data])

# Membuat model Naive Bayes dan melatihnya
model = MultinomialNB()
model.fit(X, y)

# Prediksi untuk data baru (Hujan=Ya, Temperatur=Hangat)
new_data = le.transform(['Ya', 'Hangat']).reshape(1, -1)
prediction = model.predict(new_data)

# Output prediksi
```

```
kelas_prediksi = le.inverse_transform(prediction)

print("Prediksi kelas untuk data baru:", kelas_prediksi[0])
```

Penjelasan:

1. **Preprocessing**: Dataset dikonversi ke angka menggunakan *LabelEncoder* untuk mempermudah pemrosesan.
2. **Model**: Menggunakan MultinomialNB dari Scikit-Learn yang sesuai untuk data kategori.
3. **Prediksi**: Data baru diklasifikasikan dengan model yang telah dilatih.

Output: Program akan mengeluarkan kelas prediksi untuk data baru.