Naive Bayes Classifier

Naive Bayes adalah algoritma klasifikasi yang sederhana namun efektif, berdasarkan *Teorema Bayes*. Algoritma ini mengasumsikan bahwa fitur-fitur dalam data saling independen (naif) dalam konteks kelas tertentu. Meskipun asumsi ini jarang terpenuhi dalam dunia nyata, Naive Bayes seringkali bekerja dengan baik dalam praktiknya.

# Cara Kerja Naive Bayes:

1. **Perhitungan Probabilitas Prior:** Menghitung probabilitas awal dari setiap kelas (misalnya, probabilitas seseorang membeli produk atau tidak) berdasarkan data latih.
2. **Perhitungan Probabilitas Likelihood:** Menghitung probabilitas setiap fitur dalam suatu kelas (misalnya, probabilitas seseorang berusia tertentu diberikan bahwa mereka membeli produk).
3. **Perhitungan Probabilitas Posterior:** Menggunakan *Teorema Bayes* untuk menghitung probabilitas posterior dari setiap kelas (misalnya, probabilitas seseorang membeli produk diberikan usia dan pendapatan mereka).
4. **Prediksi:** Memilih kelas dengan probabilitas posterior tertinggi sebagai prediksi.

# Jenis-jenis Naive Bayes:

1. **Gaussian Naive Bayes:** Mengasumsikan bahwa fitur numerik dalam setiap kelas mengikuti distribusi Gaussian (normal).
2. **Multinomial Naive Bayes** Mengasumsikan bahwa fitur diskrit merepresentasikan frekuensi kemunculan (seperti dalam klasifikasi teks).
3. **Bernoulli Naive Bayes:** Mengasumsikan bahwa fitur biner (0 atau 1), cocok untuk data seperti teks yang direpresentasikan dengan model ***“bag of world”***.

# Keunggulan Naive Bayes:

* **Sederhana dan Mudah Dipahami:** Konsep dan implementasinya relatif sederhana.
* **Efisien:** Cepat untuk dilatih dan membuat prediksi.
* **Cocok untuk Masalah Klasifikasi Teks:** *Multinomial dan Bernoulli Naive Bayes* sering digunakan untuk klasifikasi teks, seperti spam filtering atau analisis sentimen.
* **Bekerja Baik dengan Data Dimensi Tinggi:** Dapat menangani data dengan banyak fitur.

# Kelemahan Naive Bayes:

* **Asumsi Independensi yang Kuat:** Asumsi bahwa fitur-fitur saling independen jarang terpenuhi dalam dunia nyata.
* **Kurang Akurat pada Data yang Kompleks:** Mungkin tidak berkinerja baik pada data dengan hubungan antar fitur yang kompleks.

**Tugas Hari 27:**

Anda akan bekerja dengan dataset “Spambase” yang dapat diunduh dari UCI Machine Learning Repository: [Spambase - UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/datasets/spambase)

Dataset ini berisi email yang diberi label sebagai spam atau non-spam. Fitur-fiturnya adalah frekuensi kemunculan kata-kata tertentu dalam email.

**Tugas:**

1. **Memuat dan Menampilkan Data:**

* Unduh dataset “Spambase”.
* Gunakan Pandas untuk membaca dataset.
* Tampilkan 5 baris pertama dari dataset

1. **Membuat Model Gaussian Naive Bayes:**

* Gunakan GaussianNB dari scikit-learn untuk membuat model.
* Bagi data menjadi data latih (80%) dan data uji (20%).
* Latih model pada data latih.

1. **Prediksi dan Evaluasi:**

* Gunakan model untuk memprediksi label kelas pada data uji.
* Hitung akurasi, presisi, recall, dan F1-score dari model pada data uji.

**Contoh Kode (Scikit-Learn)**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

# ... (kode untuk membaca dan mempersiapkan dataset Spambase)

# Memisahkan fitur (X) dan target (y)

X = df.drop('Spam/ham', axis=1)

y = df['Spam/ham']

# Membagi data menjadi data latih dan data uji

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Membuat model Gaussian Naive Bayes

model = GaussianNB()

# Melatih model

model.fit(X\_train, y\_train)

# Prediksi pada data uji

y\_pred = model.predict(X\_test)

# Evaluasi model

accuracy = accuracy\_score(y\_test, y\_pred)

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

print(f'Akurasi: {accuracy:.2f}')

print(f'Presisi: {precision:.2f}')

print(f'Recall: {recall:.2f}')

print(f'F1-Score: {f1:.2f}')

**Catatan:**

* Pastikan menginstall library “scikit-learn”.
* Sesuaikan path file jika dataset Spambase.data tidak berada di direktori yang sama dengan kode.

**Selamat Mengerjakan Tugas! 😀**

**Tugas:**

1. **Memuat dan Menampilkan Data**

**import pandas as pd**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.naive\_bayes import GaussianNB**

**from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score**

**# Membaca dan Menampilkan Data**

**file\_path = 'data/spambase.data'**

**df = pd.read\_csv(file\_path, header=None)**

**# Menampilkan 5 baris pertama**

**print("Menampilkan 5 Baris:")**

**print(df.head().to\_markdown(index=False, numalign="left", stralign="left"))**

**print("\nInformasi Dataset:")**

**print(df.info())**

1. **Membuat Model Gaussian naive Bayes**

**# Memisahkan Fitur (X) dan Target (y)**

**# Kolom terakhir adalah Target**

**X = df.iloc[:, :-1] # Semua Kolom Kecuali Kolom terakhir**

**y = df.iloc[:, -1] # Kolom Terakhir (Spam atau Tidak)**

**# Membagi Data Menjadi Data Latih dan Data Uji**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

**# MEMBUAT MODEL GAUSSIAN NAIVE BAYES**

**model = GaussianNB()**

**# Melatih Model**

**model.fit(X\_train, y\_train)**

1. **Prediksi dan Evaluasi**

**# PREDIKSI PADA DATA UJI**

**y\_pred = model.predict(X\_test)**

**# EVALUASI MODEL**

**accuracy = accuracy\_score(y\_test, y\_pred)**

**precision = precision\_score(y\_test, y\_pred)**

**recall = recall\_score(y\_test, y\_pred)**

**f1 = f1\_score(y\_test, y\_pred)**

**print(f'\nAkurasi: {accuracy:.2f}')**

**print(f'\nPresisi: {precision:.2f}')**

**print(f'\nRecall: {recall:.2f}')**

**print(f'\nF1-Score: {f1:.2f}')**

**Penjelasan Kode:**

1. **Memuat dan Menampilkan Data:**
   * Anda membaca dataset "Spambase" dari file CSV yang tidak memiliki header kolom.
   * Anda menampilkan 5 baris pertama dan informasi tentang dataset menggunakan df.head() dan df.info().
2. **Membuat Model Gaussian Naive Bayes:**
   * Anda memisahkan fitur (X) dan target (y) dengan benar menggunakan iloc.
   * Anda membagi data menjadi data latih (80%) dan data uji (20%) menggunakan train\_test\_split.
   * Anda membuat objek model GaussianNB().
   * Anda melatih model menggunakan data latih (X\_train, y\_train).
3. **Prediksi dan Evaluasi:**
   * Anda menggunakan model yang sudah dilatih untuk membuat prediksi pada data uji (X\_test).
   * Anda menghitung akurasi, presisi, recall, dan F1-score menggunakan fungsi-fungsi dari sklearn.metrics.
   * Anda mencetak hasil evaluasi.

**Interpretasi Hasil:**

Hasil evaluasi akan menunjukkan seberapa baik model Naive Bayes Anda dalam mengklasifikasikan email spam dan non-spam. Anda dapat menginterpretasikan metrik evaluasi sebagai berikut:

* **Akurasi:** Proporsi prediksi yang benar dari semua prediksi.
* **Presisi:** Proporsi prediksi spam yang benar dari semua email yang diprediksi sebagai spam.
* **Recall:** Proporsi prediksi spam yang benar dari semua email yang sebenarnya spam.
* **F1-Score:** Rata-rata harmonik antara presisi dan recall, memberikan gambaran keseimbangan antara keduanya.

**Contoh Hasil:**

Akurasi: 0.85

Presisi: 0.80

Recall: 0.90

F1-Score: 0.85

Hasil ini menunjukkan bahwa model memiliki akurasi 85%, artinya 85% dari semua prediksi benar. Dari semua email yang diprediksi sebagai spam, 80% benar-benar spam (presisi). Dari semua email yang sebenarnya spam, model berhasil mengidentifikasi 90% (recall). F1-score 0.85 menunjukkan keseimbangan yang baik antara presisi dan recall.

**Kesimpulan:**

Kode Anda sudah sangat baik dan menunjukkan pemahaman yang baik tentang Naive Bayes. Anda telah berhasil membangun dan mengevaluasi model klasifikasi untuk dataset "Spambase". Jika Anda ingin meningkatkan kinerja model, Anda dapat mencoba beberapa hal berikut:

* **Tuning Hyperparameter:** Coba atur hyperparameter model Naive Bayes (seperti var\_smoothing).
* **Feature Engineering:** Coba buat fitur baru yang lebih relevan atau lakukan transformasi fitur yang sudah ada.
* **Algoritma Lain:** Coba gunakan algoritma klasifikasi lain (seperti SVM atau Random Forest) dan bandingkan kinerjanya dengan Naive Bayes.