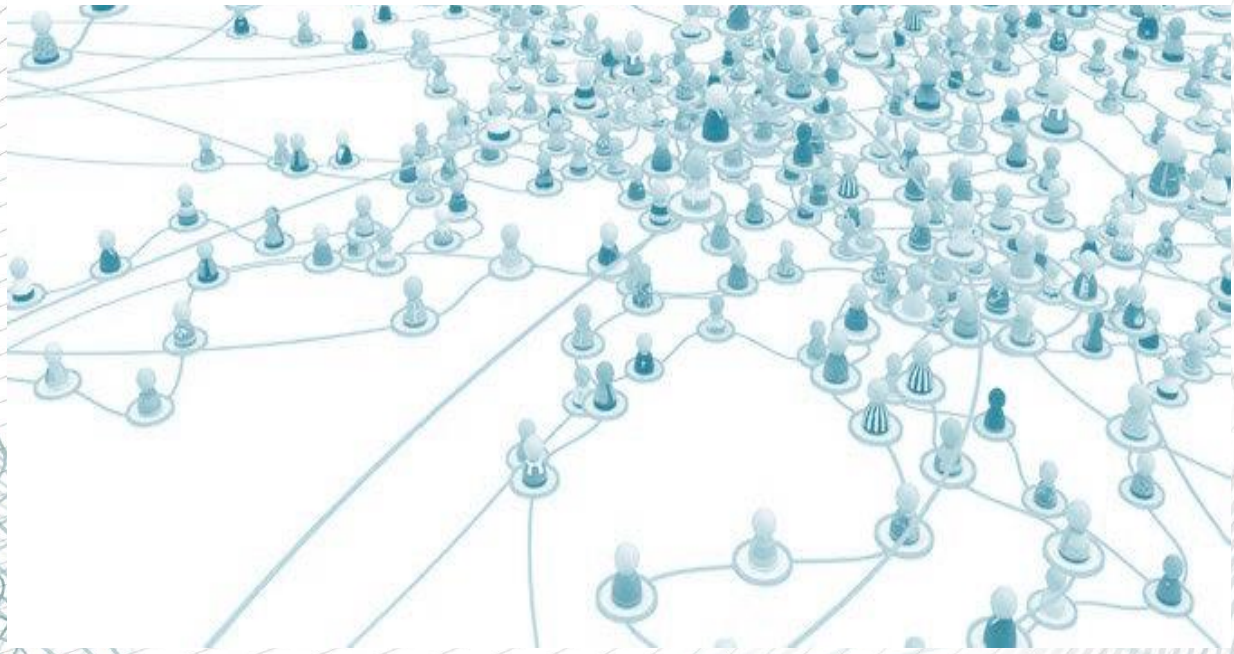
# MODUL PRAKTIKUM

## SD3107-Pembelajaran Mesin



**Program Studi Sains Data**
**Fakultas Sains**
**Institut Teknologi Sumatera**

**2024**

# MODUL 2

## **Naïve Bayes** for Classification

## A. Konsep Dasar

Klasifikasi merupakan salah satu teknik data mining yang dapat digunakan untuk menganalisis sekumpulan data dengan membangun suatu model. Klasifikasi juga dapat diartikan sebagai pemprosesan untuk menemukan sebuah model atau fungsi yang menjelaskan dan mencirikan konsep atau kelas data, untuk kepentingan tertentu. Hasil atau luaran dari penggunaan klasifikasi yaitu dapat dijadikan dasar dalam pengambilan keputusan bagi data analysis. Banyak pilihan model yang dapat digunakan untuk klasifikasi, dalam modul ini dua model yang akan digunakan yaitu *Naïve Bayes*.

*Naïve Bayes* adalah algoritma klasifikasi yang didasarkan pada teorema *Bayes*. *Teorema Bayes* menyatakan bahwa peluang suatu kejadian sama dengan peluang sebelumnya kejadian tersebut dikalikan dengan peluang kejadian tersebut jika diberi bukti. Dalam konteks klasifikasi, ini berarti kita mencoba mencari kelas yang kemungkinan besar diberi sekumpulan fitur atau atribut.

Dalam konsep *Naïve* fitur-fitur tersebut bersifat independen satu sama lain, artinya ada tidaknya suatu fitur tidak mempengaruhi ada tidaknya fitur lainnya. Hal ini menyederhanakan perhitungan kemungkinan suatu fitur, karena kita dapat menghitung kemungkinan setiap fitur secara terpisah dan kemudian mengalikannya. Berikut adalah Aturan Bayes:

$$Posterior = \frac{Likelihood \times Prior}{Evidence} \quad (1)$$

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)} \quad (2)$$

Keterangan:
Peluang Prior: $P(X)$
Peluang bersyarat: $P(X_1|X_2),\ P(X_2|X_1)$
Peluang Join: $P(X_1, X_2)$

Berbeda dengan data kategorik naive bayes dapat menyelesaikan data numerik dengan estimasi yang dapat digunakan yaitu menggunakan fungsi densitas Gauss (Distribusi Normal).

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (3)$$

$$\sigma = \sqrt{\frac{\Sigma(x_i-\mu)^2}{n-1}} \quad (4)$$

Keterangan :

$\mu$ = rata − rata
$\sigma$ = standart deviasi

## B. Tujuan Praktikum

### I. Tujuan Instruksional Umum

Praktikum bertujuan untuk menerapkan teori *Naïve Bayes* untuk klasifikasi dalam pembelajaran mesin.

### II. Tujuan Instruksional Khusus

1. Mahasiswa mampu menguasai konsep dasar *Naïve Bayes.*
2. Mahasiswa mampu menyelesaikan studi kasus data kategorik dan data numerik menggunakan metode *Naïve Bayes.*
3. Mahasiswa mampu menganalisis hasil dari studi kasus data kategorik dan data numerik menggunakan metode *Naïve Bayes.*

---

## C. Dataset dan bahasa pemrograman Python

C.1 Dataset dalam praktikum

Dalam praktikum modul 2 ini kita akan menggunakan tiga dataset dengan rincian sebagai beriku:

1) Dataset kategorik

| Harga Tanah | Jarak dari Pusat Kota | Angkutan Umum | Dipilih untuk perumahan |
|---|---|---|---|
| Murah | Dekat | Tidak | Ya |
| Sedang | Dekat | Tidak | Ya |
| Mahal | Dekat | Tidak | Ya |
| Mahal | Jauh | Tidak | Tidak |
| Mahal | Sedang | Tidak | Tidak |
| Sedang | Jauh | Ada | Tidak |
| Murah | Jauh | Ada | Tidak |
| Murah | Sedang | Tidak | Ya |
| Mahal | Jauh | Ada | Tidak |
| Sedang | Sedang | Ada | Ya |

2) Dataset kategorik dan numerik

| Harga Tanah | Jarak dari Pusat Kota | Angkutan Umum | Dipilih untuk perumahan |
|---|---|---|---|
| Murah | 4 | Tidak | Ya |
| Sedang | 9 | Tidak | Ya |
| Mahal | 3 | Tidak | Ya |
| Mahal | 20 | Tidak | Tidak |
| Mahal | 12 | Tidak | Tidak |
| Sedang | 10 | Ada | Tidak |
| Murah | 19 | Ada | Tidak |
| Murah | 7 | Tidak | Ya |
| Mahal | 8 | Ada | Tidak |
| Sedang | 2 | Ada | Ya |

3) Dataset Numerik

Dataset 3 berisi tiga kolom dan 400 baris dengan dua fitur numerik dan satu kolom target. Dataset 3 merupakan data bersih yang merupakan hasil data preprocessing. Berikut sampel dataset 3:

| | Age | EstimatedSalary | Purchased |
|---|---|---|---|
| 0 | 19 | 19000 | 0 |
| 1 | 35 | 20000 | 0 |
| 2 | 26 | 43000 | 0 |
| 3 | 27 | 57000 | 0 |
| 4 | 19 | 76000 | 0 |
| 5 | 27 | 58000 | 0 |
| 6 | 27 | 84000 | 0 |
| 7 | 32 | 150000 | 1 |
| 8 | 25 | 33000 | 0 |
| 9 | 35 | 65000 | 0 |

Informasi data umur dan gaji akan digunakan untuk memprediksi kelas apakah pengguna akan membeli asuransi atau tidak.

## C.2 Bahasa Pemrograman Python

Pada praktikum kali ini, kita akan menggunakan IDE Python Service dari Google Colab, maka dari itu Anda perlu menyiapkan akun google pribadi untuk mengakses servis ini di akun google masing-masing.

Library yang akan digunakan diantaranya numpy, pandas, dan sklearn.

1) NumPy: digunakan untuk operasi matematika berbasis array dan matriks. NumPy sangat penting dalam komputasi numerik dengan Python pada praktikum ini.
2) Pandas : library untuk pengolahan data berbasis tabel (DataFrame).

   (Keterangan: anda bisa memilih opsi menggunakan library NumPy atau pandas dengan tujuan yang sama)
3) Scikit-learn (sklearn): library untuk implementasi berbagai algoritma machine learning seperti klasifikasi, regresi, clustering, dan reduksi dimensi. Pada praktikum ini sklearn secara khusus kita gunakan untuk import model **GaussianNB** yaitu untuk perhitungan pada data numerik. Selain itu sklearn kita gunakan untuk melakukan evaluasi model.

Praktikum pembelajaran mesin pada modul 2 ini dapat maksimal anda praktikan dengan pemahaman bidang atau mata kuliah terkait seperti algoritma pemrograman, struktur data, dan data mining (data preprocessing).

*Dataset dan Source code akan diberikan saat praktikum berlangsung dan dapat anda unduh pada folder berikut* Modul 2 PM 2024 .

## D. Implementasi Naïve Bayes untuk Klasifikasi

D.1 Dataset Categorical

1) Import Library

```
import numpy as np
```

   Pada praktikum ini hanya akan menggunakan library numpy.

2) Import Dataset

```
# Dataset
dataC = np.array([
    ['Murah', 'Dekat', 'Tidak', 'Ya'],
    ['Sedang', 'Dekat', 'Tidak', 'Ya'],
    ['Mahal', 'Dekat', 'Tidak', 'Ya'],
    ['Mahal', 'Jauh', 'Tidak', 'Tidak'],
    ['Mahal', 'Sedang', 'Tidak', 'Tidak'],
    ['Sedang', 'Jauh', 'Ada', 'Tidak'],
    ['Murah', 'Jauh', 'Ada', 'Tidak'],
    ['Murah', 'Sedang', 'Tidak', 'Ya'],
    ['Mahal', 'Jauh', 'Ada', 'Tidak'],
    ['Sedang', 'Sedang', 'Ada', 'Ya']
])
```

Definisikan dataset kategorik sesuai pada tabel 1 yaitu dataset 1. Kita dapat mendefinisikan dataset kategorik berupa matriks dengan np.array.

3) Hitung probabilitas prior

```
# Hitung probabilitas prior
def calculate_prior(dataC):
    total = len(dataC)
    ya_count = sum(1 for row in dataC if row[-1] == 'Ya')
    tidak_count = total - ya_count
    return ya_count / total, tidak_count / total
```

Langkah berikutnya yaitu mengitung probabilitas prior dengan memperhatikan data target menjadi dua kelas yaitu kelas “Ya” dan “Tidak”.

Sebagai gambaran anda dapat diperhatikan tabel berikut.

| Perumahan | Jumlah Kejadian | | Probabilitas | |
|---|---|---|---|---|
| | Ya | Tidak | Ya | Tidak |
| Jumlah | 5 | 5 | 1/2 | 1/2 |

4) Hitung probabilitas kondisional untuk fitur kategori

```
# Hitung probabilitas kondisional untuk fitur kategori
def calculate_categorical_prob(dataC, feature_index, value, target_value):
    filtered_data = [row for row in dataC if row[-1] == target_value]
    count = sum(1 for row in filtered_data if row[feature_index] == value)
    return count / len(filtered_data) if len(filtered_data) > 0 else 0
```

Selanjutnya kita perlu mendifinisikan beberapa kondisi setiap fitur. Untuk dataset 1 dapat kita lihat bahwa seluruh fitur murupakan fitur kategorik, sehingga kita cukup mendefinisikan kondisi fitur kategorik sebagai calculat_categorical_prob yang berisi perhitungan probabilitas berdasarkan masing-masing kondisi disetiap fitur kategorik diinginkan (akan kita panggil dalam model) terhadap kelas pada target. Perhitungan ini perdasar pada persamaan 1 dan 2.

5) Klasifikasi Naïve Bayes

```
# Klasifikasi Naive Bayes
def naive_bayes_predict(dataC, harga, jarak, angkutan):
    prior_ya, prior_tidak = calculate_prior(dataC)

    # Probabilitas kondisional untuk Ya
    p_harga_ya = calculate_categorical_prob(dataC, 0, harga, 'Ya')
    p_jarak_ya = calculate_categorical_prob(dataC, 1, jarak, 'Ya')
    p_angkutan_ya = calculate_categorical_prob(dataC, 2, angkutan, 'Ya')

    # Probabilitas kondisional untuk Tidak
    p_harga_tidak = calculate_categorical_prob(dataC, 0, harga, 'Tidak')
    p_jarak_tidak = calculate_categorical_prob(dataC, 1, jarak, 'Tidak')
    p_angkutan_tidak = calculate_categorical_prob(dataC, 2, angkutan, 'Tidak')

    # Posterior untuk Ya dan Tidak
    p_ya = prior_ya * p_harga_ya * p_angkutan_ya * p_jarak_ya
    p_tidak = prior_tidak * p_harga_tidak * p_angkutan_tidak * p_jarak_tidak

    return 'Ya' if p_ya > p_tidak else 'Tidak', p_ya, p_tidak
```

Bagian selanjutnya adalah mendefinisikan model naive bayes dengan nama naive_bayes_predict untuk melakukan klasifikasi yang berisi perhitungan probabilitas untuk kelas “Ya” dan “Tidak” dengan data fitur kategorik harga, jarak, dan angkutan. Sebagai contoh p_harga_ya kita definisikan sebagai data kategorik yaitu dengan memanggil calculate_categorical_prob untuk menghitung probabilitas pada dataC yang ada pada fitur ke 0 (artinya fitur pertama), dengan nama fitur harga, pada kelas “Ya”. Dapat kita tuliskan sebagai (dataC, 0, harga, “Ya). Maka kita mendapatkan nilai probabilitas pada setiap fitur berdasarkan kelas “Ya” dan “Tidak”. Sebagai gambaran hasil perhitungan dapat dilihat pada tabel berikut.

| Harga Tanah | Jumlah Kejadian | | Probabilitas | |
|---|---|---|---|---|
| | Ya | Tidak | Ya | Tidak |
| Murah | 2 | 1 | 2/5 | 1/5 |
| Sedang | 2 | 1 | 2/5 | 1/5 |
| Mahal | 1 | 3 | 1/5 | 3/5 |

| Jarak Pusat Kota | Jumlah Kejadian | | Probabilitas | |
|---|---|---|---|---|
| | Ya | Tidak | Ya | Tidak |
| Dekat | 3 | 0 | 3/5 | 0 |
| Sedang | 2 | 1 | 2/5 | 1/5 |
| Jauh | 0 | 4 | 0 | 4/5 |

| Angkutan Umum | Jumlah Kejadian | | Probabilitas | |
|---|---|---|---|---|
| | Ya | Tidak | Ya | Tidak |
| Ada | 1 | 3 | 1/5 | 3/5 |
| Tidak | 4 | 2 | 4/5 | 2/5 |

Selanjutnya adalah mendifinisikan perhitungan prosterior untuk kelas “Ya” yaitu dengan mengalikan semua nilai probabilitas pada semua fitur yang masuk pada kelas “Ya” dan nilai prior_ya hasil dari perhitungan calculate_prior pada

bagian 2. Sehingga dapat kita namakan sebagai p_y yaitu hasil pengalian nilai prior_ya dengan nilai prob pada masing-masing fitur. Perhitungan prob kelas "Tidak" juga dapat dihitung dengan persamaan yang sama.

Terakhir lakukan return untuk kondisi p_ya lebih besar dari p_tidak maka data test masuk ke dalam kelas "Ya", jika sebaliknya maka masuk kelas "Tidak".

6) Klasifikasi data uji

```
# Prediksi untuk harga "Mahal", jarak "Sedang", dan angkutan "Ada"
prediction = naive_bayes_predict(dataC, 'Mahal', 'Sedang', 'Ada')
prediction
```

Sebagai contoh kita masukan data uji yaitu harga "Mahal", jarak "Sedang", dan angkutan "Ada", maka kita dapat memanggil model naive_bayes_predict untuk mendapatkan nilai probabilitas yang didapat dan menentukan prediksi kelas.

7) Tampilkan hasil

```
('Tidak', 0.008000000000000002, 0.036)
```

Dari dataset 1 sebagai data training dan data test yang kita inputkan pada bagian 6 didapatkan prediksi kelas "Tidak" dengan hasil perbandingan prob nilai p_tidak yaitu 0.036 lebih besar dari nilai p_ya yaitu 0.008. Nilai ini sesuai dengan hasil perhitungan manual sebagai berikut.

- Misalkan terdapat suatu lokasi dengan harga tanah **Mahal**, jarak dari pusat kota **Sedang**, dan **Ada** angkutan umum. Maka dapat dihitung:
- Untuk kelas Perumahan = Ya
  - P(Tanah=Mahal | Ya)=1/5
  - P(Jarak=Sedang | Ya)=2/5
  - P(Angkutan=Ada | Ya)=1/5
  - P(Ya)=1/2
- Likelihood.Prior = (1/5)(2/5)(1/5)(1/2)=0.008

- Untuk kelas Perumahan = Tidak
  - P(Tanah=Mahal | Tidak)=3/5
  - P(Jarak=Sedang | Tidak)=1/5
  - P(Angkutan=Ada | Tidak)=3/5
  - P(Ya)=1/2
- Likelihood.Prior = (3/5)(1/5)(3/5)(1/2)=0.036
- Berdasarkan Max A Posteriori, karena 0.036 > 0.008, maka lokasi dengan harga tanah mahal, jarak sedang, dan ada angkutan umum **Tidak** dibangun perumahan.

D.2 Dataset Categorical dan Numeric

1) Import Library

```
import numpy as np
from math import sqrt, exp, pi
```

Library yang kita gunakan pada topik ini adalah numpy dan math untuk pengolahan data kategorik dan numerik. Library math diperlukan untuk perhitungan pada data numerik. Kita dapat import sqrt sebagai akar, exp sebagai eksponensial dan pi untuk import nilai pi.

2) Import Dataset

```
# Dataset
data = [
    ['Murah', 4, 'Tidak', 'Ya'],
    ['Sedang', 9, 'Tidak', 'Ya'],
    ['Mahal', 3, 'Tidak', 'Ya'],
    ['Mahal', 20, 'Tidak', 'Tidak'],
    ['Mahal', 12, 'Tidak', 'Tidak'],
    ['Sedang', 10, 'Ada', 'Tidak'],
    ['Murah', 19, 'Ada', 'Tidak'],
    ['Murah', 7, 'Tidak', 'Ya'],
    ['Mahal', 8, 'Ada', 'Tidak'],
    ['Sedang', 2, 'Ada', 'Ya']
]
```

Dataset yang kita gunakan adalah data pada tabel 2 yang berisi dua fitur kategorik dan satu fitur numerik.

3) Hitung probabilitas prior

Definisikan calculate_prior sama dengan yang ada pada source code data kategorik. **Perhatikan penamaan dataset harus sesuai.**

4) Hitung probabilitas kondisional untuk fitur kategori

Definisikan calculate_categorical_prob sama dengan yang ada pada source code data kategorik. **Perhatikan penamaan dataset harus sesuai.**

5) Hitung distribusi Gaussian untuk fitur numerik

Jika terdapat fitur numerik maka perlu menambahkan bagian ini yaitu mendefinisakan perhitungan prob dengan fungsi gaussian, dapat anda lihat pada persamaan (3).

```
# Hitung distribusi Gaussian untuk fitur numerik
def calculate_gaussian_prob(x, mean, std):
    exponent = exp(-((x - mean) ** 2 / (2 * (std )**2)))
    return (1 / (sqrt(2 * pi) * std)) * exponent

# Hitung mean dan std dev untuk fitur numerik
def calculate_numerical_stats(data, feature_index, target_value):
    filtered_data = [row[feature_index] for row in data if row[-1] == target_value]

    mean = np.mean(filtered_data)
    std = np.std(filtered_data)
    return mean, std
```

6) Klasifikasi Naïve Bayes

```
# Klasifikasi Naive Bayes
def naive_bayes_predict(data, harga, jarak, angkutan):
    prior_ya, prior_tidak = calculate_prior(data)

    # Probabilitas kondisional untuk Ya
    p_harga_ya = calculate_categorical_prob(data, 0, harga, 'Ya')
    mean_ya, std_ya = calculate_numerical_stats(data, 1, 'Ya')   #fitur numerik
    p_jarak_ya = calculate_gaussian_prob(jarak, mean_ya, std_ya) #fitur numerik
    p_angkutan_ya = calculate_categorical_prob(data, 2, angkutan, 'Ya')

    # Probabilitas kondisional untuk Tidak
    p_harga_tidak = calculate_categorical_prob(data, 0, harga, 'Tidak')
    mean_tidak, std_tidak = calculate_numerical_stats(data, 1, 'Tidak') #fitur numerik
    p_jarak_tidak = calculate_gaussian_prob(jarak, mean_tidak, std_tidak) #fitur numerik
    p_angkutan_tidak = calculate_categorical_prob(data, 2, angkutan, 'Tidak')

    # Posterior untuk Ya dan Tidak
    p_ya = prior_ya * p_harga_ya * p_angkutan_ya * p_jarak_ya
    p_tidak = prior_tidak * p_harga_tidak * p_angkutan_tidak * p_jarak_tidak

    return 'Ya' if p_ya > p_tidak else 'Tidak', p_ya, p_tidak
```

Definisikan kembali model naive_bayes_predict dengan penambahan perhitungan numerik yang ada pada fitur jarak.

7) Klasifikasi data uji

```
# Prediksi untuk harga "Murah", jarak 5, dan angkutan "Tidak"
prediction = naive_bayes_predict(data, 'Murah', 5, 'Tidak')
prediction
```

Sebagai contoh kita gunakan data uji harga “Murah”, jarak 5, dan angkutan “Tidak”.

8) Tampilkan hasil

```
('Ya', 0.024477980931106355, 0.0006293259804336088)
```

Hasik prediksi kelas yaitu “Ya” dengan nilai p_ya lebih besar dari y_tidak.

D.3 Dataset Numerik dengan skLearn

1) import library

```
# Import libraries
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

2) Import dataset

```
# Read dataset
df_net = pd.read_csv('/content/new_social_network_ads.csv')
df_net.head()
```

| | Age | EstimatedSalary | Purchased |
|---|---|---|---|
| 0 | 19 | 19000 | 0 |
| 1 | 35 | 20000 | 0 |
| 2 | 26 | 43000 | 0 |
| 3 | 27 | 57000 | 0 |
| 4 | 19 | 76000 | 0 |

Dataset 3 berisi fitur numerik dan target.

3) Split dataset

Split data

Independent/Dependent variables

```
[14] # Split data into independent/dependent variables
     X = df_net.iloc[:, :-1].values
     y = df_net.iloc[:, -1].values
```

Train/Test sets

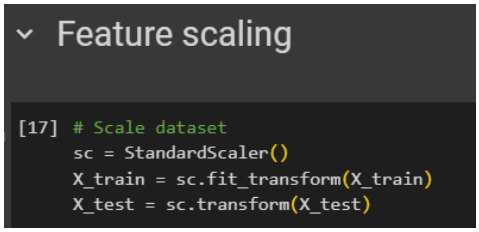
```
[16] # Split data into Train/Test sets
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = True)
```

Tahap split data menjadi variabel independen dan dependen melibatkan pemisahan fitur masukan ( variabel independen ) dari variabel target (variabel dependen ). Variabel independen digunakan untuk memprediksi nilai variabel dependen.

Data tersebut kemudian dibagi menjadi set pelatihan dan set pengujian,

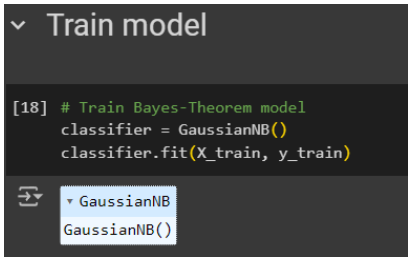
dengan set pelatihan digunakan agar sesuai dengan model dan set pengujian digunakan untuk mengevaluasi performanya.

4) Fitur Scaling




Feature scaling dilakukan sebelum melatih model, karena hal ini dapat meningkatkan performa model dan mengurangi waktu yang diperlukan untuk melatihnya, serta membantu memastikan bahwa algoritme tidak bias terhadap variabel dengan nilai lebih besar.

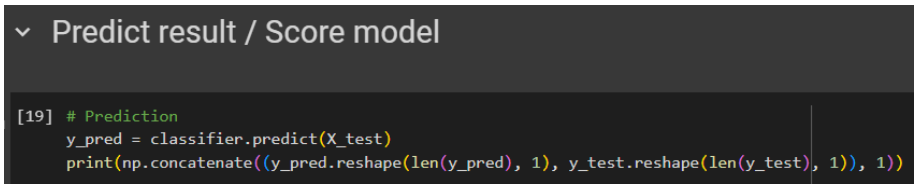
5) Train model



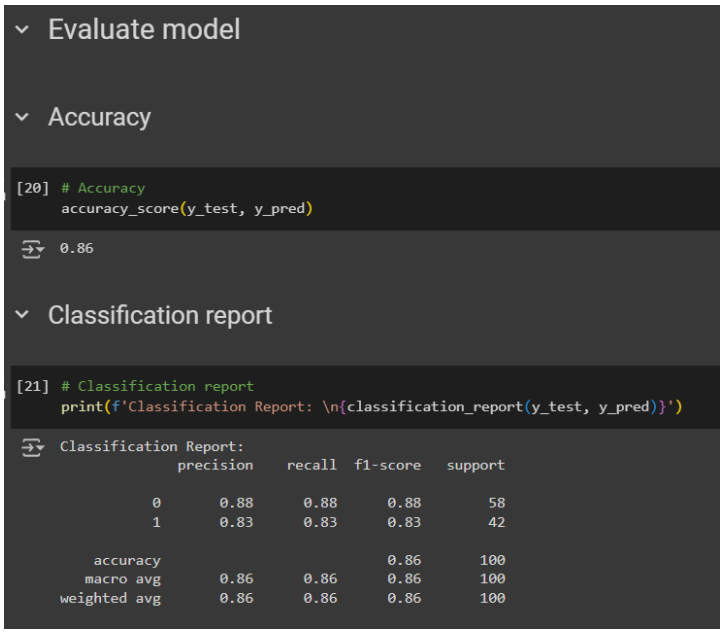

Definisikan data X_train dan y_train ke dalam model pengklasifikasi Naïve Bayes dengan classifier.fit untuk melatih model dengan data pelatihan kita.

6) Tampilkan hasil prediksi data testing

Setelah kemungkinan fitur untuk setiap kelas dihitung, algoritme mengalikan kemungkinan tersebut dengan probabilitas sebelumnya dari setiap kelas, yang diperkirakan dari data pelatihan. Kelas dengan probabilitas tertinggi kemudian dipilih sebagai kelas prediksi. Keakuratan model dapat dievaluasi pada set pengujian yang sebelumnya dilakukan selama proses pelatihan.

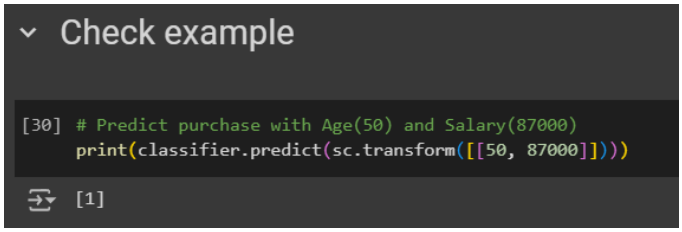



7) Evaluasi model




Hitung nilai akurasi dan print Classification report dapat memberikan rata-rata tertimbang skor masing-masing kelas (class scores ), yang memperhitungkan ketidakseimbangan distribusi kelas dalam dataset.

8) Klasifikasi data uji




Sebagai contoh kita cek untuk Usia 50 tahun dan Gaji 87000 dan periksa apakah pengguna kemungkinan akan membeli asuransi atau tidak.

Nilai prediksi [1] berarti pengguna akan membeli asuransi.

TUGAS INDIVIDU

- Tugas individu dikerjakan saat praktikum berlangsung dengan waktu yang ditentukan oleh asisten praktikum.
- Soal tugas individu terbagi menjadi tiga level soal yaitu *Basic*, *Medium*, *dan Hard*. Praktikan dapat menyelesaikan ketiganya untuk mendapatkan poin maksimal.
- Namun praktikan cukup menyelesaikan minimal satu soal jika estimasi waktu praktikum sudah akan berakhir agar tidak melewatkan penilaian test lisan.