```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df = pd.read_csv("dataset.csv")
```

```
pd.options.display.max_columns = None
```

```
df
```

	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Nac
0	1	8	5	2	1	1	
1	1	6	1	11	1	1	
2	1	1	5	5	1	1	
3	1	8	2	15	1	1	
4	2	12	1	3	0	1	
...	...	...	...	...	...	...	
4419	1	1	6	15	1	1	
4420	1	1	2	15	1	1	
4421	1	1	1	12	1	1	
4422	1	1	1	9	1	1	
4423	1	5	1	15	1	1	

4424 rows × 35 columns

```
df.dtypes
```

```
Marital status                                      int64
Application mode                                    int64
Application order                                   int64
Course                                              int64
Daytime/evening attendance                          int64
Previous qualification                              int64
Nacionality                                         int64
Mother's qualification                              int64
Father's qualification                              int64
Mother's occupation                                 int64
Father's occupation                                 int64
Displaced                                           int64
Educational special needs                           int64
Debtor                                              int64
Tuition fees up to date                             int64
Gender                                              int64
Scholarship holder                                  int64
Age at enrollment                                   int64
International                                       int64
Curricular units 1st sem (credited)                 int64
Curricular units 1st sem (enrolled)                 int64
Curricular units 1st sem (evaluations)              int64
Curricular units 1st sem (approved)                 int64
Curricular units 1st sem (grade)                  float64
Curricular units 1st sem (without evaluations)      int64
Curricular units 2nd sem (credited)                 int64
Curricular units 2nd sem (enrolled)                 int64
Curricular units 2nd sem (evaluations)              int64
Curricular units 2nd sem (approved)                 int64
Curricular units 2nd sem (grade)                  float64
Curricular units 2nd sem (without evaluations)      int64
Unemployment rate                                 float64
```

```
Inflation rate                                    float64
GDP                                               float64
Target                                             object
dtype: object
```

```
# Menampilkan jumlah data training
jumlah_data_training = df.shape[0]
print("Jumlah data training:", jumlah_data_training)
```

```
Jumlah data training: 4424
```

```
df.head(400)
```

	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Nacio
0	1	8	5	2	1	1	
1	1	6	1	11	1	1	
2	1	1	5	5	1	1	
3	1	8	2	15	1	1	
4	2	12	1	3	0	1	
...	...	...	...	...	...	...	
395	1	1	4	12	1	1	
396	1	16	1	10	1	1	
397	1	15	1	6	1	14	
398	1	8	1	14	1	1	
399	1	8	1	9	1	1	

400 rows × 35 columns

```
# Mencetak nama kolom atau atribut
print("Variabel/atribut:")
print(df.columns.tolist())

# Mencetak nilai-nilai dari dataframe
print("\nNilai-nilai:")
print(df.values)
```

```
Variabel/atribut:
['Marital status', 'Application mode', 'Application order', 'Course', 'Daytime/evening attendance', 'Previous qual

Nilai-nilai:
[[1 8 5 ... 1.4 1.74 'Dropout']
 [1 6 1 ... -0.3 0.79 'Graduate']
 [1 1 5 ... 1.4 1.74 'Dropout']
 ...
 [1 1 1 ... -0.3 0.79 'Dropout']
 [1 1 1 ... -0.8 -3.12 'Graduate']
 [1 5 1 ... 3.7 -1.7 'Graduate']]
```

```
from tabulate import tabulate

# Menghitung rata-rata atau bobot setiap kelas
nilai_kelas = df.groupby('Target').mean()

# Menampilkan tabel nilai atau bobot setiap kelas
print("Tabel Nilai atau Bobot Setiap Kelas:")
print(tabulate(nilai_kelas, headers='keys', tablefmt='psql'))
```

```
Tabel Nilai atau Bobot Setiap Kelas:
+----------+------------------+--------------------+---------------------+----------+-----------------------------
| Target   |   Marital status |   Application mode |   Application order |   Course |   Daytime/evening attendance
|----------+------------------+--------------------+---------------------+----------+-----------------------------
| Dropout  |          1.26108 |            8.34201 |             1.59324 |  9.89866 |                     0.854328
| Enrolled |          1.15239 |            7.23804 |             1.62594 |  9.733   |                     0.905542
| Graduate |          1.1349  |            5.82481 |             1.85106 |  9.95926 |                     0.909009
+----------+------------------+--------------------+---------------------+----------+-----------------------------
```

```
# Menghitung jumlah data training (80% dari total data)
jumlah_data_training = int(df.shape[0] * 0.8)
print("Jumlah data training:", jumlah_data_training)
```

```
Jumlah data training: 3539
```

```
import pandas as pd

# Data yang diberikan
data = {
    'Target': ['Graduate', 'Dropout'],
    'Course': [36, 21],
    'Tuition up to date': [0.995049, 0.717171],
    'Gender': [505, 717],
    'Scholarship Holder': [0.574, 0.272],
    'Curicular 1st sem (approved)': [25743, 73],
    'Curricular 1st sem (Grade)': [0.2970297, 0.090909],
    'Curricular 2nd sem (approved)': [3, 1],
    'Curricular 2nd sem (Grade)': [6.267327, 3.10101]
}

# Membuat DataFrame dari data
df = pd.DataFrame(data)

# Menghitung mean dan standar deviasi untuk setiap variabel dalam tiap kelas
mean_std_table = df.groupby('Target').agg({'Course': ['mean', 'std'],
                                           'Tuition up to date': ['mean', 'std'],
                                           'Gender': ['mean', 'std'],
                                           'Scholarship Holder': ['mean', 'std'],
                                           'Curicular 1st sem (approved)': ['mean', 'std'],
                                           'Curricular 1st sem (Grade)': ['mean', 'std'],
                                           'Curricular 2nd sem (approved)': ['mean', 'std'],
                                           'Curricular 2nd sem (Grade)': ['mean', 'std']}).reset_index()

# Menampilkan tabel hasil mean dan standar deviasi
print("Tabel 4: Hasil Mean dan Standar Deviasi tiap Atribut")
print(mean_std_table)
```

```
Tabel 4: Hasil Mean dan Standar Deviasi tiap Atribut
     Target Course     Tuition up to date     Gender     Scholarship Holder  \
              mean std               mean std   mean std               mean   
0   Dropout   21.0 NaN           0.717171 NaN  717.0 NaN              0.272   
1  Graduate   36.0 NaN           0.995049 NaN  505.0 NaN              0.574   

      Curicular 1st sem (approved)     Curricular 1st sem (Grade)      \
  std                         mean std                       mean std   
0 NaN                         73.0 NaN                   0.090909 NaN   
1 NaN                      25743.0 NaN                   0.297030 NaN   

  Curricular 2nd sem (approved)     Curricular 2nd sem (Grade)      
                           mean std                       mean std  
0                           1.0 NaN                   3.101010 NaN
1                           3.0 NaN                   6.267327 NaN
```

```
import numpy as np
import pandas as pd

# Membaca dataset
df = pd.read_csv("dataset.csv")

# Menghitung mean, standar deviasi, dan probabilitas untuk setiap atribut berdasarkan target
summary_stats = df.groupby('Target').agg(['mean', 'std'])
probabilities = df['Target'].value_counts(normalize=True)

print("Summary Statistics:")
print(summary_stats)

print("\nProbabilities:")
print(probabilities)

# Memisahkan data uji (20% dari dataset)
data_uji = df.sample(frac=0.2, random_state=42)

print("\nTabel Data Testing:")
print(data_uji)
```

```
...              ...   ...       ...
3162             1.4  1.74  Graduate
3281             1.4  1.74   Dropout
436              0.5  1.79  Enrolled
1434             0.6  2.02  Enrolled
1361             1.4  1.74  Graduate

[885 rows x 35 columns]
```

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

# Membaca dataset
df = pd.read_csv("dataset.csv")

# Membagi dataset menjadi data latih dan data uji sesuai dengan skenario pengujian
# Skenario Pengujian 1
X_train_1, X_test_1, y_train_1, y_test_1 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.2,

# Skenario Pengujian 2
X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.5,

# Skenario Pengujian 3
X_train_3, X_test_3, y_train_3, y_test_3 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.8,

# Membuat dan melatih model Naive Bayes untuk setiap skenario pengujian
model_1 = GaussianNB()
model_1.fit(X_train_1, y_train_1)

model_2 = GaussianNB()
model_2.fit(X_train_2, y_train_2)

model_3 = GaussianNB()
model_3.fit(X_train_3, y_train_3)

# Melakukan prediksi pada data uji untuk setiap skenario
y_pred_1 = model_1.predict(X_test_1)
y_pred_2 = model_2.predict(X_test_2)
y_pred_3 = model_3.predict(X_test_3)

# Menghitung akurasi untuk setiap skenario pengujian
accuracy_1 = accuracy_score(y_test_1, y_pred_1)
accuracy_2 = accuracy_score(y_test_2, y_pred_2)
accuracy_3 = accuracy_score(y_test_3, y_pred_3)

# Menampilkan hasil pengujian
print("Hasil Pengujian:")
print("Skenario Pengujian 1 - Akurasi:", accuracy_1)
print("Skenario Pengujian 2 - Akurasi:", accuracy_2)
print("Skenario Pengujian 3 - Akurasi:", accuracy_3)
```

```
Hasil Pengujian:
Skenario Pengujian 1 - Akurasi: 0.7073446327683616
Skenario Pengujian 2 - Akurasi: 0.6989150090415913
Skenario Pengujian 3 - Akurasi: 0.6861581920903955
```

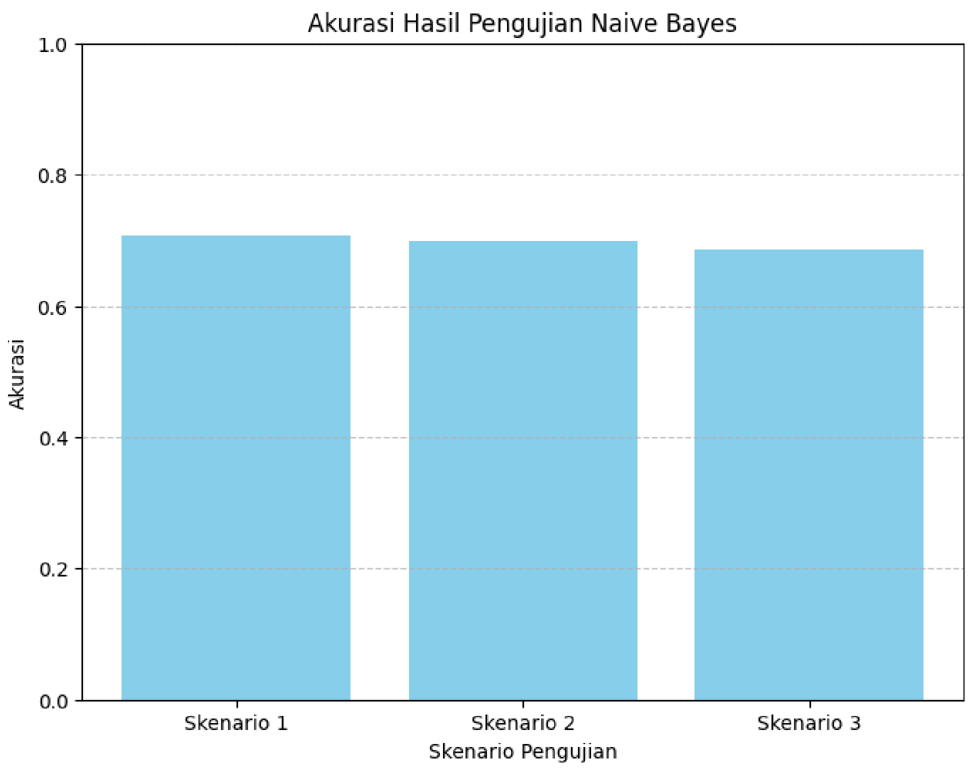
```
import matplotlib.pyplot as plt

# Definisikan data
skenario = ['Skenario 1', 'Skenario 2', 'Skenario 3']
akurasi = [accuracy_1, accuracy_2, accuracy_3]

# Plot grafik
plt.figure(figsize=(8, 6))
plt.bar(skenario, akurasi, color='skyblue')

# Tambahkan judul dan label
plt.title('Akurasi Hasil Pengujian Naive Bayes')
plt.xlabel('Skenario Pengujian')
plt.ylabel('Akurasi')

# Tampilkan grafik
plt.ylim(0, 1)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



```
import numpy as np
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 dataset
df = pd.read_csv("dataset.csv")

# Membagi dataset menjadi data latih dan data uji sesuai dengan skenario pengujian
# Skenario Pengujian 1
X_train_1, X_test_1, y_train_1, y_test_1 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.2, ra

# Skenario Pengujian 2
X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.5, ra

# Skenario Pengujian 3
X_train_3, X_test_3, y_train_3, y_test_3 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.8, ra

# Membuat dan melatih model Naive Bayes untuk setiap skenario pengujian
model_1 = GaussianNB()
```

```
model_1.fit(X_train_1, y_train_1)

model_2 = GaussianNB()
model_2.fit(X_train_2, y_train_2)

model_3 = GaussianNB()
model_3.fit(X_train_3, y_train_3)

# Melakukan prediksi pada data uji untuk setiap skenario
y_pred_1 = model_1.predict(X_test_1)
y_pred_2 = model_2.predict(X_test_2)
y_pred_3 = model_3.predict(X_test_3)

# Menghitung metrik evaluasi untuk setiap skenario
accuracy_1 = accuracy_score(y_test_1, y_pred_1)
accuracy_2 = accuracy_score(y_test_2, y_pred_2)
accuracy_3 = accuracy_score(y_test_3, y_pred_3)

precision_graduate_1 = precision_score(y_test_1, y_pred_1, average='macro', pos_label='Graduate')
precision_dropout_1 = precision_score(y_test_1, y_pred_1, average='macro', pos_label='Dropout')

recall_graduate_1 = recall_score(y_test_1, y_pred_1, average='macro', pos_label='Graduate')
recall_dropout_1 = recall_score(y_test_1, y_pred_1, average='macro', pos_label='Dropout')

f1_score_graduate_1 = f1_score(y_test_1, y_pred_1, average='macro', pos_label='Graduate')
f1_score_dropout_1 = f1_score(y_test_1, y_pred_1, average='macro', pos_label='Dropout')

precision_graduate_2 = precision_score(y_test_2, y_pred_2, average='macro', pos_label='Graduate')
precision_dropout_2 = precision_score(y_test_2, y_pred_2, average='macro', pos_label='Dropout')

recall_graduate_2 = recall_score(y_test_2, y_pred_2, average='macro', pos_label='Graduate')
recall_dropout_2 = recall_score(y_test_2, y_pred_2, average='macro', pos_label='Dropout')

f1_score_graduate_2 = f1_score(y_test_2, y_pred_2, average='macro', pos_label='Graduate')
f1_score_dropout_2 = f1_score(y_test_2, y_pred_2, average='macro', pos_label='Dropout')

precision_graduate_3 = precision_score(y_test_3, y_pred_3, average='macro', pos_label='Graduate')
precision_dropout_3 = precision_score(y_test_3, y_pred_3, average='macro', pos_label='Dropout')

recall_graduate_3 = recall_score(y_test_3, y_pred_3, average='macro', pos_label='Graduate')
recall_dropout_3 = recall_score(y_test_3, y_pred_3, average='macro', pos_label='Dropout')

f1_score_graduate_3 = f1_score(y_test_3, y_pred_3, average='macro', pos_label='Graduate')
f1_score_dropout_3 = f1_score(y_test_3, y_pred_3, average='macro', pos_label='Dropout')

# Menampilkan hasil evaluasi
print("Hasil Evaluasi Skenario 1:")
print("Akurasi:", accuracy_1)
print("Presisi untuk kelas Graduate:", precision_graduate_1)
print("Presisi untuk kelas Dropout:", precision_dropout_1)
print("Recall untuk kelas Graduate:", recall_graduate_1)
print("Recall untuk kelas Dropout:", recall_dropout_1)
print("F1-score untuk kelas Graduate:", f1_score_graduate_1)
print("F1-score untuk kelas Dropout:", f1_score_dropout_1)
print("\nHasil Evaluasi Skenario 2:")
print("Akurasi:", accuracy_2)
print("Presisi untuk kelas Graduate:", precision_graduate_2)
print("Presisi untuk kelas Dropout:", precision_dropout_2)
print("Recall untuk kelas Graduate:", recall_graduate_2)
print("Recall untuk kelas Dropout:", recall_dropout_2)
print("F1-score untuk kelas Graduate:", f1_score_graduate_2)
print("F1-score untuk kelas Dropout:", f1_score_dropout_2)
print("\nHasil Evaluasi Skenario 3:")
print("Akurasi:", accuracy_3)
print("Presisi untuk kelas Graduate:", precision_graduate_3)
print("Presisi untuk kelas Dropout:", precision_dropout_3)
print("Recall untuk kelas Graduate:", recall_graduate_3)
print("Recall untuk kelas Dropout:", recall_dropout_3)
print("F1-score untuk kelas Graduate:", f1_score_graduate_3)
print("F1-score untuk kelas Dropout:", f1_score_dropout_3)
```

```
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_labe
  warnings.warn(
Hasil Evaluasi Skenario 1:
Akurasi: 0.7073446327683616
Presisi untuk kelas Graduate: 0.6492810649038782
Presisi untuk kelas Dropout: 0.6492810649038782
Recall untuk kelas Graduate: 0.6238946595640367
Recall untuk kelas Dropout: 0.6238946595640367
F1-score untuk kelas Graduate: 0.6307389567916585
F1-score untuk kelas Dropout: 0.6307389567916585

Hasil Evaluasi Skenario 2:
Akurasi: 0.6989150090415913
Presisi untuk kelas Graduate: 0.6365550132299937
Presisi untuk kelas Dropout: 0.6365550132299937
Recall untuk kelas Graduate: 0.6166512494831305
Recall untuk kelas Dropout: 0.6166512494831305
F1-score untuk kelas Graduate: 0.6218452710743506
F1-score untuk kelas Dropout: 0.6218452710743506

Hasil Evaluasi Skenario 3:
Akurasi: 0.6861581920903955
Presisi untuk kelas Graduate: 0.6018893601563738
Presisi untuk kelas Dropout: 0.6018893601563738
Recall untuk kelas Graduate: 0.5809922748086525
Recall untuk kelas Dropout: 0.5809922748086525
F1-score untuk kelas Graduate: 0.5800493683366588
F1-score untuk kelas Dropout: 0.5800493683366588
```

```
# Impor library yang diperlukan
from sklearn.metrics import f1_score

# Hasil Evaluasi Skenario 1
akurasi_1 = 0.7073446327683616
presisi_graduate_1 = 0.6492810649038782
presisi_dropout_1 = 0.6492810649038782
recall_graduate_1 = 0.6238946595640367
recall_dropout_1 = 0.6238946595640367
f1_graduate_1 = 0.6307389567916585
f1_dropout_1 = 0.6307389567916585

# Hasil Evaluasi Skenario 2
akurasi_2 = 0.6989150090415913
presisi_graduate_2 = 0.6365550132299937
presisi_dropout_2 = 0.6365550132299937
recall_graduate_2 = 0.6166512494831305
recall_dropout_2 = 0.6166512494831305
f1_graduate_2 = 0.6218452710743506
f1_dropout_2 = 0.6218452710743506

# Hasil Evaluasi Skenario 3
akurasi_3 = 0.6861581920903955
presisi_graduate_3 = 0.6018893601563738
presisi_dropout_3 = 0.6018893601563738
recall_graduate_3 = 0.5809922748086525
recall_dropout_3 = 0.5809922748086525
f1_graduate_3 = 0.5800493683366588
f1_dropout_3 = 0.5800493683366588

# Hitung rata-rata F1-Score untuk setiap kelas
f1_score_graduate_avg = (f1_graduate_1 + f1_graduate_2 + f1_graduate_3) / 3
f1_score_dropout_avg = (f1_dropout_1 + f1_dropout_2 + f1_dropout_3) / 3

# Print hasil
print("Rata-rata F1-Score untuk kelas Graduate:", f1_score_graduate_avg)
print("Rata-rata F1-Score untuk kelas Dropout:", f1_score_dropout_avg)
```

```
Rata-rata F1-Score untuk kelas Graduate: 0.6108778654008893
Rata-rata F1-Score untuk kelas Dropout: 0.6108778654008893
```

```
# Import library
from sklearn.metrics import accuracy_score

# Data hasil evaluasi
y_true = [1, 0, 1, 0, 1]  # Label sebenarnya
y_pred = [1, 1, 0, 0, 1]  # Label yang diprediksi

# Menghitung akurasi
accuracy = accuracy_score(y_true, y_pred)

# Mencetak akurasi
print("Akurasi:", accuracy)
```

```
Akurasi: 0.6
```

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

# Membaca dataset
df = pd.read_csv("dataset.csv")

# Membagi dataset menjadi data latih dan data uji sesuai dengan skenario pengujian
# Skenario Pengujian 1
X_train_1, X_test_1, y_train_1, y_test_1 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.2, ra

# Skenario Pengujian 2
# Skenario Pengujian 2
X_train_2, X_test_2, y_train_2, y_test_2 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.5, ra

# Skenario Pengujian 3
X_train_3, X_test_3, y_train_3, y_test_3 = train_test_split(df.drop(columns=['Target']), df['Target'], test_size=0.8, ra

# Membuat dan melatih model Naive Bayes untuk setiap skenario pengujian
model_1 = GaussianNB()
model_1.fit(X_train_1, y_train_1)

model_2 = GaussianNB()
model_2.fit(X_train_2, y_train_2)

model_3 = GaussianNB()
model_3.fit(X_train_3, y_train_3)

# Melakukan prediksi pada data uji untuk setiap skenario
y_pred_1 = model_1.predict(X_test_1)
y_pred_2 = model_2.predict(X_test_2)
y_pred_3 = model_3.predict(X_test_3)

# Menghitung akurasi untuk setiap skenario
accuracy_1 = accuracy_score(y_test_1, y_pred_1)
```