```
# Akses API Kaggle
from google.colab import files
files.upload()  # Upload file kaggle.json di sini
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving kaggle (1).json to kaggle (1).json
{"kaggle (1).json": b'{"username":"nizgivegan","key":"1c7f4fd6b6bb18f371ac3ef6abcb7584"}'}
```

```
import pandas as pd  # pengolah data berkaitan data frame
import numpy as np  # manipulasi array secara mudah dan cepat
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB  # Import Gaussian Naive Bayes model
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Membaca file excel
df = pd.read_csv("/content/drive/MyDrive/Student Mental health.csv")
# Tampilkan 5 data pertama
print(df.head())
# Cek struktur data
print(df.info())
# Cek apakah ada missing value
print(df.isnull().sum())
```

```
         Timestamp Choose your gender   Age What is your course?  \
0  8/7/2020 12:02             Female  18.0          Engineering
1  8/7/2020 12:04               Male  21.0    Islamic education
2  8/7/2020 12:05               Male  19.0                  BIT
3  8/7/2020 12:06             Female  22.0                 Laws
4  8/7/2020 12:13               Male  23.0         Mathemathics

  Your current year of Study What is your CGPA? Marital status  \
0                     year 1        3.00 - 3.49            No
1                     year 2        3.00 - 3.49            No
2                     Year 1        3.00 - 3.49            No
3                     year 3        3.00 - 3.49           Yes
4                     year 4        3.00 - 3.49            No

  Do you have Depression? Do you have Anxiety? Do you have Panic attack?  \
0                     Yes                  No                       Yes
1                      No                 Yes                        No
2                     Yes                 Yes                       Yes
3                     Yes                  No                        No
4                      No                  No                        No

  Did you seek any specialist for a treatment?
0                                           No
1                                           No
2                                           No
3                                           No
4                                           No
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 11 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Timestamp                                     101 non-null    object
 1   Choose your gender                            101 non-null    object
 2   Age                                           100 non-null    float64
 3   What is your course?                          101 non-null    object
 4   Your current year of Study                    101 non-null    object
 5   What is your CGPA?                            101 non-null    object
 6   Marital status                                101 non-null    object
 7   Do you have Depression?                       101 non-null    object
 8   Do you have Anxiety?                          101 non-null    object
 9   Do you have Panic attack?                     101 non-null    object
 10  Did you seek any specialist for a treatment?  101 non-null    object
dtypes: float64(1), object(10)
memory usage: 8.8+ KB
None
Timestamp                                       0
Choose your gender                              0
Age                                             1
What is your course?                            0
Your current year of Study                      0
What is your CGPA?                              0
```

```
Marital status                                  0
Do you have Depression?                         0
Do you have Anxiety?                            0
Do you have Panic attack?                       0
Did you seek any specialist for a treatment?    0
dtype: int64
```

```
df.empty
```

```
False
```

```
df.size
```

```
1111
```

```
# 1. Drop the target column and select features
x = df.drop(['Choose your gender'], axis=1, errors='ignore')
```

```
# 2. Select the target variable
y = df['Choose your gender']
```

```
# 3. Convert categorical features to numeric using LabelEncoder
categorical_features = x.select_dtypes(include=['object']).columns
for feature in categorical_features:
    encoder = LabelEncoder() # Inisialisasi LabelEncoder di dalam loop
    x[feature] = encoder.fit_transform(x[feature].astype(str)) # ubah data type ke string
```

```
# 4. Impute missing values with the most frequent value for each column
from sklearn.impute import SimpleImputer # Import SimpleImputer here
imputer = SimpleImputer(strategy='most_frequent') # Use most frequent strategy
x = pd.DataFrame(imputer.fit_transform(x), columns=x.columns)

# --- AKHIR PERUBAHAN ---

# ... (Kode selanjutnya tetap sama) ...
```

```
# Split data
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 123)
```

```
# Mengaktifkan/memanggil/membuat fungsi klasifikasi Naive bayes
modelnb = GaussianNB()
```

```
# Memasukkan data training pada fungsi klasifikasi naive bayes
nbtrain = modelnb.fit(x_train, y_train)
```

```
# Prediksi hasil dari data uji
y_pred = nbtrain.predict(x_test)

# Hitung akurasi
print("Akurasi:", accuracy_score(y_test, y_pred))

# Tampilkan classification report
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Tampilkan confusion matrix
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Visualisasi confusion matrix
df_cm = pd.DataFrame(confusion_matrix(y_test, y_pred), index=["Gagal", "Berhasil"], columns=["Pred Gagal", "Pred Berhasil"])
sns.heatmap(df_cm, annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix")
plt.show()
```

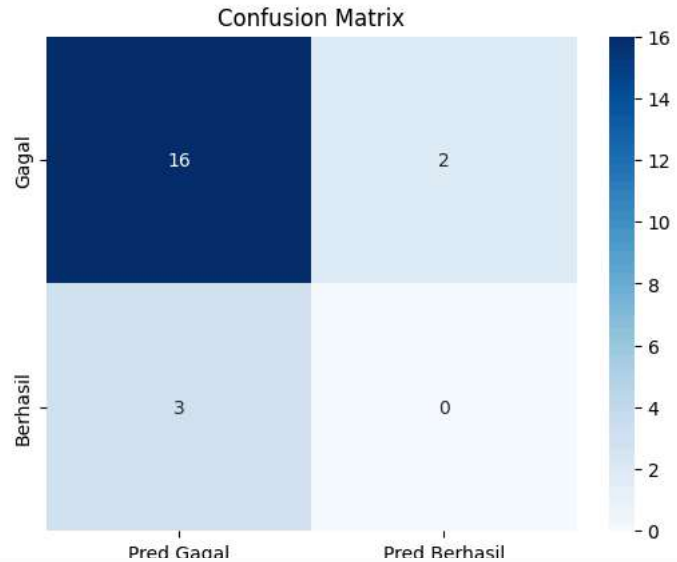
```
Akurasi: 0.7619047619047619

Classification Report:
              precision    recall  f1-score   support

      Female       0.84      0.89      0.86        18
        Male       0.00      0.00      0.00         3

    accuracy                           0.76        21
   macro avg       0.42      0.44      0.43        21
weighted avg       0.72      0.76      0.74        21


Confusion Matrix:
 [[16  2]
 [ 3  0]]
```




```
# Contoh Data Pasien Baru
# Pastikan data_pasien_baru memiliki 10 fitur, sesuai dengan data latih
# Anda perlu menyesuaikan nilai-nilai ini dengan data pasien baru yang sebenarnya
# dan urutan fitur yang sama seperti yang digunakan saat melatih model.
data_pasien_baru = [[35, 1, 10, 2, 5, 1, 0, 0, 0, 0]]  # Ganti dengan nilai yang sesuai

# Contoh:
# data_pasien_baru = [[usia, gender, durasi, jumlah, area, riwayat, fitur7, fitur8, fitur9, fitur10]]

prediksi = nbtrain.predict(data_pasien_baru)

if prediksi[0] == 1:
    print("Prediksi: Perawatan kemungkinan berhasil ✅")
else:
    print("Prediksi: Perawatan kemungkinan gagal ❌")
```

```
Prediksi: Perawatan kemungkinan gagal ❌
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Gaussian
  warnings.warn(
```

```
# Prediksi beberapa pasien
data_batch = [
    [29, 0, 5, 2, 3, 1, 0, 0, 0, 0],  # Add 4 more features here
    [45, 1, 12, 3, 7, 0, 0, 0, 0, 0],  # Add 4 more features here
    [31, 1, 8, 1, 2, 1, 0, 0, 0, 0],  # Add 4 more features here
]

hasil_batch = nbtrain.predict(data_batch)

for i, hasil in enumerate(hasil_batch):
    status = "Berhasil ✅" if hasil == 1 else "Gagal ❌"
    print(f"Pasien {i+1}: {status}")
```

```
Pasien 1: Gagal ❌
Pasien 2: Gagal ❌
Pasien 3: Gagal ❌
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Gaussian
  warnings.warn(
```

```
# Tampilkan probabilitas prediksi untuk pasien baru
proba = nbtrain.predict_proba(data_pasien_baru)
print(f"Probabilitas Gagal: {proba[0][0]:.2f}")
print(f"Probabilitas Berhasil: {proba[0][1]:.2f}")

# Gunakan probabilitas untuk pengambilan keputusan berbasis threshold
threshold = 0.7
if proba[0][1] > threshold:
    print("Rekomendasi: Lanjutkan perawatan karena kemungkinan berhasil tinggi.")
else:
    print("Rekomendasi: Pertimbangkan opsi lain karena kemungkinan berhasil rendah.")
```

```
Probabilitas Gagal: 0.60
Probabilitas Berhasil: 0.40
Rekomendasi: Pertimbangkan opsi lain karena kemungkinan berhasil rendah.
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Gaussian
  warnings.warn(
```