1. Mount Google Drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive;
```

Menghubungkan Google Colab dengan Google Drive agar file seperti diabetes.csv dapat diakses dari penyimpanan pengguna.

2. Import Library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
import sklearn.preprocessing
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
from sklearn import svm
```

Fungsi tiap pustaka:

- pandas & numpy → manipulasi data.
- matplotlib, seaborn → visualisasi grafik.
- sklearn → alat machine learning (preprocessing, evaluasi, model).
- warnings.filterwarnings('ignore') → menonaktifkan peringatan agar output lebih bersih.
- svm → mengimpor algoritma Support Vector Machine.

3. Membaca Dataset

```
df = pd.read_csv('/content/drive/MyDrive/praktikum_ml/praktikum06/data/diabetes.csv')
```

Membaca file CSV dari Google Drive dan menyimpannya sebagai DataFrame df.

4. Melihat Data Awal

```
df.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Menampilkan 5 baris pertama dari dataset agar tahu struktur kolom dan data yang tersedia.

5. Statistik Deskriptif

```
1 df.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81

Menampilkan statistik (mean, min, max, std, dll) untuk setiap kolom numerik.

6. Mengecek Missing Value

```
1 df.isnull().values.any()
```

```
np.False_
```

Mengecek apakah ada nilai kosong (NaN) di dataset.
Jika output False, berarti tidak ada nilai kosong.

7. Membersihkan Data

```
1 zero_not_allowed = ["Glucose","BloodPressure","SkinThickness"]
2
3 for column in zero_not_allowed:
4     df[column] = df[column].replace(0, np.nan)
5     mean = int(df[column].mean(skipna = True))
6     df[column] = df[column].replace(np.nan, mean)
```

Semua 0 diubah menjadi NaN (kosong).
Nilai kosong diisi dengan rata-rata kolom tersebut.

8. Split Data: Train dan Test

```
1 x = df.iloc[:, :-2]
2 y = df.iloc[:, -1]
3 x_train, x_test, y_train, y_test = train_test_split(x, y, random_state = 0, test_size = 0.2)
```

x = fitur input (semua kolom kecuali dua terakhir)
y = kolom target (Outcome)
80% data untuk pelatihan, 20% untuk pengujian.

9. Membuat dan Melatih Model SVM

```
1 clf = svm.SVC(kernel='rbf')
2 clf.fit(x_train,y_train)
3 y_pred = clf.predict(x_test)
```

SVC(kernel='rbf') → membuat model Support Vector Classifier dengan Radial Basis Function.
fit() → melatih model dengan data training.
predict() → memprediksi hasil pada data uji.

10. Mengukur Akurasi

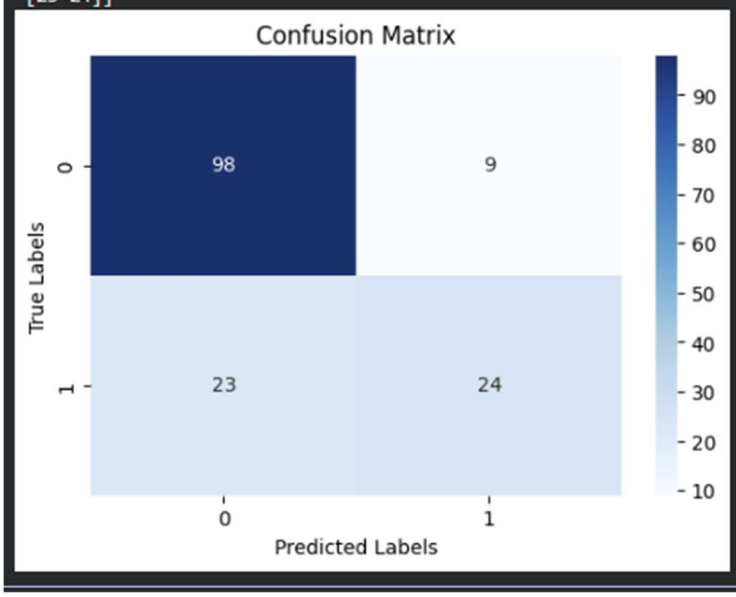
```
1 confusion_matrix(y_test,y_pred)

array([[98,  9],
       [23, 24]])
```

Menghitung persentase prediksi yang benar dibanding data sebenarnya.

11. Confusion Matrix (Numerik)

```
1 from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4
5 print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
6 cm = confusion_matrix(y_test, y_pred)
7 plt.figure(figsize=(6,4))
8 sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
9 plt.title("Confusion Matrix")
10 plt.xlabel("Predicted Labels")
11 plt.ylabel("True Labels")
12 plt.show()
```

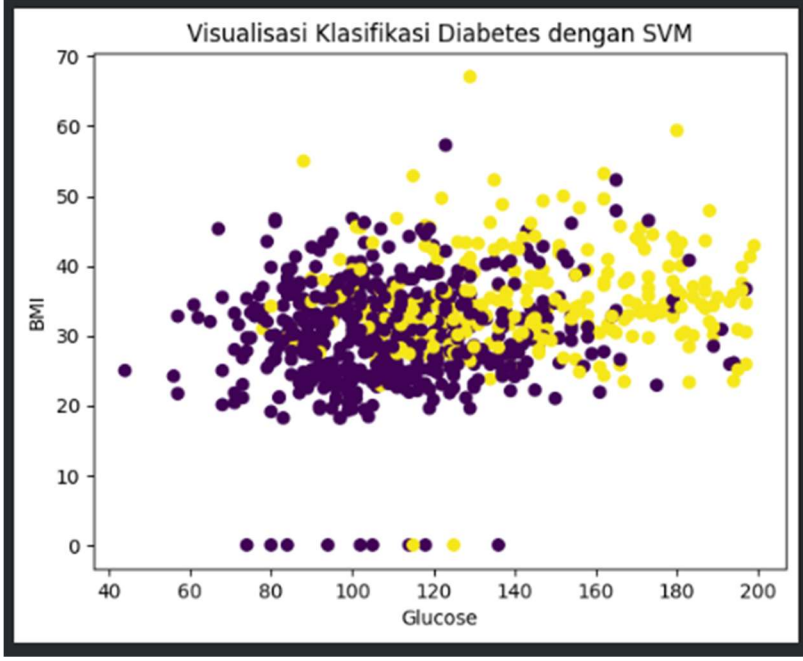
```
Confusion Matrix:
 [[98  9]
 [23 24]]
```



Menampilkan matriks kesalahan (TP, TN, FP, FN) dalam bentuk angka.
Menampilkan confusion matrix dalam bentuk heatmap untuk memudahkan interpretasi visual.
Warna biru menunjukkan jumlah prediksi benar/salah per kelas.

12. Visualisasi scatterplot

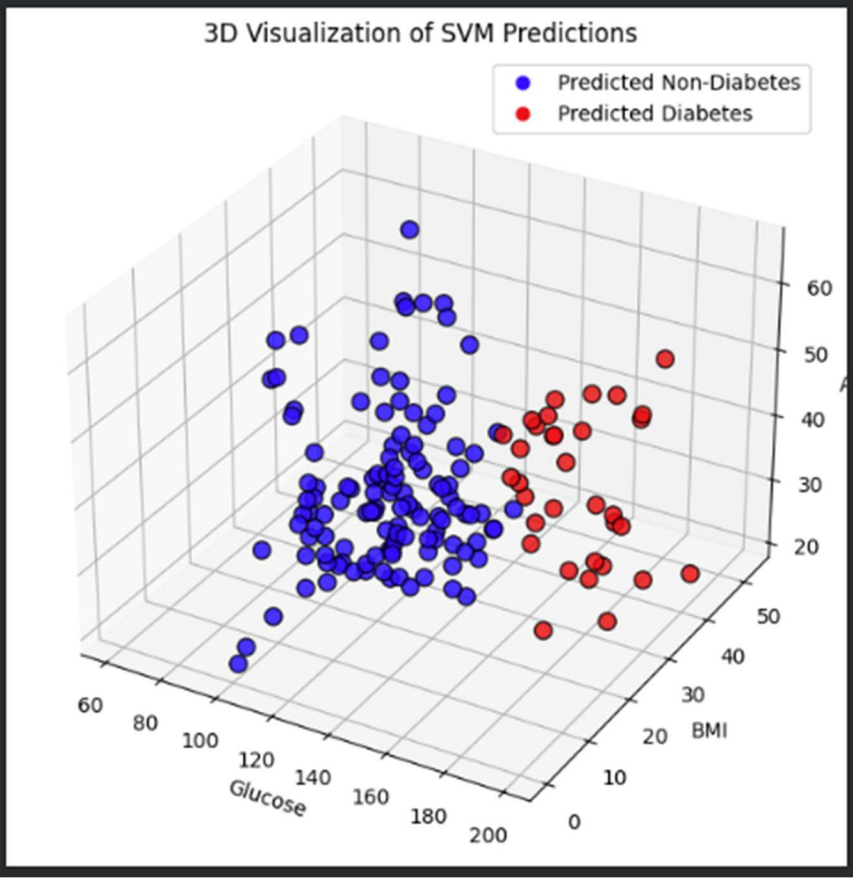
```python
import matplotlib.pyplot as plt

# Scatter plot menggunakan dua fitur numerik, misalnya Glucose dan BMI
plt.scatter(df['Glucose'], df['BMI'], c=df['Outcome'], cmap='viridis')

# Label sumbu dan judul
plt.xlabel('Glucose')
plt.ylabel('BMI')
plt.title('Visualisasi Klasifikasi Diabetes dengan SVM')
plt.show()
```



Menampilkan persebaran data pasien berdasarkan dua fitur utama, yaitu Glucose dan BMI.
plt.scatter() → membuat scatter plot dengan warna titik ditentukan oleh kolom Outcome.
cmap='viridis' → memberikan gradasi warna agar perbedaan kelas terlihat jelas.
plt.xlabel() dan plt.ylabel() → memberi label pada sumbu X (Glucose) dan Y (BMI).
plt.show() → menampilkan grafik ke layar.

13. Visualisasi 3D

```python
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import numpy as np

x = x_test['Glucose']
y = x_test['BMI']

colors = ['red' if label == 1 else 'blue' for label in y_pred]

fig = plt.figure(figsize=(9,7))
ax = fig.add_subplot(111, projection='3d')

ax.scatter(x, y, z, c=colors, s=70, alpha=0.8, edgecolor='k')

ax.set_xlabel('Glucose')
ax.set_ylabel('BMI')
ax.set_zlabel('Age')
ax.set_title('3D Visualization of SVM Predictions')

from matplotlib.lines import Line2D
legend_elements = [
    Line2D([0], [0], marker='o', color='w', label='Predicted Non-Diabetes',
           markerfacecolor='blue', markersize=8),
    Line2D([0], [0], marker='o', color='w', label='Predicted Diabetes',
           markerfacecolor='red', markersize=8)
]
ax.legend(handles=legend_elements, loc='best')

plt.show()
```



Membuat visualisasi 3D hasil prediksi SVM, agar terlihat perbedaan antara data pasien diabetes dan non-diabetes.
Axes3D → membuat plot 3 dimensi.
matplotlib.pyplot → menampilkan grafik.

Kesimpulan
Kode dalam praktikum ini berhasil membangun, melatih, dan mengevaluasi model SVM untuk klasifikasi diabetes, sekaligus menampilkan visualisasi 2D dan 3D yang memperjelas hasil prediksi model serta hubungan antar fitur utama pasien.