

## Praktikum 6 – Machine Learning Support Vector Machine (SVM)

Prepared By:

Dr. Sirojul Munir S.Si,M.Kom

Diah Ayu Puspasari

Tujuan :

1. Menjelaskan konsep dasar algoritma Support Vector Machine (SVM) dan mengimplementasikannya untuk kasus klasifikasi data bunga Iris, mulai dari tahap preprocessing hingga evaluasi model.

Dateline : 1 Pekan

Gitlab/Github :

Branch Repository : [PRODI ROMBEL]\_[NAMASINGKAT]\_[NIM] ( contoh: ti01\_budi\_0110112001)

### Aturan Pengerjaan:

1. Gunakan text editor yang nyaman bagi anda
2. Diperkenankan mengerjakan langsung bagi yang sudah memahami dan menguasai materi
3. Dilarang melakukan tindakan plagiarism (asisten lab akan mengecek hasil pekerjaan)
  - a. 1x nilai praktikum terkait bernilai 0
  - b. 2x nilai matakuliah pemrograman web E
  - c. 3x mahasiswa akan di sidang komite etik kampus

## Pendahuluan

Support vector machine (SVM) adalah salah satu algoritma machine learning yang digunakan untuk klasifikasi dan regresi. Prinsip kerja SVM adalah mencari hyperplane terbaik yang dapat memisahkan data dari dua kelas atau lebih dengan margin terbesar. Margin adalah jarak antara hyperplane dan data terdekat dari masing-masing kelas yang disebut support vector. Semakin besar margin antar kelas, semakin baik kemampuan model untuk melakukan generalisasi terhadap data baru.

Beberapa jenis kernel pada SVM:

- Linear → cocok untuk data yang dapat dipisahkan dengan garis lurus.
- Polynomial → memisahkan data menggunakan fungsi polinomial.
- RBF (Radial Basis Function) → cocok untuk data non-linear.

Klasifikasi data Iris dengan Support Vector Machine (SVM) salah satu tugas pembelajaran mesin yang melibatkan penggunaan algoritma SVM untuk memprediksi kelas dari dataset Iris. Tujuannya untuk mengembangkan model yang dapat memisahkan dan mengklasifikasikan data ke dalam tiga kelas tersebut.

SVM mencoba menemukan hyperplane (garis atau bidang dalam dimensi yang lebih tinggi) yang paling baik dalam memisahkan kelas-kelas data. Algoritma SVM akan memilih hyperplane dengan margin terbesar di antara kelas, sehingga model cenderung memiliki performa generalisasi yang baik.

### Langkah Awal:

Sebelum memulai praktikum ini, pastikan Anda telah menyiapkan struktur folder di Google Drive / Jupyter seperti pada praktikum sebelumnya. Untuk praktikum ke-6, buat sub-folder baru dengan nama praktikum06/ di dalam direktori utama praktikum\_ml/.

Struktur folder yang digunakan adalah sebagai berikut:

```
praktikum_ml/
├── praktikum01/
├── praktikum02/
├── praktikum03/
|   ├── data/
|   ├── notebooks/
|   ├── model/
|   └── reports/
```

#### Catatan:

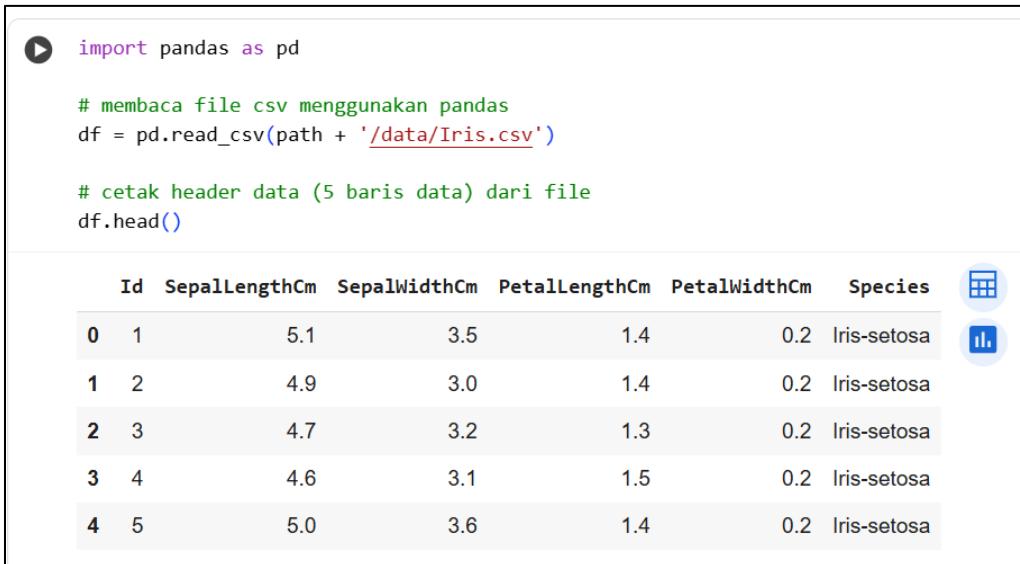
1. Jika telah memiliki folder praktikum\_ml, cukup tambahkan folder praktikum06.
2. Semua file notebook pada praktikum ini disimpan di dalam folder notebooks/ dengan nama praktikum06.ipynb.
3. Dataset ditempatkan di folder data/.
4. Model hasil training disimpan di folder model/ (format .pkl atau .joblib).
5. Visualisasi atau laporan hasil analisis disimpan di folder reports/. Jalankan file installer yang sudah diunduh.

Setelah itu, lakukan Langkah-langkah seperti biasa untuk Loading dataset. Mulai dari menghubungkannya dengan google drive sampai membaca dataset.

## Praktikum Support Vector Machine (SVM)

### 1. Loading Dataset

Pada tahap ini dilakukan proses membaca dan menampilkan dataset yang akan digunakan untuk membangun model



```
import pandas as pd

# membaca file csv menggunakan pandas
df = pd.read_csv(path + '/data/Iris.csv')

# cetak header data (5 baris data) dari file
df.head()
```

	ID	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Mengimpor library pandas, yaitu library Python yang digunakan untuk membaca, memproses, dan menganalisis data dalam bentuk tabel (DataFrame). Lalu, menampilkan 5 baris pertama dengan .head()

- Menampilkan informasi detail dengan df.info()

```
df.info()  
  
-> <class 'pandas.core.frame.DataFrame'>  
RangeIndex: 150 entries, 0 to 149  
Data columns (total 7 columns):  
 #   Column           Non-Null Count  Dtype     
---  --  
 0   Id               150 non-null    int64    
 1   SepalLengthCm   150 non-null    float64  
 2   SepalWidthCm    150 non-null    float64  
 3   PetalLengthCm   150 non-null    float64  
 4   PetalWidthCm    150 non-null    float64  
 5   Species          150 non-null    object    
 6   SpeciesEncoded  150 non-null    int64    
dtypes: float64(4), int64(2), object(1)  
memory usage: 8.3+ KB
```

Perintah ini memberikan ringkasan lengkap dari dataset, seperti:

- Jumlah baris dan kolom
- Nama kolom beserta tipe datanya (object, int64, float64)
- Jumlah nilai yang tidak kosong (Non-Null Count)
- Total penggunaan memori dataset

- Menampilkan statistika deskriptif dari dataset

```
df.describe()  
  
->      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  
count  150.000000    150.000000    150.000000    150.000000    150.000000  
mean   75.500000    5.843333     3.054000     3.758667     1.198667  
std    43.445368    0.828066     0.433594     1.764420     0.763161  
min    1.000000    4.300000     2.000000     1.000000     0.100000  
25%   38.250000    5.100000     2.800000     1.600000     0.300000  
50%   75.500000    5.800000     3.000000     4.350000     1.300000  
75%   112.750000   6.400000     3.300000     5.100000     1.800000  
max   150.000000   7.900000     4.400000     6.900000     2.500000
```

- Cek nilai pada kolom Species

Perintah `df["Species"].unique()` digunakan untuk menampilkan daftar nilai unik pada kolom Species. Hasilnya menunjukkan bahwa dataset Iris memiliki tiga jenis bunga, yaitu Iris-setosa, Iris-versicolor, dan Iris-virginica.

```
df["Species"].unique()  
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

- Menghitung jumlah pada kolom Species

Selanjutnya, perintah `df["Species"].value_counts()` digunakan untuk menghitung jumlah data pada setiap kelas. Hasil yang ditampilkan menunjukkan bahwa masing-masing spesies memiliki 50 data, sehingga total keseluruhan adalah 150 data.

```
df["Species"].value_counts()  
  
count  
Species  
Iris-setosa      50  
Iris-versicolor  50  
Iris-virginica   50  
  
dtype: int64
```

Dari hasil ini dapat disimpulkan bahwa dataset Iris bersifat seimbang (balanced) karena setiap kelas memiliki jumlah data yang sama, sehingga model SVM dapat dilatih tanpa bias terhadap salah satu kelas.

## 2. Pemilihan Fitur

Pada tahap ini dilakukan pemisahan antara fitur (X) dan label (y).

```
X = df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]  
  
# Kolom target (label)  
y = df['Species']
```

X.head()

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Next steps: [Generate code with X](#) [New interactive sheet](#)

y.head()

	Species
0	Iris-setosa
1	Iris-setosa
2	Iris-setosa
3	Iris-setosa
4	Iris-setosa

dtype: object

- Fitur (X) terdiri dari empat kolom numerik yaitu SepalLengthCm, SepalWidthCm, PetalLengthCm, dan PetalWidthCm, yang akan digunakan sebagai input model.
- Sedangkan label (y) berisi kolom Species, yaitu target keluaran berupa jenis bunga: Iris-setosa, Iris-versicolor, dan Iris-virginica.

Tahap ini bertujuan agar model SVM dapat belajar membedakan setiap spesies bunga berdasarkan nilai fitur-fiturnya.

### 3. Split dan Bangun Model SVM

Pada tahap ini dataset dibagi menjadi data training (80%) dan data testing (20%) menggunakan fungsi `train_test_split()`. Data training digunakan untuk melatih model, sedangkan data testing digunakan untuk menguji performa model.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Membuat model SVM dengan kernel linear
model = SVC(kernel='linear')
model.fit(X_train, y_train)

▼ SVC ⓘ ?  
SVC(kernel='linear')
```

Selanjutnya dibuat model Support Vector Machine (SVM) dengan kernel linear menggunakan `SVC(kernel='linear')`. Model kemudian dilatih dengan data training menggunakan `model.fit(X_train, y_train)` agar dapat mempelajari pola hubungan antara fitur dan label.

Tahapan ini merupakan dasar dari proses pembelajaran mesin di mana model mulai belajar membedakan setiap kelas bunga berdasarkan data latih yang diberikan.

### 4. Evaluasi Akurasi dan Report Klasifikasi

Tahap ini bertujuan untuk mengevaluasi performa model SVM yang telah dilatih. Proses dilakukan dengan memprediksi data uji menggunakan perintah `model.predict(X_test)` dan menyimpan hasilnya ke variabel `y_pred`.

```
▶ y_pred = model.predict(X_test)
# Akurasi
print(f"Akurasi: {accuracy_score(y_test, y_pred) * 100:.2f}%")
# Laporan klasifikasi
print("\nLaporan Klasifikasi:\n", classification_report(y_test, y_pred))

⇒ Akurasi: 100.00%

Laporan Klasifikasi:
      precision    recall  f1-score   support
Iris-setosa       1.00     1.00     1.00      10
Iris-versicolor   1.00     1.00     1.00       9
Iris-virginica    1.00     1.00     1.00      11

      accuracy         1.00      30
     macro avg       1.00     1.00     1.00      30
weighted avg      1.00     1.00     1.00      30
```

Nilai akurasi dihitung dengan fungsi `accuracy_score(y_test, y_pred)`, yang menunjukkan persentase prediksi benar terhadap total data uji. Hasil menunjukkan akurasi 100%, menandakan bahwa model mampu mengklasifikasikan semua data uji dengan tepat.

Selain itu, fungsi `classification_report()` digunakan untuk menampilkan metrik evaluasi lain seperti precision, recall, dan f1-score untuk setiap kelas (Iris-setosa, Iris-versicolor, dan Iris-virginica). Ketiga metrik tersebut juga bernilai sempurna (1.00), yang berarti model memiliki performa sangat baik dalam mengenali setiap jenis bunga.

## 5. Confusion Matrix

Tahap ini digunakan untuk menampilkan hasil klasifikasi model dalam bentuk Confusion Matrix, yaitu tabel yang menunjukkan jumlah prediksi benar dan salah untuk setiap kelas.

Kode `confusion_matrix(y_test, y_pred)` menghasilkan matriks perbandingan antara label sebenarnya (True Labels) dan label hasil prediksi (Predicted Labels). Visualisasi dibuat menggunakan `seaborn.heatmap()` agar lebih mudah dibaca.

```
▶ from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
  import matplotlib.pyplot as plt
  import seaborn as sns

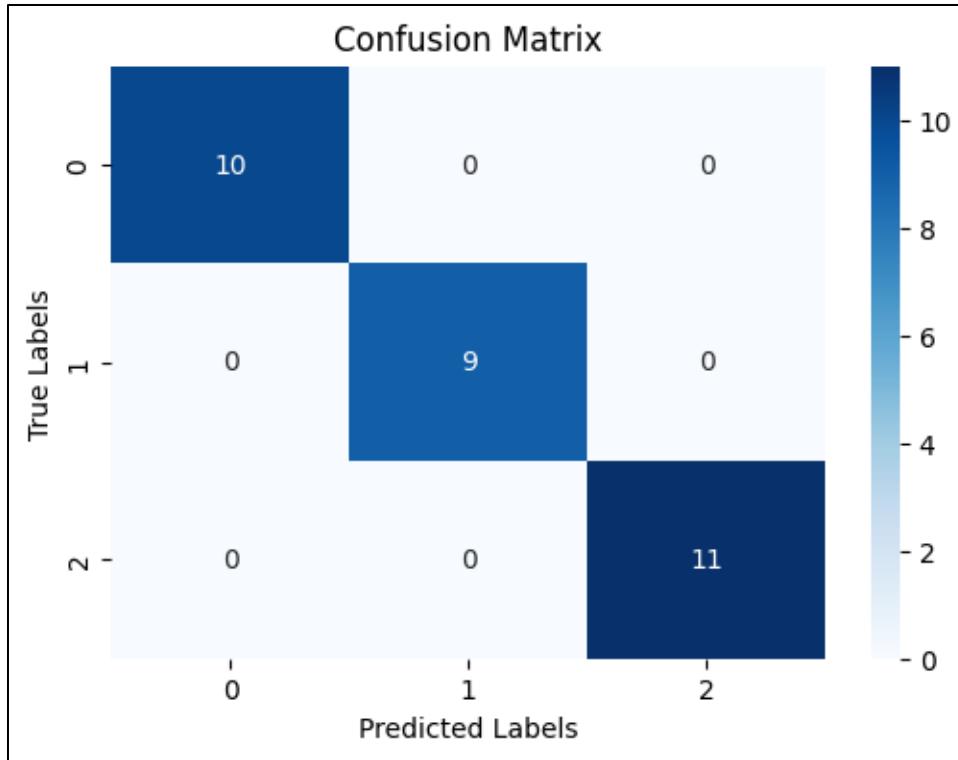
  print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))

  # Buat confusion matrix
  cm = confusion_matrix(y_test, y_pred)

  # Jika kamu tahu nama kelas (opsional, agar lebih informatif)
  # misalnya: class_names = ['Negatif', 'Positif']
  # maka tambahkan ke heatmap di bagian "xticklabels" dan "yticklabels"

  plt.figure(figsize=(6,4))
  sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
  plt.title("Confusion Matrix")
  plt.xlabel("Predicted Labels")
  plt.ylabel("True Labels")
  plt.show()

→
Confusion Matrix:
[[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```

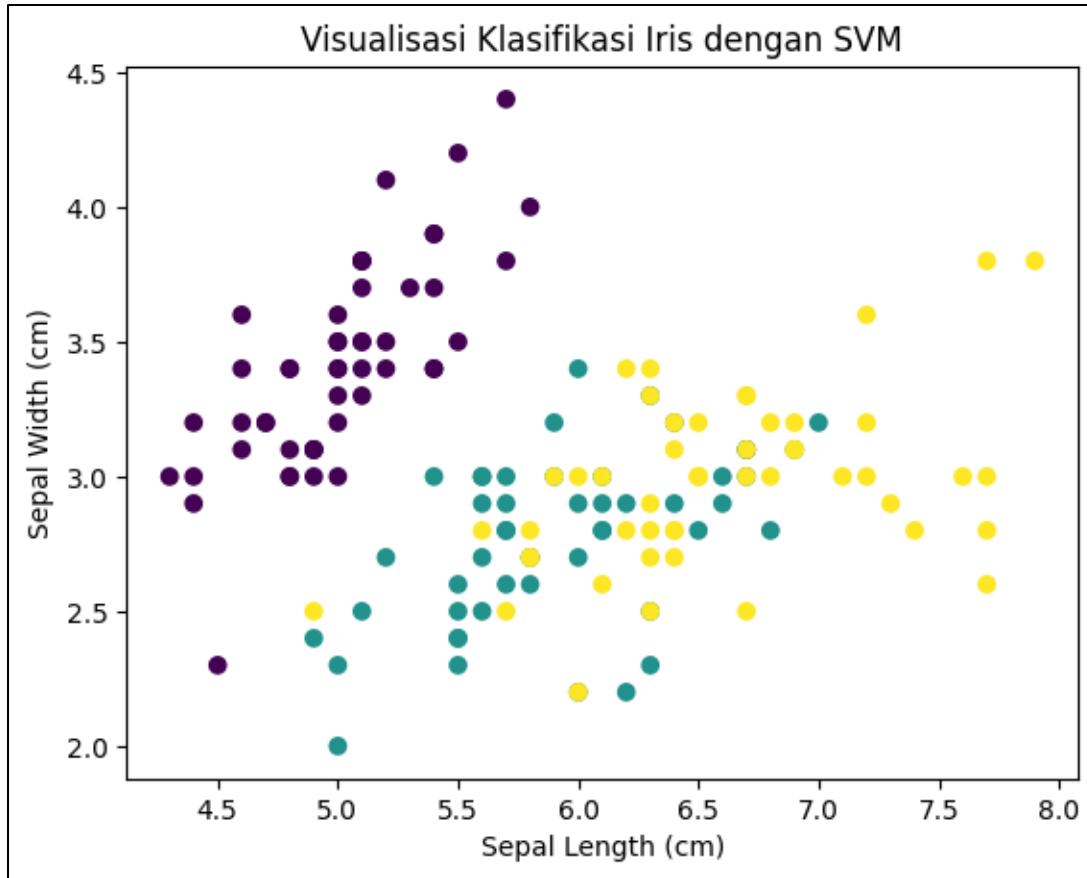


Berdasarkan hasil yang ditampilkan, seluruh data uji terklasifikasi dengan benar pada diagonal utama matriks (tanpa nilai kesalahan). Artinya, model SVM berhasil mengenali ketiga kelas bunga — Iris-setosa, Iris-versicolor, dan Iris-virginica — dengan akurasi sempurna (100%).

## 6. Visualisasi Hasil Model SVM

Tahap ini digunakan untuk menampilkan visualisasi sebaran data pada dataset Iris berdasarkan dua fitur utama yaitu Sepal Length dan Sepal Width. Visualisasi dibuat menggunakan fungsi plt.scatter() dari library Matplotlib, dengan warna titik (c) merepresentasikan masing-masing spesies bunga.

```
▶ import matplotlib.pyplot as plt
    plt.scatter(df['SepalLengthCm'], df['SepalWidthCm'], c=df['Species'].astype('category').cat.codes)
    plt.xlabel('Sepal Length (cm)')
    plt.ylabel('Sepal Width (cm)')
    plt.title('Visualisasi Klasifikasi Iris dengan SVM')
    plt.show()
```



Dari hasil grafik, terlihat bahwa setiap warna mewakili satu jenis bunga Iris yang berbeda. Titik-titik berwarna ungu, hijau, dan kuning membentuk kelompok yang terpisah, menunjukkan bahwa fitur-fitur tersebut cukup efektif dalam membedakan ketiga kelas bunga (Iris-setosa, Iris-versicolor, dan Iris-virginica).

Visualisasi ini membantu memverifikasi bahwa model SVM dapat mengklasifikasikan data dengan baik karena batas antar kelas terlihat cukup jelas.

## 7. 3D Visualisasi Hasil Model SVM

Tahap ini bertujuan untuk menampilkan hasil klasifikasi dataset Iris dalam bentuk grafik tiga dimensi (3D) agar pola antar kelas terlihat lebih jelas.

```
▶ from sklearn.preprocessing import LabelEncoder
  from sklearn.metrics import accuracy_score
  from mpl_toolkits.mplot3d import Axes3D

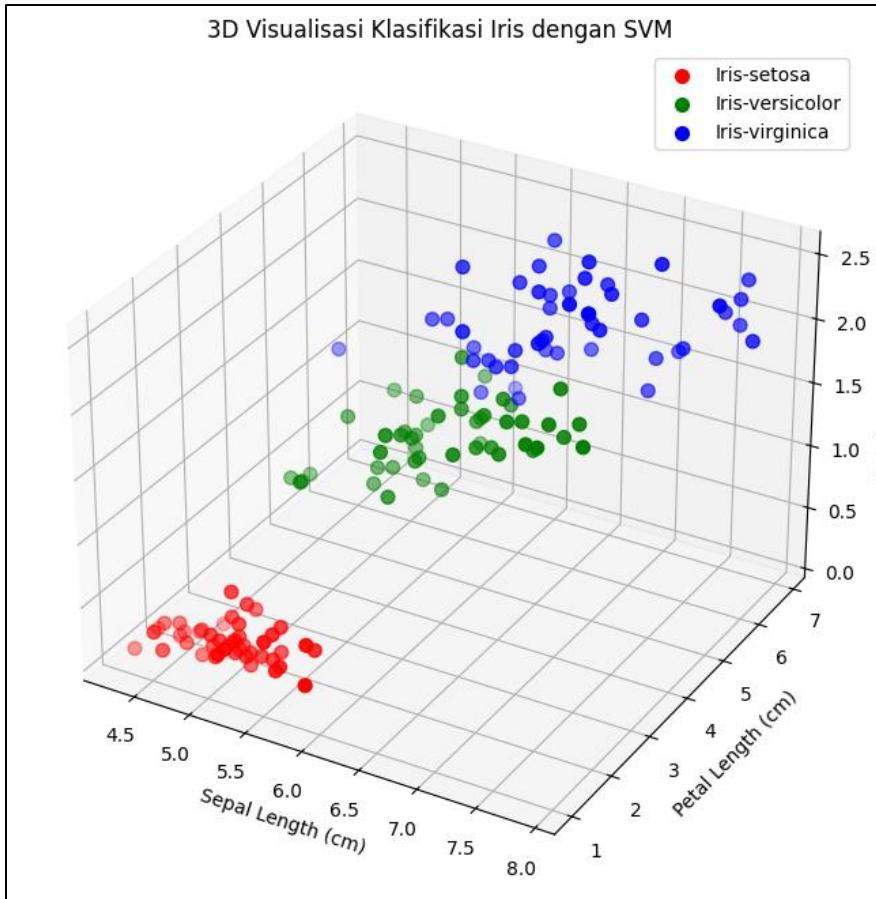
  # 3. Encode label (ubah teks jadi angka)
  le = LabelEncoder()
  df['SpeciesEncoded'] = le.fit_transform(df['Species'])

  # 8. Plot 3D hasil klasifikasi
  fig = plt.figure(figsize=(10, 8))
  ax = fig.add_subplot(111, projection='3d')

  # Warna untuk tiap kelas
  colors = ['r', 'g', 'b']
  labels = le.classes_

  # Plot tiap spesies dengan warna berbeda
  for i, species in enumerate(labels):
      subset = df[df['SpeciesEncoded'] == i]
      ax.scatter(
          subset['SepalLengthCm'],
          subset['PetalLengthCm'],
          subset['PetalWidthCm'],
          color=colors[i],
          label=species,
          s=50
      )

  ax.set_xlabel('Sepal Length (cm)')
  ax.set_ylabel('Petal Length (cm)')
  ax.set_zlabel('Petal Width (cm)')
  ax.set_title('3D Visualisasi Klasifikasi Iris dengan SVM')
  ax.legend()
  plt.show()
```



Pertama, data label pada kolom Species diubah menjadi bentuk numerik menggunakan LabelEncoder agar setiap spesies memiliki nilai angka berbeda. Kemudian dibuat visualisasi 3D menggunakan Axes3D dari library Matplotlib.

Sumbu X, Y, dan Z masing-masing merepresentasikan fitur Sepal Length, Petal Length, dan Petal Width. Setiap warna menandakan kelas bunga yang berbeda:

- ● Iris-setosa
- ● Iris-versicolor
- ● Iris-virginica

Dari hasil plot, terlihat ketiga spesies bunga membentuk kelompok yang terpisah dengan jelas, menandakan bahwa fitur-fitur pada dataset Iris sangat efektif digunakan oleh model SVM untuk melakukan klasifikasi.

## Tugas Praktikum Mandiri

1. Explore dataset di platform Kaggle, temukan dataset untuk solusi klasifikasi menggunakan SVM!
2. Gunakan SVM untuk klasifikasi dataset tersebut!