Nama: Noer Muhammad Ayub

NIM: 0110222142

Tugas: Manchine Learning

Laporan praktikum 6

1. **Import Library**

```
[15]
# Import library
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

Bagian import library ini digunakan untuk memanggil pustaka Python yang berfungsi dalam pengolahan data, visualisasi, pembagian data, standarisasi, pembangunan model SVM, serta evaluasi kinerja model machine learning.

2. **Menghubungkan colab dengan drive**

```
[2]
#menghubungkan colab dengan google drive
from google.colab import drive
drive.mount('/content/drive')

Mounted at /content/drive
```

Bagian kode ini digunakan untuk menghubungkan Google Colab dengan Google Drive agar file yang tersimpan di Drive dapat diakses dan digunakan langsung dalam lingkungan kerja Colab.

3. **Membaca file SCV dengan perintah delimiter**

```
[3]
#Membaca file SCV dengan perintah delimiter
path = ('/content/drive/MyDrive/praktikum/praktikum6')
```

Bagian kode ini digunakan untuk menentukan lokasi atau jalur folder di Google Drive tempat file CSV praktikum disimpan, sehingga dapat dibaca dan diakses dalam program.

4. **Membaca file csv**

```
[16]
#Membaca file csv
df = pd.read_csv(path + '/data/train.csv')
#ceetak data header (5 baris data dari file)
df.head()
```

Bagian kode ini digunakan untuk membaca file dataset berformat CSV dari folder yang telah ditentukan dan menampilkan lima baris pertama data sebagai contoh isi dari dataset tersebut.

5. **Menampilkan informasi umum**

```
[5]
✓ 0s
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
```

Kode df.info() digunakan untuk menampilkan informasi umum tentang dataset, seperti jumlah baris dan kolom, nama setiap kolom, tipe data masing-masing kolom, serta jumlah nilai yang tidak kosong (non-null) di setiap kolom.

6. **menampilkan statistik deskriptif**

```
[6]
✓ 0s
df.describe()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores
count	2000.000000	2000.0000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	1238.518500	0.4950	1.522250	0.509500	4.309500	0.521500	32.046500	0.501750	140.249000	4.520500
std	439.418206	0.5001	0.816004	0.500035	4.341444	0.499662	18.145715	0.288416	35.399655	2.287837
min	501.000000	0.0000	0.500000	0.000000	0.000000	0.000000	2.000000	0.100000	80.000000	1.000000
25%	851.750000	0.0000	0.700000	0.000000	1.000000	0.000000	16.000000	0.200000	109.000000	3.000000
50%	1226.000000	0.0000	1.500000	1.000000	3.000000	1.000000	32.000000	0.500000	141.000000	4.000000

Kode df.describe() digunakan untuk **menampilkan statistik deskriptif** dari data numerik dalam dataset, seperti nilai rata-rata (mean), standar deviasi (std), nilai minimum, maksimum, serta kuartil (25%, 50%, 75%) untuk membantu memahami sebaran data.

7. **mengecek jumlah nilai kosong**

```
[7]
✓ 0s
df.isnull().sum()
```

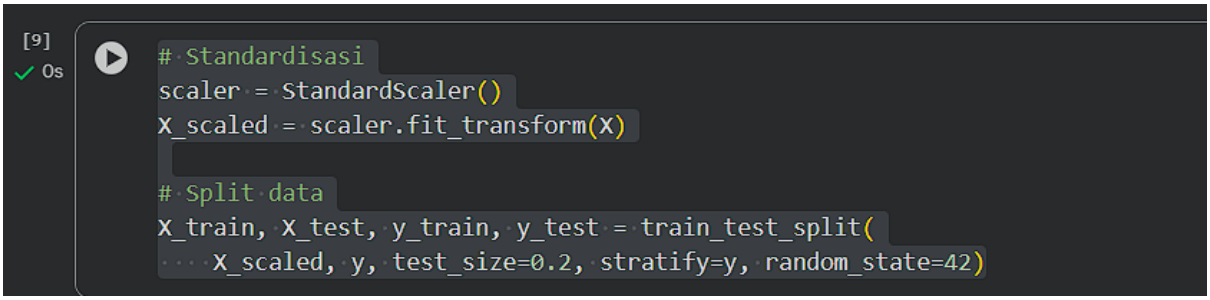
	0
battery_power	0
blue	0
clock_speed	0

Kode df.isnull().sum() digunakan untuk mengecek jumlah nilai kosong (missing values) di setiap kolom dataset, sehingga dapat diketahui apakah ada data yang perlu dibersihkan atau dilengkapi sebelum proses analisis.

8. memisahkan data fitur

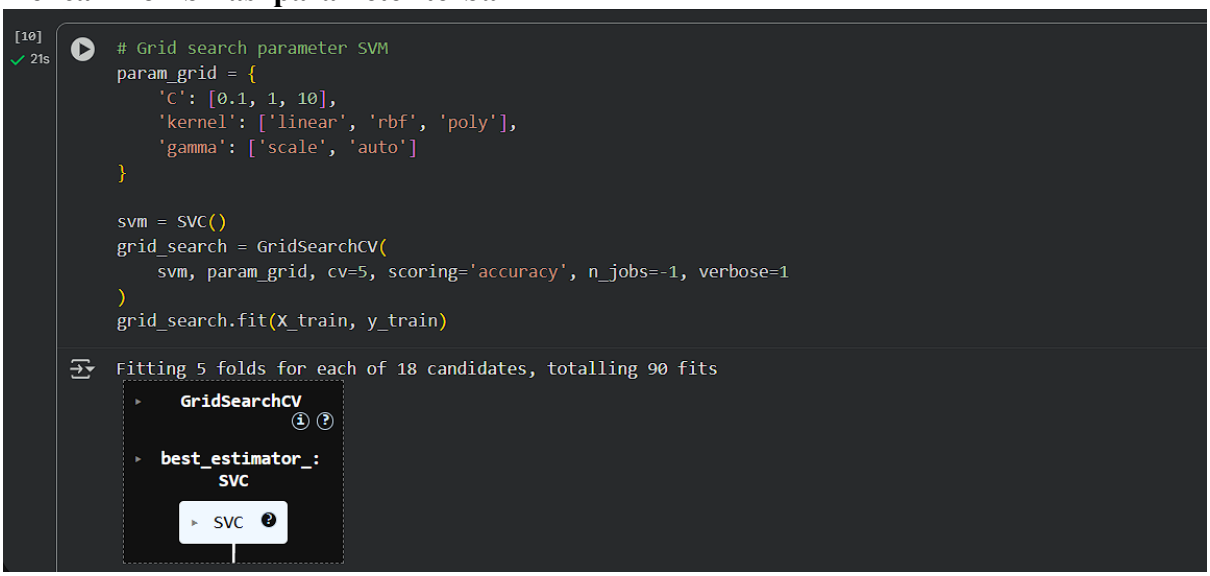
Kode tersebut digunakan untuk memisahkan data fitur (X) dan data target (y), di mana X berisi semua kolom kecuali price_range yang akan digunakan untuk memprediksi, sedangkan y berisi kolom price_range sebagai label atau variabel yang ingin diklasifikasikan.

9. menstandarkan data fitur



Kode tersebut digunakan untuk menstandarkan data fitur agar memiliki skala yang sama menggunakan StandardScaler, kemudian membagi dataset menjadi data latih (80%) dan data uji (20%) dengan menjaga proporsi kelas yang seimbang melalui parameter stratify=y.

10. mencari kombinasi parameter terbaik



Kode tersebut digunakan untuk mencari kombinasi parameter terbaik pada model SVM menggunakan GridSearchCV, dengan menguji berbagai nilai C, jenis kernel, dan gamma melalui proses cross-validation 5 lipatan (cv=5) agar diperoleh model dengan akurasi tertinggi.

11. mengambil model SVM terbaik

```
[11]
✓ 0s
# Model terbaik
best_model = grid_search.best_estimator_
print("Best Parameters:", grid_search.best_params_)

Best Parameters: {'C': 10, 'gamma': 'scale', 'kernel': 'linear'}
```

Kode tersebut digunakan untuk mengambil model SVM terbaik hasil dari GridSearchCV dan menampilkan kombinasi parameter terbaik (best_params_) yang memberikan performa akurasi tertinggi pada proses pencarian.

12. mengukur kinerja model SVM terbaik

```
[12]
✓ 0s
# Evaluasi
y_pred = best_model.predict(X_test)
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("Akurasi Model:", accuracy_score(y_test, y_pred))
```

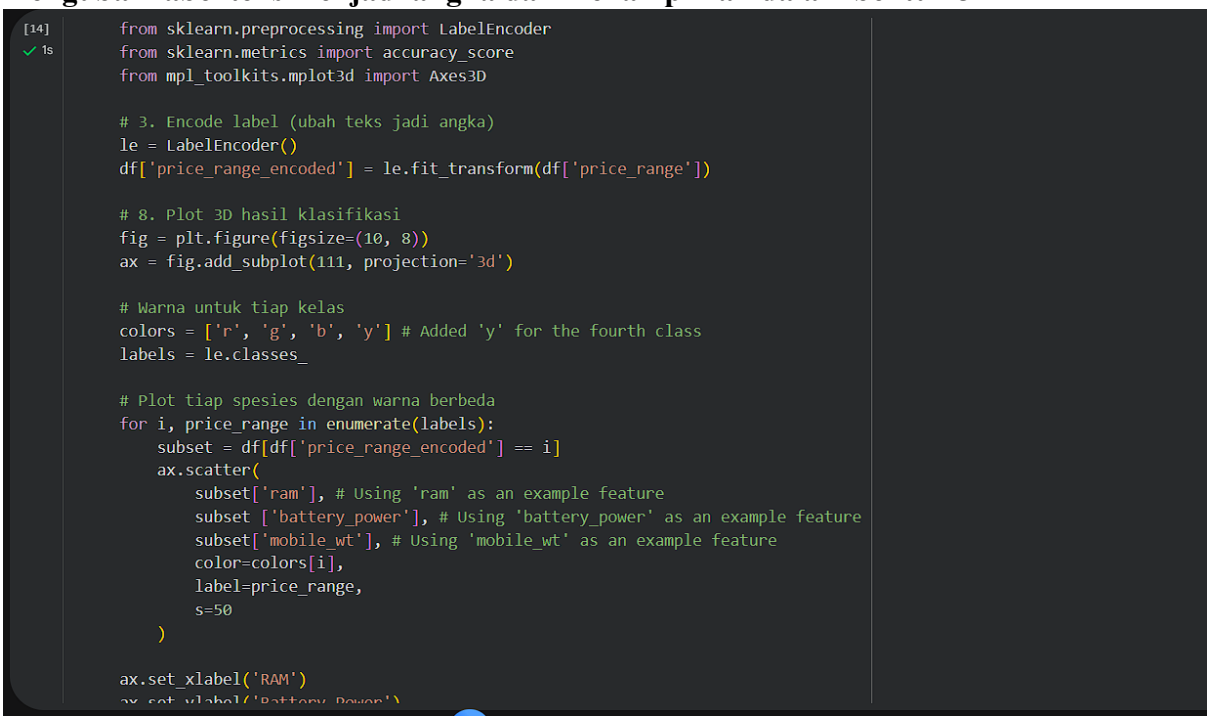
Kode tersebut digunakan untuk mengukur kinerja model SVM terbaik dengan cara memprediksi data uji (y_pred) lalu menampilkan laporan klasifikasi yang berisi nilai precision, recall, dan f1-score, serta menghitung akurasi total model menggunakan accuracy_score.

13. menampilkan hasil prediksi

```
[13]
✓ 0s
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix SVM")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

Kode tersebut digunakan untuk menampilkan hasil prediksi model dalam bentuk confusion matrix, yaitu tabel yang memperlihatkan perbandingan antara kelas sebenarnya dan hasil prediksi, kemudian divisualisasikan menggunakan heatmap Seaborn agar lebih mudah dibaca dan dianalisis.

14. mengubah label teks menjadi angka dan menampilkan dalam bentuk 3D



```
[14]
✓ 1s
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['price_range_encoded'] = le.fit_transform(df['price_range'])

# 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', 'y'] # Added 'y' for the fourth class
labels = le.classes_

# Plot tiap spesies dengan warna berbeda
for i, price_range in enumerate(labels):
    subset = df[df['price_range_encoded'] == i]
    ax.scatter(
        subset['ram'], # Using 'ram' as an example feature
        subset ['battery_power'], # Using 'battery_power' as an example feature
        subset['mobile_wt'], # Using 'mobile_wt' as an example feature
        color=colors[i],
        label=price_range,
        s=50
    )

ax.set_xlabel('RAM')
ax.set_ylabel('Battery Power')
```

Kode tersebut digunakan untuk mengubah label teks menjadi angka dan menampilkan hasil klasifikasi dalam bentuk visualisasi 3D.

Secara rinci:

- LabelEncoder mengonversi kolom price_range yang berbentuk teks menjadi angka agar dapat diproses oleh model machine learning.
- Setelah itu, dibuat grafik 3D menggunakan fitur seperti ram, battery_power, dan mobile_wt untuk menggambarkan persebaran data berdasarkan kelas harga ponsel.
- Tiap kelas harga ditampilkan dengan warna berbeda sehingga memudahkan dalam melihat pola dan perbedaan antar kelompok data.