

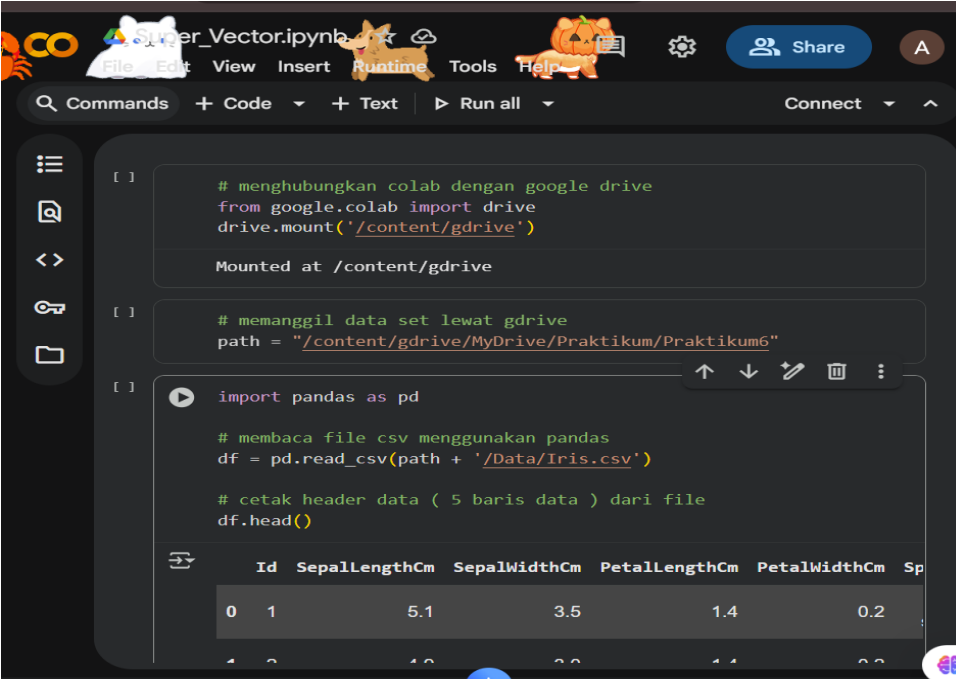
# Tugas 6: Tugas Praktikum Mandiri 6 – Machine Learning

Alia Maisyarah 1 - 0110224095 <sup>1\*</sup>

<sup>1</sup> Teknik Informatika, STT Terpadu Nurul Fikri, Depok

\*E-mail: [40110224095@student.nurulfikri.ac.id](mailto:40110224095@student.nurulfikri.ac.id) – email mahasiswa 1

## 1. Latihan Mandiri 6



The screenshot shows a Google Colab notebook interface. The code is as follows:

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

Mounted at /content/gdrive

[ ] # memanggil data set lewat gdrive
path = "/content/gdrive/MyDrive/Praktikum/Praktikum6"

[ ] 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()
```

Below the code, the output of `df.head()` is displayed as a table:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Sp
0	1	5.1	3.5	1.4	0.2	
1	2	4.9	3.0	1.4	0.2	
2	3	4.7	3.2	1.3	0.2	
3	4	4.6	3.1	1.5	0.2	
4	5	5.0	3.6	1.4	0.2	

1. # menghubungkan colab dengan google drive  
from google.colab import drive  
drive.mount('/content/gdrive')

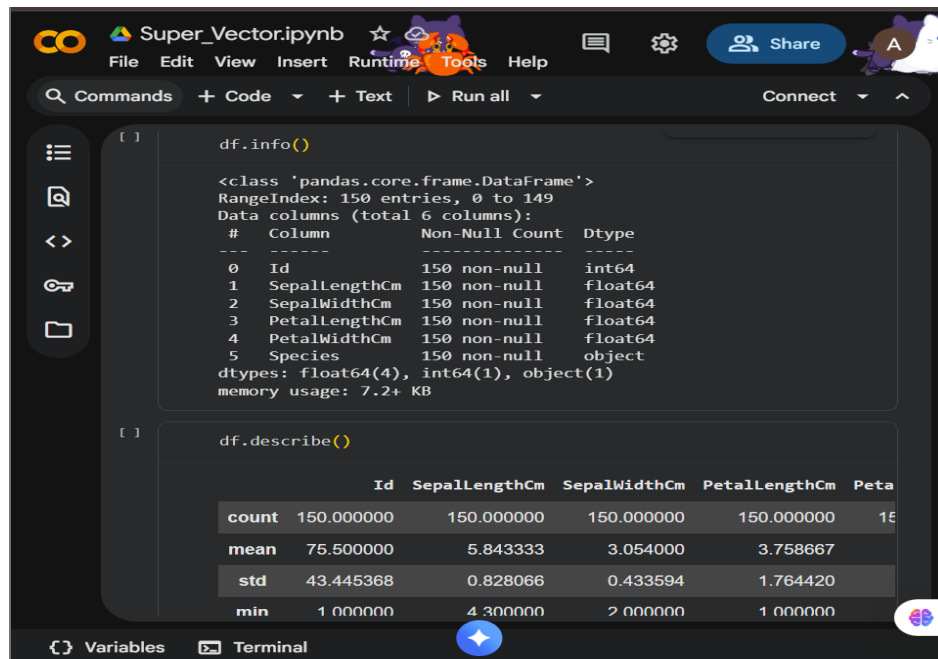
(Perintah ini digunakan supaya file di Google Drive bisa dipakai di Colab — misalnya buat membaca dataset (day.csv, hour.csv), menyimpan hasil, atau memuat model.)

2. # memanggil data set lewat gdrive  
path = "/content/gdrive/MyDrive/Praktikum/Praktikum4"

(Kode ini digunakan untuk menunjukkan lokasi dataset di Google Drive, supaya nanti ketika kamu memanggil data (misalnya dengan `pd.read_csv(path + '/day.csv')`), Python tahu harus mencari file-nya di mana.)

3. Import library  
import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.tree import DecisionTreeClassifier, plot\_tree  
from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix  
import matplotlib.pyplot as plt

(Kode ini berfungsi untuk menyiapkan semua pustaka yang diperlukan dalam pembuatan, pelatihan, evaluasi, dan visualisasi model *Decision Tree* menggunakan dataset Iris.)



The screenshot shows a Jupyter Notebook titled 'Super\_Vector.ipynb'. The first code cell contains `df.info()`, which outputs the following information:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 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 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

The second code cell contains `df.describe()`, which outputs a summary statistics table:

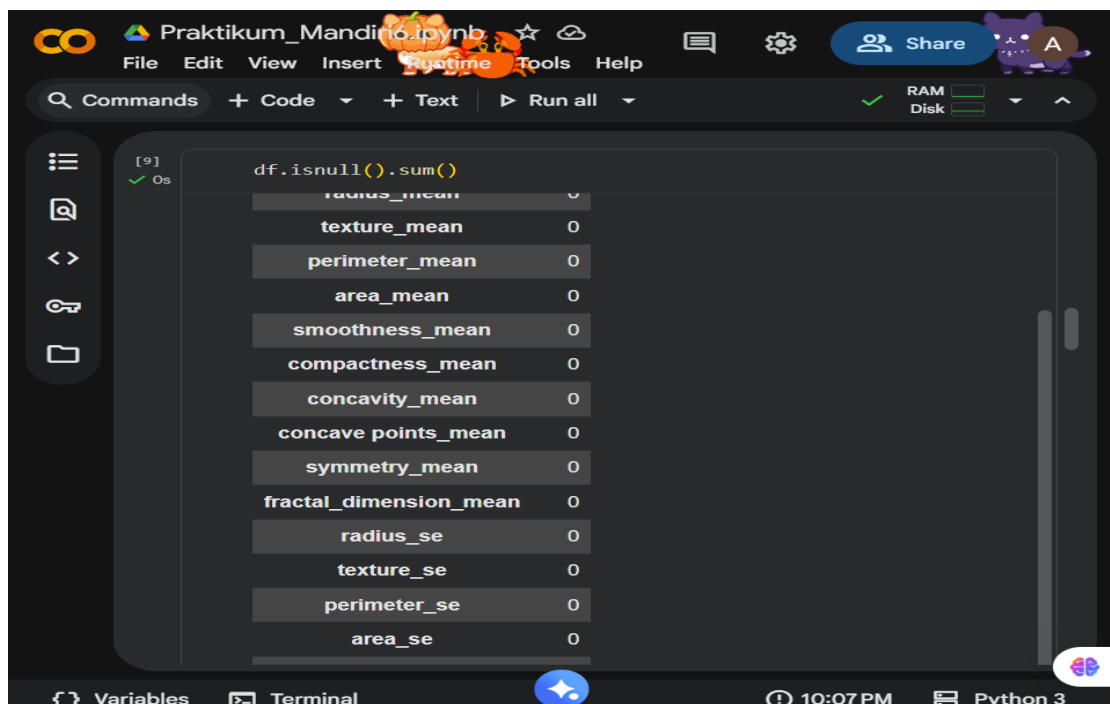
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.591333
std	43.445368	0.828066	0.433594	1.764420	0.762238
min	1.000000	4.300000	2.000000	1.000000	0.100000

#### 4. `df.info()`

(Dipakai buat melihat struktur data, tipe data, dan ada tidaknya data kosong di dataset kamu. Biasanya ini dijalankan di awal analisis supaya kamu paham dulu bentuk data yang kamu hadapi.)

#### 5. `df.describe()`

(dipakai untuk melihat ringkasan statistik dari data numerik membantu kamu memahami sebaran data, rata-rata, dan rentang nilainya sebelum analisis lebih lanjut.)

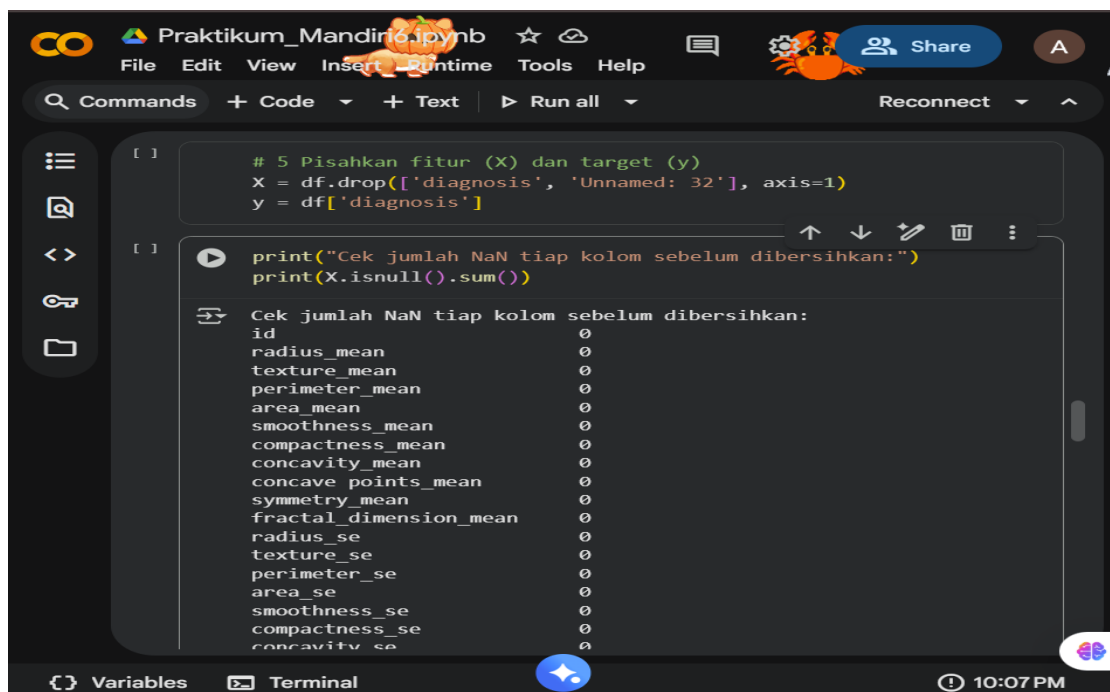


```
df.isnull().sum()
```

Feature	Count
radius_mean	0
texture_mean	0
perimeter_mean	0
area_mean	0
smoothness_mean	0
compactness_mean	0
concavity_mean	0
concave points_mean	0
symmetry_mean	0
fractal_dimension_mean	0
radius_se	0
texture_se	0
perimeter_se	0
area_se	0

6. `df.isnull().sum()`

(Artinya semua kolom tidak ada data yang kosong (0 = tidak ada nilai hilang))



```
# 5 Pisahkan fitur (X) dan target (y)
X = df.drop(['diagnosis', 'Unnamed: 32'], axis=1)
y = df['diagnosis']

print("Cek jumlah NaN tiap kolom sebelum dibersihkan:")
print(X.isnull().sum())
```

Cek jumlah NaN tiap kolom sebelum dibersihkan:

Feature	Count
id	0
radius_mean	0
texture_mean	0
perimeter_mean	0
area_mean	0
smoothness_mean	0
compactness_mean	0
concavity_mean	0
concave points_mean	0
symmetry_mean	0
fractal_dimension_mean	0
radius_se	0
texture_se	0
perimeter_se	0
area_se	0
smoothness_se	0
compactness_se	0
concavity_se	0

7. Pisahkan fitur (X) dan target (y)

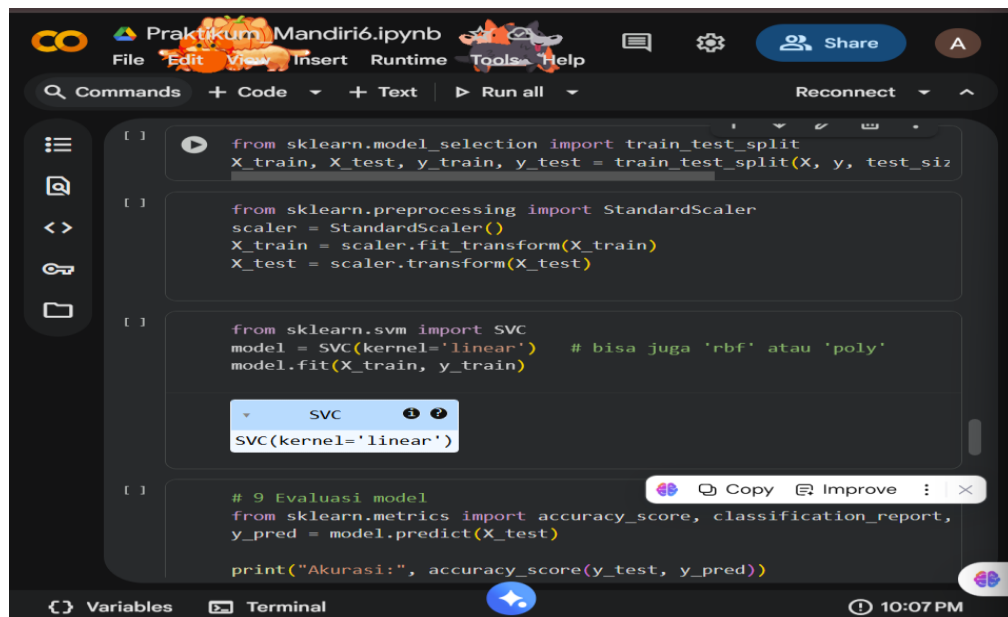
```
X = df.drop(['diagnosis', 'Unnamed: 32'], axis=1)
```

```
y = df['diagnosis']
```

(Kode ini digunakan untuk memisahkan data fitur (X) dan data target (y) pada dataset. Biasanya dilakukan sebelum pelatihan model Machine Learning.)

8. `print("Cek jumlah NaN tiap kolom sebelum dibersihkan:")`  
`print(X.isnull().sum())`

(digunakan untuk mengecek apakah ada data kosong (NaN) di setiap kolom pada dataset fitur X.)



The screenshot shows a Jupyter Notebook titled 'Praktikum Mandiri6.ipynb'. It contains four code cells. The first cell imports `train_test_split` from `sklearn.model_selection` and splits the data. The second cell imports `StandardScaler` from `sklearn.preprocessing` and scales the training and testing data. The third cell imports `SVC` from `sklearn.svm` and fits the model. A variable inspector shows the `SVC` object with `kernel='linear'`. The fourth cell, titled '# 9 Evaluasi model', imports `accuracy_score` and `classification_report` from `sklearn.metrics`, predicts on the test data, and prints the accuracy.

```
[ ] from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

[ ] from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

[ ] from sklearn.svm import SVC
model = SVC(kernel='linear') # bisa juga 'rbf' atau 'poly'
model.fit(X_train, y_train)

SVC
SVC(kernel='linear')

[ ] # 9 Evaluasi model
from sklearn.metrics import accuracy_score, classification_report
y_pred = model.predict(X_test)

print("Akurasi:", accuracy_score(y_test, y_pred))
```

9. `from sklearn.model_selection import train_test_split`  
`X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)`

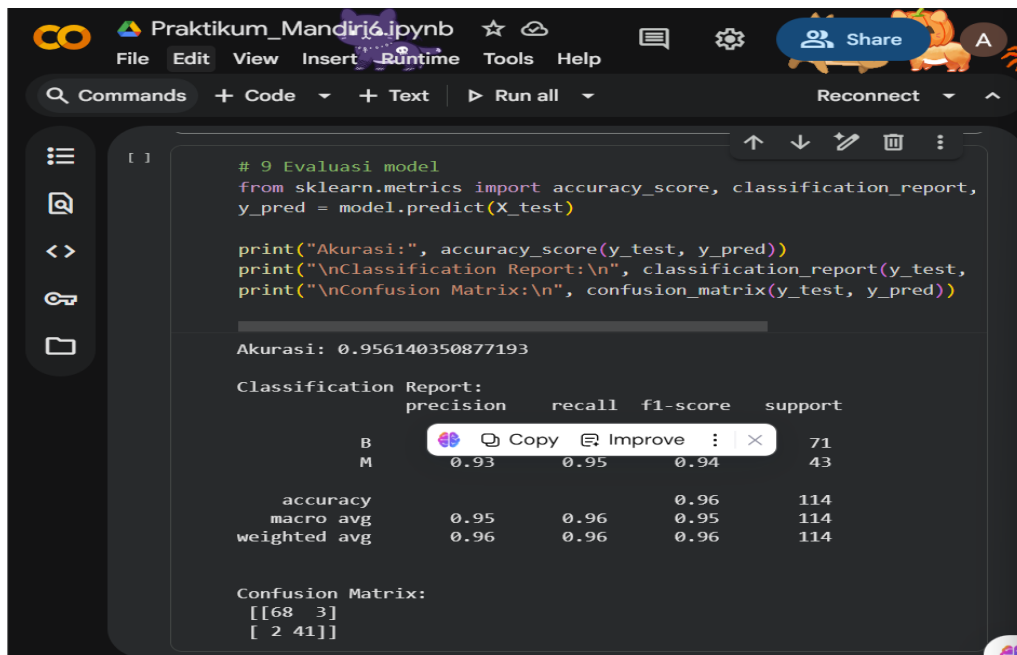
(Kode ini fungsinya untuk memisahkan data menjadi data latih dan data uji sebelum model dilatih.)

10. `from sklearn.preprocessing import StandardScaler`  
`scaler = StandardScaler()`  
`X_train = scaler.fit_transform(X_train)`  
`X_test = scaler.transform(X_test)`

(Kode ini digunakan untuk menormalkan atau menstandarkan data fitur (X) sebelum dimasukkan ke model.)

11. `from sklearn.svm import SVC`  
`model = SVC(kernel='linear') # bisa juga 'rbf' atau 'poly'`  
`model.fit(X_train, y_train)`

(Kode ini digunakan untuk membuat dan melatih model klasifikasi menggunakan algoritma Support Vector Machine (SVM).)



The screenshot shows a Jupyter Notebook titled 'Praktikum\_Mandiri.ipynb'. The code cell contains the following Python code:

```
# 9 Evaluasi model
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
y_pred = model.predict(X_test)

print("Akurasi:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

The output of the code is displayed below the code cell:

```
Akurasi: 0.956140350877193

Classification Report:
              precision    recall  f1-score   support

      B         0.93         0.95         0.94         71
      M         0.96         0.96         0.96         43

   accuracy          0.95
  macro avg          0.96
 weighted avg          0.96
```

The Confusion Matrix is also displayed:

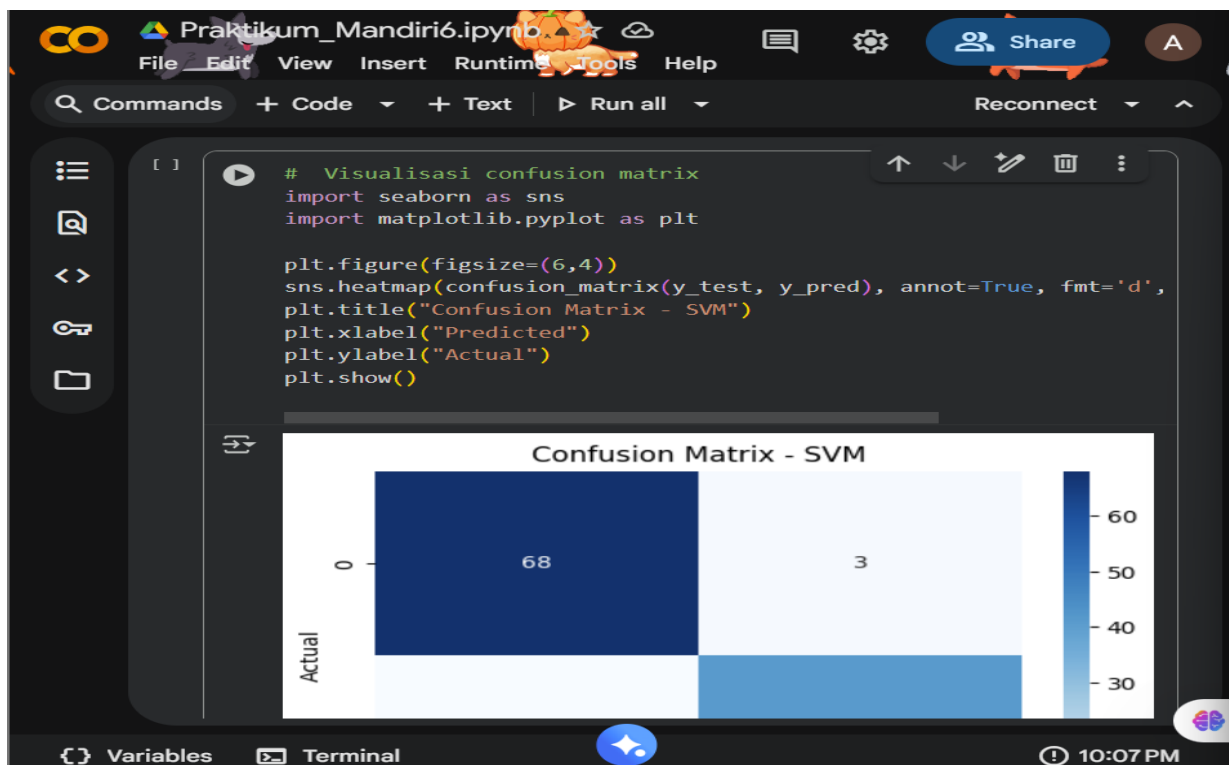
```
Confusion Matrix:
[[68  3]
 [ 2 41]]
```

## 12. 9 Evaluasi model

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
y_pred = model.predict(X_test)
```

```
print("Akurasi:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

(Kode ini digunakan untuk mengevaluasi seberapa baik model SVM kamu bekerja dalam memprediksi data uji ( $X_{test}$ )).



### 13. Visualisasi confusion matrix

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
plt.figure(figsize=(6,4))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix - SVM")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
```

(Kode ini digunakan untuk menampilkan hasil evaluasi model dalam bentuk visual (grafik), yaitu confusion matrix.)

### LINK GITHUB UPLOAD TUGAS :

<https://github.com/AliaMaisyarah14/Praktikum02ML.git>

