Practical NO: 2

Name: Sayali S. Shedge

PRN: RBTL23CS146

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import classification_report
```

[15] ✓ 0.0s

```
# Load iris dataset
iris = datasets.load_iris()
X = iris.data[:, :2]   # only first 2 features
y = iris.target
```

[11] ✓ 0.0s

```
# Use only two classes for binary classification
X = X[y != 2]
y = y[y != 2]
```

[10] ✓ 0.0s

```
# Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)
```

[12] ✓ 0.0s

```
# Create SVM model
svm_clf = SVC(kernel='linear', C=1.0)

# Train model
svm_clf.fit(X_train, y_train)
```

[5] ✓ 0.0s

```
SVC
▶ Parameters
```

```python
# Predictions
y_pred = svm_clf.predict(X_test)

# Performance
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Confusion Matrix:
 [[17  0]
 [ 0 13]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        17
           1       1.00      1.00      1.00        13

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

Plot decision boundary

def plot_svm_decision_boundary(model, X, y):

plt.scatter(X[:, 0], X[:, 1], c=y, cmap='coolwarm', s=30)

Get the separating hyperplane

ax = plt.gca()

xlim = ax.get_xlim()

ylim = ax.get_ylim()

Create grid

xx = np.linspace(xlim[0], xlim[1], 30)

yy = np.linspace(ylim[0], ylim[1], 30)

YY, XX = np.meshgrid(yy, xx)

xy = np.vstack([XX.ravel(), YY.ravel()]).T

Z = model.decision_function(xy).reshape(XX.shape)

```
# Plot decision boundary and margins
ax.contour(XX, YY, Z, colors='k', levels=[-1, 0, 1],
alpha=0.7, linestyles=['--', '-', '--'])
# Plot support vectors
ax.scatter(model.support_vectors_[:, 0],
model.support_vectors_[:, 1],
s=100, linewidth=1, facecolors='none', edgecolors='k')
plt.xlabel("Sepal length")
plt.ylabel("Sepal width")
plt.title("SVM Optimal Separating Hyperplane")
plt.show()
plot_svm_decision_boundary(svm_clf, X_train, y_train)
```