Task 7: Support Vector Machines (SVM)

Mini Guide with Code

Objective: Use SVMs for linear and non-linear classification.

Tools: Scikit-learn, NumPy, Matplotlib

Step 1: Load and prepare a dataset for binary classification

```
from sklearn.datasets import make_moons
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

X, y = make_moons(n_samples=300, noise=0.2, random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Step 2: Train an SVM with linear and RBF kernel

```
from sklearn.svm import SVC

# Linear Kernel
svm_linear = SVC(kernel='linear')
svm_linear.fit(X_train, y_train)
```

Task 7: Support Vector Machines (SVM)

```
# RBF Kernel
svm_rbf = SVC(kernel='rbf', gamma='scale')
svm_rbf.fit(X_train, y_train)

Step 3: Visualize decision boundary using 2D data
-------------------------------------------------
import matplotlib.pyplot as plt
import numpy as np

def plot_decision_boundary(model, X, y):
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.linspace(x_min, x_max, 100),
                         np.linspace(y_min, y_max, 100))
    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, alpha=0.3)
    plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k')
    plt.title("Decision Boundary")
    plt.show()

plot_decision_boundary(svm_rbf, X_test, y_test)
```

Task 7: Support Vector Machines (SVM)

Step 4: Tune hyperparameters like C and gamma

```
from sklearn.model_selection import GridSearchCV


param_grid = {
    'C': [0.1, 1, 10],
    'gamma': ['scale', 0.1, 1, 10]
}


grid = GridSearchCV(SVC(kernel='rbf'), param_grid, cv=5)
grid.fit(X_train, y_train)
print("Best parameters:", grid.best_params_)
```

Step 5: Use cross-validation to evaluate performance

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report, confusion_matrix


scores = cross_val_score(grid.best_estimator_, X, y, cv=5)
print("Cross-validation scores:", scores)
print("Mean accuracy:", scores.mean())


y_pred = grid.best_estimator_.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

Task 7: Support Vector Machines (SVM)

Outcome:

- Understand the difference between linear and non-linear SVM.

- Visual interpretation of decision boundaries.

- Impact of hyperparameters on model accuracy.