|  |  |
| --- | --- |
| **EXP NO: 2** | **SUPPORT VECTOR MACHINE (SVM) AND RANDOM FOREST FOR BINARY & MULTICLASS CLASSIFICATION** |

# AIM

To build classification models using **Support Vector Machines (SVM)** and **Random Forest**, apply them to a dataset, and evaluate the models using performance metrics like accuracy and confusion matrix.

# ALGORITHM

Part A: SVM Model

1. Import necessary libraries
2. Load and explore the dataset
3. Handle missing values if any
4. Encode categorical variables
5. Split dataset into training and testing sets
6. Build SVM classifier using SVC()
7. Train and predict
8. Evaluate the model using accuracy and confusion matrix Part B: Random Forest Model
9. Initialize Random Forest using RandomForestClassifier()
10. Train and predict
11. Evaluate and compare with SVM

# CODE:

# 1. Import libraries import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.model\_selection import train\_test\_split from sklearn.preprocessing import StandardScaler from sklearn.svm import SVC

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, confusion\_matrix import seaborn as sns

import matplotlib.pyplot as plt

# 2. Load dataset iris = load\_iris()

X = iris.data y = iris.target

# 3. Feature scaling scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# 4. Train-test split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.3, random\_state=42)

#

# Part A: SUPPORT VECTOR MACHINE #

# 5. Initialize and train SVM

svm\_model = SVC(kernel='linear') # You can also try 'rbf', 'poly' svm\_model.fit(X\_train, y\_train)

# 6. Predict and evaluate SVM y\_pred\_svm = svm\_model.predict(X\_test)

print("SVM Accuracy:", accuracy\_score(y\_test, y\_pred\_svm)) print("SVM Confusion Matrix:\n", confusion\_matrix(y\_test, y\_pred\_svm))

#

# Part B: RANDOM FOREST #

# 7. Initialize and train Random Forest

rf\_model = RandomForestClassifier(n\_estimators=100, random\_state=42) rf\_model.fit(X\_train, y\_train)

# 8. Predict and evaluate Random Forest y\_pred\_rf = rf\_model.predict(X\_test)

print("Random Forest Accuracy:", accuracy\_score(y\_test, y\_pred\_rf)) print("Random Forest Confusion Matrix:\n", confusion\_matrix(y\_test, y\_pred\_rf))

#

# 9. Visual comparison using seaborn heatmap #

plt.figure(figsize=(10, 4))

plt.subplot(1, 2, 1)

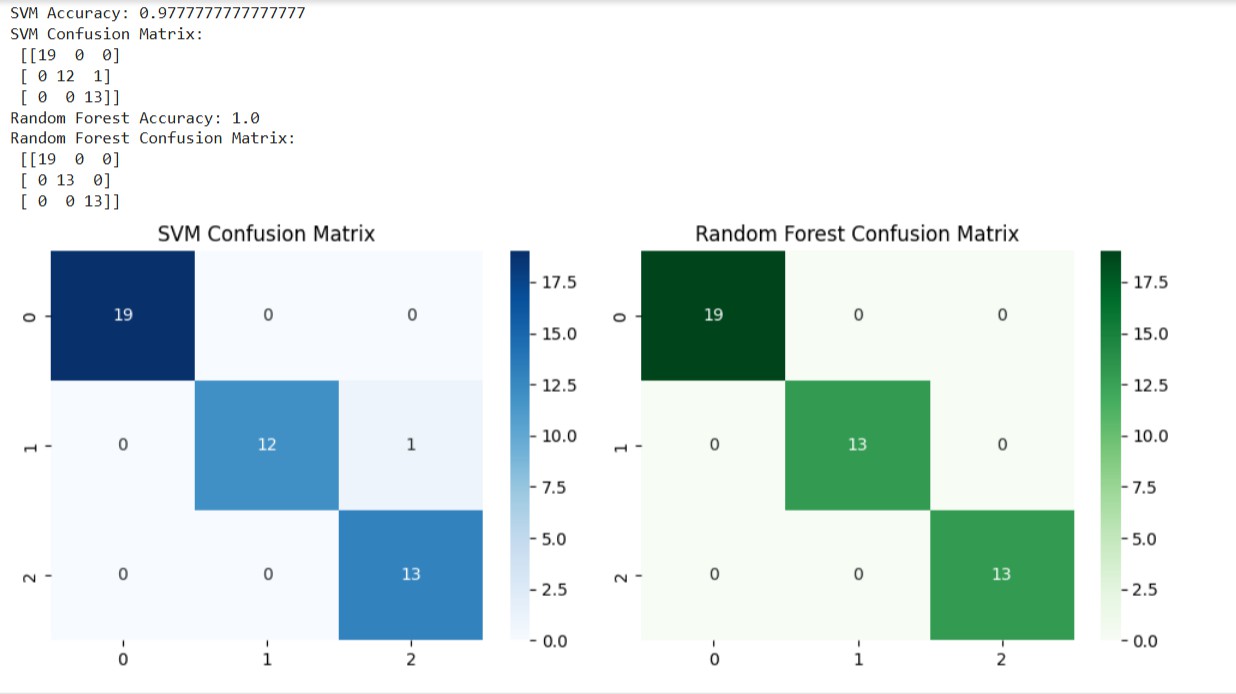
sns.heatmap(confusion\_matrix(y\_test, y\_pred\_svm), annot=True, cmap='Blues', fmt='d') plt.title("SVM Confusion Matrix")

plt.subplot(1, 2, 2)

sns.heatmap(confusion\_matrix(y\_test, y\_pred\_rf), annot=True, cmap='Greens', fmt='d') plt.title("Random Forest Confusion Matrix")

plt.tight\_layout() plt.show()

# OUTPUT:

****

**RESULT:**

The Support Vector Machine (SVM) and Random Forest algorithms were successfully implemented for both binary and multiclass classification tasks. The models were trained and tested on the given dataset, and both achieved good accuracy.