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

1. Initialize Random Forest using RandomForestClassifier()
2. Train and predict
3. Evaluate and compare with SVM

CODE:

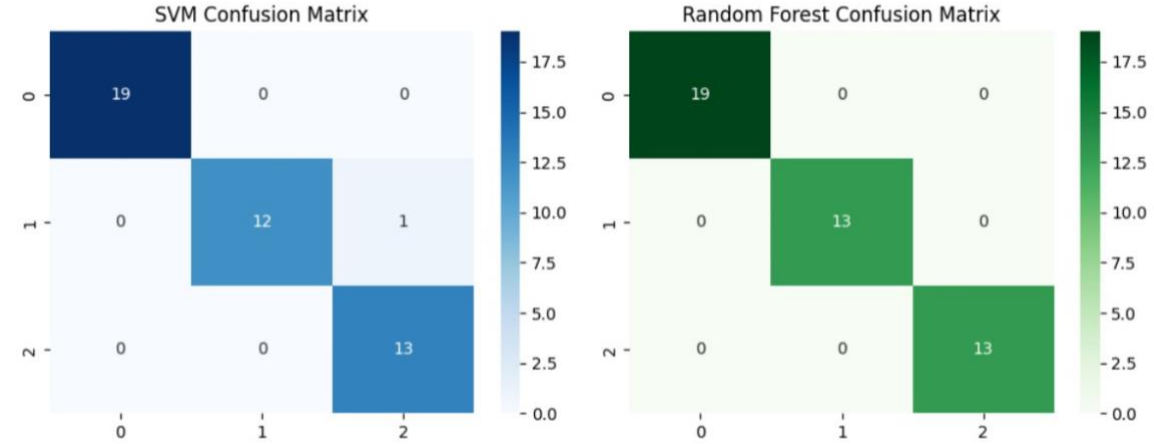
```
# 1. Import libraries

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:

```
SVM Accuracy: 0.9777777777777777
SVM Confusion Matrix:
 [[19  0  0]
 [ 0 12  1]
 [ 0  0 13]]
Random Forest Accuracy: 1.0
Random Forest Confusion Matrix:
 [[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
```



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.