**Ex. No.7** **SVM MODELS**

**Date:**

**Aim:**

To write a Python program to build SVM model.

**Algorithm:**

```
Step 1. Import the necessary libraries (matplotlib.pyplot, numpy, and svm from sklearn).
Step 2. Define the features (X) and labels (y) for the fruit dataset.
Step 3. Create an SVM classifier with a linear kernel using svm.SVC(kernel='linear').
Step 4. Train the classifier on the fruit data using clf.fit(X, y).
Step 5. Plot the fruits and decision boundary using plt.scatter(X[:, 0], X[:, 1], c=colors),
        where colors is a list of colors assigned to each fruit based on its label.
Step 6. Create a meshgrid to evaluate the decision function using
        np.meshgrid(np.linspace(xlim[0], xlim[1], 100), np.linspace(ylim[0], ylim[1], 100)).
Step 7. Use the decision function to create a contour plot of the decision boundary and
        margins using ax.contour(xx, yy, Z, colors='k', levels=[-1, 0, 1], alpha=0.5,
        linestyles=['--', '-', '--']).
Step 8. Show the plot using plt.show().
```

**Program:**

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import svm

# Define the fruit features (size and color)
X = np.array([[5, 2], [4, 3], [1, 7], [2, 6], [5, 5], [7, 1], [6, 2], [5, 3], [3, 6], [2, 7], [6, 3], [3, 3],
[1, 5], [7, 3], [6, 5], [2, 5], [3, 2], [7, 5], [1, 3], [4, 2]])

# Define the fruit labels (0=apples, 1=oranges)
y = np.array([0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0])

# Create an SVM classifier with a linear kernel
clf = svm.SVC(kernel='linear')

# Train the classifier on the fruit data
clf.fit(X, y)

# Plot the fruits and decision boundary
colors = ['red' if label == 0 else 'yellow' for label in y]
plt.scatter(X[:, 0], X[:, 1], c=colors)
ax = plt.gca()
ax.set_xlabel('Size')
ax.set_ylabel('Color')
xlim = ax.get_xlim()
ylim = ax.get_ylim()

# Create a meshgrid to evaluate the decision function
xx, yy = np.meshgrid(np.linspace(xlim[0], xlim[1], 100), np.linspace(ylim[0], ylim[1], 100))
Z = clf.decision_function(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# Plot the decision boundary and margins
ax.contour(xx, yy, Z, colors='k', levels=[-1, 0, 1], alpha=0.5, linestyles=['--', '-', '--'])
plt.show()
```

**Viva Questions:**

1. What is an SVM model?
2. What is the kernel function in SVM?
3. How do you choose the optimal value of C in SVM?
4. What is the decision boundary in SVM?
5. What is the purpose of the mesh grid in the code?

**Result:**

Thus, the Python program to build an SVM model was developed, and the output was successfully verified.