AML Lab Assignment-3:

Name: Gurvinder Kaur Matharu
PRN: 1032230432
Roll No.: PA14

AIM:

Implementation of SVM, Comparison with tree based Classifier

THEORY:

Support Vector Machine (SVM) is a supervised learning algorithm used for classification and regression tasks. It finds the optimal hyperplane in a high-dimensional space that best separates different classes by maximizing the margin between the classes. SVM can handle complex datasets by mapping them into higher dimensions via kernel functions, enabling effective separation even when the data isn't linearly separable in its original space.

Tree-based classifiers, like Random Forest or Gradient Boosting, create decision trees as their base models, making decisions based on a series of hierarchical rules. They're versatile, handle non-linear relationships well, and provide interpretability at each decision step. On the other hand, SVM aims to find the best separating hyperplane in a high-dimensional space, suitable for complex datasets. While SVMs might offer higher accuracy in certain scenarios and handle outliers better, they can be less interpretable compared to tree-based models and might be computationally intensive for larger datasets.

CODE EXECUTION & OUTPUT:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score, roc_auc_score, roc_curve, auc
import seaborn as sns
import matplotlib.pyplot as plt
```

```
data = pd.read_csv('Fish.csv')
```

```
data.head()
```

	Species	Weight	Length1	Length2	Length3	Height	Width
0	Bream	242.0	23.2	25.4	30.0	11.5200	4.0200
1	Bream	290.0	24.0	26.3	31.2	12.4800	4.3056
2	Bream	340.0	23.9	26.5	31.1	12.3778	4.6961
3	Bream	363.0	26.3	29.0	33.5	12.7300	4.4555
4	Bream	430.0	26.5	29.0	34.0	12.4440	5.1340

```
data.shape
```

```
(159, 7)
```

```
data['Species'].unique()
```

```
array(['Bream', 'Roach', 'Whitefish', 'Parkki', 'Perch', 'Pike', 'Smelt'],
      dtype=object)
```

```
X = data.iloc[:,1:]
y = data.iloc[:,0]
```

```
encoder = LabelEncoder()
y = encoder.fit_transform(y)
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)
```

```
model = SVC(kernel = 'linear')
```

```
model.fit(X_train, y_train)
```

```
SVC
SVC(kernel='linear')
```

Prediction

```
y_pred = model.predict(X_test)
```

```
y_pred
```

```
array([1, 3, 3, 2, 3, 2, 5, 5, 0, 3, 5, 0, 4, 3, 2, 2, 0, 0, 3, 0, 2, 3,
       2, 3, 1, 0, 2, 4, 4, 5, 2, 0, 2, 5, 1, 4, 3, 2, 2, 4])
```

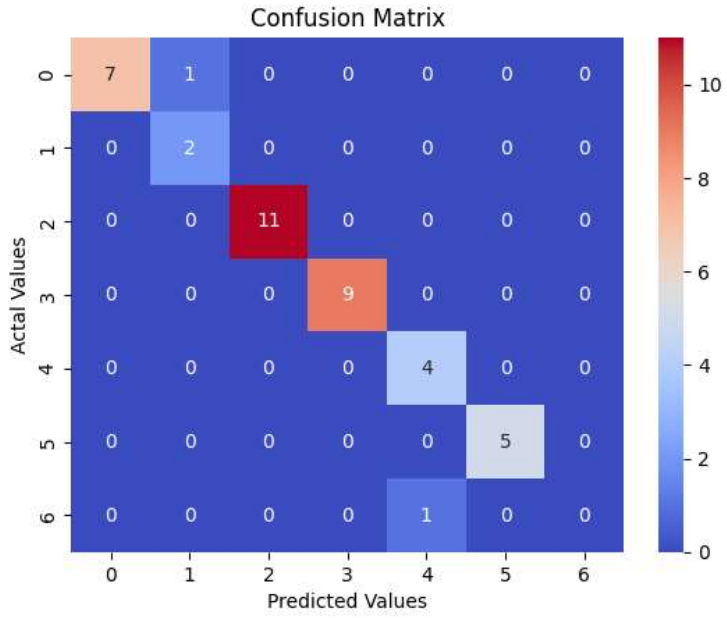
Accuracy

```
accuracy_score(y_test, y_pred)
```

```
0.95
```

```
cm = confusion_matrix(y_test, y_pred)
```

```
sns.heatmap(cm, annot=True, cmap='coolwarm')
plt.title('Confusion Matrix')
plt.ylabel('Actal Values')
plt.xlabel('Predicted Values')
plt.show()
```



Decision Trees

```
model_dt = DecisionTreeClassifier(criterion='entropy')
```

```
model_dt.fit(X_train, y_train)
```

```
y_pred1 = model_dt.predict(X_test)
```

```
accuracy_score(y_test, y_pred1)
```

```
0.8
```

```
cm = confusion_matrix(y_test, y_pred1)
```

```
sns.heatmap(cm, annot=True, cmap='coolwarm')
plt.title('Confusion Matrix')
plt.ylabel('Actal Values')
plt.xlabel('Predicted Values')
plt.show()
```