Exercise 10 Date: 05/03/2025

Support Vector Recognition - Two Dimensional Dataset

Aim: To build a Support Vector Regression (SVR) on a two-dimensional dataset.

Algorithm:

Support Vector Regression (SVR) is a supervised learning algorithm used for regression tasks. Unlike traditional regression models that aim to minimize the error, SVR tries to fit the best hyperplane within a margin of tolerance, controlled by the epsilon (ε) parameter.

The model works by finding a function (f(x)) that has at most ε deviation from the actual target values while ensuring model complexity remains minimal. The optimization problem is formulated as:

$$\min \frac{1}{2}||w||^2 \quad \text{subject to} \quad |y_i - f(x_i)| \leq \epsilon$$

where:

- y_i represents the actual target values,
- $f(x_i)$ is the predicted function,
- ϵ defines the margin of tolerance.

The RBF kernel is commonly used in SVR because it captures non-linear patterns by transforming input features into higher-dimensional space, making it effective for complex relationships.

Step 1: Import Libraries

- Import necessary Python libraries such as NumPy for numerical operations, Matplotlib for visualization, and scikit-learn modules for model training and evaluation.

Step 2: Load the Dataset

- Generate a synthetic dataset by creating 100 random feature values ranging from 0 to 5.
- Compute the target variable y using the sine function with added Gaussian noise.

Step 3: Split the Data

- Divide the dataset into training (80%) and testing (20%) sets using train_test_split().

Step 4: Define the SVR Model

- Initialize an SVR model with the RBF kernel.
- Set C = 100, Gamma = 0.1, and Epsilon = 0.1 as hyperparameters.
- Train the SVR model on the training dataset.

Step 5: Make Predictions

- Use the trained SVR model to predict target values for the test dataset.

Step 6: Evaluate the Model

- Compute Mean Squared Error (MSE) to measure the average squared difference between actual and predicted values.
- Compute R^2 Score to indicate how well the model explains variance in the data.

Step 7: Visualize Results

- Plot the original data points and the SVR regression curve to observe how well the model captures the data pattern.

Import the libraries

In [11]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
```

Load the dataset

In [12]:
```
np.random.seed(42)
X = np.sort(5 * np.random.rand(100, 1), axis=0)
y = np.sin(X).ravel() + np.random.normal(0, 0.1, X.shape[0])
```

Split the Data

In [13]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

RBF Kernel

In [14]:
```
svr_rbf = SVR(kernel='rbf', C=100, gamma=0.1, epsilon=0.1)
svr_rbf.fit(X_train, y_train)
```

Out[14]:
```
SVR(C=100, gamma=0.1)
```

Evaluating the model

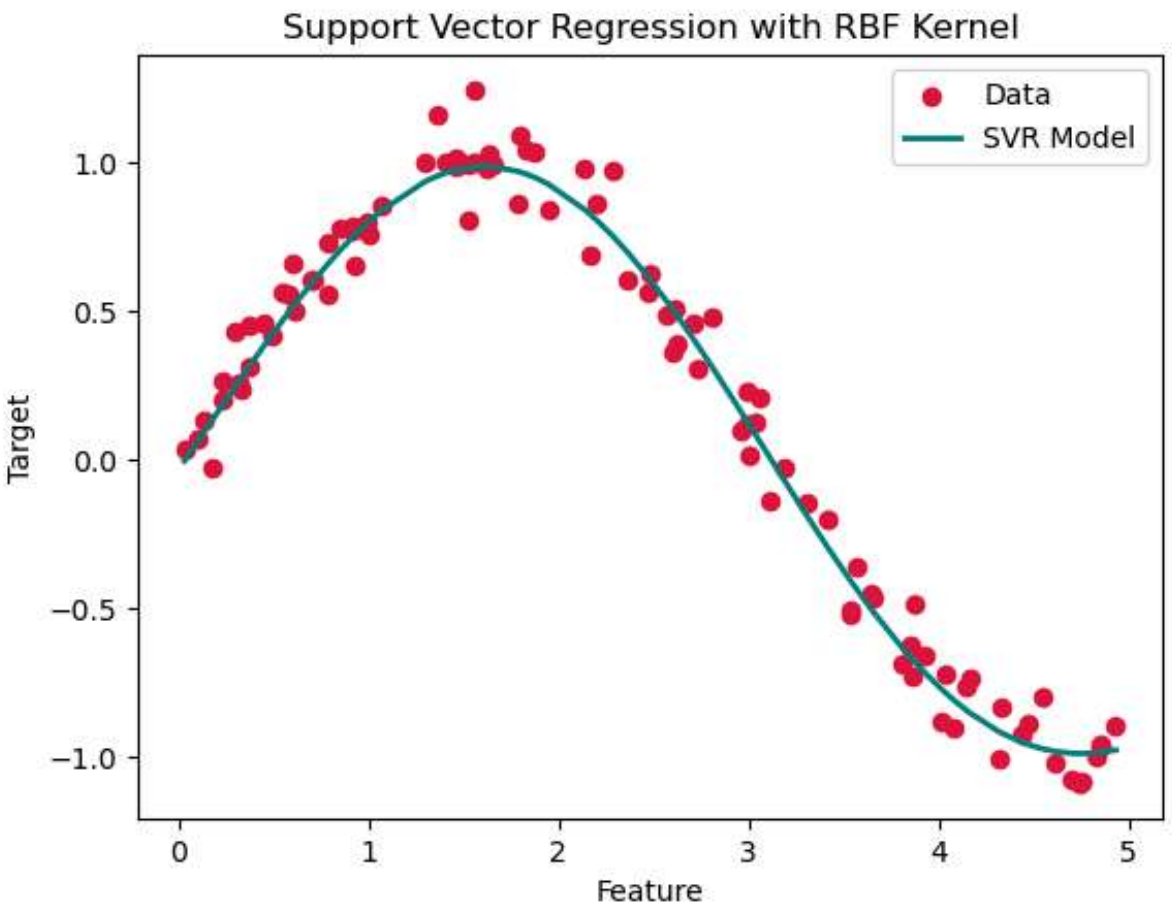
In [15]:
```
y_pred = svr_rbf.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse:.4f}")
print(f"R2 Score: {r2:.4f}")
```

```
Mean Squared Error: 0.0076
R2 Score: 0.9842
```

Visualisation

In [16]:
```
plt.scatter(X, y, color='crimson', label='Data')
plt.plot(X, svr_rbf.predict(X), color='teal', lw=2, label='SVR Model')
plt.xlabel("Feature")
plt.ylabel("Target")
plt.title("Support Vector Regression with RBF Kernel")
plt.legend()
plt.show()
```



Result

A Support Vector Regression (SVR) model was successfully implemented using the Radial Basis Function (RBF) kernel to capture non-linear relationships in the data. The model effectively learns the underlying pattern with a Mean Squared Error (MSE) of 0.0076 and an R^2 Score of 0.9842.
