The Iris Dataset: SVM with Gaussian RBF and Polynomial Kernels

Importing necessary libraries

First, we need to import the libraries required for our analysis: `pandas` and `numpy` for data manipulation, `sklearn` for building the support vector machines (SVMs) and evaluating their performance, and `matplotlib` and `seaborn` for creating confusion matrix heatmaps.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

2. Loading and preparing the dataset

Here, we load the Iris dataset, which contains features of three different species of Iris flowers. The features are stored in `X`, while the target labels are in `y`. We also split the dataset into training and testing sets with a 70-30 ratio.

```
iris = load_iris()
X = iris.data
y = iris.target
target_names = iris.target_names

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

3. Creating and training the models

Now, we define a function to encapsulate the training and evaluation process for the SVM classifier. It takes a kernel type as input, initialises the SVM model, fits it to the training data, and predicts the labels for the test data. It returns the accuracy, classification report, and confusion matrix.

```
def evaluate_svm(kernel):
  svm = SVC(kernel=kernel)
  svm.fit(X_train, y_train)
  y_pred = svm.predict(X_test)

  accuracy = accuracy_score(y_test, y_pred)
  report = classification_report(y_test, y_pred, target_names=target_names)
  cm = confusion_matrix(y_test, y_pred)

  return accuracy, report, cm
```

4. Evaluating the models

We first evaluate the SVM model using the RBF kernel by calling the `evaluate_svm` function with `rbf` as an argument. The accuracy, classification report, and confusion matrix are stored for later use. Next, we evaluate the model with a polynomial kernel by calling the same function with `poly`. The results are also stored for comparison.

```
accuracy_rbf, report_rbf, cm_rbf = evaluate_svm(kernel='rbf')
accuracy_poly, report_poly, cm_poly = evaluate_svm(kernel='poly')
```

5. Generating the confusion matrix heatmaps

In this section, a helper function is defined to visualise the confusion matrices for each kernel using Seaborn's `heatmap` function. This function enhances interpretability of model performance.

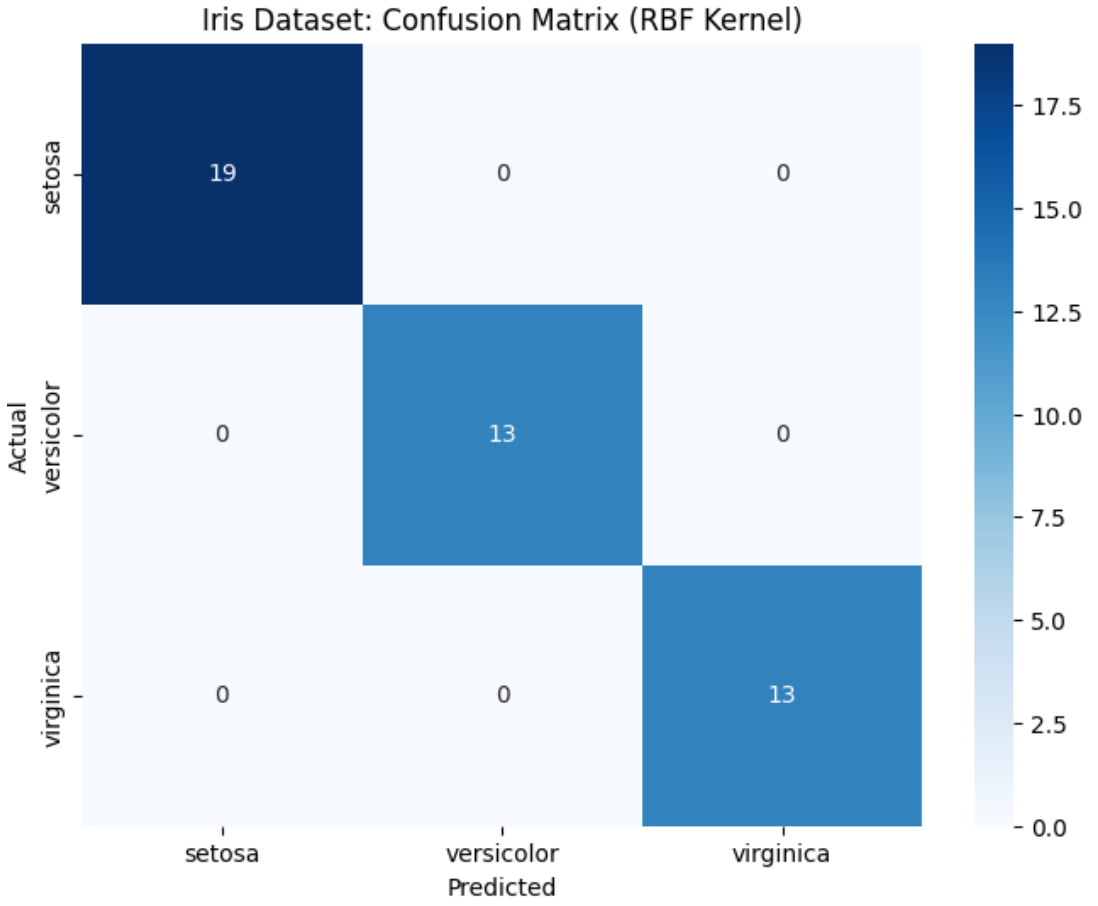
```
def plot_confusion_matrix(cm, title):
  plt.figure(figsize=(8,6))
  sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=target_names, yticklabels=target_names)
  plt.title(title)
  plt.xlabel('Predicted')
  plt.ylabel('Actual')
  plt.show()
```

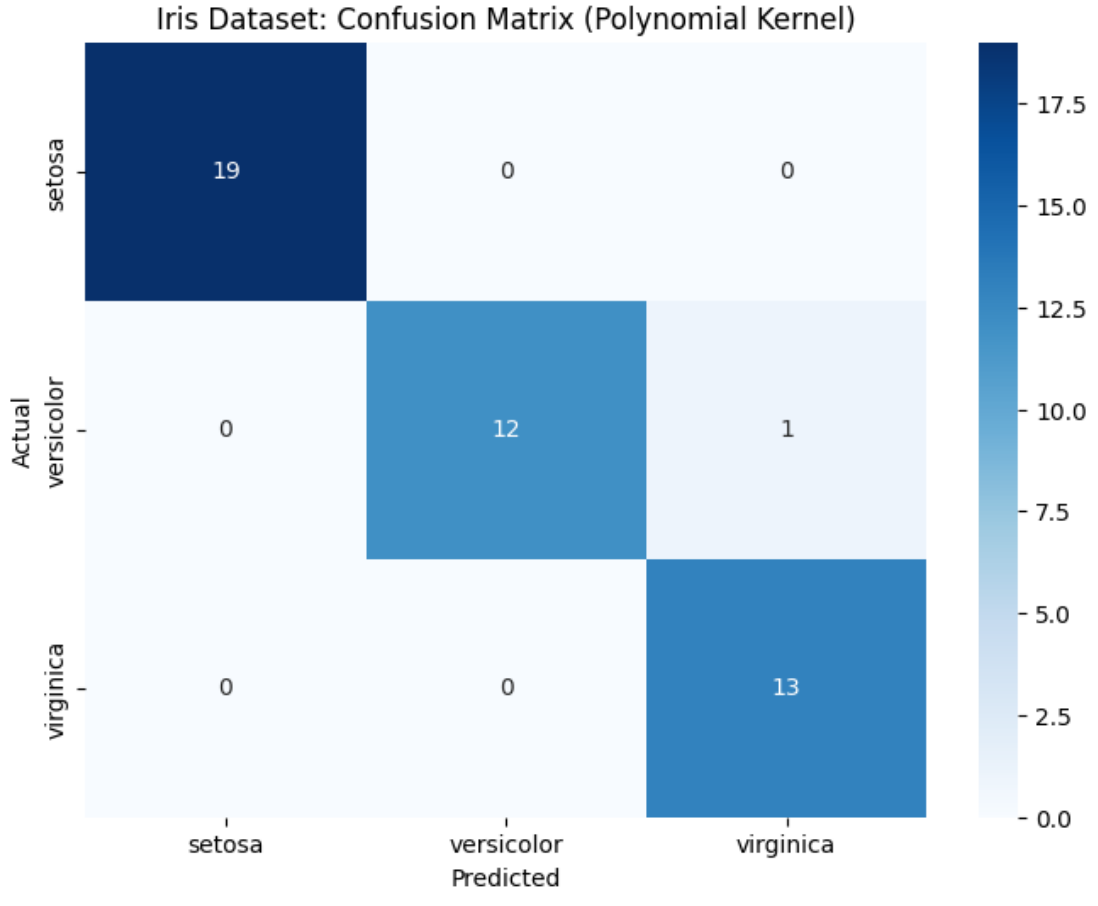
6. Visualising the results

Finally, the confusion matrices for both the RBF and polynomial kernels are plotted, providing visual insights into the classification results. The accuracy scores and classification reports for both kernels are also printed. This allows for a straightforward comparison of how each kernel performed.

```
plot_confusion_matrix(cm_rbf, 'Iris Dataset: Confusion Matrix (RBF Kernel)')
plot_confusion_matrix(cm_poly, 'Iris Dataset: Confusion Matrix (Polynomial Kernel)')

print(f'RBF kernel accuracy: {accuracy_rbf:.2f}')
print('Classification report:\n', report_rbf)
print(f'Polynomial kernel accuracy: {accuracy_poly:.2f}')
print('Classification report:\n', report_poly)
```


Iris Dataset: Confusion Matrix (RBF Kernel)
setosa
versicolor
virginica
Actual
19
0
0
0
13
0
0
0
13
setosa
versicolor
virginica
Predicted
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0


Iris Dataset: Confusion Matrix (Polynomial Kernel)
setosa
versicolor
virginica
Actual
19
0
0
0
12
1
0
0
13
setosa
versicolor
virginica
Predicted
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0

```
RBF kernel accuracy: 1.00
Classification report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        19
  versicolor       1.00      1.00      1.00        13
   virginica       1.00      1.00      1.00        13

    accuracy                           1.00        45
   macro avg       1.00      1.00      1.00        45
weighted avg       1.00      1.00      1.00        45

Polynomial kernel accuracy: 0.98
Classification report:
              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        19
  versicolor       1.00      0.92      0.96        13
   virginica       0.93      1.00      0.96        13

    accuracy                           0.98        45
   macro avg       0.98      0.97      0.97        45
weighted avg       0.98      0.98      0.98        45
```