Practical No: 3

Multiclass classification (Problem based Learning)

AIM: Support vector machine (SVM) algorithm for multiclass classification using Iris.csv and wine dataset from sklearn.

Description:

Calculate the TP, TN, FP, FN values for the class Setosa using the confusion matrix / contingency table and also calculate precision and recall for data file 'wine' from sklearn dataset:

TP (True Positives): The number of correctly predicted positive instances in a binary classification problem.

TN (True Negatives): The number of correctly predicted negative instances in a binary classification problem.

FP (False Positives): The number of instances that were predicted as positive but are actually negative in a binary classification problem.

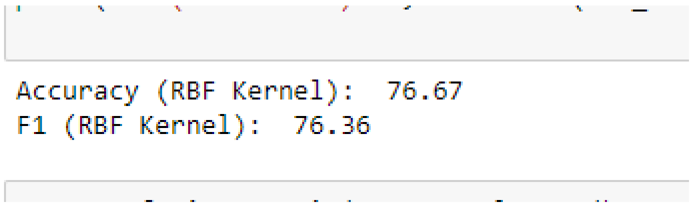
FN (False Negatives): The number of instances that were predicted as negative but are actually positive in a binary classification problem.

Support Vector Machine (SVM): A supervised machine learning algorithm that finds a hyperplane to maximize the margin between different classes in a dataset, making it effective for classification and regression tasks.

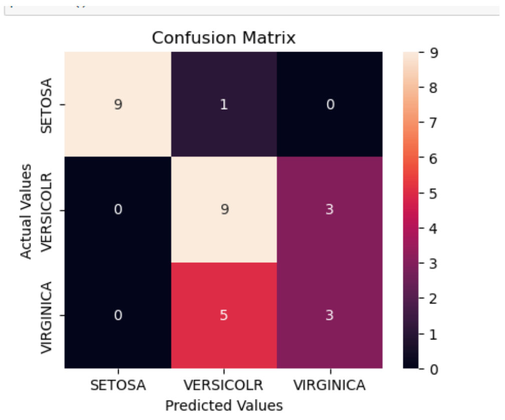
Code with output

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import svm, datasets
import sklearn.model_selection as model_selection
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
iris = datasets.load_iris()
#print(iris.data)
X = iris.data[:, :2]
```

```
y = iris.target
X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, train_size=0.80,
test_size=0.20, random_state=101)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
rbf = svm.SVC(kernel='rbf', gamma=0.5, C=0.1).fit(X_train, y_train)
poly = svm.SVC(kernel='poly', degree=3, C=1).fit(X_train, y_train)
poly_pred = poly.predict(X_test)
rbf_pred = rbf.predict(X_test)
poly_accuracy = accuracy_score(y_test, poly_pred)
poly_f1 = f1_score(y_test, poly_pred, average='weighted')
print('Accuracy (Polynomial Kernel): ', "%.2f" % (poly_accuracy*100))
print('F1 (Polynomial Kernel): ', "%.2f" % (poly_f1*100))
rbf_accuracy = accuracy_score(y_test, rbf_pred)
rbf_f1 = f1_score(y_test, rbf_pred, average='weighted')
print('Accuracy (RBF Kernel): ', "%.2f" % (rbf_accuracy*100))
print('F1 (RBF Kernel): ', "%.2f" % (rbf_f1*100))
cm = confusion_matrix(y_test,poly_pred)
cm_df = pd.DataFrame(cm,
                index = ['SETOSA','VERSICOLR','VIRGINICA'],
                columns = ['SETOSA','VERSICOLR','VIRGINICA'])
plt.figure(figsize=(5,4))
#print(cm_df)
sns.heatmap(cm_df, annot=True)
plt.title('Confusion Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
```



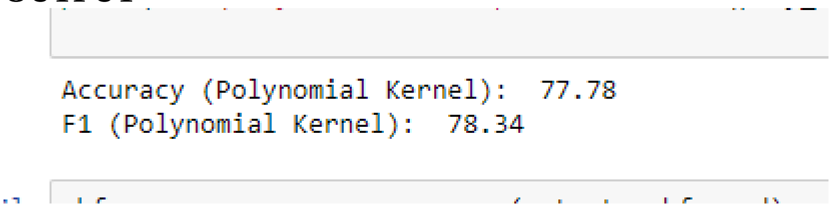
```
Accuracy (RBF Kernel):  76.67
F1 (RBF Kernel):  76.36
```



Code and output for wine dataset from sklearn:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import svm, datasets
import sklearn.model_selection as model_selection
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
wine = datasets.load_wine()
X = wine.data[:, :2]
y = wine.target
X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, train_size=0.80,
test_size=0.20, random_state=101)
rbf = svm.SVC(kernel='rbf', gamma=0.5, C=0.1).fit(X_train, y_train)
poly = svm.SVC(kernel='poly', degree=3, C=1).fit(X_train, y_train)
poly_pred = poly.predict(X_test)
rbf_pred = rbf.predict(X_test)
poly_accuracy = accuracy_score(y_test, poly_pred)
poly_f1 = f1_score(y_test, poly_pred, average='weighted')
print('Accuracy (Polynomial Kernel): ', "%.2f" % (poly_accuracy*100))
print('F1 (Polynomial Kernel): ', "%.2f" % (poly_f1*100))
rbf_accuracy = accuracy_score(y_test, rbf_pred)
rbf_f1 = f1_score(y_test, rbf_pred, average='weighted')
print('Accuracy (RBF Kernel): ', "%.2f" % (rbf_accuracy*100))
print('F1 (RBF Kernel): ', "%.2f" % (rbf_f1*100))
cm = confusion_matrix(y_test,poly_pred)
cm_df = pd.DataFrame(cm)
plt.figure(figsize=(5,4))
#print(cm_df)
sns.heatmap(cm_df, annot=True)
plt.title('Confusion Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
```

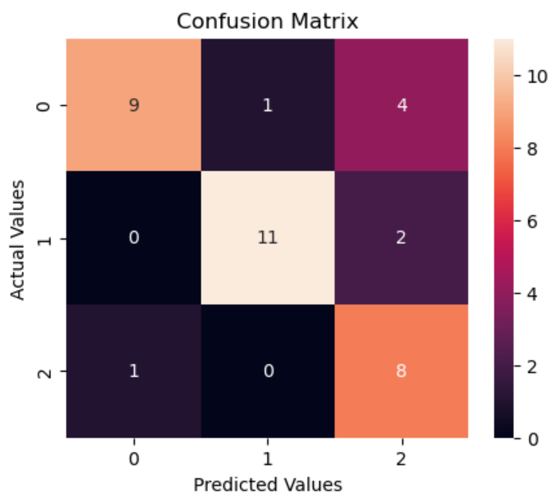
OUTPUT



```
Accuracy (Polynomial Kernel):  77.78
F1 (Polynomial Kernel):  78.34
```

```
Accuracy (RBF Kernel):  77.78
F1 (RBF Kernel):  79.18
```

```
plt.show()
```



Learnings

The provided code performs machine learning classification using two SVM kernels (Polynomial and RBF) on the wine dataset.

The code also generates and visualizes a confusion matrix, which is a valuable tool for evaluating the classification model's performance. The confusion matrix displays the number of true positive (correctly predicted positive class), true negative (correctly predicted negative class), false positive (predicted positive but actual negative), and false negative (predicted negative but actual positive) instances. It provides a clear summary of the model's classification accuracy and potential errors, helping to assess its strengths and weaknesses in differentiating between the wine categories.