Name: Tanushri Mohan Chougale

SRN: PES2UG24CS826

Section: F

Moons Dataset Question

1. Inferences about the Linear Kernel's Performance:

•The Linear kernel performs well when the data is linearly separable or nearly so.

•It produces a straight decision boundary, offering good interpretability and low computational cost.

•However, its performance may decline on complex, non-linear datasets since it cannot capture curved or intricate class boundaries.

•In such cases, it tends to underfit, resulting in lower accuracy compared to non-linear kernels.

2. Comparison between RBF and Polynomial Kernel Decision Boundaries:

•The RBF (Radial Basis Function) kernel creates smooth, flexible, and non-linear boundaries, adapting well to complex data patterns.

•The Polynomial kernel also models non-linear relationships but often produces more rigid and oscillating boundaries, especially with higher degrees.

•RBF generally performs better on most datasets due to its ability to localize decision regions and handle overlapping classes effectively.

•In contrast, the Polynomial kernel can overfit if the degree is too high or the data is noisy.

Banknote Dataset Questions

1. Which kernel was most effective for this dataset?

•The RBF (Radial Basis Function) kernel was the most effective for this dataset.

•It achieved the highest accuracy and balanced classification metrics, effectively handling the nonlinear patterns in the data.

•Its ability to map inputs into a higher-dimensional space allowed it to separate complex class boundaries that the Linear and Polynomial kernels could not.

2. Why might the Polynomial kernel have underperformed here?

•The Polynomial kernel may have underperformed because it can create overly complex and oscillating decision boundaries, especially when the data is noisy or not strongly polynomial in nature.

•It tends to overfit small variations in the training data, reducing its generalization ability on unseen test samples.

•Additionally, its hyperparameters (degree, coef0, etc.) require fine-tuning; without proper tuning, it may fail to capture the true data structure effectively.

Hard vs. Soft Margin Questions 1.

Which margin (soft or hard) is wider?

•The soft margin is wider.

•Because it allows some misclassifications (slack), the decision boundary is more flexible and not tightly fitted to all data points.

2. Why does the soft margin model allow "mistakes"?

•The soft margin SVM allows some points to be misclassified to achieve a better generalization on unseen data.

•Allowing “mistakes” (via slack variables) helps the model handle noisy or overlapping data, preventing it from forcing a perfectly rigid separation.

3. Which model is more likely to be overfitting and why?

•The hard margin model is more likely to overfit.

•It tries to perfectly separate all training points without allowing any errors, which makes it highly sensitive to noise or outliers in the data.

4. Which model would you trust more for new data and why?

•The soft margin model is more trustworthy for new (unseen) data.

•Its flexibility helps it generalize better, reducing overfitting and improving performance on realworld, imperfect datasets.

Moons Dataset

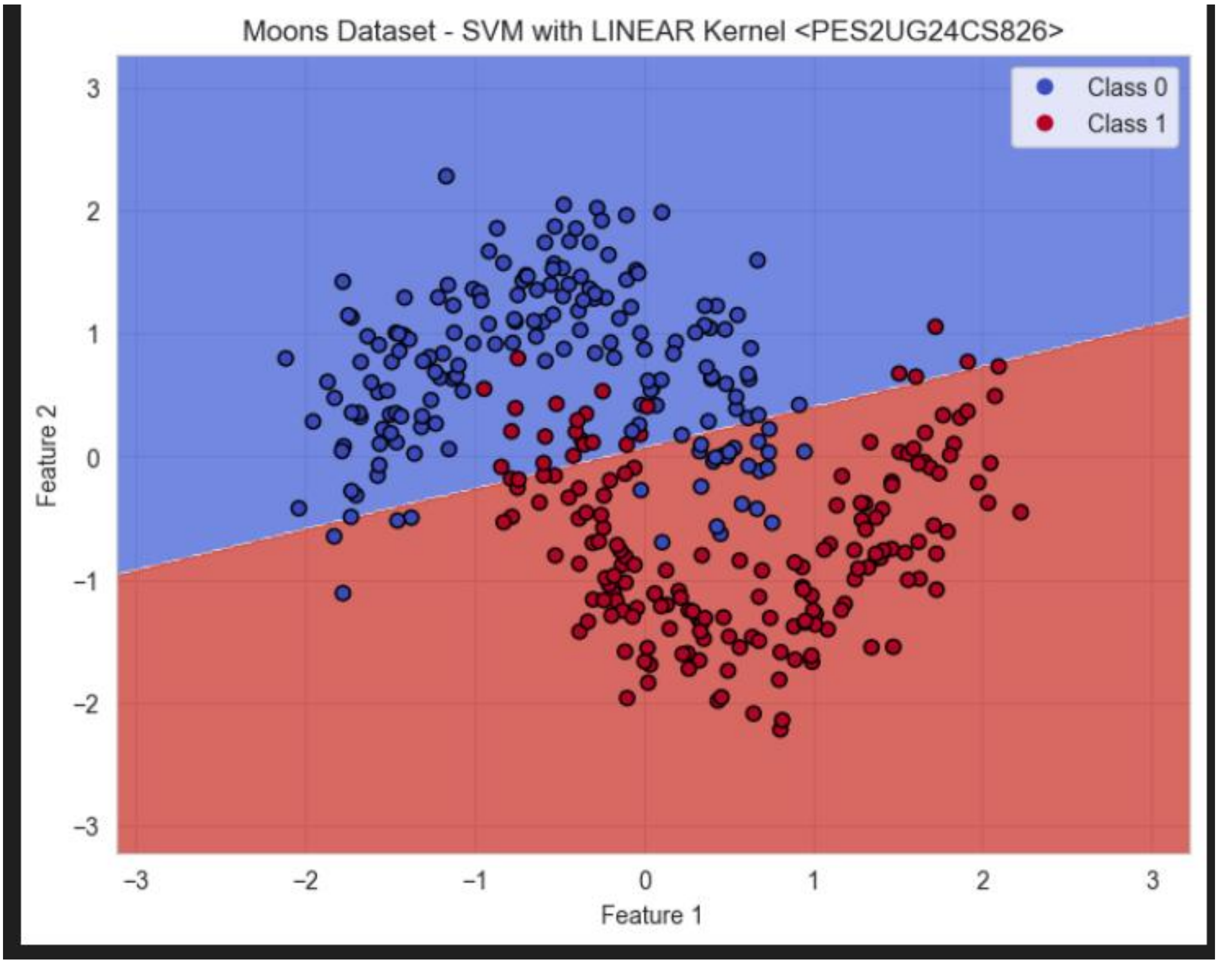
```
SVM with LINEAR Kernel <PES2UG24CS826>
              precision    recall  f1-score   support

           0       0.85      0.89      0.87        75
           1       0.89      0.84      0.86        75

    accuracy                           0.87       150
   macro avg       0.87      0.87      0.87       150
weighted avg       0.87      0.87      0.87       150

----------------------------------------

SVM with RBF Kernel <PES2UG24CS826>
              precision    recall  f1-score   support

           0       0.96      1.00      0.98        75
           1       1.00      0.96      0.98        75

    accuracy                           0.98       150
   macro avg       0.98      0.98      0.98       150
weighted avg       0.98      0.98      0.98       150

----------------------------------------

SVM with POLY Kernel <PES2UG24CS826>
              precision    recall  f1-score   support

           0       0.93      0.88      0.90        75
           1       0.89      0.93      0.91        75

    accuracy                           0.91       150
   macro avg       0.91      0.91      0.91       150
weighted avg       0.91      0.91      0.91       150

----------------------------------------
```

Banknote Dataset

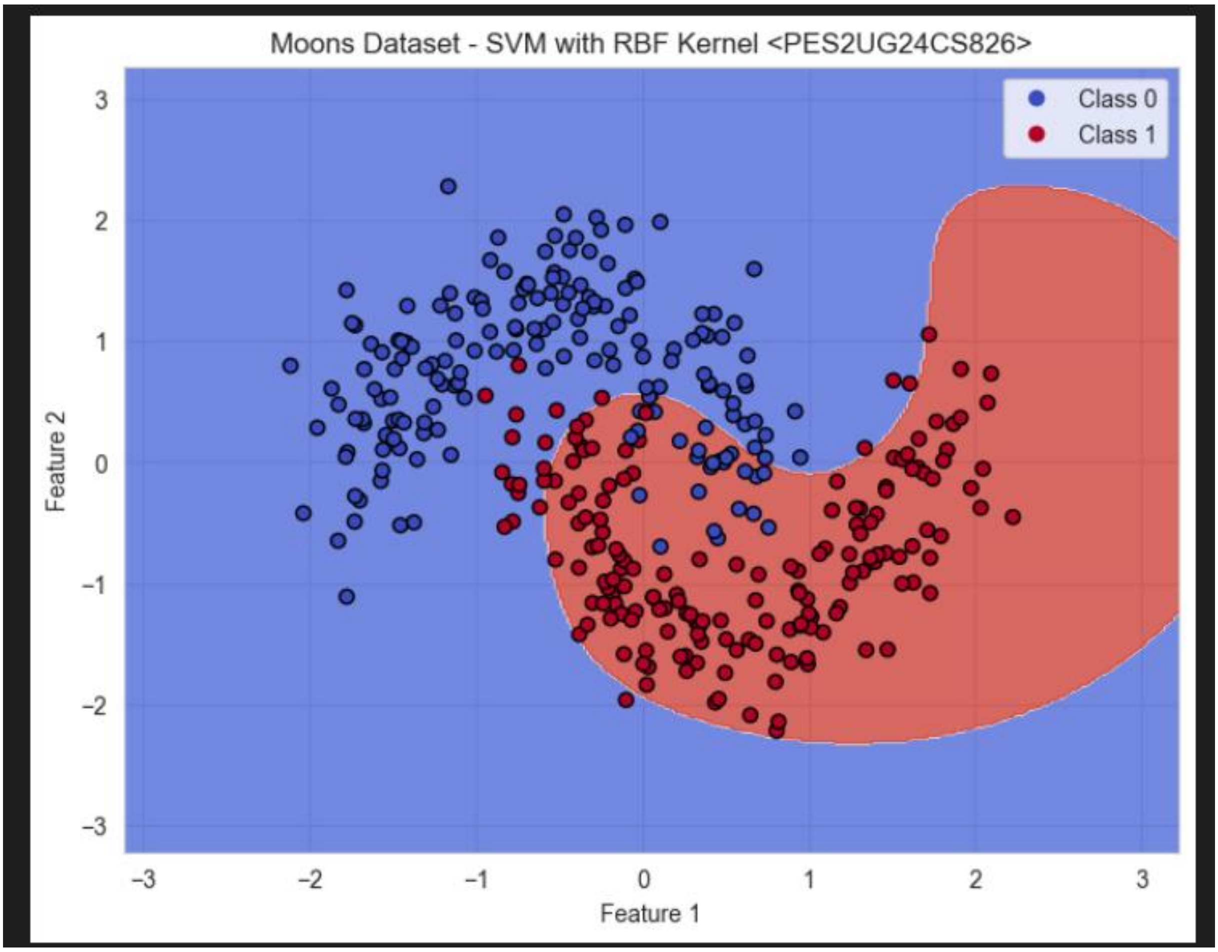
```
SVM with LINEAR Kernel <PES2UG24CS826>
              precision    recall  f1-score   support

      Forged       0.90      0.88      0.89       229
     Genuine       0.86      0.88      0.87       183

    accuracy                           0.88       412
   macro avg       0.88      0.88      0.88       412
weighted avg       0.88      0.88      0.88       412

----------------------------------------

SVM with RBF Kernel <PES2UG24CS826>
              precision    recall  f1-score   support

      Forged       0.96      0.91      0.94       229
     Genuine       0.90      0.96      0.93       183

    accuracy                           0.93       412
   macro avg       0.93      0.93      0.93       412
weighted avg       0.93      0.93      0.93       412

----------------------------------------
```

```
SVM with POLY Kernel <PES2UG24CS826>
              precision    recall  f1-score   support

      Forged       0.96      0.81      0.88       229
     Genuine       0.80      0.96      0.88       183

    accuracy                           0.88       412
   macro avg       0.88      0.89      0.88       412
weighted avg       0.89      0.88      0.88       412

----------------------------------------
```

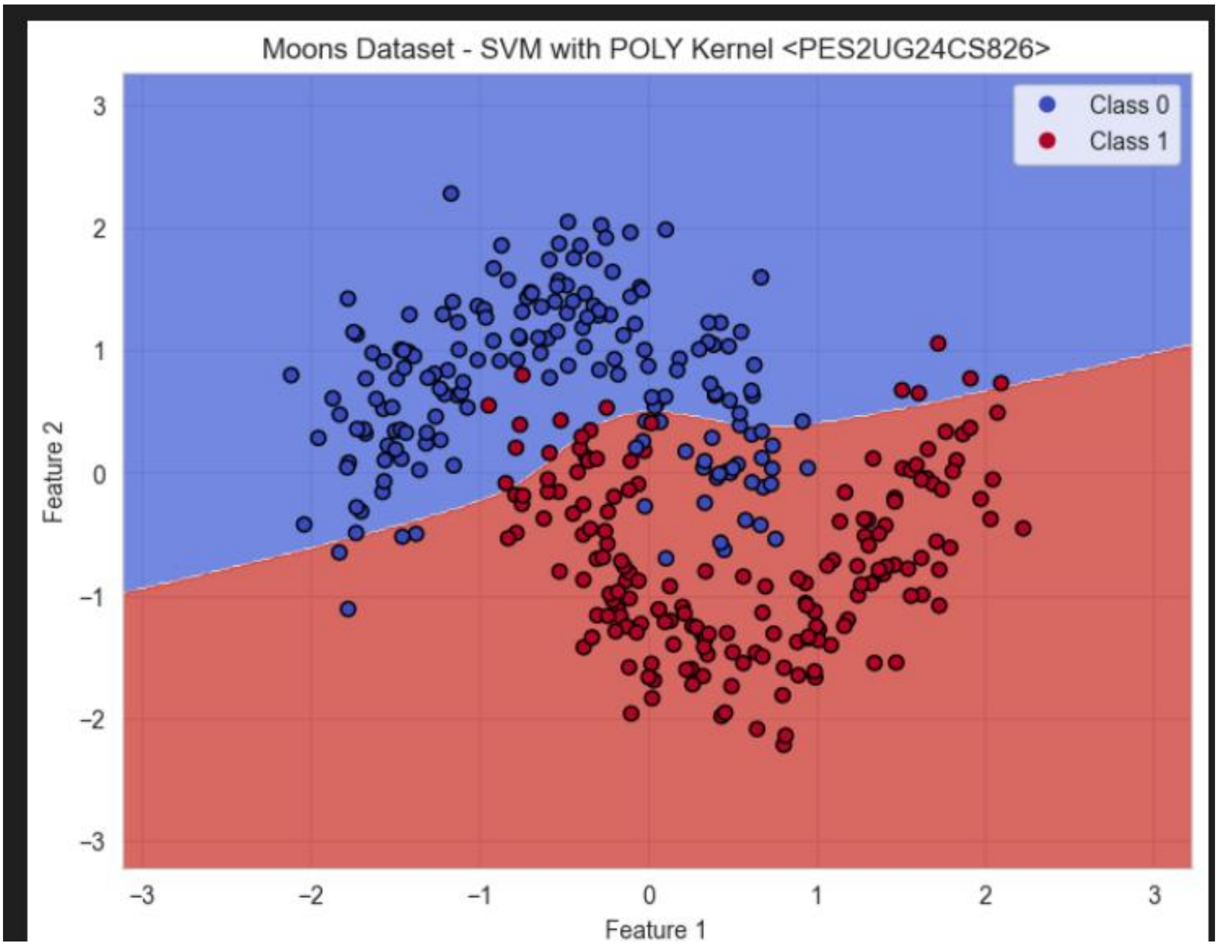
7. Moons Dataset - SVM with LINEAR Kernel



8. Moons Dataset - SVM with RBF Kernel

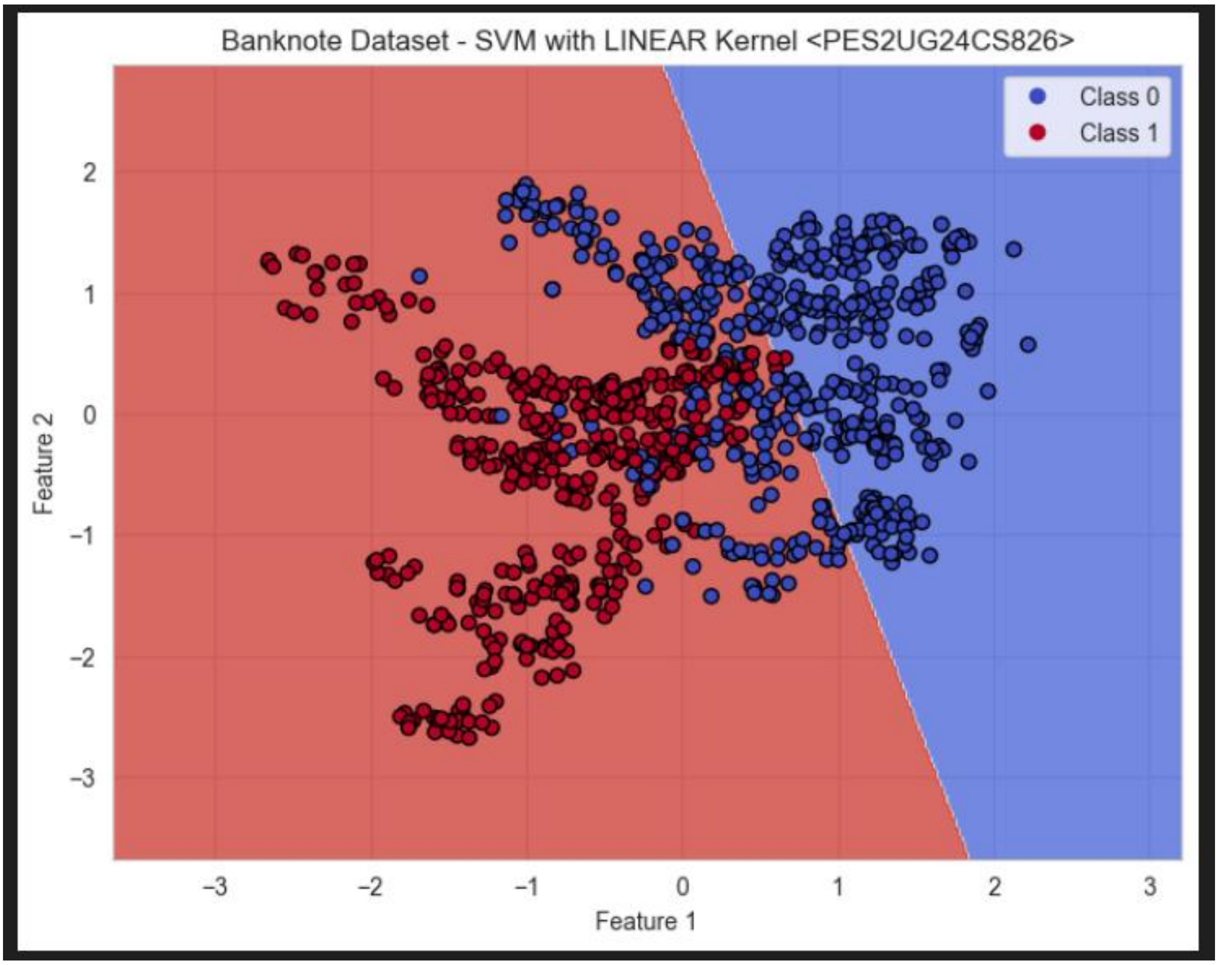


9. Moons Dataset - SVM with POLY Kernel

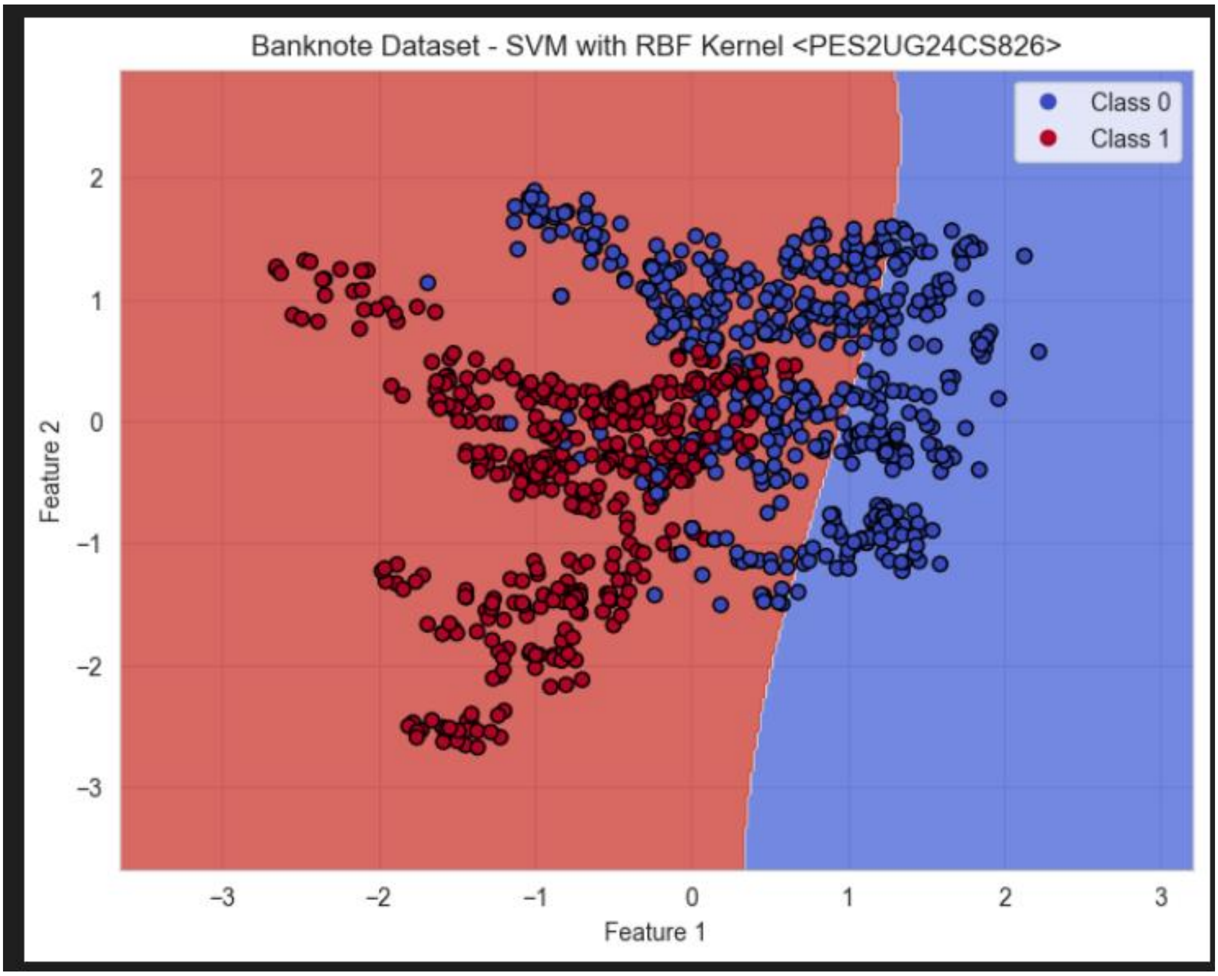


Banknote Dataset:=

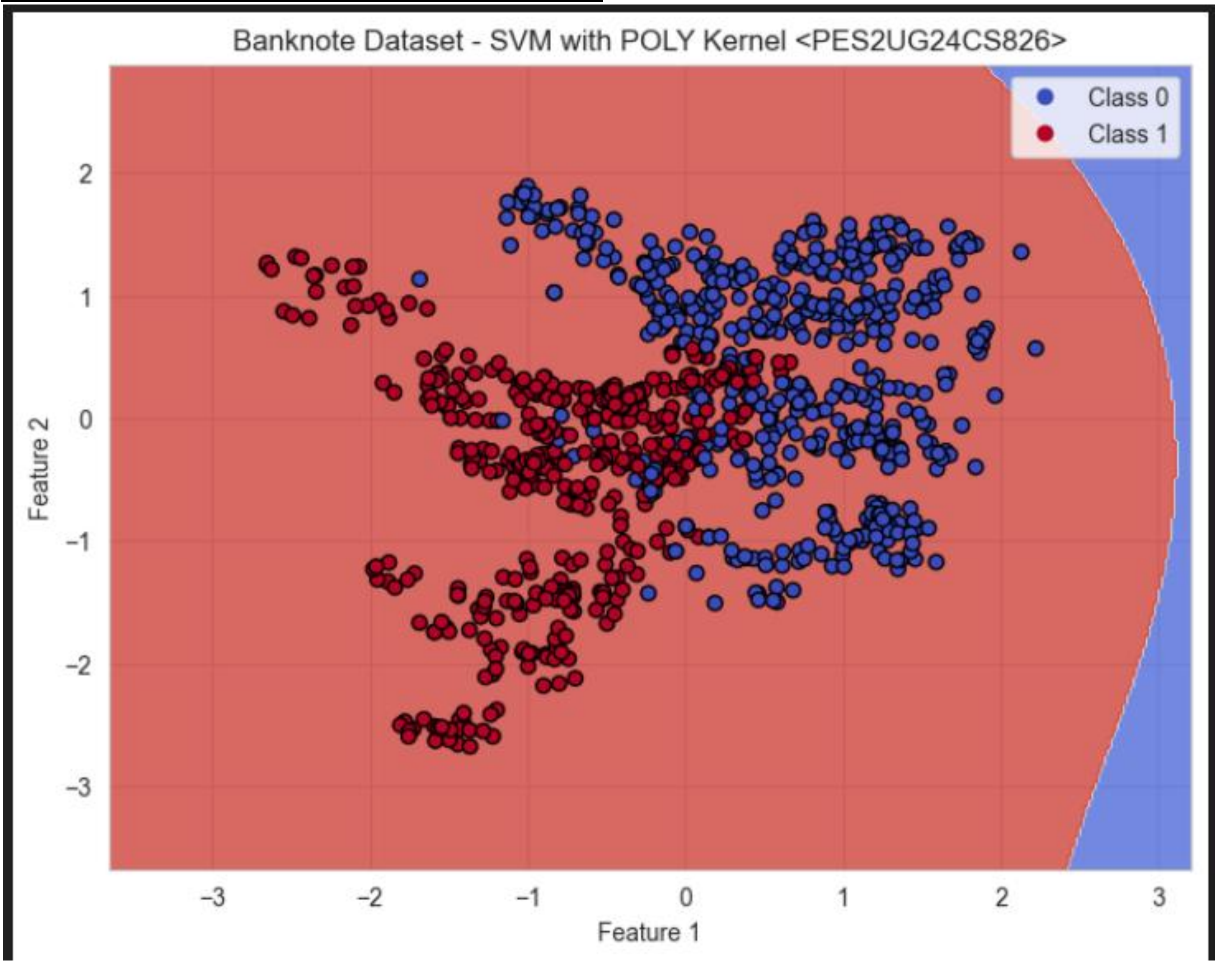
10. Banknote Dataset - SVM with LINEAR Kernel



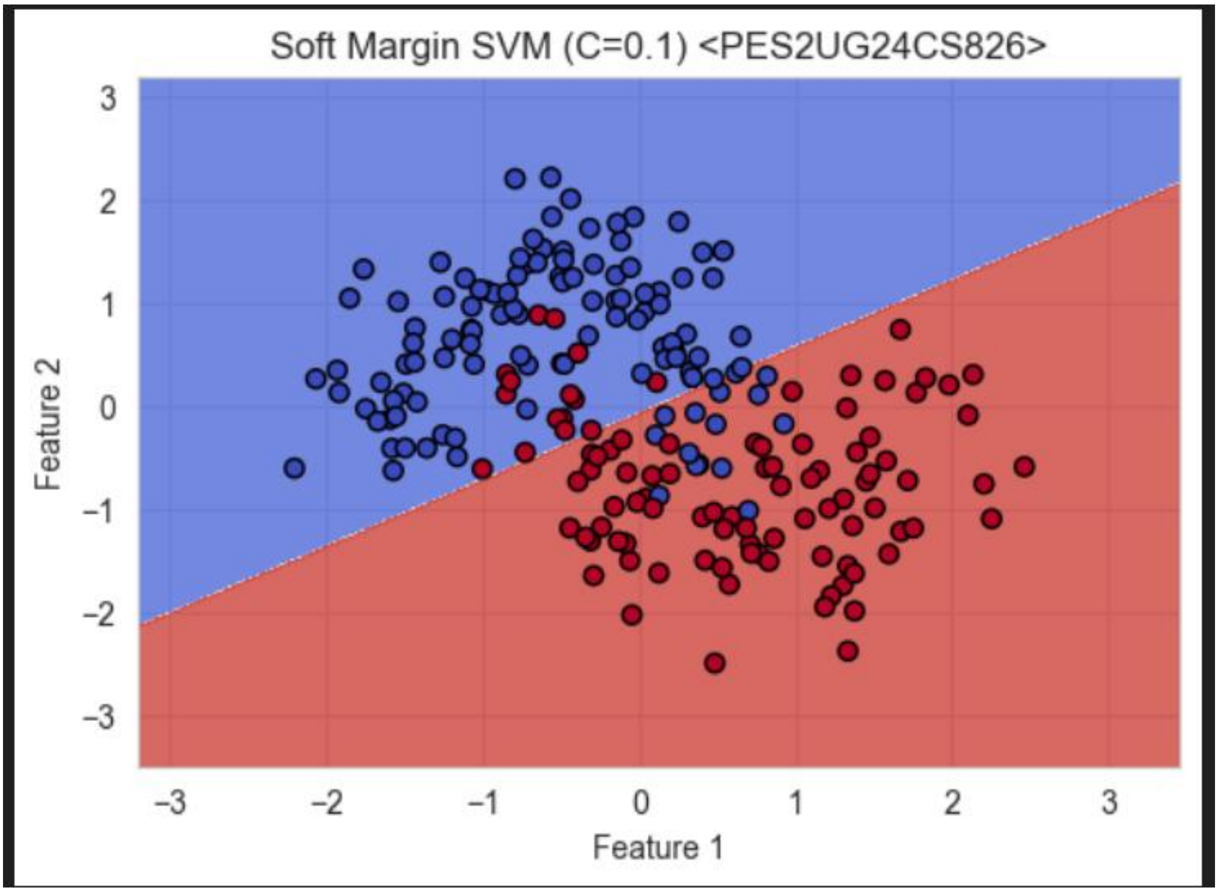
11. Banknote Dataset - SVM with RBF Kernel



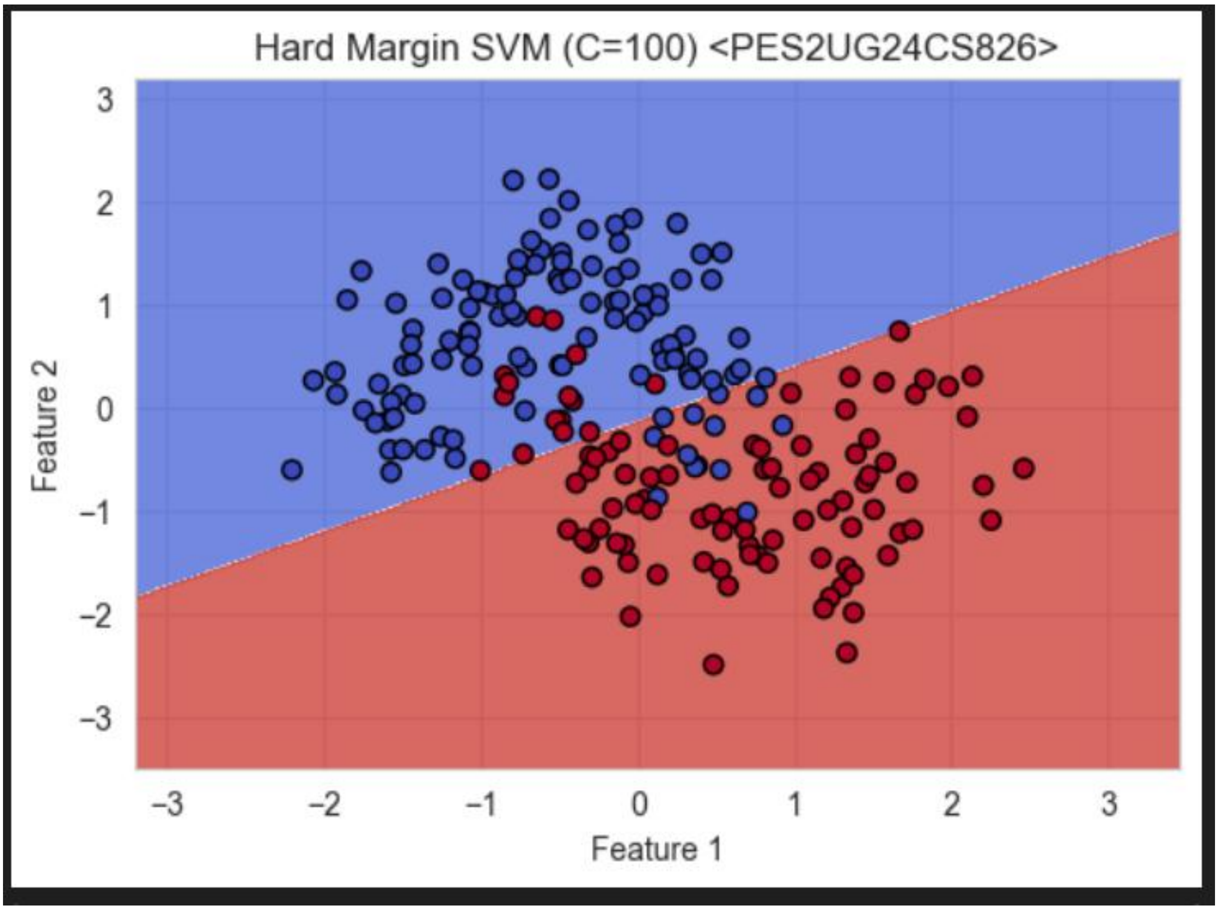
12. Banknote Dataset - SVM with POLY Kernel



13.Soft Margin SVM (C=0.1)



Hard Margin SVM (C=100)



: