# Report——Chapter 18

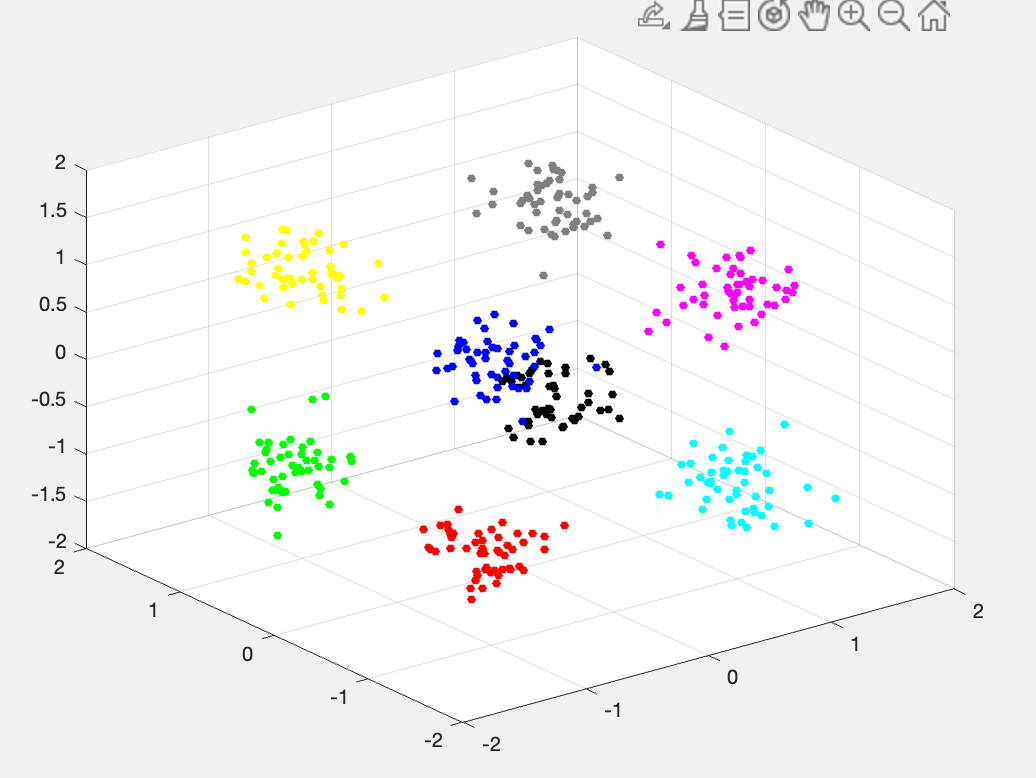
Perceptron, a basic neural network model, can be used to learn the linear decision boundary of binary classification tasks. The code will first change the weight using the Perceptron learning technique to reduce the inaccuracy in the training data. When dealing with nonlinear separable data, we use the kernel Perceptron technique. The kernel Perceptron can make previously indivisible data linearly separable by translating it to a higher dimensional space using the kernel function. The two kernel functions now in use are RBF kernel and linear kernel. RBF kernel functions, unlike linear kernel functions, can map data to an infinite dimensional space.

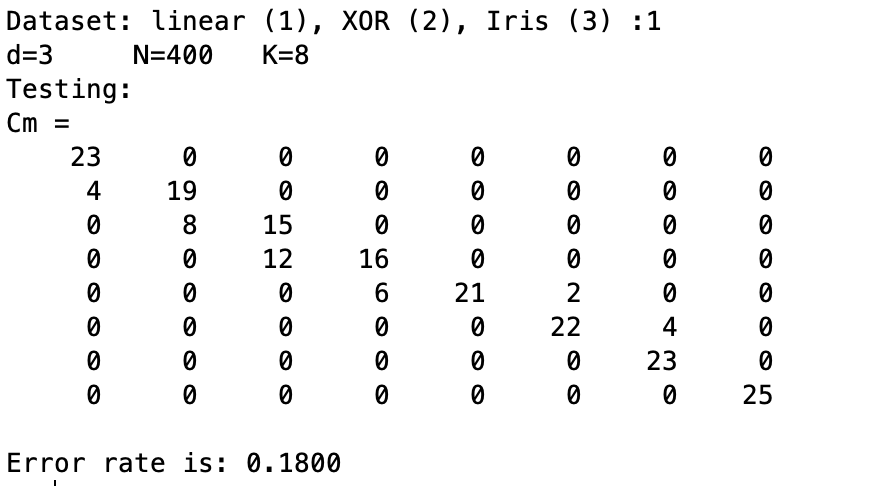
We employ two synthetic datasets (1 linearly separable data created by Gaussian distribution and 1 dataset with 2D points belonging to two classes in an XOR pattern) and one external dataset “iris.txt” in our code, and the program permits Perceptron learning on diverse data sets. The program separates the data set into training and test sets, then trains the Perceptron model on the test set, and lastly utilizes the Confusion matrix on the test set to evaluate the model's prediction accuracy and determine the error rate. In summary, this program demonstrates how the Compute kernel may be used to extend the Perceptron model to handle data sets that were previously indivisible.

**Linear Kernal**:

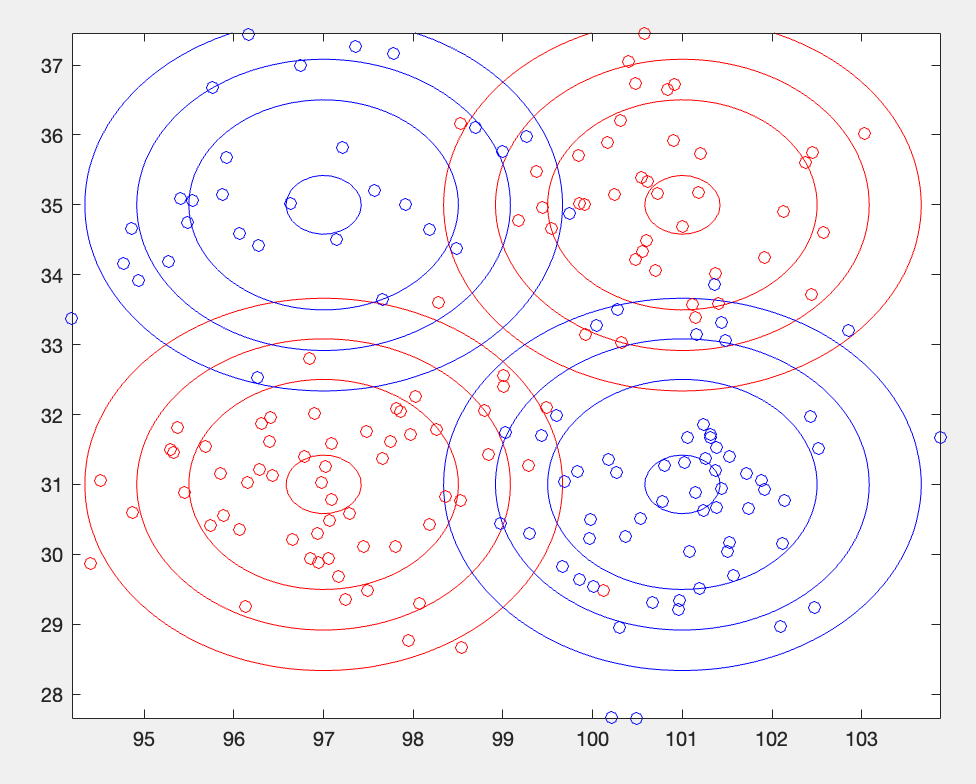
Output:

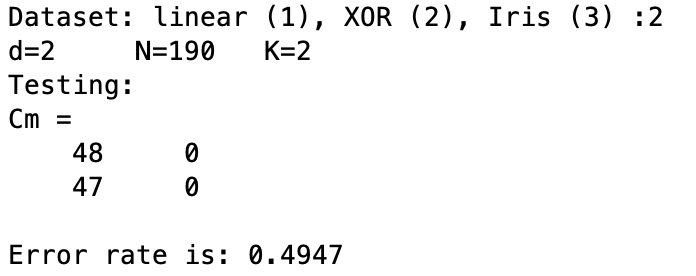
* Linear:



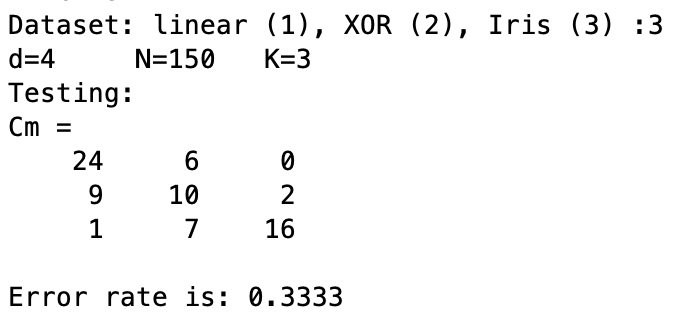


* XOR:





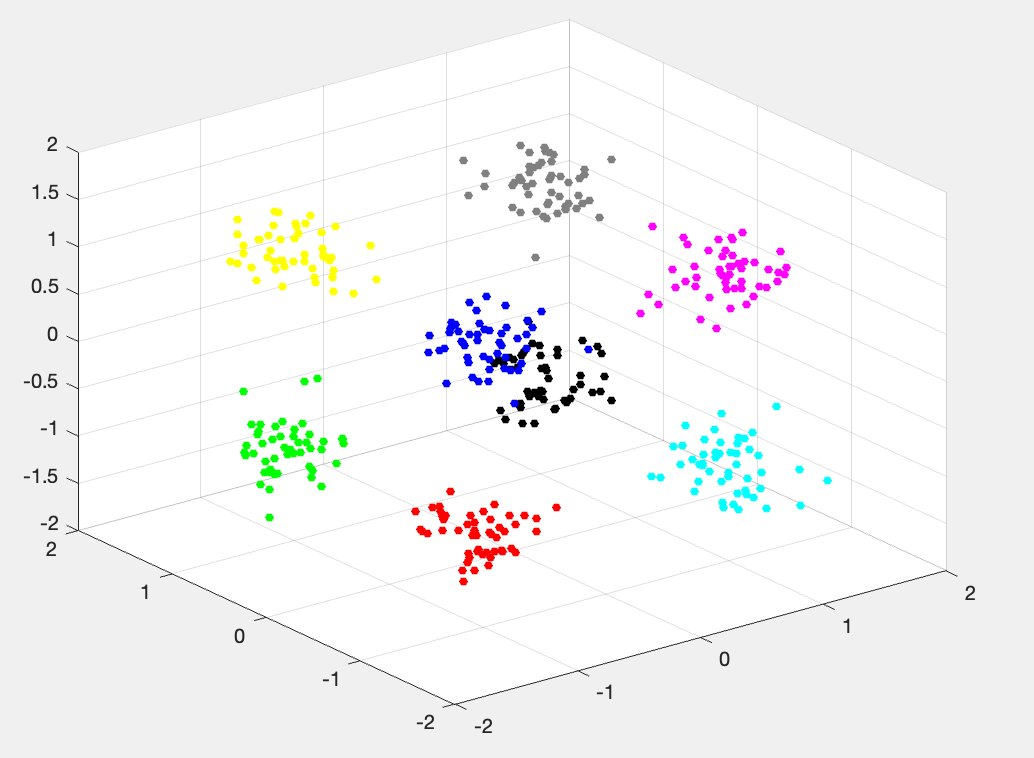
* Iris

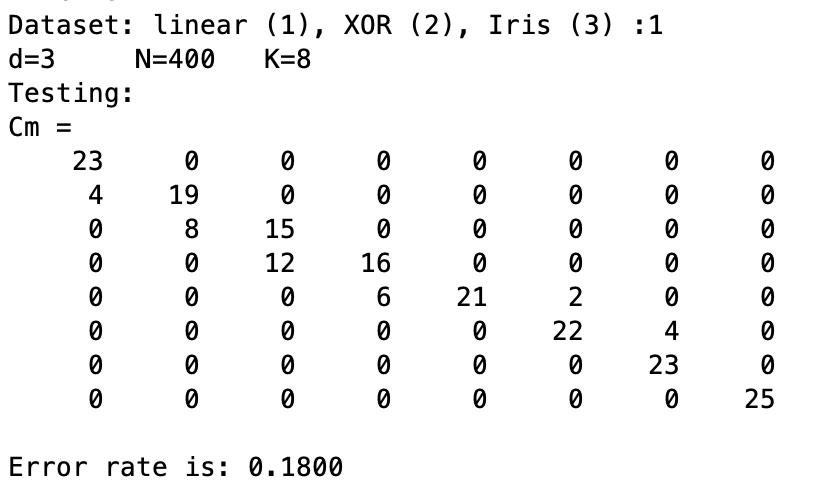


**RBF Kernal**

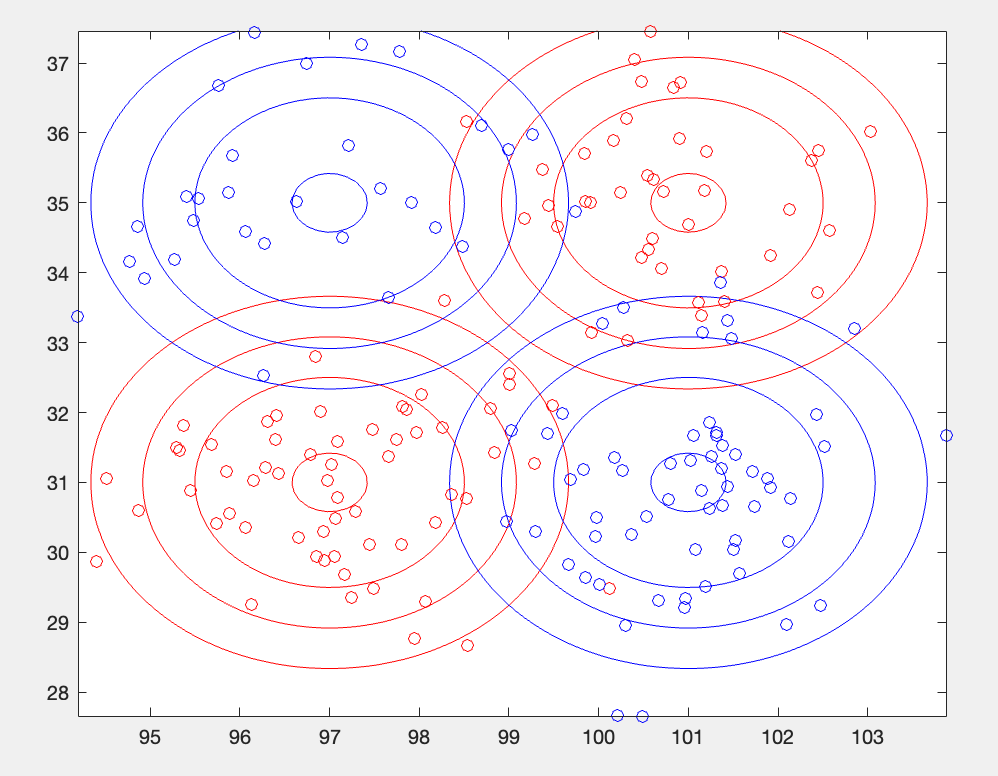
Output:

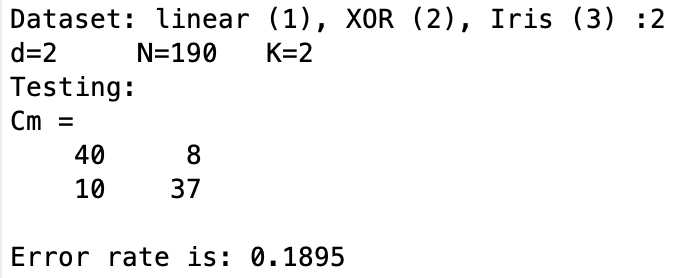
* Linear



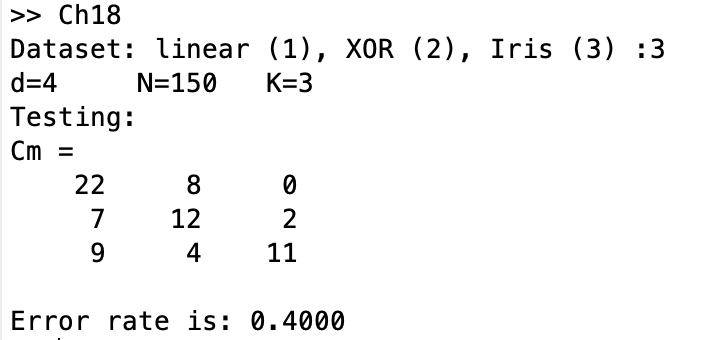


* XOR





* Iris



**Observations:**

* For the linearly separable data, both linear and RBF kernels achieve the same 18% error rate. This is expected as the data is linearly separable.
* For XOR data, the linear kernel has a much higher error rate of 49.47% compared to 18.95% for RBF kernel. This demonstrates the power of the RBF kernel in learning complex non-linear decision boundaries.
* For the Iris data, the linear kernel has a lower error rate of 33.33% compared to 40% for the RBF kernel. This suggests the Iris classes may be close to linearly separable in the original feature space.
* The charts that represent the linearly separable synthetic 3D data with 8 Gaussian clusters are shown in the figures of linear dataset. Each cluster is represented by a different hue and is uniformly spaced along the three axes. The plot shows how the classes are linearly separable, allowing both linear and Kernal Perceptron to perform well.
* From the figures of XOR dataset, displays a plot that shows the 2D XOR dataset with binary labels. The XOR pattern of the two classes is clearly visible. This highlights the need for kernels when data is not linearly separable, as evidenced by the poor performance of the linear perceptron on this dataset.

**Explanations:**

The RBF kernel is a non-linear kernel function that can handle non-linearly separable datasets. It maps each sample to a higher-dimensional space, making the data more separable. However, due to its flexibility, the RBF kernel may overfit the data and result in higher error rates.

In contrast, the linear kernel is limited to handling linearly separable data, but is more stable and less prone to overfitting.

On the linearly separable Iris dataset, the linear kernel performs better, while the RBF kernel likely overfits some data points, leading to higher error rates. This suggests the RBF kernel is learning a more complex boundary than necessary, when a simpler linear boundary suffices for this dataset.

Overall, the RBF kernel excels on non-linear data but may overfit linear data. The linear kernel is limited by linear assumptions but is stable. Their performance depends on the separability of the dataset.

**Conclusion**：

We can conclude that kernelized Perceptron outperform linear and RBF kernel Perceptron on more complicated data (nonlinear data regimes) by implementing and comparing them on synthetic and actual datasets. The much lower mistake rates demonstrate that by projecting the data into higher dimensions, RBF kernel Perceptron may learn effective nonlinear decision boundaries. Linear Perceptron, on the other hand, struggle with nonlinear input, resulting in significantly larger mistakes.

On simple data, however, kernelized Perceptron are prone to overfitting, resulting in higher error rates than linear Perceptron.

Overall, linear Perceptron are simpler and faster than kernelized Perceptron, but they are limited by their linear boundary assumption and perform poorly on nonlinear data with complex decision boundaries.