Mathematics for AI Assignment 3

RBF Network

Group Members	ID
1. Dawit Getahun	UGR/9220/13
2. Metsakal Zeleke	UGR/1027/13
3. Tewodros Berhanu	UGR/9715/13
4. Tinsae Shemalise	UGR/6075/13
5. Yohannes Dessie	UGR/7612/13

Introduction

In this document, we discuss our process and implementation of an RBF network that uses K-means clustering for classification. We have implemented our RBF network on the EEG Eye State Data Set (available at https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State# from UCI
– ML database). We have gained results we believe to be satisfactory, and those will also be analyzed in this document.

Implementation

We wrote a K-means algorithm that utilizes choosing centers, calculating the cluster mean, and then updating the centers until the centers stop changing or the maximum number of iterations is reached. And compared to the built-in implementation found on Scikit-learn, it performed well on randomly generated data as can be visualized as follows.

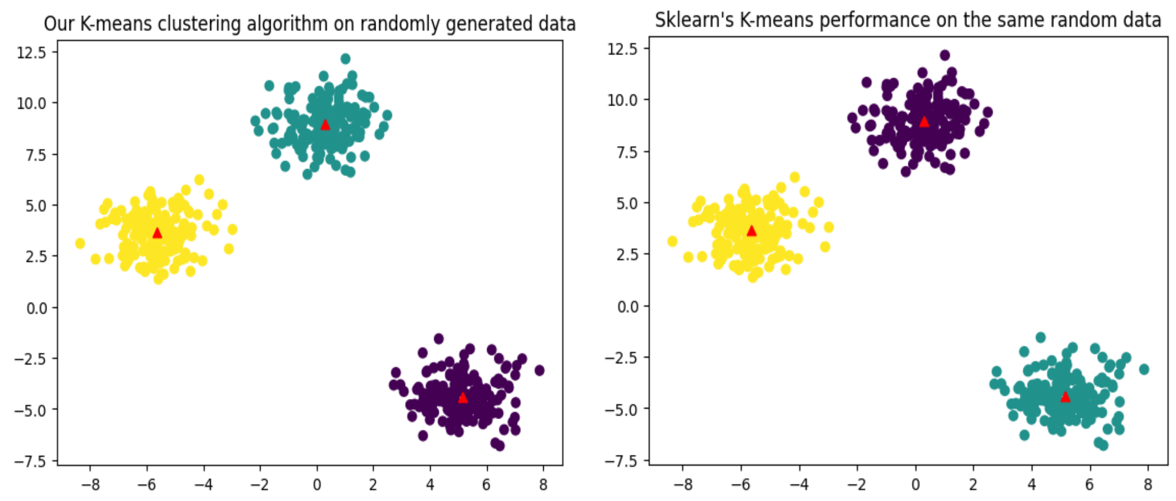

Fig 1. Comparison of our K-means clustering (*left*) with Scikit-learn's K-means (*right*) on randomly generated data

Now if we look at the EEG Eye State Test dataset, it has more than 14,000 entries and to apply our K-means algorithms to it, we would need to get a K value that can get us a good result. And we used **the elbow method** to help us decide the optimal k we need to use. The basic idea behind this method is to load the data and run the clustering algorithm for a range of K values and then plot the inertia (given by the with-in-cluster sum of distances) versus the k graph and look for the elbow point. The elbow point is where the plot stops decreasing quickly and starts decreasing slowly.

To get the best out of our RBF network, K has to be a fraction of the number of samples in the dataset. The exact RBF method would give the optimal performance because it will use the same number of neurons in its hidden layer as the input layer. And that would be heavily taxing in computation power. But what we aspire to do here is create a many-to-one

mapping between the input layer and the hidden layer so that we can the computation cost. So that is why we need to select a reasonable k-value that puts that into consideration.
So what we did was choose a reasonable range of values and select k based on plotting the elbow graph for each of those values. The range we have chosen was between 1, 100, 200, 300, 400, ….., 1000. And the graph can be seen in the following image.

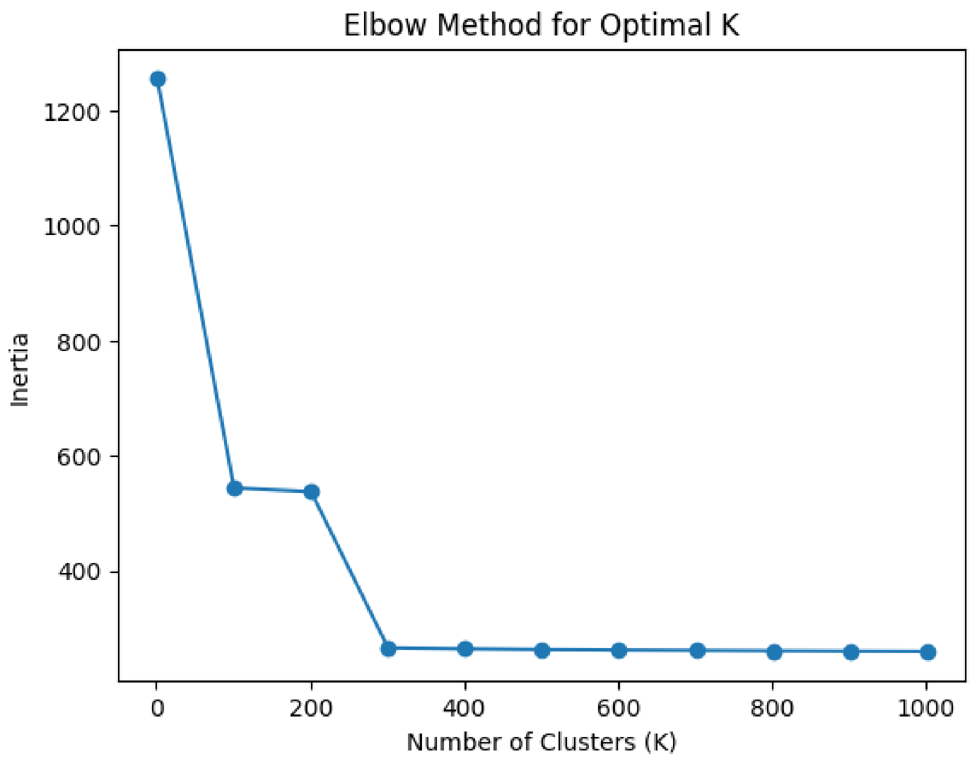

Fig 2. The elbow graph of Inertia vs K for the range 1, 100, 200, …., 1000.

And by looking at the graph we can see that k=300 is the point where the plot stops rapidly decreasing and starts decreasing slowly. So after training the network on the training dataset, we have got a testing accuracy of 85%. But is that really the best we can do? The answer is NO. But first, let’s discuss briefly our implementation of the RBF Network.

Our RBF network will accept each input from the data in its input layer, and then use our K-means clustering algorithm to select k centers. By applying the Gaussian radial basis function we get new feature values. This will leave us with a set of linear equations we can use to solve to get the weights attributed to the new values. The problem can be solved by using NumPy or implementing a Least Mean Squared algorithm to learn the weights. In our implementation, we have used NumPy to solve for the weights.

Back to our discussion of selecting the optimal K, we realize the question of the radial basis network is really an optimization problem. The exact RBF method will guarantee correct classification for all training inputs but there is no learning involved and it becomes impossible to solve for a data with very large number of samples. So we are using the

K-means approach to optimize in terms of computation and also get the best possible performance on the training data. So the question is what is the best K to realize that? The answer is there is no guaranteed way we can say this K is the best for very large datasets. The value depends on the performance metrics (accuracy etc) and the computation metrics (time etc). So for the dataset we have at hand, we have selected a few K values and plotted the accuracy vs K and the training time vs K graphs as shown below. The k values (number of clusters) selected are 1, 100, 200, 300, 400, 500, 1000, 2000 and 3000.

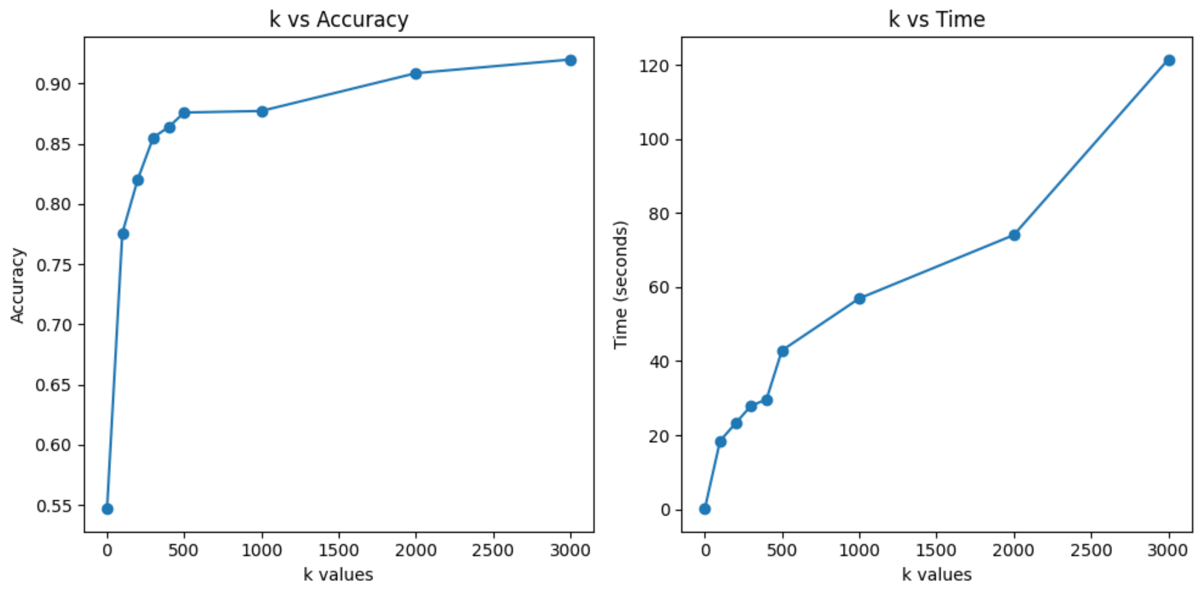

Fig 3. Shows K (number of clusters) vs Accuracy on training data for our RBF network *(left)* and K vs training time taken by the network (*right*)

As can be seen from the figure, the accuracy increases as we go well past our initial elbow point, but it does so very slowly. We can get accuracy on the testing set as high as 91% but it will be computationally taxing. The graph on the right shows that the computational time will be very high as the number of clusters increases. So it is really about balancing the two: computation power and performance.

Conclusion

By working on this assignment, we have learned how to implement the K-means clustering method for unsupervised learning, and how to implement it alongside a Radial Basis Network to apply it to a linearly inseparable dataset. We have applied our own RBF Network to the given dataset. We have also seen the elbow method to get the best K value and experimented with different K values for our network. In the end, the conclusion we have reached demonstrates how selecting the number of clusters in the network is an optimization question and the number should be chosen by analyzing the performance and the computation power at hand.