CSE 5526 - AUTUMN 2020 INTRODUCTION TO NEURAL NETWORKS PROGRAMMING ASSIGNMENT 2

A PREPRINT

Ibrahim M. Koc
Department of Electrical and Computer Engineering
The Ohio State University
Columbus, OH 43201
koc.15@osu.edu

October 9, 2020

1 Discussion

RBF networks assumes that clustering a dataset with different Gaussian distributions could help us predict that dataset. In other words, a Gaussian distribution-like function will give a similar result around its mean point, and will quickly decay to zero as getting further away from the mean. The assumption here is that we are dealing with a symmetric function around the mean (A sine function is a great example of such a function).

For this problem we sample from a function $h(x)$ as described in Eq. 1. First, we sample 75 numbers from a uniform distribution [0.0, 1.0]. Then, we obtain the output of those 75 points from h function, and we corrupt the output by an additive uniform noise in range of [-0.1, 0.1].

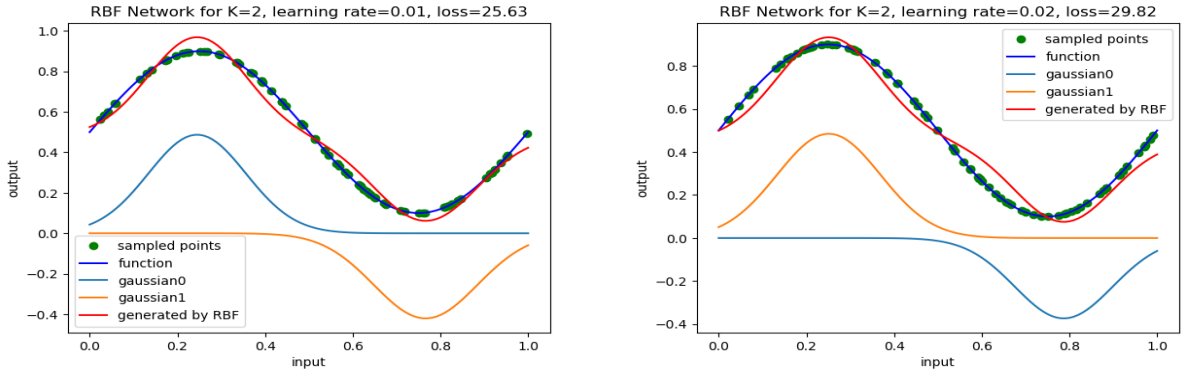
Then we try to first cluster obtained samples, then try to fit a RBF network to predict the latent function. We will use:

- Number of bases in the range of 2,4,11, and 16
- Two values of learning rate (η): 0.01 and 0.02

$$h(x) = 0.5 + 0.4\sin(2\pi x) \tag{1}$$

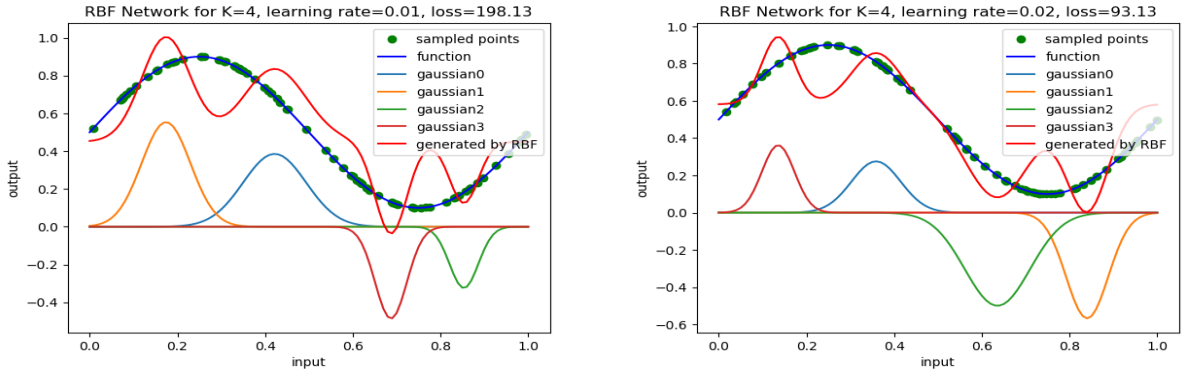
2 Results

Here you can see the results. As one can see increasing the learning rate may reduce the total loss, but not necessarily since it is not the case for Fig. 1. However, one can say that using a learning rate, neither too small nor too big may help to increase prediction performance. Furthermore, looking at Fig. 4 and Fig. 9, one can see that both Fig. 9a and Fig. 9b predicted the actual function very well whereas both Fig. 4a and Fig. 4b performed poorly and overfitted. Therefore, using a constant variance for larger number of clusters prevents overfitting.



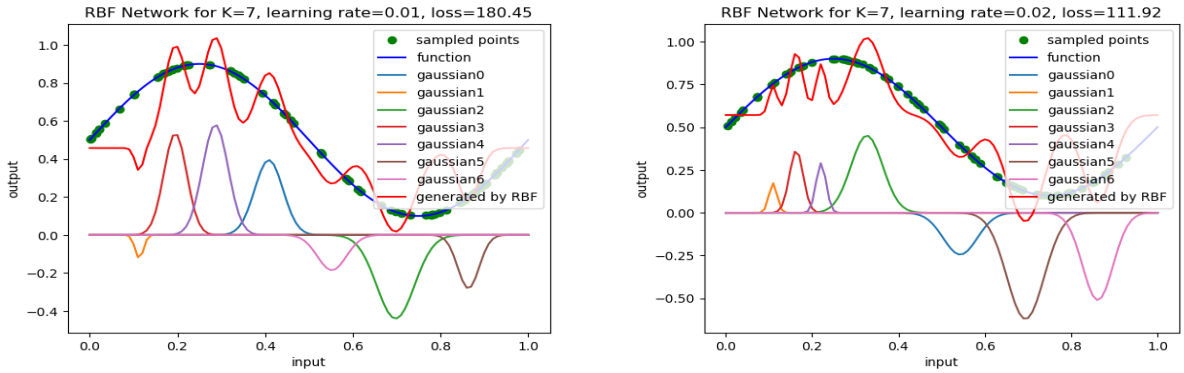
(a) (b)

Figure 1: Different learning rates for 2-means



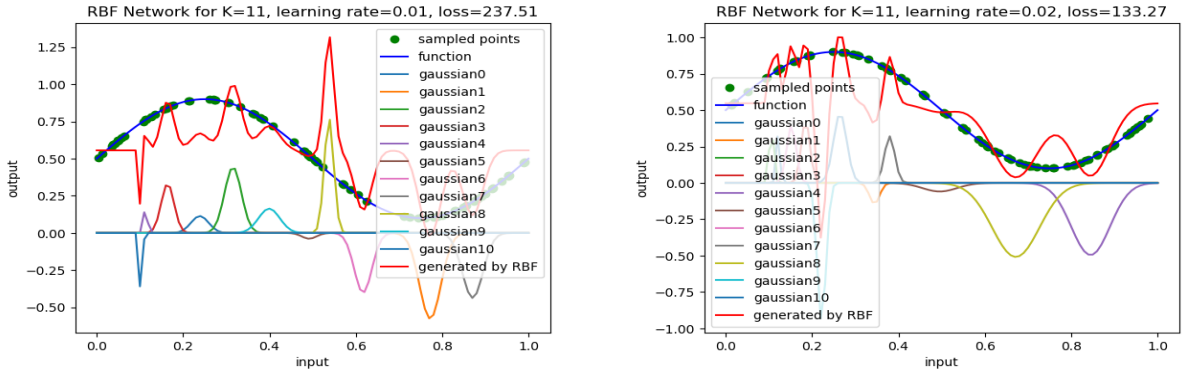
(a) (b)

Figure 2: Different learning rates for 4-means



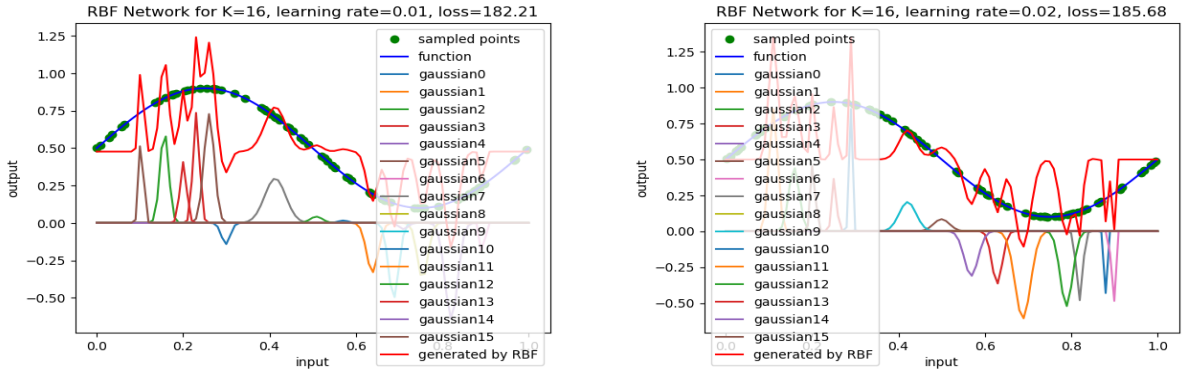
(a) (b)

Figure 3: Different learning rates for 7-means



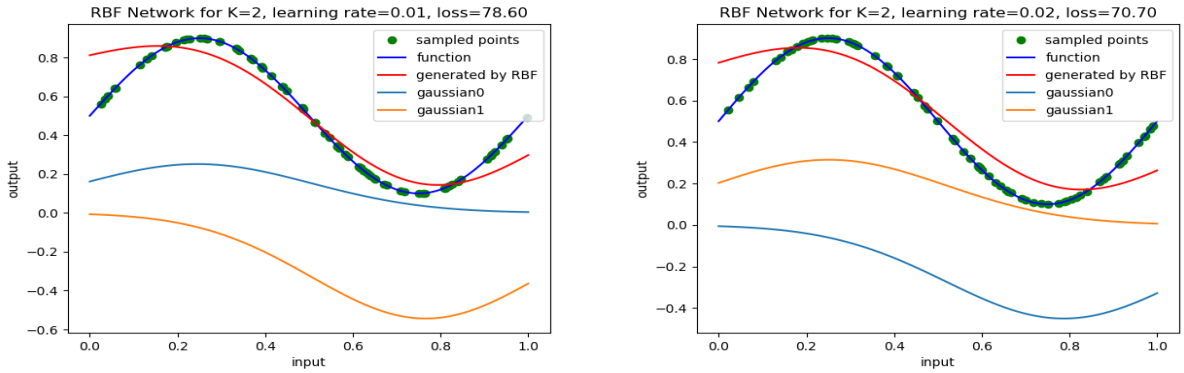
(a) (b)

Figure 4: Different learning rates for 11-means



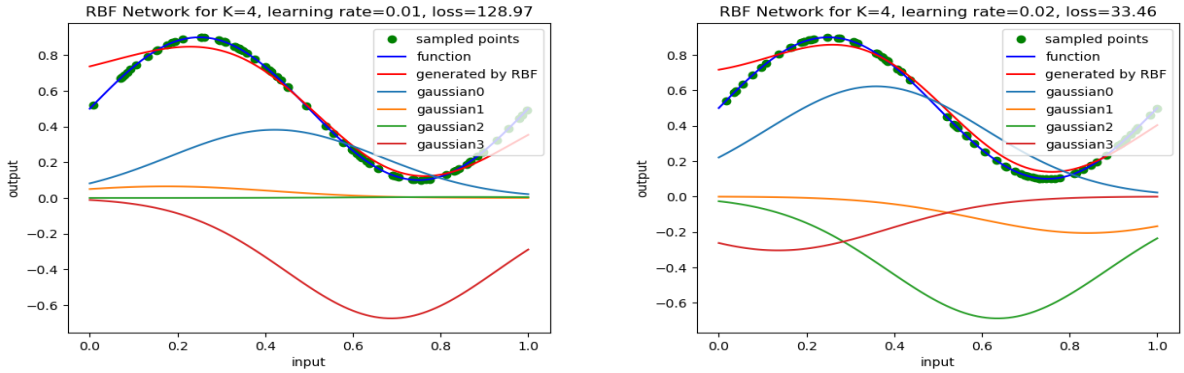
(a) (b)

Figure 5: Different learning rates for 16-means



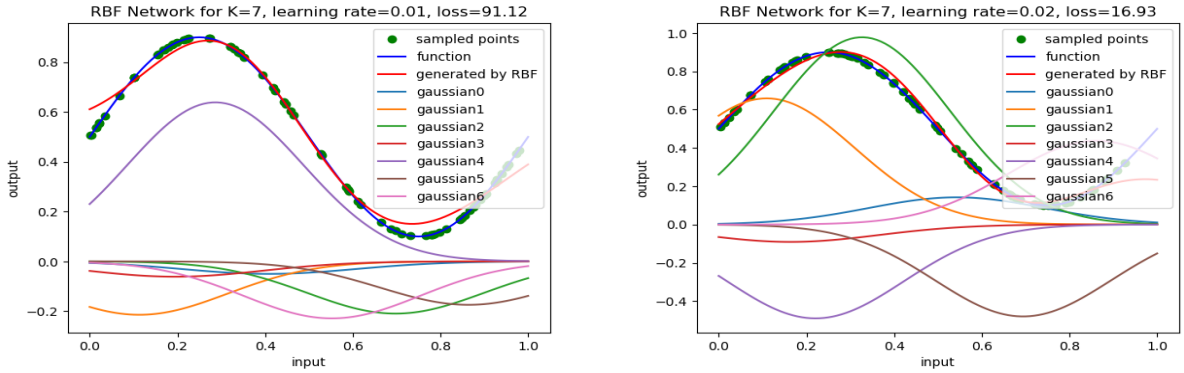
(a) (b)

Figure 6: Different learning rates for 2-means with constant variance



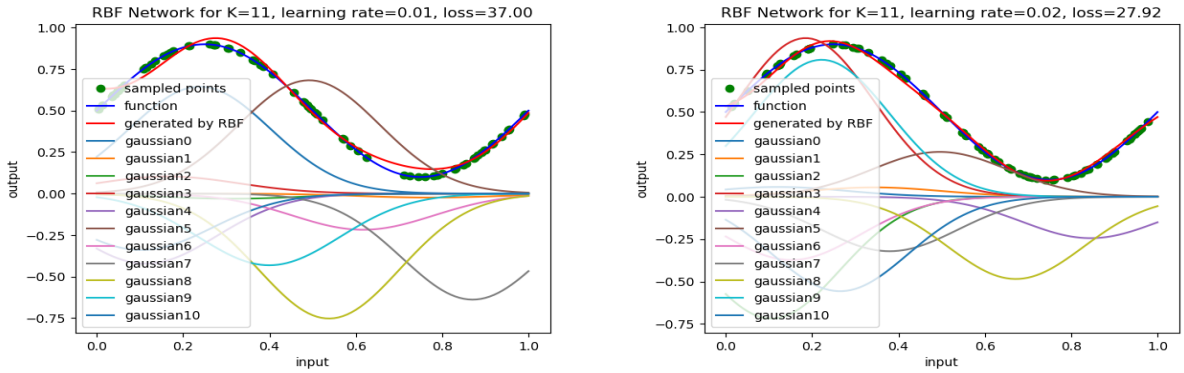
(a) (b)

Figure 7: Different learning rates for 4-means with constant variance



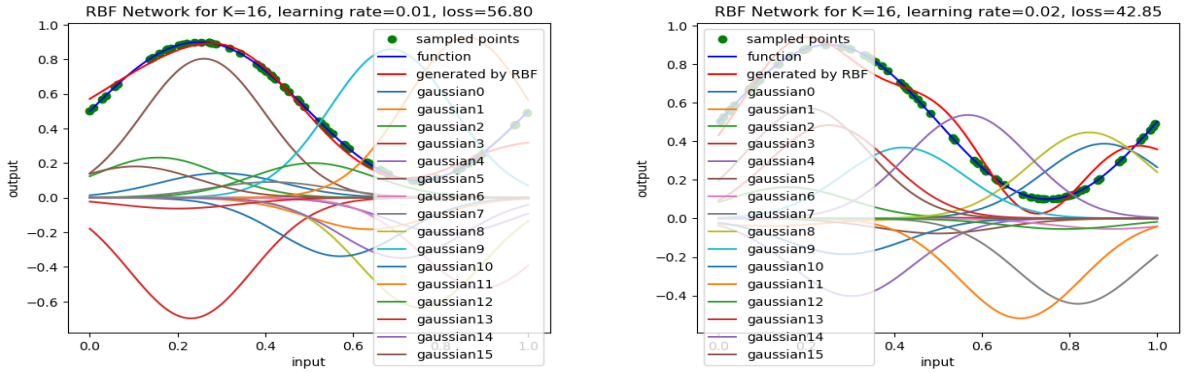
(a) (b)

Figure 8: Different learning rates for 7-means with constant variance



(a) (b)

Figure 9: Different learning rates for 11-means with constant variance



(a) (b)

Figure 10: Different learning rates for 16-means with constant variance

3 Conclusion

As one can see from the figures, constant variance case gets better for higher number of clusters although there is still a number that works the best. Furthermore, constant variance cases tend to less overfit and these cases act less unexpected around unsampled data regions whereas assigning variance to each cluster results in overfitting and a poor performance. Therefore, for storage efficiency, ease of calculation and higher performance, it could be helpful to use a constant variance and a higher number of clusters to predict the dataset.

For project's github link: `https://github.com/mertkoc/simpleNeuralNetwork` (I will upload the code to github after the submission ends)