The Catholic University of America

CSC 527: Fundamental of Neural Networks

Project 2 – Report

The Least-Mean-Square Algorithm

Instructor: Dr. Hieu Bui

Student: Tien Pham

Table of Contents

I. Introduction of Least-Mean-Square Algorithm

- The least-mean-square (LMS) algorithm was developed by Widrow and Hoff (1960) and was the first linear adaptive-filtering algorithm for solving problems such as making predictions and communication-channel equalization.
- The LMS algorithm was developed based on the perceptron algorithm. Although these two algorithms have different applications, they share a common feature: Both involve the use of a *linear combiner*, therefore, the designation of "linear".
- Some advantages of the LMS Algorithm:
  + The LMS Algorithm's complexity is linear
  + The algorithm is simple to code and build
  + The algorithm is robust with respect to external disturbances

II. Project Report

1. Task 1 – Linear Prediction

- For this task, I perform the verification of the statistical learning theory of the LMS algorithm, assuming the learning rate is small.
- The learning rates used in this task are η = 0.001, η = 0.002, η = 0.01, and η = 0.02.
- The results are obtained as follows:

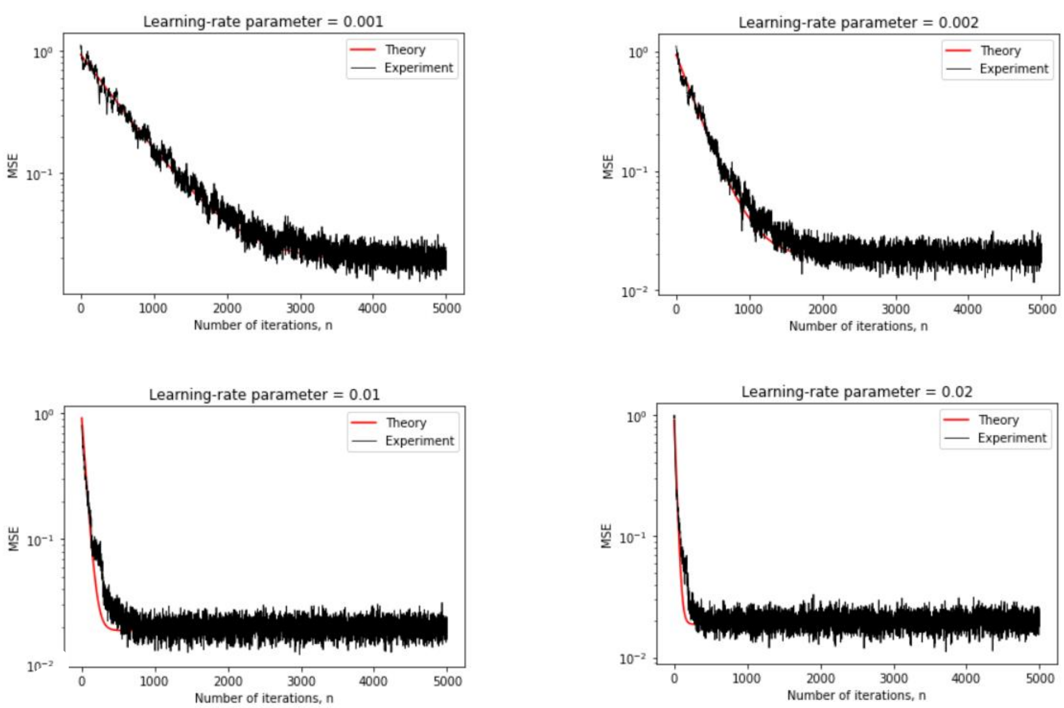


Fig 1. Experimental verification of the small-learning-rate-parameter theory of the LMS algorithm applied to an autoregressive process of order one

- The table below shows the summary of results obtained from the above plots:

Learning Rate	Number of time-steps for convergence (approx.)
0.001	3500
0.002	2200
0.01	600
0.02	400

2. Task 2 – Double Moon Classification

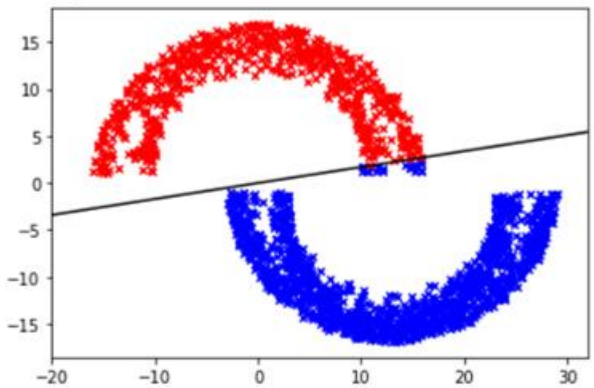
- For each distance (d = 1, 0, -4), the program generates a training dataset and a test dataset using the given "moon" function.
- Then, I apply the LMS Algorithm to train the "train dataset" (Here, number of epochs is 20 and learning rate is 0.005)

```python
def Train(dataset, epochs, learningRate):
    w = np.random.rand(2)/2 - 0.25
    mseArr = []
    for epoch in range(epochs):
        mse = 0.0
        np.random.shuffle(dataset)
        for row in dataset:
            rowLabel = row[:2]
            rowLabel = np.asarray(rowLabel)
            prediction = np.dot(w, rowLabel)
            expected = row[-1]
            error = expected - prediction
            mse += error ** 2
            w = w + learningRate*error*rowLabel

        mse /= len(dataset)
        mseArr.append(mse)

        if mse == 0:
            break

    return w, mseArr
```

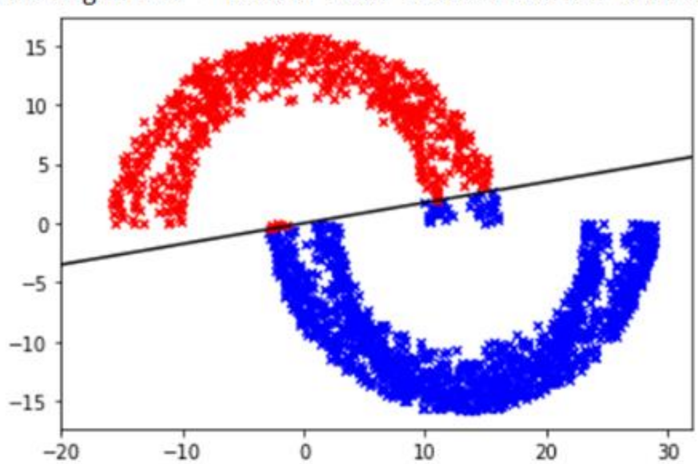
Fig 2. The LMS Algorithm for training the dataset

- Then, I use the trained result to apply on the test dataset and the results are obtained as follows:

LMS Algorithm - Double Moon Classification with distance = 1 :



LMS Algorithm - Double Moon Classification with distance = 0 :



LMS Algorithm - Double Moon Classification with distance = -4 :

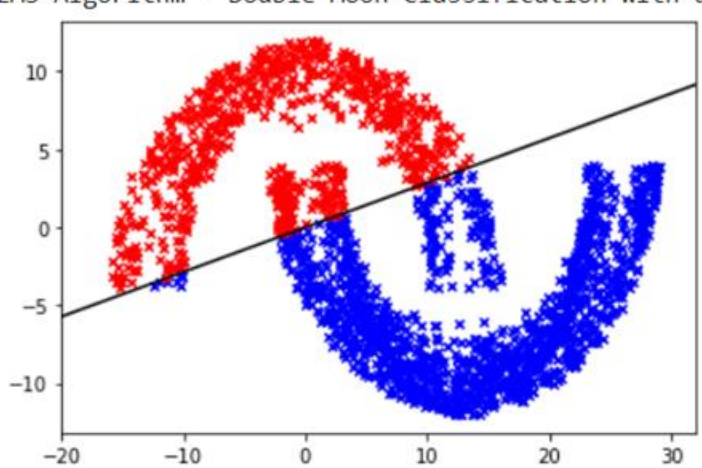


Fig 3. Double Moon Classification using LMS with distance d = 1, d = 0, and d = -4

3. Task 3

- Below is the Double Moon Classification result from the Rosenblatt Perceptron method (Source: Tien Pham – csc527 – Project1 - https://github.com/tienpham2103/csc527/tree/master/Project1)

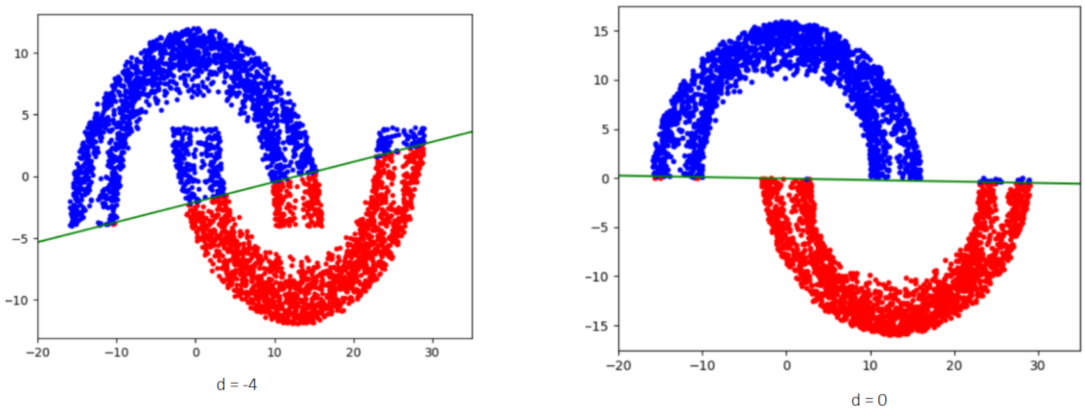


Fig 4. Double Moon Classification using Rosenblatt Perception with distance d = -4 and d = 0

- Below is the Double Moon Classification result from the method of Least Square (Source: Tien Pham – csc527 – Homework4 - https://raw.githubusercontent.com/tienpham2103/csc527/master/Homework4/2%20moon%20classificaction.png)

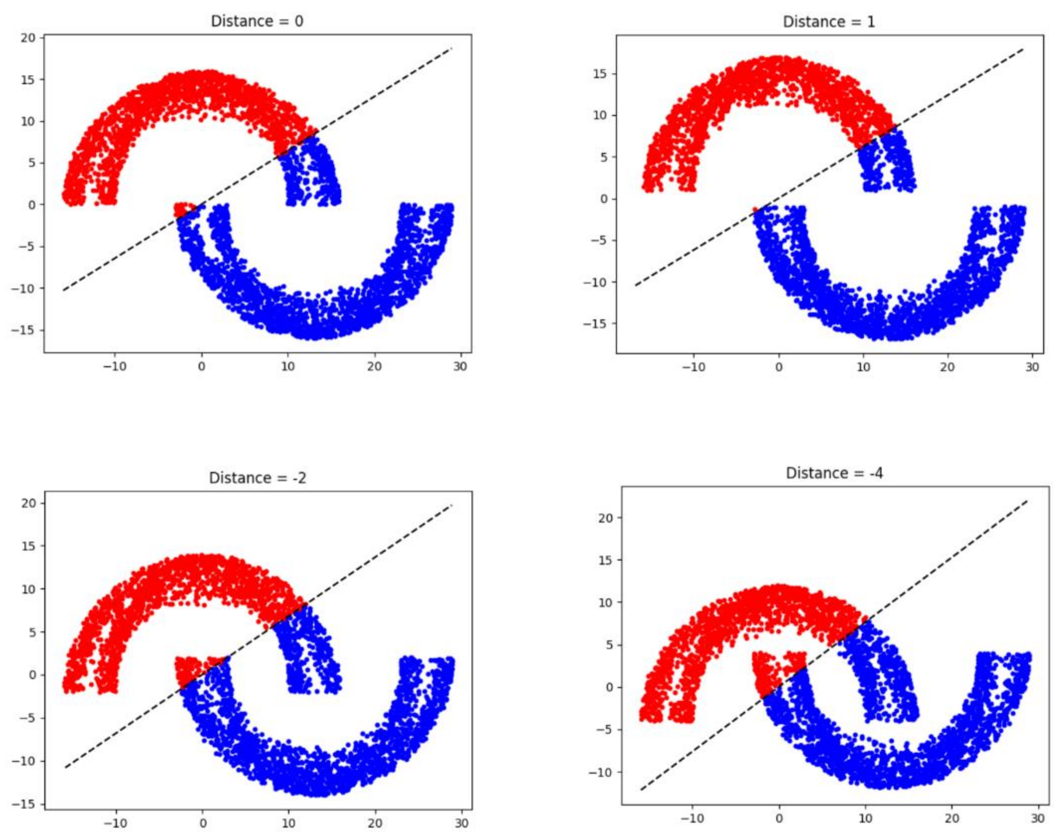


Fig 5. Double Moon Classification using Least Squares method with distance d = 0, d = 1, d = -2, and d = -4

- In order to make better comparison, let us check the Mean Square Error of each method/algorithm:

Distance (d)	Average Mean Square Error		
	Rosenblatt Perceptron	Least Squares	Least Mean Square
1	N/A	0.36	0.28
0	0.015	0.42	0.32
-4	0.7	0.72	0.48

- It can be seen that the LMS algorithm always performs better than the Least Squares method, that the LMS algorithm performs better than the Rosenblatt Perceptron method for d = 0 (and presumably, d = 1 as well), which means the problem is linearly classifiable, and that the LMS algorithm performs worse than the Rosenblatt Perceptron method for d = -4, which means the problem is non-linearly classifiable.

4. Task 4

- Below is the learning curves of the LMS algorithm applied to the Double Moon Classification for different values of distance:

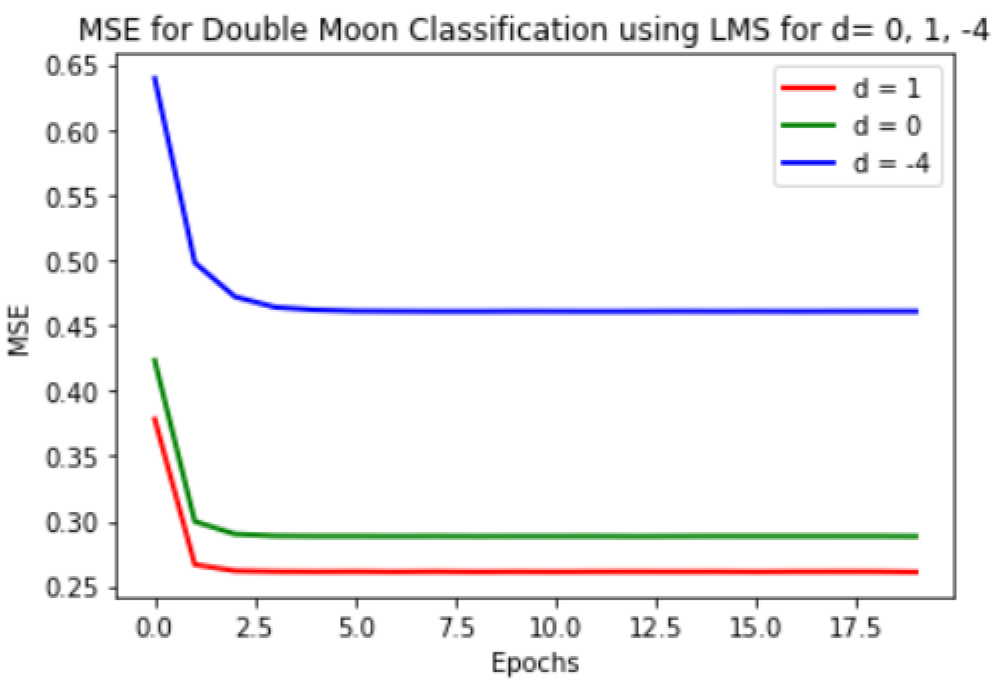


Fig 6. Learning curves for different values of distance

- On the verge of linear separability (d = 0), the mean square error of LMS algorithm is approximately 0.32 and the mean square error of Rosenblatt Perceptron is 0.015.

III. References

1. Simon, H. (2009). Neural Network and Learning Machine (3rd Edition)
2. Least-Mean-Squares Python. Retrieved from: https://matousc89.github.io/padasip/sources/filters/lms.html

The codes for this project can be found at my GitHub (https://github.com/tienpham2103/csc527/tree/master/Project%202)