Charles Bangert

11/1/19

Problem Set 6: Expectation Maximization

Problem 1:

i. x=0:0.05:1; y=2*x+1

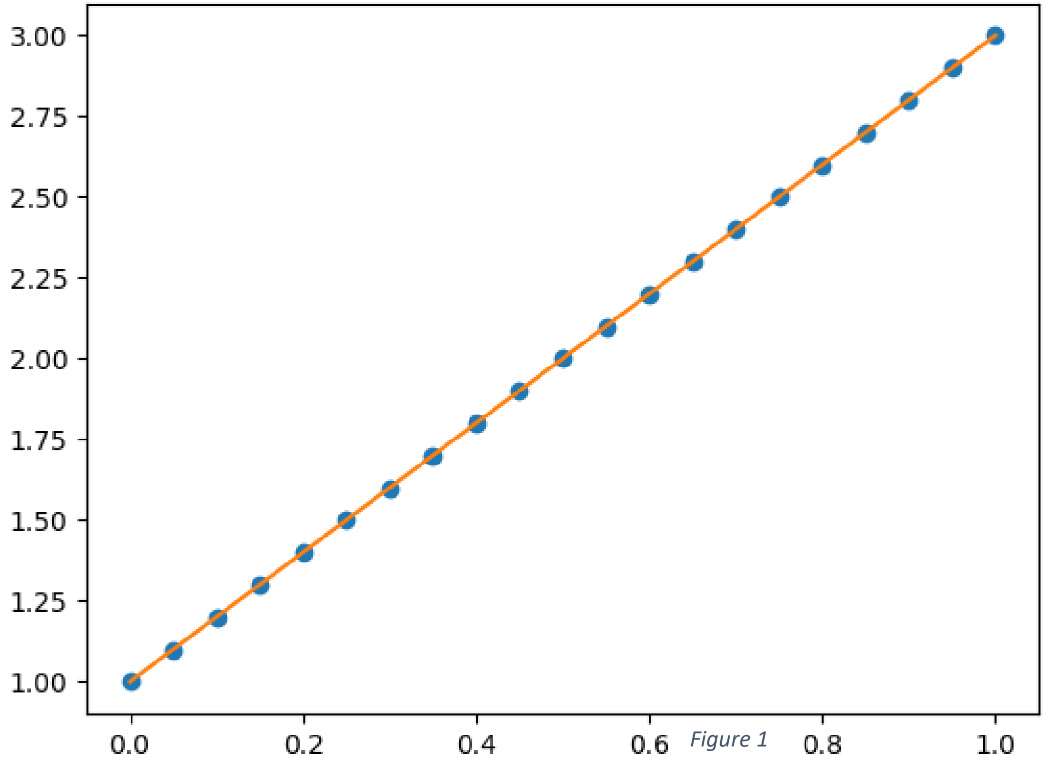
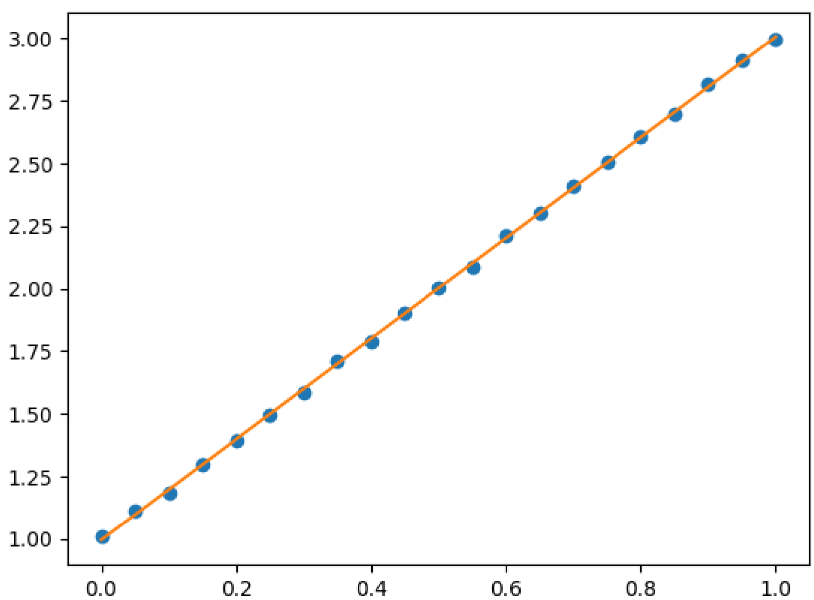
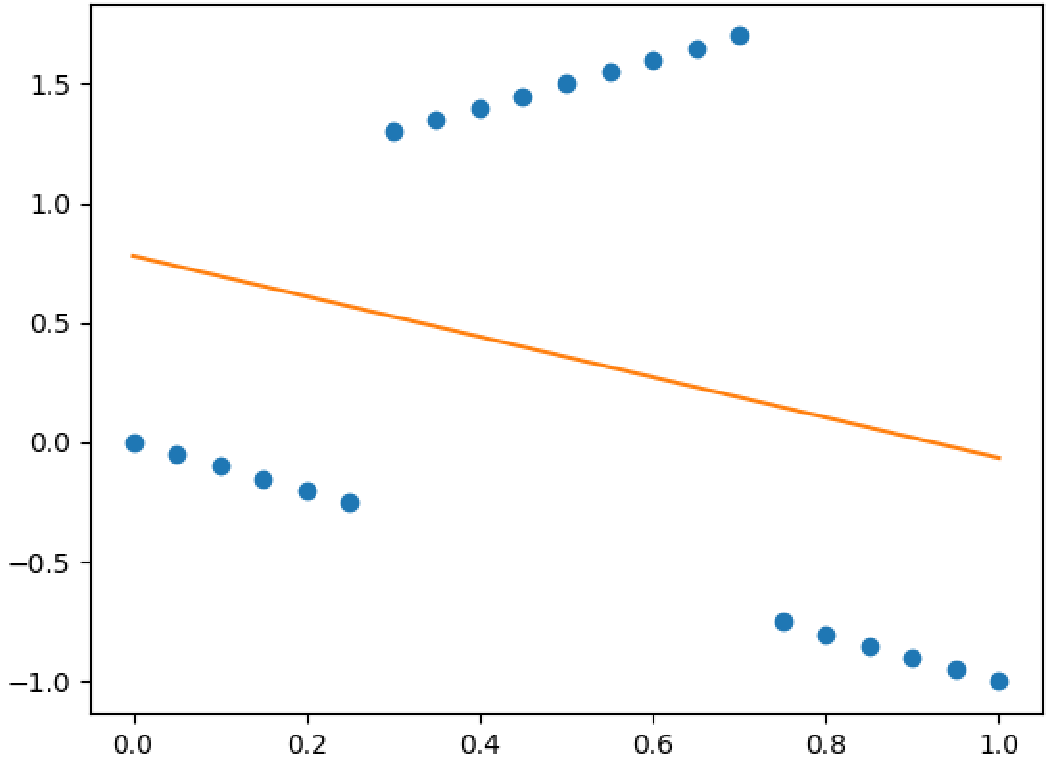


Figure 1

ii. x=0:0.05:1; y=2*x+1+0.1*randn(size(x))



iii. x=0:0.05:1; y=(abs(x-0.5) < 0.25).*(x+1)+(abs(x-0.5) >=0.25).*(-x);

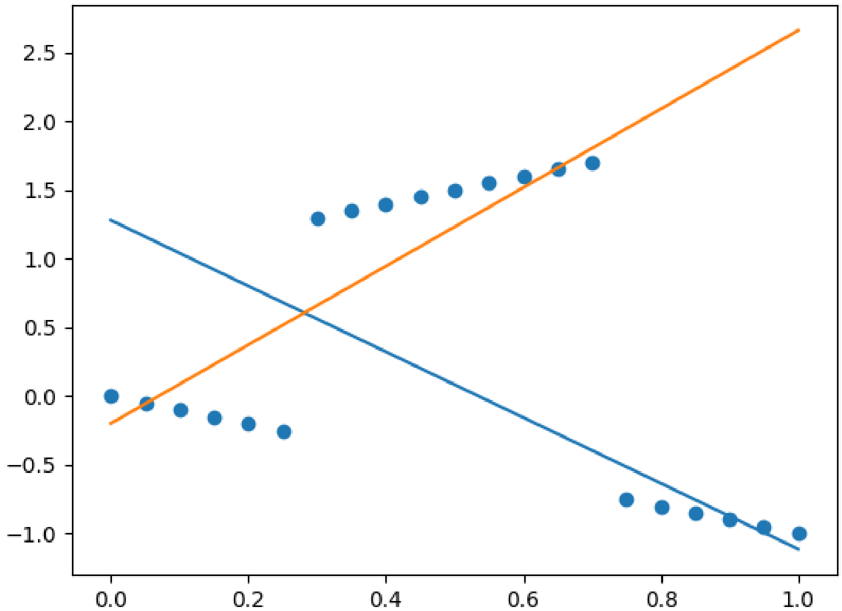


For problem one, the least squares method was used to fit a line to the points. As the plots show, this follows intuition that it works best when the points most closely resemble a line.

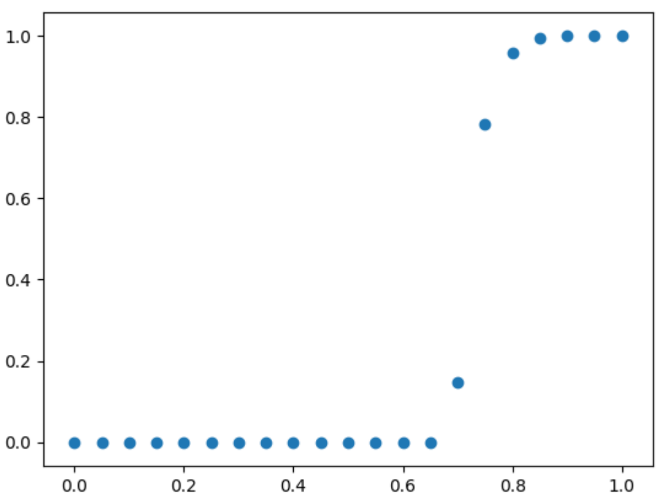
Problem 2:

a.

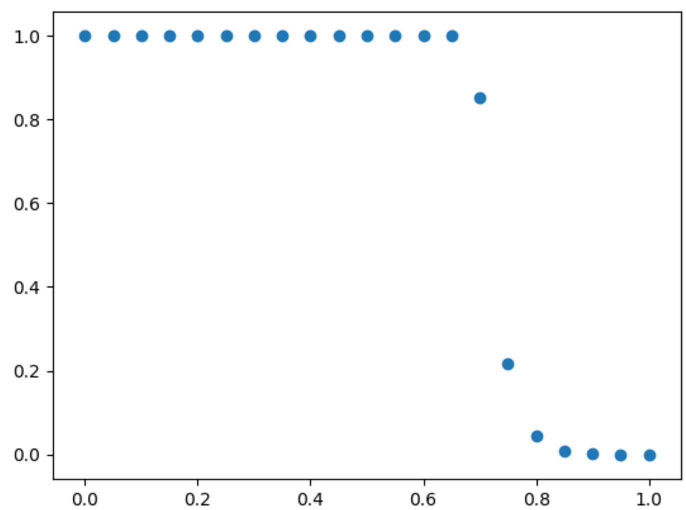
Iteration 1:



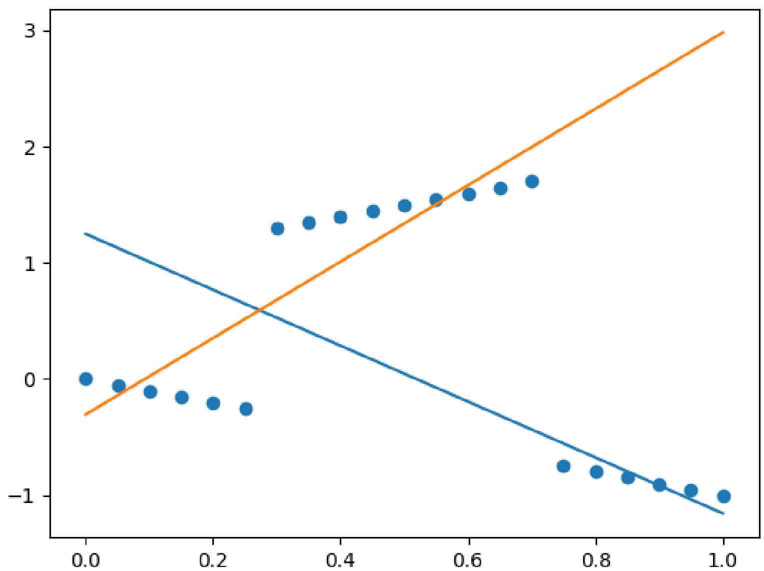
Weight 1:



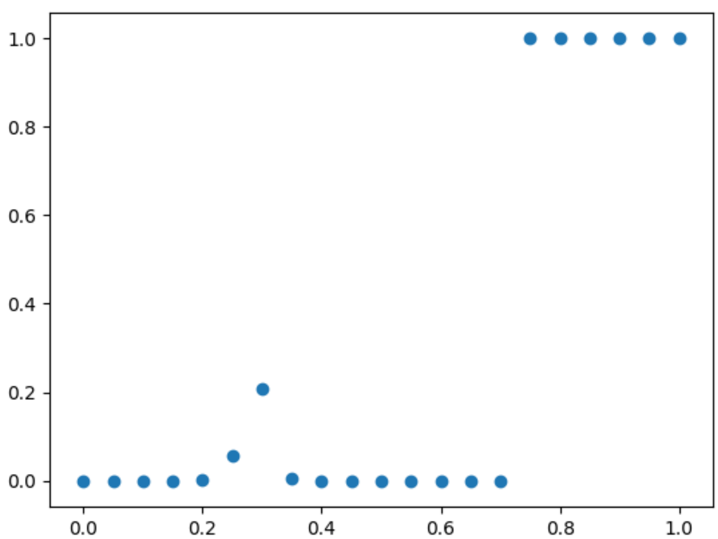
Weight 2:



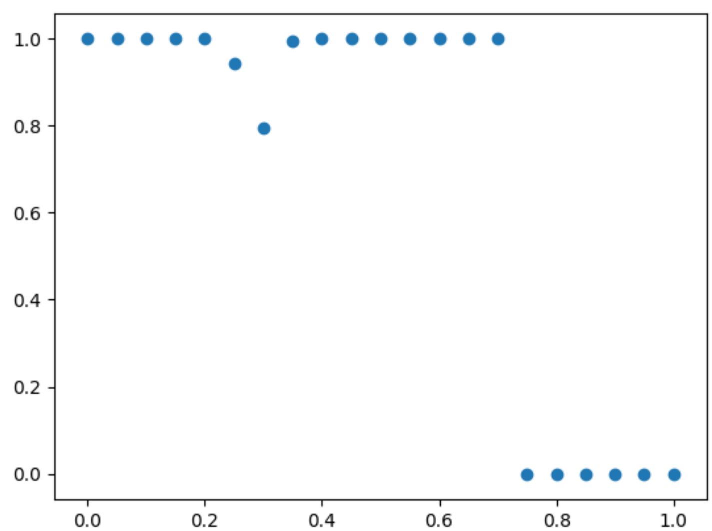
Iteration 2:



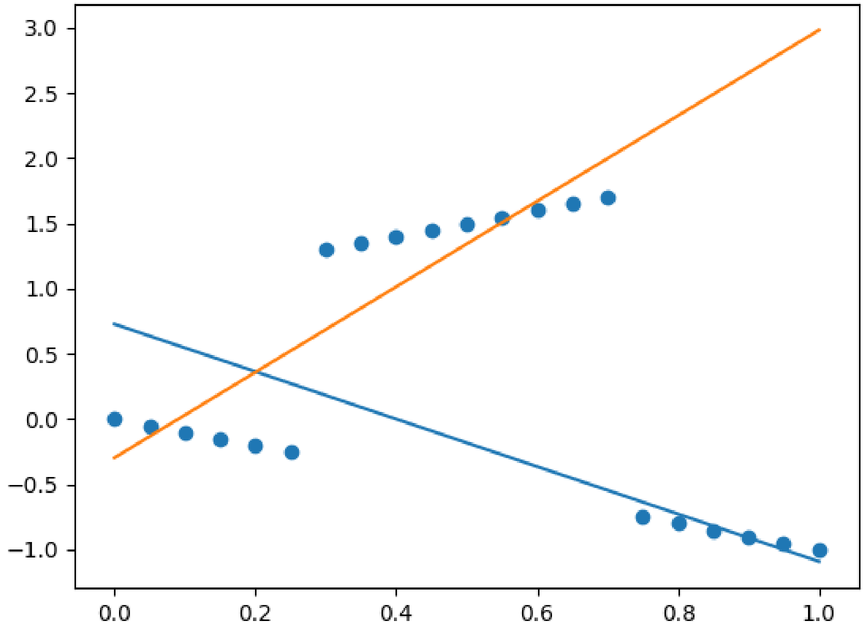
Weight 1:



Weight 2:

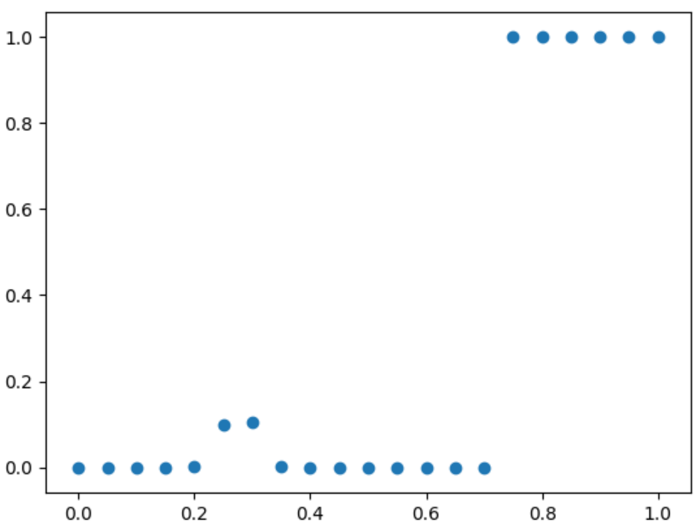


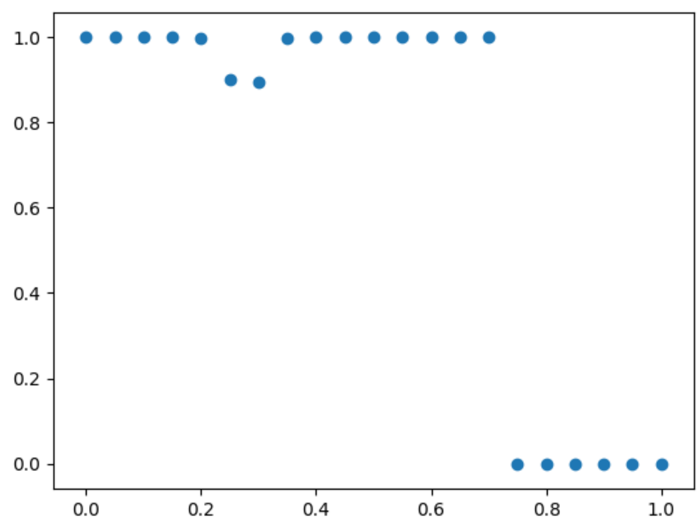
Iteration 3:



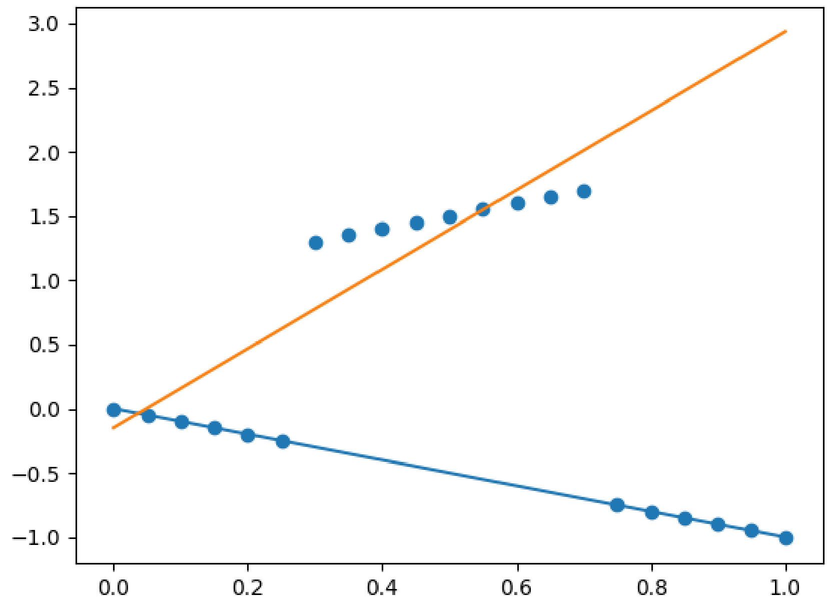
Weight 1:

Weight 2:

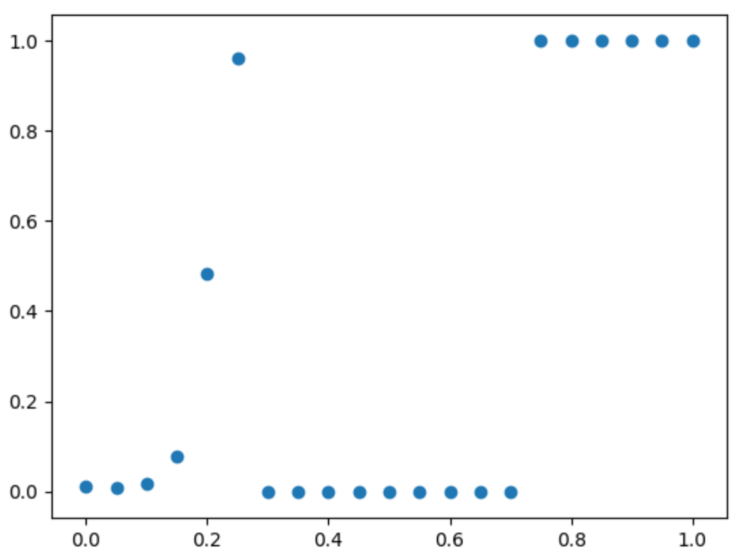




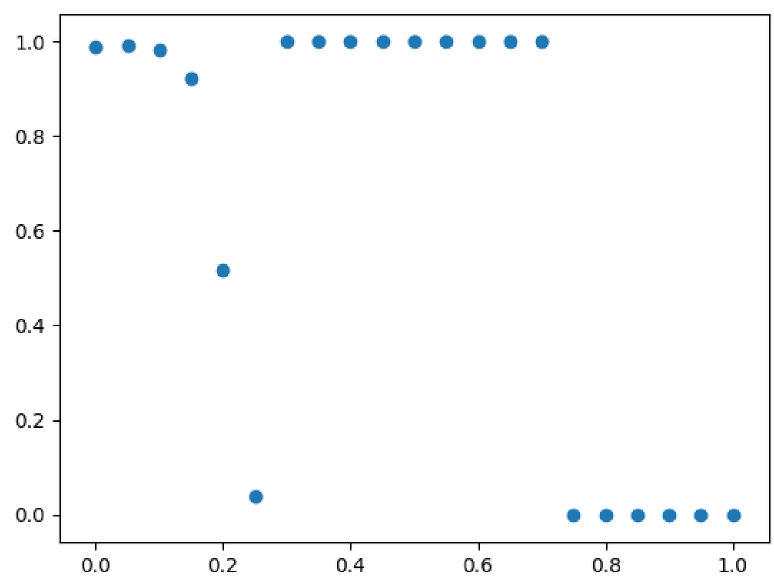
Iteration 4:



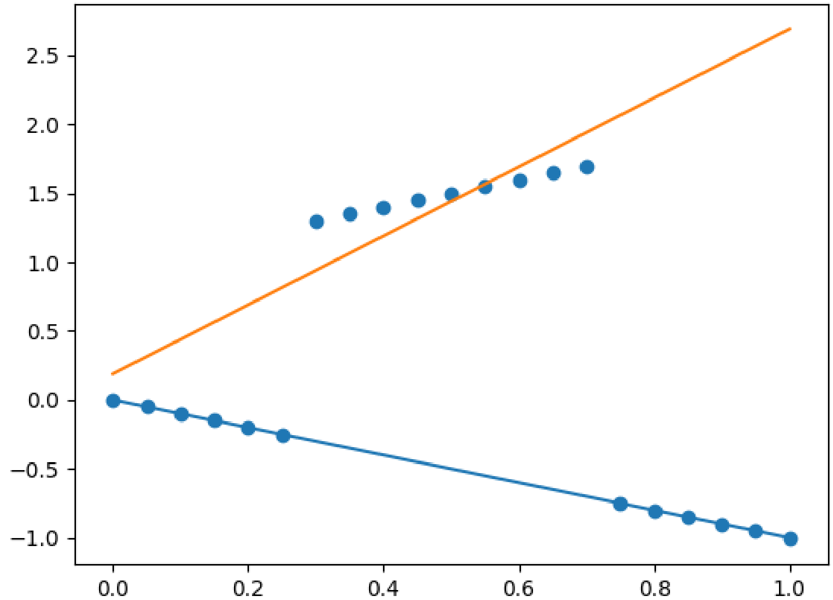
Weight 1:



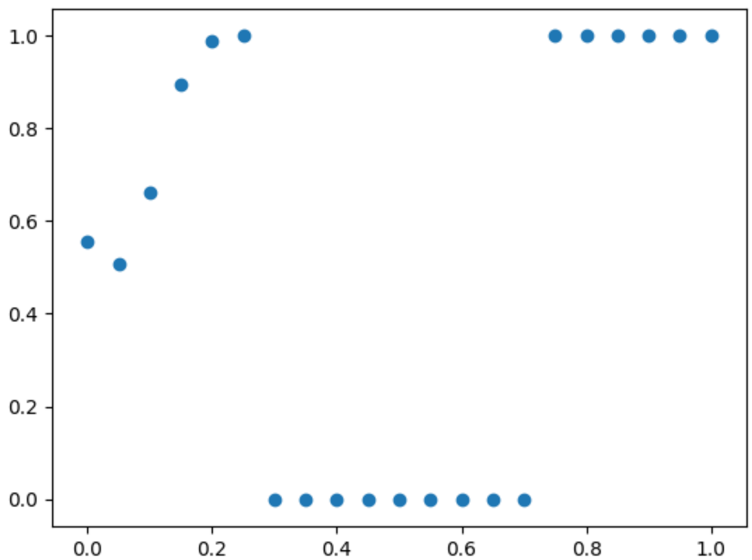
Weight 2:



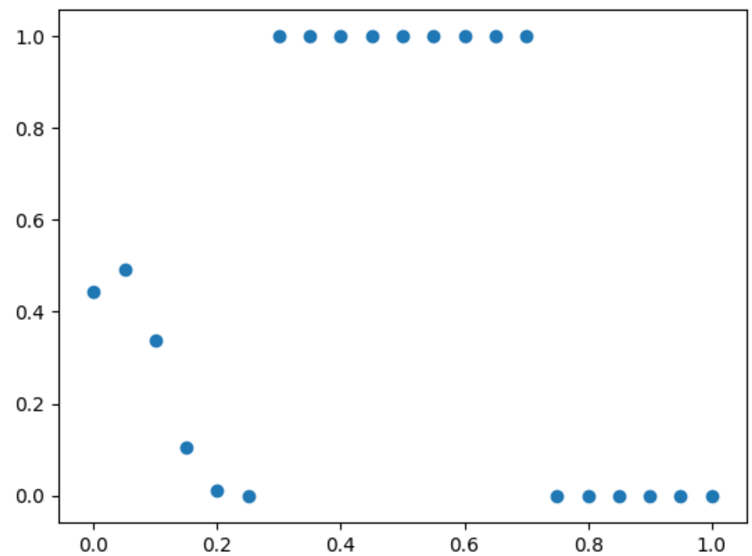
Iteration 5:



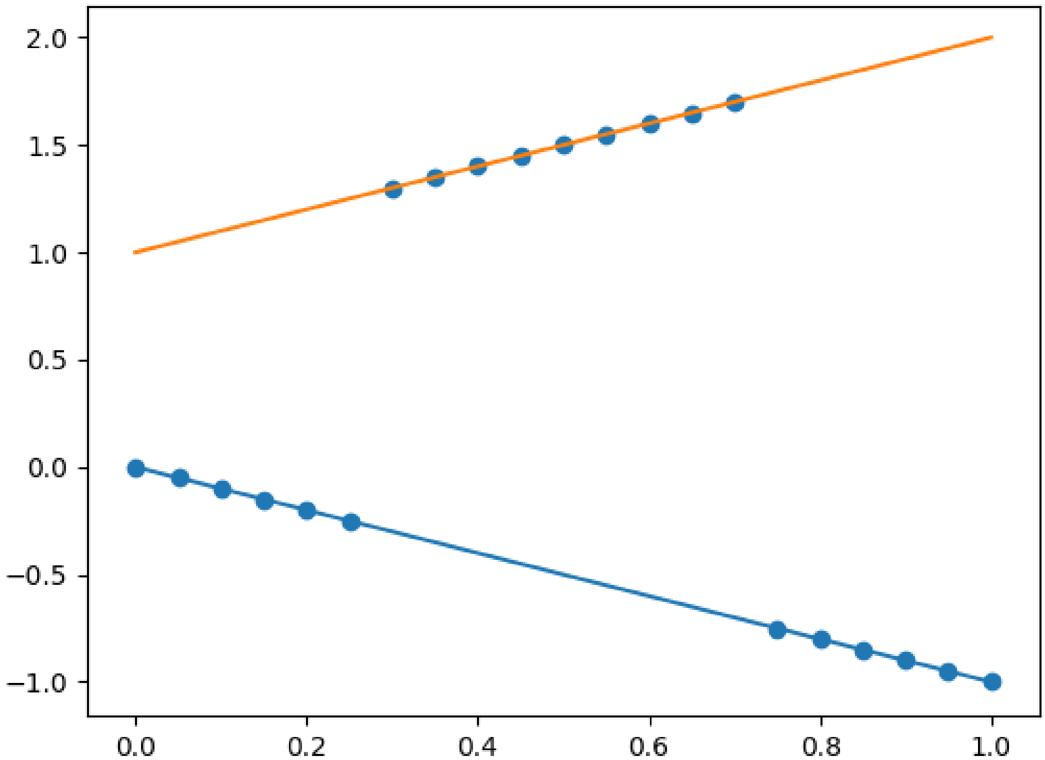
Weight 1:



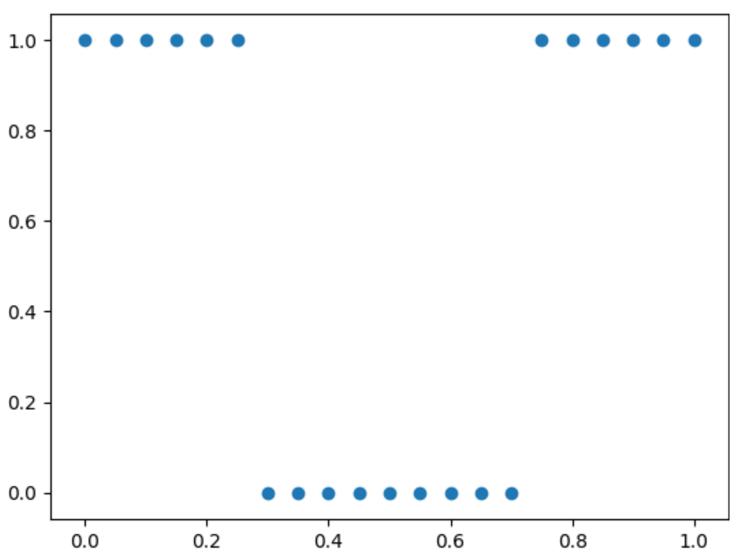
Weight 2:



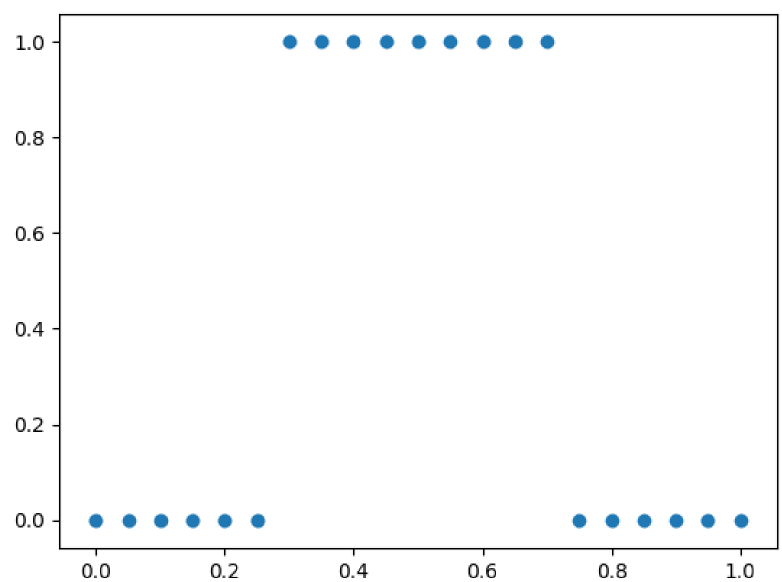
Final Plot:



Final Weight 1:

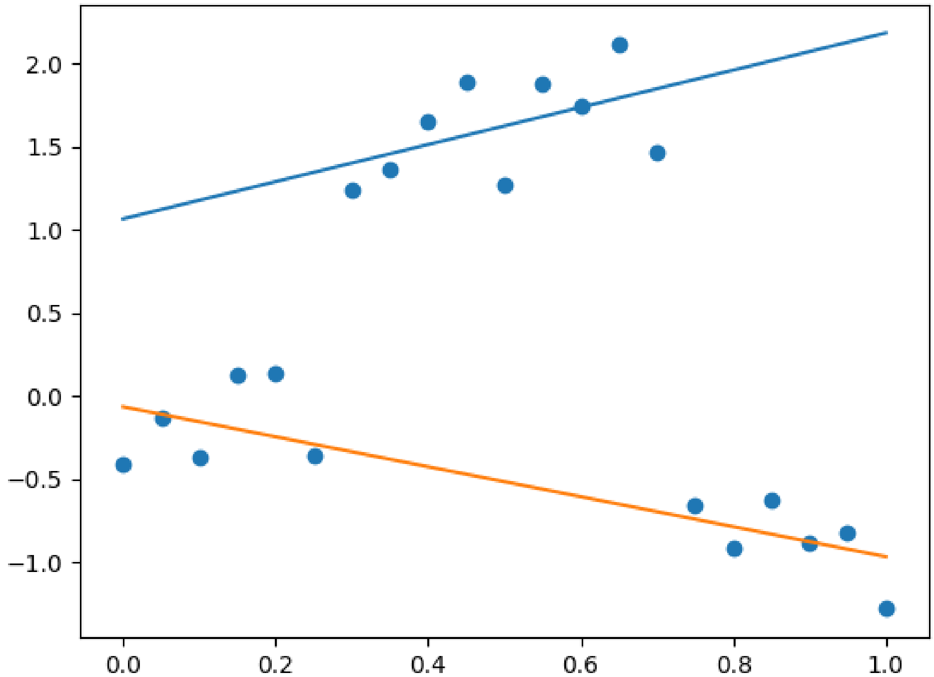


Final Weight 2:



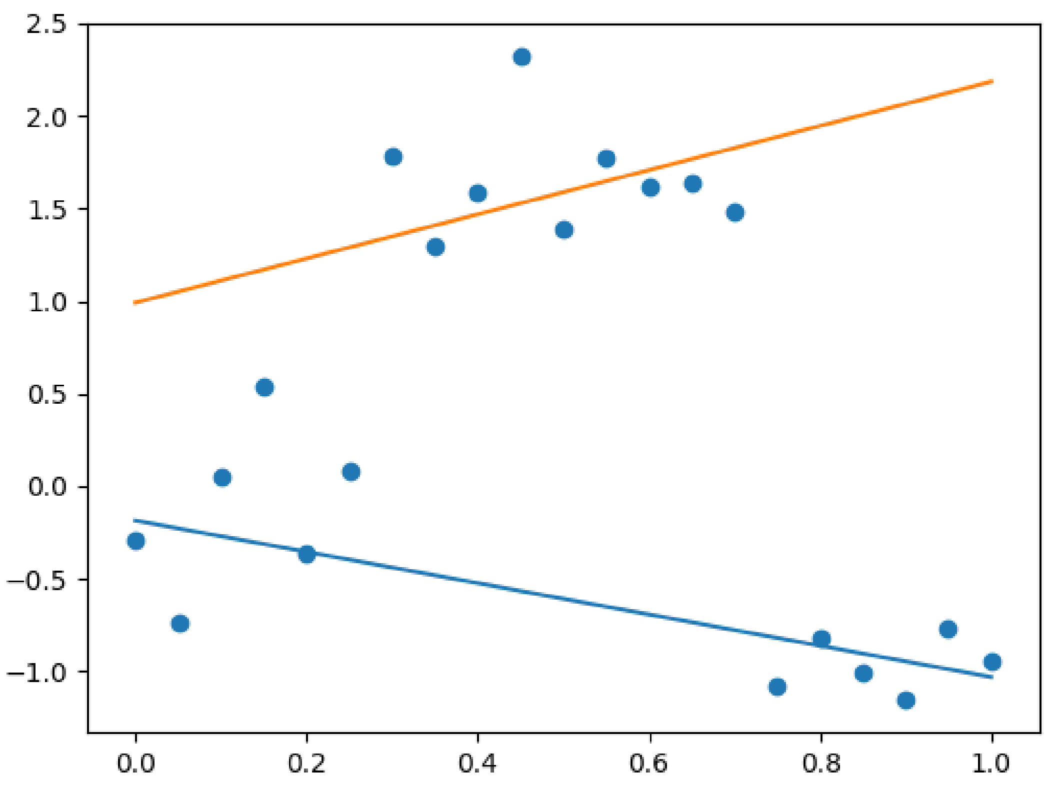
b. <u>Adding Random Gaussian Noise</u>

<u>Experiment 1: Gaussian noise centered at 0 with standard deviation .25</u>:

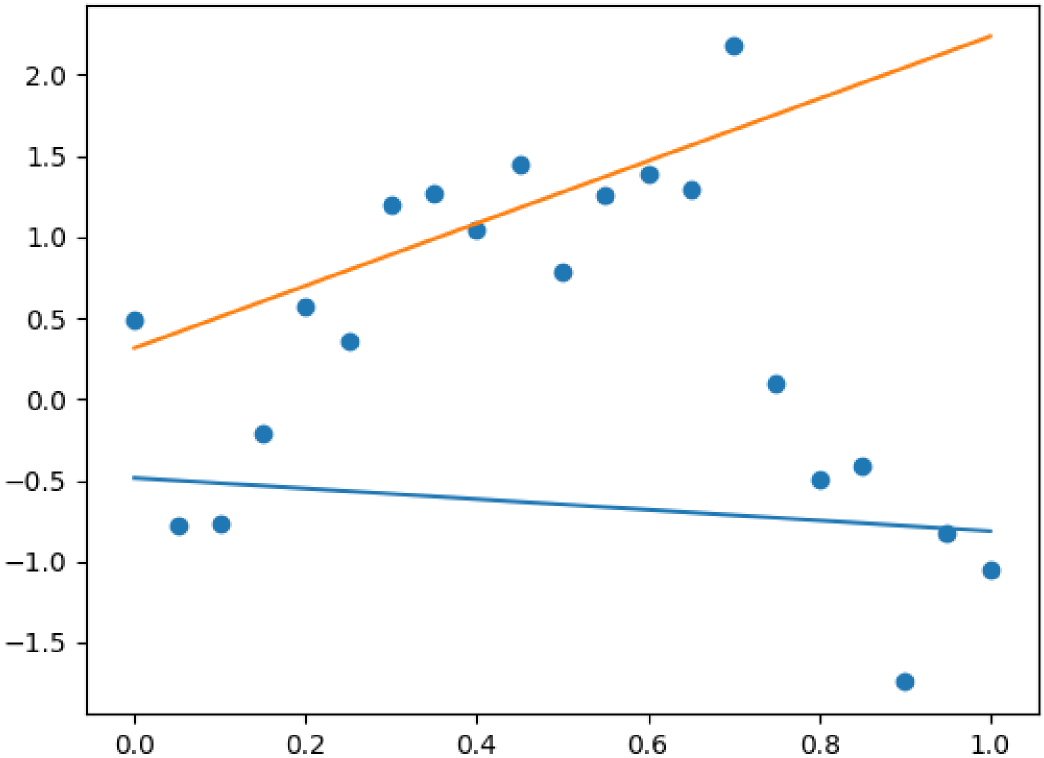


With this amount of Gaussian noise added, it appears the algorithm still converges to the correct lines. This is demonstrated in the above plot.

<u>Experiment 2: Gaussian noise centered at 0 with a standard deviation of .5:</u>

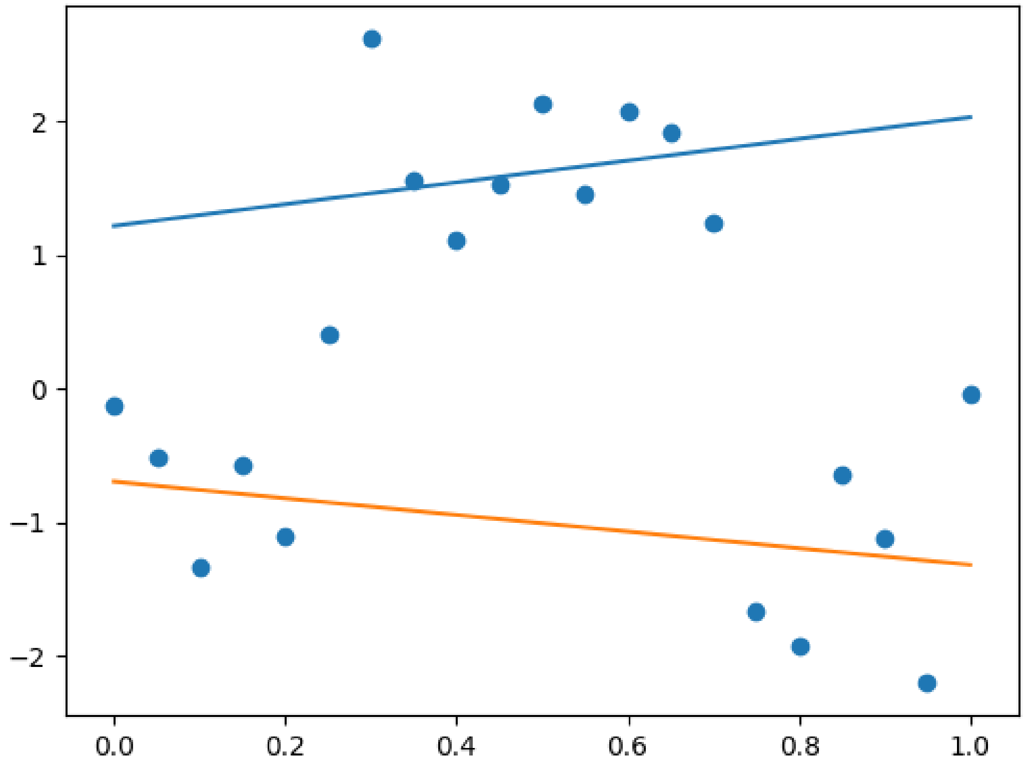


With added gaussian noise with standard deviation of .5, the algorithm still can converge to the correct lines. Above is an example of a time where the random noise added didn't come from any outliers in the Gaussian distribution, so the line fit worked very well. Below is a plot of a run of the experiment where some of the random numbers were further from the mean in the Gaussian distribution:

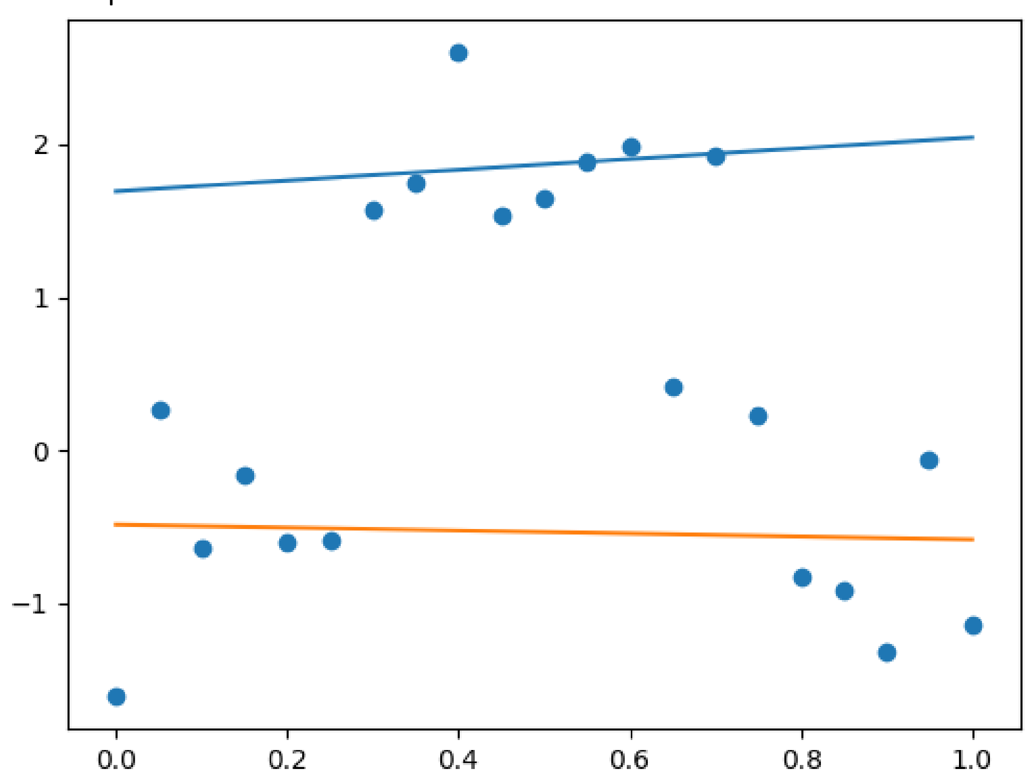


The outputted lines still look close to what we expect, though.

Experiment 3: Gaussian noise centered at 0 with standard deviation .75:

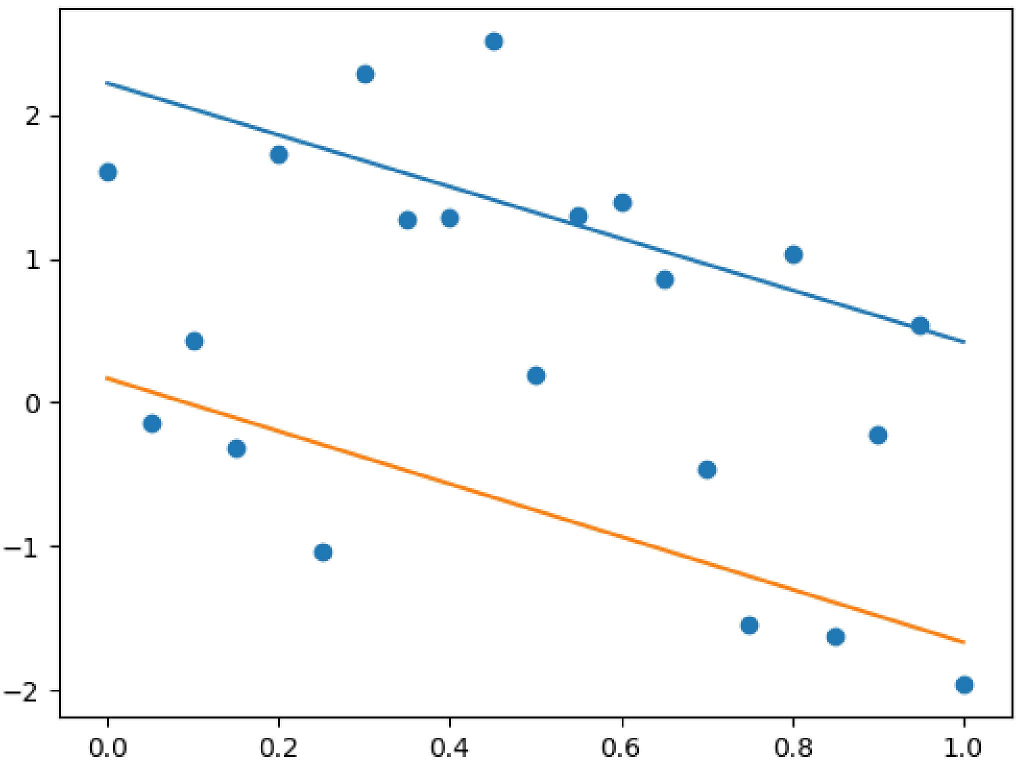


With this much noise, we can still visually pick out 3 clusters of points that the generating function outputs. After several runs of the experiment, the above plot was best representative of what I was expecting in terms of the fitted lines. This is because the random data, as previously mentioned, came from points closer to the center of the Gaussian distribution. Below is a plot where this wasn't the case:



The lines obtained still were reasonably close to expected, as the bottom one still has a slight negative slope and is represented by the line y = -.097x - .483. While this isn't that close to y = -x as expected, its not too bad yet. Same goes for the top line, y = .35x + 1.69. We were expecting y = x + 1, which isn't too far off.

<u>Experiment 4: Gaussian noise centered at 0 with standard deviation of 1:</u>



This is the point where I would say that the algorithm has broken. Additional runs of the experiment yield slightly better lines, but they can also be much worse, such as the following: