James Folk

DATA 527 – Predictive Modeling

Assignment 3

DEADLINE: March 28, 2024

Spring 2024

Overview

The purpose of this study is to implement a feed forward network solves the problem of the XOR logic function.

Methodology

- **Creation of the data structure of the Neural Network:** I chose to create the neural network using a class data structure.
- **Viewing the data:** I wanted to be able to view the neural network, so I added functions to draw the neural network using `matplotlib`.
- **Writing the feed forward:** Following the algorithm of the lecture, I wrote the feed forward process
- **Writing the back propagation:** Following the algorithm of the lecture, I wrote the back propagation process.
- **Testing the algorithm:** I utilized python's `unittest` package to test the different functionality. The two functions `test_batch_learning` and `test_stochastic` run the batch learning process and the stochastic process.
- **Calculation the error:** Implemented the estimate_r_squared function, utilizing the trained model to predict dependent values based on the dependent values.

Implementation

articlealgorithmalgorithmic
Feedforward Neural Network

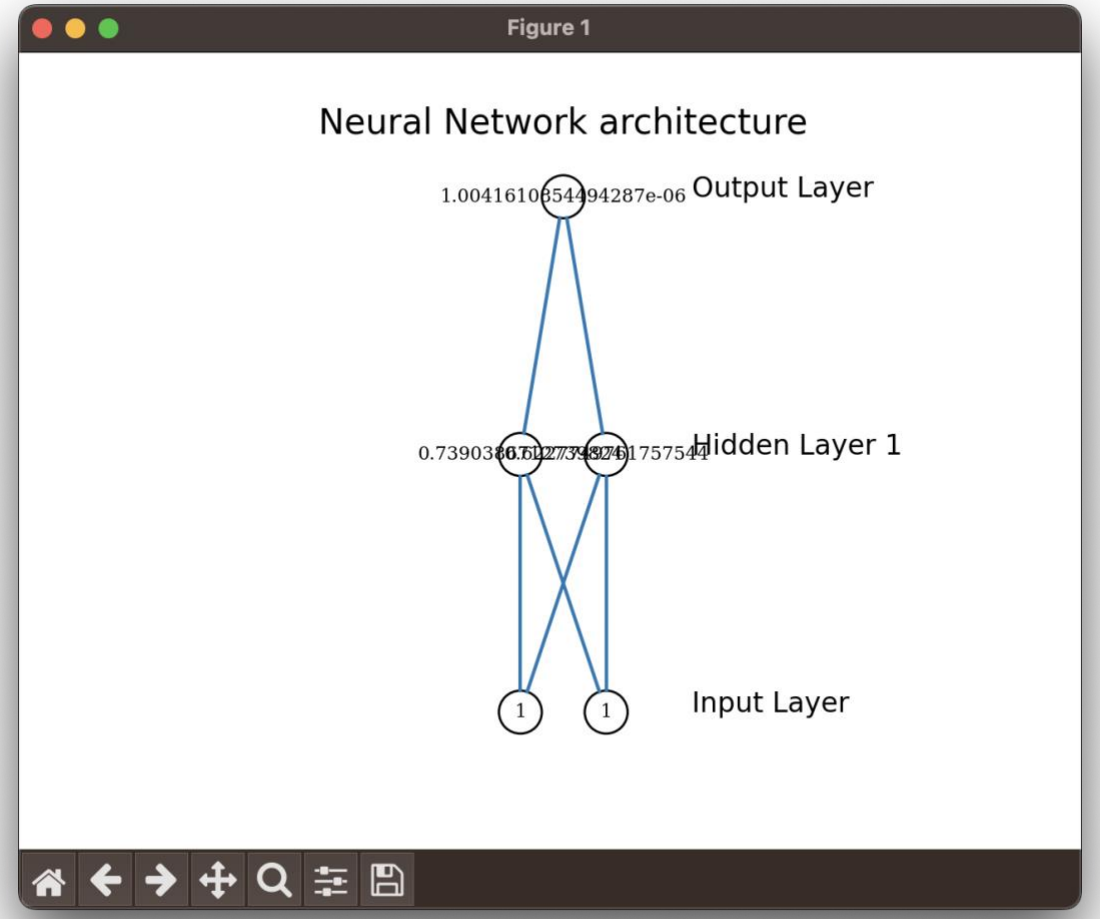
[1]
function feedforward_network(network, inputs):
each layer l in network
new_inputs ← []
each node n in layer
activation ← node['weights'][-1] *// Bias*
i from 0 to length(inputs) - 1
activation += node['weights'][i] * inputs[i]

node['output'] ← activation_function(activation)
new_inputs.append(node['output'])

inputs ← new_inputs

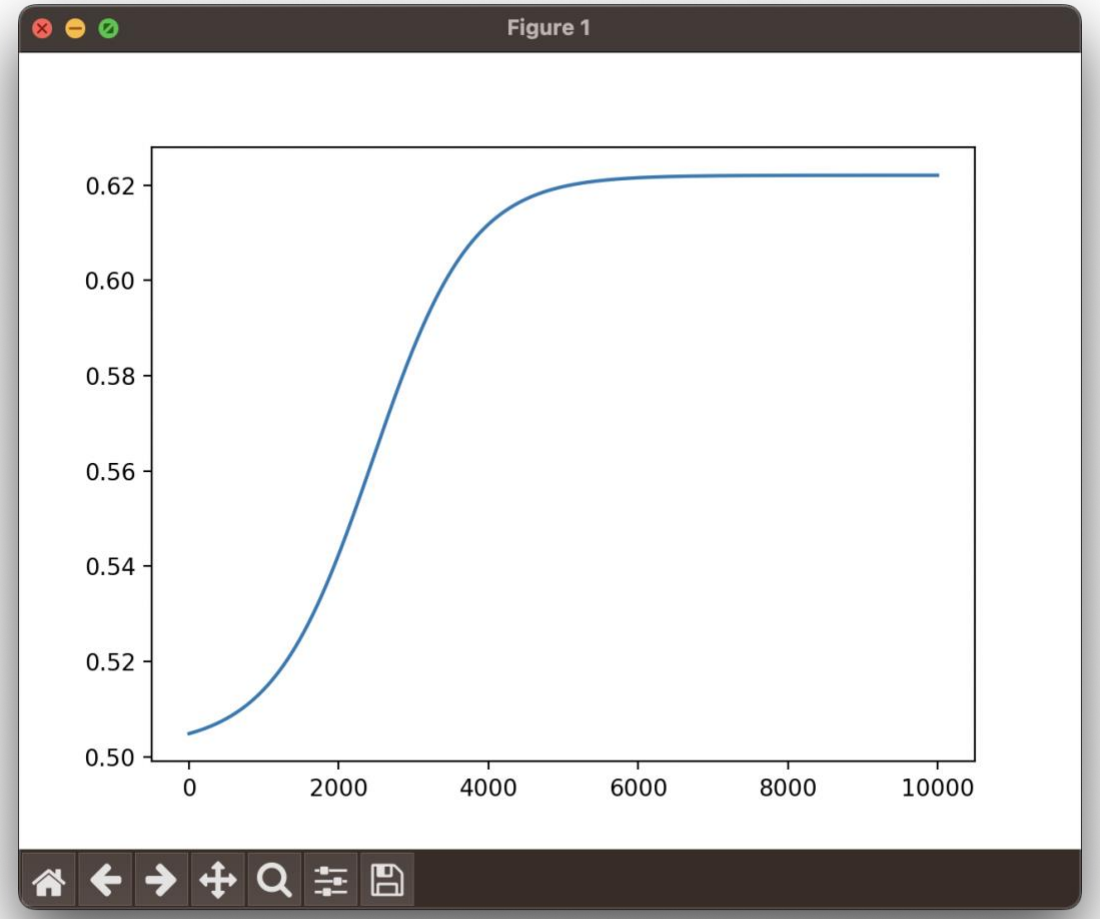
inputs

Results



Neural Network Topography

There are three layers.



Errors per iteration

{

"learningRate": 0.001,

"iterations": 10000,

"final mse": 0.6220643031721588,

"r value": -0.5478559975622943,

"neural network": {

```
"layers": [
 {
  "neurons": [
   {
    "bias": 1,
    "weights": [
     0.18144856003867058,
     0.5915557182523438
    ]
   },
   {
    "bias": 1,
    "weights": [
     0.5705911571075168,
     0.34895927025115503
    ]
   }
  ]
 },
 {
  "neurons": [
   {
```

```
        "bias": 0.6796229808028796,
        "weights": [
          0.6849640803187985
        ]
      },
      {
        "bias": 0.7192036709557639,
        "weights": [
          0.7431559071997575
        ]
      }
    ]
  },
  {
    "neurons": [
      {
        "bias": 0.9999977865596916,
        "weights": []
      }
    ]
  }
]
```

```
  }

}
```

Discussion

Challenges Faced and Solutions

- **Determining difference between stochastic and batch gradient descent:** I am not quite sure if I am doing what is expected for the stochastic and batch gradient descent functionality in the tests.
- **Finding the correct learning rate and number of iterations:** As I was trying to determine the best learning rate, it seemed not follow a linear cause and effect.
- **Neural network topology:** It might have not been the best topology to have only one hidden layer with two nodes.

Conclusion

In this work, we investigated the prediction of dependent values using a neural network, with a focus on independent values.

References

The assignment specification.