



Programming: The Backpropagation Algorithm

[Submit Assignment](#)

Due No Due Date **Points** 20 **Submitting** a file upload **File Types** py

In this programming project, you will extend existing code that builds and uses a simple neural network for solving a classification problem. The code, the data, and a tutorial can be found [here](https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/) (<https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/>). (A local copy of the code can be found [here](#) )

Start by executing the program as is, and make sure you understand its main functions. If you get errors with CSV file available on that page, use [this one](#)  instead.

As you can see, the network that's being built is not that deep, as it has just one hard-coded, hidden layer:

```
# Initialize a network
def initialize_network(n_inputs, n_hidden, n_outputs):
    network = list()
    hidden_layer = [{'weights':[random() for i in range(n_inputs + 1)]] for i in range(n_hidden)]
    network.append(hidden_layer)
    output_layer = [{'weights':[random() for i in range(n_hidden + 1)]] for i in range(n_outputs)]
    network.append(output_layer)
    return network
```

Your tasks is as follows:

Modify the existing program so to be able to build deep neural networks with any number of layers, and any number of neurons in those layers. The arguments to the `initialize_network` function should be as follows:

```
def initialize_network(n_inputs, hidden_layers, n_outputs)
```

where the parameter `hidden_layers` is a list of numbers, each defining the number of neurons in a layer. For example,

```
initialize_network(100, (50, 25, 10), 2)
```

builds a DNN with an input layer of 50 neurons, an output layer of 2 neurons, and 3 hidden layers in between: the first one has 50 neurons, the second 25, and the third 10.

Note that the exercise does not end with the change in the `initialize_network` function. That change will have consequences in other functions that traverse the network back and forth. You need to change all of those functions too.

Finally, run the same classification task using three different DNNs, and compare the results:

a) `hidden_layers=(5)`

b) `hidden_layers=(50, 25, 10)`

c) `hidden_layers=(50, 25, 10, 5)`

WHAT TO TURN IN:

Turn in your modified program configured for the network that produced the best results.

WHAT TO SHOW:

Besides turning in the code, you must show your program running to the TA. The TA will ask you questions about it, and may want to see it executing in different configurations.

Some Rubric			
Criteria	Ratings		Pts
Program runs correctly	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
Student shows understanding of the code	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
			Total Points: 20.0