Project-3 Documentation

Anudeep Reddy

December 10, 2018

1 Neural Networks

1.1 build_model() function

First, variables W1, W2, b1 & b2 are initialized with size according to the number of layers and size of input & random weights along with a dictionary 'model'. Next, I reshaped y into 2-D numpy array. Next, I've iterated number of times that 'num_passes' variable specified which is passed as an argument to this function.

Inside each iteration, forward and backward propagation is done. Layer-1 net value is stored in 'a' and activation (tanh) result in 'h'. Layer-2 net value is stored in 'z' and activation in 'y_pred'. Layer-1 & Layer-2 calculations are stored in the above mentioned variables (Forward propagation). Next, in backward propagation, derivatives are performed. 'dL_dy_pred' stores difference between y_pred and y(original label). 'dL_da' stores the derivative value of loss of y over a. 'dL_dW2' stores loss over weights W2. 'dL_db2' stores loss over bias b2. 'dL_dW1' stores loss over weights W1. 'dL_db1' stores loss over bias b1. This is done over the entire dataset in each iteration.

Next, variables W1,b1, W2 & b2 are updated through gradient descent where eta is given as 0.001. And these values are stored in 'model' dictionary with key same as variable names. Next, loss is printed (multiplied by 100) in percent in an 'if statement' only if 'print_loss' variable is set 'True' which is passed as an argument to this function. Finally, 'model' is returned.

1.2 calculate_loss() function

Model, X & y are passed as an arguments to this function where 'model' containing weights and bias, X with samples and y with labels. A forward propagation is carried out with same variables as above function and an extra step is done where 'y_pred' is passed to an cross_entropy loss function where the loss is stored in 'loss' variable and returned.

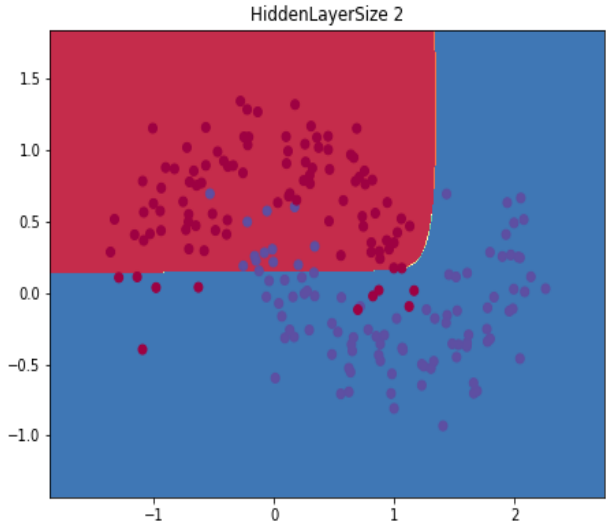
1.3 predict() function

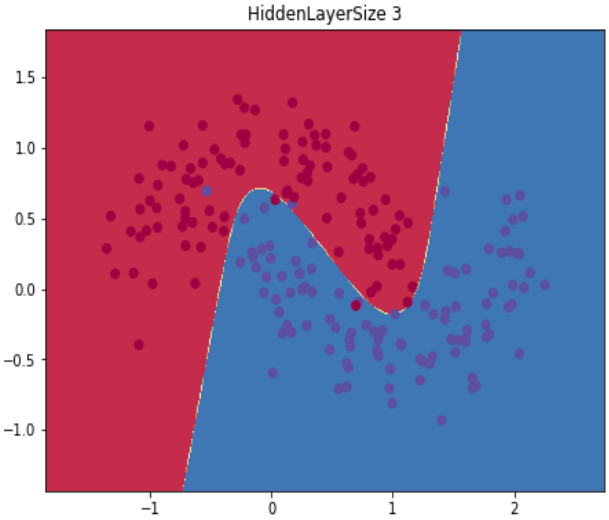
A 'model' and a sample (x) is passed as an arguments. w1, b1, w2 & b2 are extracted from the model and stored in their respective variables. Forward propagation is done and results are stored in same variable names as above. Finally 'y_pred' has the probability of the output class. And an argmax() is performed on 'y_pred' and the label is returned.

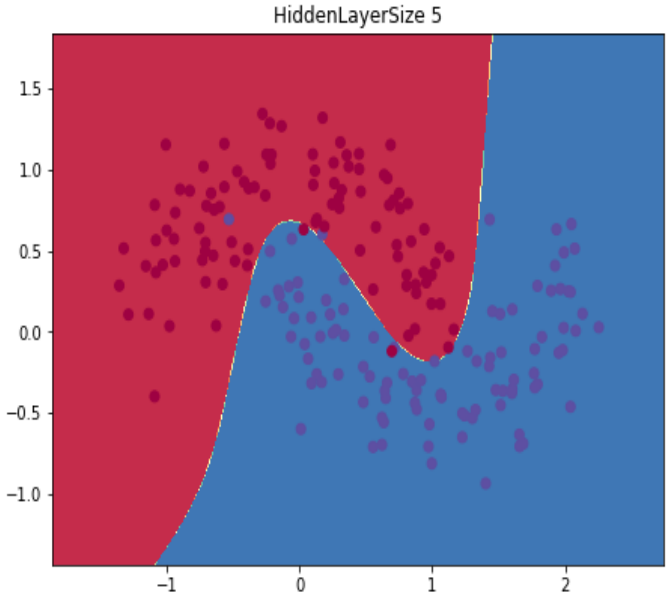
1.4 build_model_691() function

Everything is same as 2D predict function except y is reshaped as 3D output because input features are three.

1.5 Decision boundary for 2D







1.6 Decision boundary for 3D