PROJECT

Your first neural network

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW | CODE REVIEW | NOTES

Requires Changes

SHARE YOUR ACCOMPLISHMENT

1 SPECIFICATION REQUIRES CHANGES

Great work in general!
You've done a great job for your first neural net :)
Please take a look at my feedback, correct what I commented on and you'll be good to go

Good luck! I look forward to see the amazing things you build next :)

Code Functionality

✓	**All the code in the notebook runs in Python 3 without failing, and all unit tests pass.**
	Good job! All unit tests passed!

✓	**The sigmoid activation function is implemented correctly**
	Great job! Nice implementation of the sigmoid function!

Forward Pass

✓	**The input to the hidden layer is implemented correctly in both the train and run methods.**

Well done!

✓ **The output of the hidden layer is implemented correctly in both the `train` and `run` methods.**

Great job! Good use of the activation function!

✓ **The input to the output layer is implemented correctly in both the train and run methods.**

Well done!

✓ **The output of the network is implemented correctly in both the train and run methods.**

Great work! In this case the activation function is indeed f(x)=x

Backward Pass

✓ **The network output error is implemented correctly**

Good job!

✓ **Updates to both the weights are implemented correctly.**

Great work updating the weights!

Hyperparameters

🔁 **The number of epochs is chosen such the network is trained well enough to accurately make predictions but is not overfitting to the training data.**

Good effort here!
Please note that it's very hard to know whether you converged already or not because of the scale of the graph, your current loss is ~0.1 while your ymax is 1.5. Please change the ylim to be 0.5, to make sure we really reached convergence.
(Hint: you should reach a ~0.05 loss for training and ~0.15 for validation)

✓ **The number of hidden units is chosen such that the network is able to accurately predict the number of bike riders, is able to generalize, and is not overfitting.**

Good job! This number of hidden units satisfies the assignment requirements! :)
For future reference, here's a good guide on how to determine a good number of hidden units - https://www.quora.com/How-do-I-decide-the-number-of-nodes-in-a-hidden-layer-of-a-neural-network

✓ **The learning rate is chosen such that the network successfully converges, but is still time efficient.**

Good job here!
The effective learn rate in this case is self.lr/n_records, which in this case is equal to 0.5 / 1440 which is quite small, but still OK :)

RESUBMIT PROJECT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH