UDACITY

PROJECT

Your first neural network

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Requires Changes

2 SPECIFICATIONS REQUIRE CHANGES

Code Functionality

All the code in the notebook runs in Python 3 without failing, and all unit tests pass.

The sigmoid activation function is implemented correctly

Good, you can also use a lambda function as follow:
`self.activation_function = lambda x: 1/(1+np.exp(-x))`

Forward Pass

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

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

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

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

Good as this is a regression model, the output should be the raw input to the output layer.

Backward Pass

The network output error is implemented correctly

Updates to both the weights are implemented correctly.

Good !

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.

Almost good, when you look at your graph, you can see than both the training and validation still decreasing. Thus with this set of hyperparameters your should increase the number of epochs.
But I would just to modify the learning rate instead.

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.

Your number of hidden units is too low, here some advices:
The number of nodes is fairly open; if validation loss is low, it should meet specifications. You can recommend that it should probably be no more than twice the number of input units, and enough that the network can generalize, so probably at least 8. A good rule of thumb is the half way in between the number of input and output units.

There's a good answer here for how to decide the number of nodes in the hidden layer. 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, even if your learning rate is very low. You should know that below 0.001 the learning rate lead to a very low speed of learning, and above 0.1 to very high speed of learning, with the risk of not converging.
At around 0.05, it should be better and mainly faster.

RESUBMIT

DOWNLOAD PROJECT