UDACITY

PROJECT

Your first neural network

A part of the Deep Learning Nanodegree Foundation Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Good job overall implementing your first neural network and trying to fine tune the parameters!

You are precisely on point!

You correctly observed that the model predictions seem to fail at the end of December.

There reason is because there's not enough data for the network to learn about lower ridership during the holiday season.

The dataset only has two years of data (from January 1 2011 to December 31 2012) so the network saw the holiday period only once for training (the second holiday period in the 2-year-dataset is not used for training).

Congratulations and good luck with your Nanodegree!

Code Functionality

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

Well done running Python 3 without errors and all 5 unit tests are passing!

The sigmoid activation function is implemented correctly

Good job implementing the sigmoid activation function!

Forward Pass

The forward pass is correctly implemented for the network's training.

The run method correctly produces the desired regression output for the neural network.

Great job passing the input through a sigmoid here!

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.

Backward Pass

The network correctly implements the backward pass for each batch, correctly updating the weight change.

Well done here! The output error is the target minus the network output.

Updates to both the input-to-hidden and hidden-to-output weights are implemented correctly.

Impressive work - that was the hard part!

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.

Great! Looks like 2500 will do the job here.

Check out the plot of training and validation loss. They are still decreasing so you want to increase the number of epochs until the curves flatten and don't decrease anymore.

Experiment with the iterations hyperparameter to observe how the model behaves.
Be careful not to get the loss on the validation set to start increasing while the training set decreasing because then the network is overfit to the training data which is bad.

You should choose the number of iterations such that the loss on the training set is low and the loss on the validation set isn't increasing.

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! 16 hidden nodes will suffice because the minimum is 8 so that the network can generalize!

A good rule of thumb is the halfway in between the number of input and output units or less.

How many input units are there?

How many 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.

Great! Learning rate of 0.85 works in this scenario!

DOWNLOAD PROJECT

RETURN TO PATH