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

1 SPECIFICATION REQUIRES CHANGES

Well Done !!! This is a fantastic submission. You only have to tune your learning rate. However, when you adjust the learning rate, it is imperative that the number of epochs would have to be adjusted. Both go hand in hand. Please experiment extensively before the next submission.

Your observation about the model making poor predictions starting December 21 is correct. However, it is not only a matter of best learning curve that you would use a smaller learning rate. It does have an effect on the generalization of your model and its predictive abilities. Although it seems as if it has similar performance on the test set, if measured quantitatively, it might have larger error over the last 10 days. Also, if you are going to use a model to predict the hourly rate on a daily basis, it is better to choose a model that overpredicts or underpredicts slightly even if this model has a similar or slightly larger overall error as compared to the other model. Also, why do you think the actual bike ridership demand decreases during this period? Is it a recurring pattern every year? If it is, why is the model not able to account for the same? What can you do to improve the prediction?

All the best for your next submission. Keep learning !!!

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 Job!! If you want to learn about other kinds of activation functions, please refer to this.

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 Job!!! The forward pass has been implemented correctly. Please note that we are using an identity function as an activation function for the final output layer as the task is a regression one.

Backward Pass

The network output error is implemented correctly

Updates to both the weights are implemented correctly.

Well Done !!! The backward pass has been implemented correctly.

However, I have a few suggestions to make:

- Always try to use explicit matrix multiplication instead of implicit one using multiplication operator. The multiplication operator does work here as numpy uses broadcasting rules to achieve the same. However, sometimes, the results might not be as expected. You can learn more about broadcasting here.
- You are using slicing using `None` to reshape the array. Instead, use `np.reshape` or `np.newaxis` to enhance the readability of the code.

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.

The network has converged successfully. So, the number of epochs chosen is optimal.

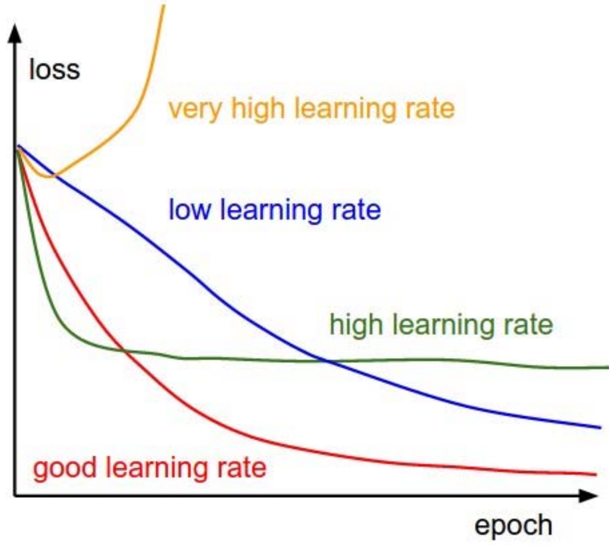
The number of hidden units is chosen such that the network is able to accurately predict the number of bike riders, is able to generalize, and is not overfitting.

You have chosen a good number of hidden units. Well Done !!!

Anything between 8 and 20 works quite well for this problem. By now, you must have understood that the number of hidden units to be used in a network is fairly open and there is no mathematical formula that provides the exact size of the hidden layer. However, there have been empirical studies that suggest some heuristics that can be used to help make this decision. One of the most common heuristics is to use the average of the input and output nodes. You can take a look at the whole discussion.

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

The learning rate that you are using is very high. This can be seen from the learning curve where the error decreases rapidly initially and then flattens out with a neck. Below is a cartoon representation of different learning rates taken from cs231n class notes.



For this hidden layer size, 0.2 - 0.3 learning rate should work fine. It would not take longer and will have similar performance.

RESUBMIT

DOWNLOAD PROJECT

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH

Student FAQ