CS57300 Spring 2018 Homework — Homework 3 — Last update: April 6, 2018

https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/index.html Due: April 16th 6:00am, 2018

Instructions and Policy: Each student should write up their own solutions independently, no copying of any form is allowed. You MUST to indicate the names of the people you discussed a problem with; ideally you should discuss with no more than two other people.
YOU MUST INCLUDE YOUR NAME IN THE HOMEWORK
You need to submit your answer in PDF. LaTeX is typesetting is encouraged but not required. Please write clearly and concisely - clarity and brevity will be rewarded. Refer to known facts as necessary.

The code of Homework Y, question QX, item i should be named HWY_QX_i.py. Your code is REQUIRED to run on Python 3 at scholar.rcac.purdue.edu. The TAs will help you with the use of the scholar cluster. If the name of the executable is incorrect, it wont be graded. Please make sure you didnt use any library/source explicitly forbidden to use. If such library/source code is used, you will get 0 pt for the coding part of the assignment. If your code doesnt run on scholar.rcac.purdue.edu, then even if it compiles in another computer, your code will still be considered not-running and the respective part of the assignment will receive 0 pt.

Q0 (0pts correct answer, -1,000pts incorrect answer: (0,-1,000) pts): A correct answer to the following questions is worth 0pts. An incorrect answer is worth -1,000pts, which carries over to other homeworks and exams, and can result in an F grade in the course.

(1) Student interaction with other students / individuals:

(a) I have copied part of my homework from another student or another person (plagiarism).

(b) Yes, I discussed the homework with another person but came up with my own answers. Their name(s) is (are) ____

(c) No, I did not discuss the homework with anyone

(2) On using online resources:

(a) I have copied one of my answers directly from a website (plagiarism).

(b) I have used online resources to help me answer this question, but I came up with my own answers (you are allowed to use online resources as long as the answer is your own). Here is a list of the websites I have used in this homework:
____

(c) I have not used any online resources except the ones provided in the course website.

Q1 (5 pts): Deep Learning

In Lecture 15 we have implemented a Feedforward neural network (a.k.a., Multilayer Perceptron) with one hidden layer and without biases in any of the layers. Training uses Stochastic Gradient Ascent.
The code is at `https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/lectures/MiniBatch_GD_FeedForward_ReLU.html`
and the iPython notebook is at
`https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/lectures/MiniBatch_GD_FeedForward_ReLU.ipynb.`

In this part of the homework, using only the libraries used in the original code, you need to add biases for all neurons and ONE extra fully connected hidden layer to the neural network (same number of units). The bias must be added for **all neurons**, including the existing ones. As in the original code, the learning has to be performed using only numpy array and matrices and Stochastic Gradient Ascent.

Note: If you want to run it on the scholar.rcac.purdue.edu cluster, you will need the command
`module load anaconda/5.0.0-py36`
to load python3 + numpy.

1. (1pt total, PDF) Describe the new equations of your neural network as we did in sections (0.5pt) "Define the forward pass" and (0.5pt) "Define backpropagation" at
   `https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/lectures/MiniBatch_GD_FeedForward_ReLU.html.`

2. (1pt, PDF + CODE) Report the accuracy of your implementation after 10 mini-batches (at PDF) and the new test scatter plot in line 11 (at PDF). Comment these results. Give one reason why the model is more accurate now.

   Code output (REQUIRED, not other output in your code):

   ```
   Iteration 0
   Iteration 1
   Iteration 2
   Iteration 3
   Iteration 4
   Iteration 5
   Iteration 6
   Iteration 7
   Iteration 8
   Iteration 9
   Accuracy after 10 iterations with new neural network: (your accuracy)
   ```

   Replacing (your accuracy) by whatever you get with your new algorithm.

   Hint: For each neuron: If the neuron activation function is $\sigma()$ and the input to the neuron is z, output of the neuron is $\sigma(z)$. The bias is an extra parameter b, such that the new neuron output is $\sigma(z + b)$.

   (Python Source) Turn in your python source as HW3_Q1_2.py (this is your code with all biases and the new layer) using turn-in (see instructions on page 1).

3. (1pt total, PDF) We now compare the previous code (with the extra layer and bias) with the original code. Report the average and standard deviation classification accuracy of 20 runs with 100 iterations each over validation data. Sample new training (3000 points) and validation (100 points) data points at each run.

   (0.2pt) Plot these 20 accuracy values using a boxplot.

(0.8pt) Is there a significant difference to merit using the test data? Formalize the null and alternative hypotheses. Explain using paired t-tests to support your decision. Quantify the confidence in your answer.

Hint: Example code to plot boxplots in python:
https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/hw/boxplot.html

4. (1pt) Divide the original training data into 3000 examples for training, remaining for validation. Using your code (the one with the bias + extra layer), plot the learning curves (training and validation accuracy) for training data sizes of 100, 500, 1000, 2000, and 3000. The learning curves should use the zero-one loss (accuracy).

   Hint: The plot has 100, 500, 1000, 2000, and 3000 in the horizontal axis. And two curves: the accuracy over the training data, and the accuracy over the test data.

5. (1pt) In the iPython notebook used in lecture
   `https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/lectures/MiniBatch_GD_FeedForward_ReLU.html`
   "Define backpropagation", we have seen how the gradient of the weight matrix $\mathbf{W}$ is computed by averaging the gradient computed for each training example in the minibatch. Clearly, however, if the gradient of a parameter is zero, this parameter will not change in the next gradient step. **Consider the first fully connected layer**. Let the training dataset $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and the minibatches be of size N/K, where K is a constant (assuming N is a multiple of K). Define the conditions under which the gradient of a ReLU unit $\mathbf{h}(r)$ of the first hidden layer is zero for all minibatches. When this happens, we say that the neuron has died.
   Hints: (1) Pay attention to the fact that the gradient of a parameter is the average of the gradient over all training examples of the minibatch. (2) Also note that a minibatch is a sampled from the training data. (3) The values of K and N are not relevant to your answer.

Q2 (5 pts): **Cross-validation**

Here we will use a neural network model using PyTorch to determine whether a customer is likely to repay their loan request or not. You can download the data from `https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/data/data.zip`.

1. (1pt, PDF + CODE) Using the skeleton code at `https://www.cs.purdue.edu/homes/ribeirob/courses/Spring2018/data/hw3_Q2.py` , fix the learning rate and batch size and vary the number of training examples: $[10\%, 30\%, 50\%, 70\%, 90\%, 100\%]$ of the whole training set. Plot learning curves using the AUC scores.

   Hint: You are allowed to use a library to compute the AUC score, but you can also use the code above.

   (Python Source) Turn in your python source as HW3_Q2_1.py (this is your code with all biases and the new layer) using turn-in (see instructions on page 1).

2. (1pt, PDF) Now, we compare the neural network with two other models we have seen in class: a logistic regression with L2 regularization and boosted decision trees (adjust their hyper-parameters for better performance). Plot the learning curves of these two models.
   Hint: You can using xgboost and sklearn package to build your logistic and boosted decision tree models.

3. (1pt, PDF) Using 10-fold cross validation, compare the models `neural network`, `logistic regression` and `boosted decision trees`. Is the boosted decistion tree the best model? Formalize the null and alternative hypotheses. Explain using paired t-tests to support your decision, correcting for the fact that we are testing multiple hypotheses. Quantify the confidence in your answer.

4. (1pt full, PDF + CODE) Generally, to train a neural network model, we will perform hyper-parameter tuning of the model or the optimization parameters. The procedure described in this question can be thought of as a discrete-optimization-within-validation approach for hyper-parameter tuning.

   When training the neural network with the 9 "training" folds of your 10-fold cross-validation we will perform a discrete optimization of the training hyper-parameters. The procedure works as follows: call the 9-folds used for training the full_training data. Split full_training into training' (80%) and validation' (20%). Now, with training' and validation', find the best learning rate and batch size over the following grid: learning rates: $\{10^{-2}, 10^{-3}\}$ and the batch sizes $\{100, 1000\}$.

   (0.2pt) Select the best performing neural network model to be compared with the other two models. Report the hyper-parameters of choice in each fold.

   (0.8pt) Detail the complete test procedure between the `neural network`, `logistic regression` and `boosted decision trees`. We are trying to tell whether the neural network wins. Formalize the null and alternative hypotheses. Does this procedure change the hypothesis test?

   Hint: Note that inner-loop validation (training' + validation') can be seeing as single discrete+continuous hybrid optimization algorithm.

   (Python Source) Turn in your python source as HW3_Q2_4.py (this is your code with all biases and the new layer) using turn-in (see instructions on page 1).

5. (1pt, PDF) Consider the above procedure where instead of doing our discrete-optimization-within-validation, we are comparing `logistic regression` and `boosted decision trees` with 4 distinct trained neural networks, one for each combination of the hyper-parameters. We are trying to tell whether the neural network wins. Formalize the null and alternative hypotheses. Does this procedure

change the hypothesis test from question Q2(4)? Describe how it changes and the potential issues.
Hint: Note that this is often what researchers and practitioners do when trying to improve their methods “manually”. We haven’t seen how to test this type of hypothesis but you should still be able to write it down and identify the issue.