CSCI 447 — Machine Learning

Project #3

Assigned: October 7, 2020
Project Due: October 30, 2020

This assignment requires you to implement several neural network training algorithms to perform classification and regression on several data sets from the UCI Machine Learning Repository. In addition to implementing and testing the algorithms, you are required to write a research paper describing the results of your experiments.

For this assignment, you will use three classification datasets and three regression data sets that you will download from the UCI Machine Learning Repository, namely:

1. Breast Cancer [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29`

   This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

2. Glass [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Glass+Identification`

   The study of classification of types of glass was motivated by criminological investigation.

3. Soybean (small) [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Soybean+%28Small%29`

   A small subset of the original soybean database.

4. Abalone [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Abalone`

   Predicting the age of abalone from physical measurements.

5. Computer Hardware [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Computer+Hardware`

   The estimated relative performance values were estimated by the authors using a linear regression method. The gives you a chance to see how well you can replicate the results with these two models.

6. Forest Fires [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Forest+Fires`

   This is a difficult regression task, where the aim is to predict the burned area of forest fires, in the northeast region of Portugal, by using meteorological and other data .

When using these data sets, be careful of some issues.

1. Some of the data sets have missing attribute values, which is usually indicated by "?". When this occurs in low numbers, you may simply edit the corresponding data items out of the data sets. For more occurrences, you should do some kind of "data imputation" where, basically, you generate a value of some kind. A naïve approach is to impute the missing value with a random number or the attribute's mean (or median). A better approach is to sample according to the conditional probability of the values occurring, given the underlying class for that example. The choice of strategy is yours, but be sure to document your choice.

2. For networks with multiple outputs, you should use what is called a "multi-net." This is where you train a single network with multiple outputs, one for each outcome you need to predict. This is distinct from training separate networks for each output, which is effectively what you have done with prior linear models.

3. The attributes should not require any special handling with either model. It is highly recommended that you normalize numerical attributes first to be in the range -1 to $+1$ or by using z-score normalization (i.e., $z = (x - \mu)/\sigma$) and apply the inputs directly.

4. You will need to determine the number of hidden nodes per layer via a tuning process. Note that you can use a rule of thumb that you have fewer hidden nodes than inputs, but be careful that this may not always work. In addition, when you go to two hidden layers, you should not need as many hidden nodes per layer as you needed in the one hidden layer case.

Your assignment consists of the following steps:

1. Download the six (6) data sets from the UCI Machine Learning repository. You can download from the URLs above or from Brightspace. You can also find this repository at `http://archive.ics.uci.edu/ml/`

2. Pre-process each data set as necessary to handle missing data and normalize as needed.

3. Implement a multi-layer feedforward network with backpropagation learning capable of training a network with an arbitrary number of inputs, an arbitrary number of hidden layers, an arbitrary number of hidden nodes by layer, and an arbitrary number of outputs. In other words, the number of inputs, hidden layers, hidden units by layer, and outputs should be furnished as inputs to your program. Be able to specify whether a node uses a linear activation function for regression or a sigmoidal activation function for classification (you may choose between logistic or hyperbolic tangent). Do not use ReLU or other similar activation functions. Remember that these choices affect the update rules because of having different derivatives. Implement learning such that momentum is provided as an option.

4. Develop a hypothesis focusing on convergence rate and final performance of each of the chosen algorithms for each of the various problems.

5. Test the MLP algorithm with 0, 1, and 2 hidden layers.

6. Write a very brief paper summarizing the results of your experiments. Your paper is required to be at least 5 pages and no more than 10 pages using the JMLR format You can find templates for this format at `http://www.jmlr.org/format/format.html`. The format is also available within Overleaf. Make sure you explain the experimental setup, the tuning process, and the final parameters used for each algorithm.

7. Your paper should contain the following elements:

   (a) Title and author name

   (b) Problem statement, including hypothesis

   (c) Description of your experimental approach

   (d) Presentation of the results of your experiments

   (e) A discussion of the behavior of your algorithms, combined with any conclusions you can draw

   (f) Summary

   (g) References (Only required if you use a resource other than the course content.)

8. Create a video that is no longer than 5 minutes long demonstrating the functioning of your code. For the video, the following constitute minimal requirements that must be satisfied:

   - The video is to be no longer than 5 minutes long.
   - The video should be provided in mp4 format. Alternatively, it can be uploaded to a streaming service such as YouTube with a link provided.
   - Fast forwarding is permitted through long computational cycles. Fast forwarding is *not permitted* whenever there is a voice-over or when results are being presented.
   - Be sure to provide verbal commentary or explanation on all of the elements you are demonstrating.

- Provide sample outputs from one test set showing performance on your networks. Show results for each of the cases where you have no hidden layers, one hidden layer, and two hidden layers.
- Show a sample model for the smallest of each of your neural network types. This will consist of showing the weight matrices with the inputs/outputs of the layer labeled in some way.
- Demonstrate and explain how an example is propagated through a two hidden layer network. Be sure to show the activations at each layer being calculated correctly.
- Demonstrate the weight updates occurring on a two-layer network for each of the layers.
- Demonstrate the gradient calculation at the output for any one of your networks.
- Show the average performance over the ten folds for one of the data sets for each of the types of networks.

9. Submit your fully documented code with the outputs from running your programs, your video, and your paper.