Farhad Kamangar kamangar@uta.edu

Assignment 02 (Due date: Oct. 6, 2019)

Neural Networks

Assignment 02

Due date: Oct. 6, 2019

The purpose of the this assignment is to:

- Practice with different learning rules and use them to classify input vectors. Note that the network in this assignment may also be used for regression but you will only use it for classification.
- Practice with batch processing.

To begin this assignment download the Kamangar-02.zip and unzip it to your computer.

Implementing Model and Training

- In this assignment you will implement a single layer neural network and use different learning rules to train the network.
- The single layer neural network in this assignment will be used as a classifier. This means that you will consider the largest output of the network as the correct class.
- Your model weights **should include the bias**.
- Your code should be vectorized using numpy.
- There are two files in the unzipped directory. The "hebbian.py" file is where you are to implement the network model and all necessary helper functions. The "test_hebbian.py" file includes the unit test modules.
- You DO NOT need to rename these two files according the submission guidelines. Just modify the first four lines of these files according to the assignment submission guidelines.
- The "Hebbian" class is 'stubbed out' and your task is to implement the unimplemented functionality **within the given structure**.
- **DO NOT** alter the structure that is given. You may introduce additional helper methods, but do not alter function names or argument structures.
- The API structure of the given file is inspired by (but not an exact copy of) modules such as **scikit-learn** and **keras** that are very commonly used in practice.
- The comments and docstrings provide additional information that should help with your implementation.
- Methods that you must implement:
  - `_initialize_weights` - This is where you create the weights for your model (bias is included in the weights).
  - `initialize_all_weights_to_zeros` - This method initializes all weights to zero (bias is included in the weights).
  - `predict` - Given array of inputs this method predicts array of corresponding outputs.

- `print_weights` - This method prints the weight matrix (bias is included in the weights).
- `train` - Given array of inputs, desired outputs as class indexes, and other parameters, this method adjusts the weights using the appropriate learning rule. Note that this method should implement batch training.
- `calculate_percent_error` - Given array of inputs and desired outputs as class indexes this method calculates percent error.
- `calculate_confusion_matrix` - Given array of inputs and desired outputs as class indexes array this method calculates the confusion matrix

Notes:

- There are three different transfer functions that you should implement
- There are three different learning method that you should implement.
- The test_hebbian.py file includes a very minimal set of unit tests for the hebbian.py part of the assignment. Part of the assignment grade will be based on your code passing these tests (and some other unspecified tests)
- **You may modify the "test_hebbian.py" to include more tests. You may add additional tests to help you during development of your code. The changes that may make to the "test_hebbian.py" file will not be graded.**
- You may run these tests using the command: `py.test --verbose test_hebbian.py`
- If the last batch in an epoch does not have enough samples, you should use a smaller batch which includes all the left over samples.

The following is roughly what your output should look like if all tests pass

```
collected 4 items

test_hebbian.py::test_weight_dimension PASSED        [ 25%]

test_hebbian.py::test_weight_initialization PASSED   [ 50%]

test_hebbian.py::test_predict PASSED                 [ 75%]

test_hebbian.py::test_error_calculation PASSED       [100%]


========================== 4 passed in 0.04 seconds
===========================
```

Grading Criteria

- Passing Unit Tests - 80 points Note: Not all tests are given.
- Qualitative Evaluation - 20 points (Grader may examine your code and subjectively award as many as 20 points.)

Submission Guidelines

- Modify the first four lines of the two Python files according to the assignment submission guidelines.

```
# Your name (last-name, first-name)
# Your student ID (100x-xxx-xxx)
# Date of submission (yyyy-mm-dd)
# Assignment-nn-kk
```

- You **DO NOT** need to rename the two Python files. The name of the two Python files should remain as "hebbian.py" and "test_hebbian.py" .
- Create a directory and name it according to the submission guidelines and include your files in that directory.
- Zip the directory and upload it to Canvas according to the submission guidelines.

Top