Assignment 2: Binary classifiers using various loss functions and regularisers

In the uploaded doc there are 2 files main.py and utils.py

Description of **main.py**

The description of the classes and functions are described below.

(1) class DataLoader (Modelled for reading csv files. This file reads the csvfile and make batches from the datafile)
Initialisation parameters- path_to_file,batch_size
Functions
(a) __len__ () - returns number of batches
(b) __getitem() - returns a tuple of input,output for the current batch indexed by i. The shape of the train feature matrix for a batch is batch_size x number_of_features, and the target is of the size batch_size x 1

(2) classify() - Given a vector of weight and a batch of the input returns the w^Tx.
(3) get_objective_function() - Returns a pointer to the loss function according to the combination of the input.
(4) get_gradient_function() - Returns a pointer to the gradient function according to the combination of the input.
(5) train()- The input to the function is an object from the DataLoader class(used to read a train file). Additionally it also takes as input the loss_type, regularisation type and the value of C.
(6) test() - Given a set of input feature and weights, it returns the classified labels.
(7) get_data() - For reading the test file.
(8) main() - This function initialises the DataLoader object for the train file. The arguments passed from the terminal are the loss_type,regulariser,value of C, train_file,test_file.

Description of **utils.py**

You are to required to implement this module. This module computes the necessary loss and regulariser values. Also it should contain the gradients of the loss function and various regularisers

Additional Description- The bias term can be appended along with the weight vector, as its last term and to accommodate that you need to append a ‘1’ in the input feature vector.

A command line argument to run the code is

python2 main.py --loss logistic_loss --batch_size 64 --train-data train.csv --test-data test.csv --loss-weight 1

Note:

You can change any function, in “main.py” , modify them or use them however you like, or rewrite the entire thing from scratch. The “main.py” is just given for reference. As long as your code takes the following command line arguments and generates an “output.csv”

The “main.py” is given for reference just for modularity and to make the code easy to understand.

Please go through scipy.optimise function and understand the necessary arguments that needs to be passed, to the optimise function.

Do not change the function declarations in utils.py. Even if you write a code in other language, the utils.py section should be there with all the functions.

Implement all the functions in “utils.py” , please do not change their declaration (the function name, the accepted parameters and the returned parameters)

All the loss functions take input as (target,output) and returns the loss.
Similarly all grad functions take input as weights,batch_X,batch_Y and predicted outputs
The regularisers only take weight as their parameters
Suggestions:

So that the model can train better you can normalise the data. It is encouraged to see how the results vary for different batch sizes (tweak the batch size parameters to get better convergence). Observe the difference between updating gradients per sample vs updating gradients for a batch vs updating gradients from the entire training set. Try out various values of C , and see the impact on the validation set. It will also be interesting to see how a regularised vs non- regularised model performs on the validation set.