CENG 501 - Deep Learning - Fall 2022

Instructor: Emre Akbaş

Oct 6, 2022

Assignment 1

Due date is Oct 16, 23:55. Submit online via ODTUClass.

IMPORTANT NOTE: What you submit should be **solely your own work**! By submitting your solution to this assignment, you agree to the METU Code of Honor:

> "As reliable, responsible and honorable individuals, all members of Middle East Technical University embrace only the success and recognition they deserve, and act with integrity in the use, evaluation and presentation of facts, data and documents."

Good luck!

Part 1 [90 pts] Implementation, training and testing of a classifier

In this part, you are required to

- implement,
- train,
- test and
- report the performance of

a custom two-class classification method. In the following, we are going to refer to this method as the "classifier".

The classification model of the classifier is as follows

$$f(\mathbf{x}; \mathbf{w}, b) = \sigma(\mathbf{w}^T \mathbf{x} + b)$$

where $\mathbf{x}$ is a D dimensional input vector, $\mathbf{w}$ is a D dimensional weight vector, b is a scalar, and $\sigma(\cdot)$ is the following function:

$$\sigma(t) = \frac{2}{1 + e^{-t}} - 1.$$

The classification rule for the classifier, i.e. the estimated label $\hat{y}$ for a given input vector $\mathbf{x}$ is

$$\hat{y} = \begin{cases} +1, & \text{if } f(\mathbf{x}) \geq 0 \\ -1, & \text{otherwise} \end{cases}$$

where 1 is considered to be the positive label and -1 is the negative label.

You are provided with two Python files: `main.py` and `hw1_sol.py`. You will implement your solution in `hw1_sol.py`. You should not change `main.py`.

Q1.1 [15 pts] Implement the function $f(\cdot)$ inside `hw1_sol.py`. This function should work with an arbitrary number of input examples. Follow the instructions inside `hw1_sol.py`.

Q1.2 [25 pts] Consider the L_2 loss function to train the classifier:

$$L(\mathbf{w}, b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i; \mathbf{w}, b))^2$$

Derive the partial derivatives of the loss function with respect to $\mathbf{w}$ and b. Show the main steps of your derivation.

$\frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b) = ?$

$\frac{\partial}{\partial b} L(\mathbf{w}, b) = ?$

Q1.3 [35 pts] Implement the `l2loss()` and `minimize_l2loss()` functions in `hw1_sol.py` by following the instructions given in `hw1_sol.py`. The function `l2loss()` should return three items: the loss value given the current $\mathbf{w}$, the gradient with respect to $\mathbf{w}$ and the gradient with respect to b. The function `minimize_l2loss()` should implement the gradient descent method with a constant stepsize.

In gradient descent, you update the parameters of the model in the negative direction of the gradient of the loss function. For example, this is the update rule for w:

$$\mathbf{w}^{\text{next}} = \mathbf{w}^{\text{current}} - \eta \nabla_{\mathbf{w}} L$$

In order to test your implementation, a main driver file, i.e. `main.py` is provided. If your implementation is correct, then running `main.py` with appropriate arguments should train and test your classifier successfully.

Run the following two calls:

```
python main.py data/linear_data_train.pickle.gz data/linear_data_test.pickle.gz 100 1
```

```
python main.py data/mnist_01_train.pickle.gz data/mnist_01_test.pickle.gz 100 1
```

Each call will show two plots, one for the loss-vs-iterations and the other for visualizing the learned weights. You will be submitting these plots.

Q1.4 [15 pts] Is the L_2 loss function given in **Q1.2** convex in terms of the parameters of the classifier, i.e., $\mathbf{w}$ and b? Why / why not? Explain.

Part 2 [10 pts] Understanding cross-validation

Q2.1 [5 pts] Suppose you are given a classifier training function `f = train(S,C)` which takes a training set `S` and a parameter value `C` as input and returns the trained classifier `f`.

Write a pseudo-code for k-fold cross-validation to find the best value for parameter `C`.

Q2.2 [5 pts] Suppose you are given a dataset D and a classification model M with its train and predict functions. You are asked to estimate the future performance of the classification model M on *unseen* data that come from the same population as did set D. How would you use cross-validation in such a case? Show your idea by writing a pseudo-code.

What to submit?

- Your `hw1_sol.py` file.
- A PDF file that contains:
  - The `losses.png` and `weights.png` for each of the two calls (a total of 4 plots) in Q1.3.
  - Your written answers to **Q1.2**, **Q1.4** and **Part 2** (Submit a digital copy please. You can either use a word processor or scan your answers).