Deep learning & applications

Practice#2-1

Tae Hyun Kim

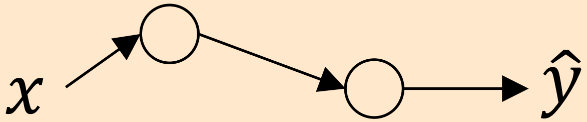
Reference

- Python + Numpy tutorial
  - http://cs231n.github.io/python-numpy-tutorial

Task2: binary classification using 2-layered net (cross-entropy loss)

(train/test samples acquired from practice#1-2)
Input: 1-dim vector, $\boldsymbol{x}$
Output: label of the input, $\mathbf{y} \in \{0,1\}$

Pseudo code #you can use numpy module!



Step 1. Load generated ‘m’ train samples, ‘n’ test samples in **practice#1-2**

Step 2. Update W = $[w_1], b$ with ‘m’ samples for 5000 (=K) iterations: #K updates with the grad descent (Thr. = 0.5)

- **Step 2-1.** print W, b every 500 iterations
- **Step 2-2.** calculate the cost on the ‘m’ train samples!
- **Step 2-3.** calculate the cost with the ‘n’ test samples!
- **Step 2-4.** print accuracy for the ‘m’ train samples! (display the number of correctly predicted outputs/m*100)
- **Step 2-5.** print accuracy with the ‘n’ test samples! (display the number of correctly predicted outputs/n*100)

Report

- You need to submit a short report; (Due: 4/14, 3pm)
  - Format: studentid_name.pdf + single source file (.py or .ipynb)
  - Should not be more than 2 pages
  - Should include
    - Estimated unknown function parameters W & b
    - Empirically determined (best) hyper parameter, α
    - Accuracy (fill in the blanks in the tables below and add them to the report)
    - Discussion (what you've learned in this experiment)

	m=10, n=1000, K=5000	m= 100, n=1000, K=5000	m=10000, n=1000, K=5000
Accuracy (with 'm' train samples)			
Accuracy (with 'n' test samples)			

	m=10000, n=1000, K=10	m=10000, n=1000, K=100	m=10000, n=1000, K=5000
Accuracy (with 'm' train set)			
Accuracy (with 'n' test samples)			