Deep learning & applications

Practice#1-0

Tae Hyun Kim

Input: 2-dim vector, $x = \{x_1, x_2\}$
Output: label of the input, $y \in \{0,1\}$

Pseudo code

Step 1. Generate 10000(=m) train samples:

```
x1_train=[], x2_train=[], y_train=[]

for i in range(m):

    x1_train.append(random.uniform(-10, 10))

    x2_train.append(random.uniform(-10, 10))

    if x1_train[-1] + x2_train[-1] > 0:

        y_train.append(1)

    else:

        y_train.append(0)
```

Input: 2-dim vector, $x = \{x_1, x_2\}$
Output: label of the input, **y** $\in$ {0,1}

Step 2. Initialize W $= [w_1, w_2]$ and b (small random values? large random values? zeros?)

Step 3. Determine the learning rate α (let's tart with 1/100)

Step 4. Update W $= [w_1, w_2], b$ with 'm' samples for 5000 (=K) iterations: #K updates with the grad descent

 Step 3-1. Print W, b every 500 iterations

 Step 3-2. Calculate the cost on the 'm' train samples!

 Step 3-3. Assume that predicted output is 1 if $\tilde{y} > \mathbf{0.5}$, other wise 0. Then, print accuracy for the 'm' train samples! (display the number of correctly predicted outputs/m*100)