Towson University:

Department of Computer and Information Sciences.

COSC 750: NEURAL NETWORKS AND DEEP LEARNING

Title: Module 1 Project – Bike Sharing.

By:

Abubakar Musa 088112

Almustapha Wakili – 0815154662

**PART A1: SOLUTIONS**

A1-1) (B) b.shape=(1, 4, 5) c.shape=(4, 5)

A1-2) (A) torch.Size([60])

A1-3) (B) torch.Size([1, 3, 4, 5])

A1-4) (C) torch.Size([1, 1, 3, 4, 5])

A1-5) (D) torch.Size([3, 1, 4, 5])

**PART A2: SOLUTIONS**

A)

w0 > 0

w0 + w1 + w2 > 0

w0 + w1 < 0

B)

Given that the scale is arbitrary, we can bigin with w0 = 1. And the Third equation says 1 + w1 < 0, Thus we to set w1 = -2.

Similarly, the second equation indicates 1 - 2 + w2 > 0,

Thus we select w2 = 2.

Therefore, a feasible solution would be w0 = 1, w1 = -2, and w2 = 2.

**PART A3: SOLUTIONS**

**∂J∂h​λi=L(w)+∂w∂ww2​**

∂J∂h​λi2=(σ(wx+b)−t)+∂w2w2​

iλ∂hi​∂h​∂h2​∂h1​​=(σ(wx+b)−t)+2∂w∂w2​∂ww2​

=(σ(wx+b)−t)(σ(wx+b)−t)+λw∂w∂w​

=(σ(wx+b)−t)(σ′(wx+b))[wx+b]+λw

=(σ(wx+b)−t)(σ′(wx+b))x+λw

**PART A4: SOLUTIONS**

**A:**

x1

θ

w1

y L

w2

x2 t

**B:**

dL/dL = L = 1

dL/dy = y = L \* (y − t)

dL/dw1 = w1 = y \* x1

dL/dw2 = w2 = y \* x2

dL/dθ = θ = −w1 \* sin θ + w2 \* cos θ

**PART A5: SOLUTIONS**

Applying the Quotient Rule,

i.e., f(x)/g(x) = f'(x)g(x)-g'(x)f(x) / g(x)^2,

σ'(z) = ∂/∂z ( 1/ (1 + e^(-z)))

= 0 - (-e^(-z)) / (1 + e^(-z))^2

= 1/ (1 + e^(-z)) \* e^(-z) / (1 + e^(-z))

= σ(z) \* (1 - σ(z))

**Part C Follow-up Questions of Part B**

**C1:**

The complete dataset records information from January 1, 2011, and December 31, 2012. For testing, we keep the last 21 days' data. We also keep the last 60 days or so as a validation set from the remaining data.

Training Set: January 1, 2011 to October 10, 2012

Validation Set: October 11, 2012, to December 10, 2012, is the starting date.

Testing Set: 12/31/2012 to 12/11/2012 as the end date

**C2**:

One well-known feature that occurs when your trained model converges is that training loss does not continue to diminish. In the subsequent training rounds, the training loss varies within a narrow range. There are several reasons why the model might not converge, including improper hyper-parameter values and flawed back propagation function logics (even when the model passes).

**C3:**

It took us 45 Minutes to finish Training our Model.

The Unit test Ran 5 tests in 0.006s;

unittest.runner.TextTestResult run=5 errors=0 failures=0