Neural Networks

|  |  |  |
| --- | --- | --- |
| Jan Carlo D. Rabacca  11106344 | Suzette Dela Cruz  11118113 | Marc Janssen Chiu  11114525 |

1. Introduction

Neural Network is a computing system made up of simple, highly interconnected processing elements that process information by their dynamic state response to external inputs. Artificial neural networks are devices or algorithms that are modelled after the neuronal structure of the mammalian cerebral cortex but is a more basic form.

Neural networks, typically have layers made up of interconnected ‘nodes’ which contain an ‘activation function’. There is an input layer where patterns are presented. The input layer communicates to the hidden layers where the processing is being done with a system of weighted connections. The hidden layers link to the output layer.

1. Data and Results

*Part 3*

In [52]:

%matplotlib inline

import matplotlib.pyplot as plt

#Compare estimate, yHat, to actually score

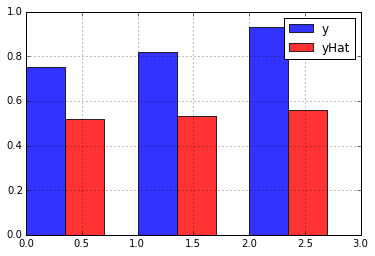
plt.bar([0,1,2], y, width = 0.35, alpha=0.8)

plt.bar([0.35,1.35,2.35],yHat, width = 0.35, color='r', alpha=0.8)

plt.grid(1)

plt.legend(['y', 'yHat'])

Out [52]:



In [26]:

timeElapsed = endTime-startTime

timeElapsed

0.023830502161373935

In [55]:

%matplotlib inline

import matplotlib.pyplot as plt

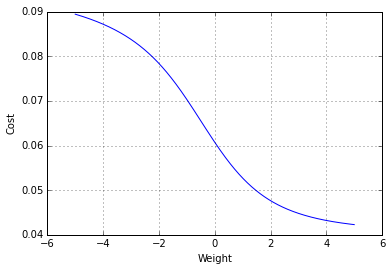
plt.plot(weightsToTry, costs)

plt.grid(1)

plt.ylabel('Cost')

plt.xlabel('Weight')

Out [55]:



*Part 4*

In [6]:

%matplotlib inline

import matplotlib.pyplot as plt

testValues = np.arange(-5,5,0.01)

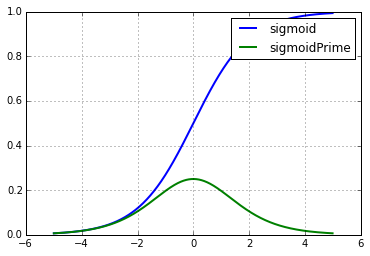
plt.plot(testValues, sigmoid(testValues), linewidth=2)

plt.plot(testValues, sigmoidPrime(testValues), linewidth=2)

plt.grid(1)

plt.legend(['sigmoid', 'sigmoidPrime'])

Out [6]:



In [15]:

print cost1, cost2

Out [15]:

[ 0.40023281] [ 0.70519492]

In [17]:

print cost2, cost3

Out [17]:

[ 0.70519492] [ 0.45518129]

*Part 5*

In [5]:

numericalGradient, 2\*x

Out [5]:

(2.9999999999996696, 3.0)

In [9]:

numgrad = computeNumericalGradient(NN, X, y)

numgrad

Out[9]:

array([ 0.01247989, -0.01838624, -0.00560211, 0.00574306, -0.00828112,

-0.00246826, -0.05745107, -0.05743181, -0.06648923])

In [10]:

grad = NN.computeGradients(X,y)

grad

Out[10]:

array([ 0.01247989, -0.01838624, -0.00560211, 0.00574306, -0.00828112,

-0.00246826, -0.05745107, -0.05743181, -0.06648923])

In [21]:

from numpy import linalg as LA

LA.norm(grad-numgrad)/LA.norm(grad+numgrad)

Out [21]:

8.2427921372074015e-10

1. Analysis and Conclusion