Ryan Helmlinger

Period 3

5/19/16

NOT:

Error: 3.000\*e-101

AND:

Error: 7.678\*e-55

OR:

Error: 3.245\*e-101

import math, sys, random, decimal

from decimal import Decimal, localcontext

def sigCalc(x, w, b):

(x1, x2) = x #for AND and OR

return 1/(1+Decimal(math.e)\*\*Decimal(-w\*(x1+x2)+b)) #for AND and OR

#return 1/(1+Decimal(math.e)\*\*Decimal(-w\*x+b)) #for NOT

def calcError(rules, w, b):

totalError = 0

for part in rules:

num = sigCalc(part, w, b)

error = rules[part]-num

totalError += error\*\*2

return totalError

def hillClimb(rules, w, b):

while True:

diff, newbies = 10, []

for i in range(-1,2):

for j in range(-1,2):

if not i+j==1:

w2,b2 = w,b

w2 += diff\*i

b2 += diff\*j

newbies.append((w2,b2))

minEQ, minError = 0, calcError(rules, w, b)

edited = False

if minError < 1.0\*10\*\*-100:

return (w,b)

for eq in newbies:

(w2,b2) = eq

error = calcError(rules, w2, b2)

if error<minError:

edited = True

minError = error

minEQ = eq

if edited == True:

(w,b) = minEQ

continue

else: return (w,b)

#probDict = {1:0, 0:1} #NOT

probDict = {(0,0):0, (1,0):0, (1,1):1} #AND

#probDict = {(0,0):0, (1,0):1, (1,1):1} #OR

minOverError = float('inf')

while True:

(w,b)=hillClimb(probDict, random.uniform(-10,10), random.uniform(-10,10))

error = calcError(probDict, w, b)

if error<minOverError:

minOverError = error

print()

print("w: ", w, "\nb: ", b, "\nerror: ", error)

if error==0: break