In [137]:

```
import numpy as np
```

In [138]:

```
data=[]
res=0
for i in range(0,1024):

  res=i

  #print("")
  re=np.binary_repr(res, width=10)
  temp=[]
  for k in re:
      temp.append(int(k))

  data.append(temp)

data=np.reshape(data,(1024,10))
print(data)
```

```
[[0 0 0 ..., 0 0 0]
 [0 0 0 ..., 0 0 1]
 [0 0 0 ..., 0 1 0]
 ...,
 [1 1 1 ..., 1 0 1]
 [1 1 1 ..., 1 1 0]
 [1 1 1 ..., 1 1 1]]
```

In [139]:

```
#data=np.random.randint(2,size=(50,10))

#print(data)
#extra=np.ones((1024,1),dtype=int)
#print(extra)
#X= np.concatenate((data, extra), axis=1)
X=data
print(X)
```

```
[[0 0 0 ..., 0 0 0]
 [0 0 0 ..., 0 0 1]
 [0 0 0 ..., 0 1 0]
 ...,
 [1 1 1 ..., 1 0 1]
 [1 1 1 ..., 1 1 0]
 [1 1 1 ..., 1 1 1]]
```

In [140]:

```
ans=[]
for i in range(0,1024):
    temp=[]
    i=i%8
    re=np.binary_repr(i,width=3)
    for k in re:
        temp.append(int(k))

    ans.append(temp)

#ans=np.array(ans)
y=np.reshape(ans,(1024,3))
print(y)
```

```
[[0 0 0]
 [0 0 1]
 [0 1 0]
```

```
 ...,
 [1 0 1]
 [1 1 0]
 [1 1 1]]
```

In [159]:

```
w1=2*np.random.random((len(X[0]),2))-1
w2=2*np.random.random((2,3))-1
print(w1)
print(w2)
```

```
[[ 0.35952209  0.83185773]
 [ 0.58842727 -0.48981136]
 [ 0.87579884  0.62075646]
 [-0.32517376 -0.8940269 ]
 [ 0.96608894 -0.51651674]
 [-0.90286206 -0.58874045]
 [ 0.54875069 -0.20370135]
 [ 0.37303503 -0.1834349 ]
 [-0.00176486  0.68894695]
 [ 0.28652741 -0.94087365]]
[[ 0.97368008 -0.65062761  0.71495783]
 [-0.96514486  0.76809884 -0.20339941]]
```

In [160]:

```
def sigmoid(x, deriv=False):
  if(deriv):
    return x*(1-x)
  else:
    return (1/(1+np.exp(-x)))
```

In [161]:

```
#Training
for epoch in range(0,100000):

  eta=0.2
  #Weighted Sum

  z_h=np.dot(X,w1)
  a_h=sigmoid(z_h)

  # second weighted sum
  z_o=np.dot(a_h,w2)
  a_o=sigmoid(z_o)
  #Error Calculate
  Ea=(0.5*(np.power((a_o-y),2)))
  #BackPropagation dE/dw2
  #dE/da0
  dEda0=(a_o-y)
  #da0/dz0
  da0dz0=sigmoid(a_o,deriv=True)
  #dz0/dw
  dz0dw=a_h
  #dE/dw2
  st=da0dz0*dEda0
  dEdw2=np.dot(dz0dw.T,st)
  #Weight update
  w2=w2-(eta*dEdw2)
  #hidden Layer dEdw1
  #dE/dah
  dEdah=np.dot(st,w2.T)
  #dzh/dw
  dzhdw=X
  #dah/dzh
  dahdzh=sigmoid(a_h,deriv=True)
  fs=dEdah*dahdzh
  dEdw1=np.dot(dzhdw.T,fs)
  #Weight Update
  w1=w1-(eta*dEdw1)

print(w1)
```

```
print(w2)
```

```
[[  4.43148972   4.51053474]
 [  4.43148972   4.51053474]
 [  4.43148972   4.51053474]
 [  4.43148972   4.51053474]
 [  4.43148972   4.51053474]
 [  4.43148972   4.51053474]
 [  4.43148972   4.51053474]
 [-48.94428166   0.81280369]
 [  4.72856028   5.08607353]
 [  5.40149503 -48.72693181]]
[[-18.67086221  -2.30794685   8.90697189]
 [  9.42443324  -0.60857091 -17.1480007 ]]
```

In [162]:

```
np.set_printoptions(suppress=True)
test_ah=sigmoid(np.dot(X,w1))
prediction=sigmoid(np.dot(test_ah,w2))
print(prediction)
```

```
[[ 0.00972566  0.18873377  0.01597676]
 [ 0.00000001  0.09132316  0.99985905]
 [ 0.0001072   0.05252273  0.00027084]
 ...,
 [ 0.49998318  0.49999789  0.50000777]
 [ 0.99991928  0.35238428  0.00000004]
 [ 0.49810689  0.49976025  0.50086215]]
```

In [173]:

```
result=[]
for i in range(0,1024):
    tp=[]
    for j in range(0,3):
        if(prediction[i][j]>0.40):
            tp.append(1)
        else:
            tp.append(0)
    result.append(tp)
target=np.reshape(result,(1024,3))
print(target)
```

```
[[0 0 0]
 [0 0 1]
 [0 0 0]
 ...,
 [1 1 1]
 [1 0 0]
 [1 1 1]]
```

In [174]:

```
#Training Accuracy
cnt=0

for i in range(0,1024):
    for j in range(0,3):
        if(target[i][j]==y[i][j]):
            cnt=cnt+1

accuracy=(cnt/3072.00)*100.00
print(accuracy,"%")
```

```
83.33333333333334 %
```

In [216]:

In [216]:

```
#Testing Accuracy
#Case 2
X_train=[]
for i in range(0,614):
    X_train.append(X[i])
X_train=np.array(X_train)
print(X_train)
X_train.shape
```

```
[[0 0 0 ..., 0 0 0]
 [0 0 0 ..., 0 0 1]
 [0 0 0 ..., 0 1 0]
 ...,
 [1 0 0 ..., 0 1 1]
 [1 0 0 ..., 1 0 0]
 [1 0 0 ..., 1 0 1]]
```

Out[216]:

```
(614, 10)
```

In [217]:

```
y_train=[]
for i in range(0,614):
    y_train.append(y[i])
y_train=np.array(y_train)
print(y_train)
y_train.shape
```

```
[[0 0 0]
 [0 0 1]
 [0 1 0]
 ...,
 [0 1 1]
 [1 0 0]
 [1 0 1]]
```

Out[217]:

```
(614, 3)
```

In [218]:

```
w3=2*np.random.random((len(X_train[0]),2))-1
w4=2*np.random.random((2,3))-1
print(w3)
print(w4)
```

```
[[-0.62733608  0.67013078]
 [ 0.39338393  0.39376368]
 [-0.02430889 -0.52565891]
 [-0.10692442 -0.67054252]
 [-0.14186614 -0.22941268]
 [ 0.3975741  -0.08429866]
 [-0.22348553  0.30979906]
 [ 0.30291585  0.5746871 ]
 [ 0.49845281  0.19927141]
 [ 0.51213463 -0.86889614]]
[[-0.92890949 -0.31766341  0.71822142]
 [-0.6000111   0.16430128 -0.4757969 ]]
```

In [219]:

```
X_test=[]
for i in range(614,1024):
    X_test.append(X[i])
X_test=np.array(X_test)
print(X_test)
X_test.shape
```

```
[[1 0 0 ..., 1 1 0]
 [1 0 0 ..., 1 1 1]
 [1 0 0 ..., 0 0 0]
 ...,
 [1 1 1 ..., 1 0 1]
 [1 1 1 ..., 1 1 0]
 [1 1 1 ..., 1 1 1]]
```

Out[219]:

```
(410, 10)
```

In [220]:

```
y_test=[]
for i in range(614,1024):
    y_test.append(y[i])
y_test=np.array(y_test)
print(y_test)
y_test.shape
```

```
[[1 1 0]
 [1 1 1]
 [0 0 0]
 ...,
 [1 0 1]
 [1 1 0]
 [1 1 1]]
```

Out[220]:

```
(410, 3)
```

In [221]:

```
for epoch in range(0,120000):

  eta=0.2
  #Weighted Sum

  z_h=np.dot(X_train,w3)
  a_h=sigmoid(z_h)

  # second weighted sum
  z_o=np.dot(a_h,w4)
  a_o=sigmoid(z_o)
  #Error Calculate
  Ea=(0.5*(np.power((a_o-y_train),2)))
  #BackPropagation dE/dw2
  #dE/da0
  dEda0=(a_o-y_train)
  #da0/dz0
  da0dz0=sigmoid(a_o,deriv=True)
  #dz0/dw
  dz0dw=a_h
  #dE/dw2
  st=da0dz0*dEda0
  dEdw2=np.dot(dz0dw.T,st)
  #Weight update
  w4=w4-(eta*dEdw2)
  #hidden Layer dEdw1
  #dE/dah
  dEdah=np.dot(st,w4.T)
  #dzh/dw
  dzhdw=X_train
  #dah/dzh
  dahdzh=sigmoid(a_h,deriv=True)
  fs=dEdah*dahdzh
  dEdw1=np.dot(dzhdw.T,fs)
  #Weight Update
  w3=w3-(eta*dEdw1)

print(w3)
print(w4)
```

```
print(w4)
```

```
[[  7.57678671   7.26286314]
 [  7.57314325   4.91007832]
 [  7.57166314   4.9100777 ]
 [  7.57653396   4.91929238]
 [  7.57664511   4.91929242]
 [  7.57594079   4.92781975]
 [  7.57467325   4.92781921]
 [  7.64218351   4.91776362]
 [  7.56535972   5.06370899]
 [  7.29498846 -39.98351389]]
[[ -2.94307604  -4.4242976   14.97270063]
 [  0.64056074  -2.19610254 -23.05698099]]
```

In [222]:

```
np.set_printoptions(suppress=True)
test_ah=sigmoid(np.dot(X_test,w3))
prediction=sigmoid(np.dot(test_ah,w4))
print(prediction)
prediction.shape
```

```
[[ 0.09091486  0.00133112  0.00030825]
 [ 0.05006486  0.01184068  0.99999969]
 [ 0.09091486  0.00133112  0.00030825]
 ...,
 [ 0.08313442  0.00186159  0.01038196]
 [ 0.09091486  0.00133112  0.00030825]
 [ 0.0908545   0.00133446  0.00031646]]
```

Out[222]:

```
(410, 3)
```

In [223]:

```
result=[]
for i in range(0,410):
    tp=[]
    for j in range(0,3):
        if(prediction[i][j]>0.40):
            tp.append(1)
        else:
            tp.append(0)
    #print(result)
    result.append(tp)
result=np.array(result)
target=np.reshape(result,(410,3))
print(target)
```

```
[[0 0 0]
 [0 0 1]
 [0 0 0]
 ...,
 [0 0 0]
 [0 0 0]
 [0 0 0]]
```

In [224]:

```
#Testing Accuracy
cnt=0

for i in range(0,410):
    for j in range(0,3):
        if(target[i][j]==y_test[i][j]):
            cnt=cnt+1

accuracy=(cnt/1230.00)*100.00
print(accuracy,"%")
```

```
65.77235772357723 %
```

In [206]:

```
#Case 3
#Testing Accuracy
X_train2=[]
for i in range(0,922):
    X_train2.append(X[i])
X_train2=np.array(X_train2)
print(X_train2)
X_train2.shape
```

```
[[0 0 0 ..., 0 0 0]
 [0 0 0 ..., 0 0 1]
 [0 0 0 ..., 0 1 0]
 ...,
 [1 1 1 ..., 1 1 1]
 [1 1 1 ..., 0 0 0]
 [1 1 1 ..., 0 0 1]]
```

Out[206]:

```
(922, 10)
```

In [207]:

```
X_test2=[]
for i in range(922,1024):
    X_test2.append(X[i])
X_test2=np.array(X_test2)
print(X_test2)
X_test2.shape
```

```
[[1 1 1 ..., 0 1 0]
 [1 1 1 ..., 0 1 1]
 [1 1 1 ..., 1 0 0]
 ...,
 [1 1 1 ..., 1 0 1]
 [1 1 1 ..., 1 1 0]
 [1 1 1 ..., 1 1 1]]
```

Out[207]:

```
(102, 10)
```

In [208]:

```
y_train2=[]
for i in range(0,922):
    y_train2.append(y[i])
y_train2=np.array(y_train2)
print(y_train2)
y_train2.shape
```

```
[[0 0 0]
 [0 0 1]
 [0 1 0]
 ...,
 [1 1 1]
 [0 0 0]
 [0 0 1]]
```

Out[208]:

```
(922, 3)
```

In [ ]:

```
y_test2=[]
```

```
for i in range(922,1024):
    y_test2.append(y[i])
y_test2=np.array(y_test2)
print(y_test2)
y_test2.shape
```

In [210]:

```
w5=np.random.random((len(X_train2[0]),2))
w6=np.random.random((2,3))
print(w5)
print(w6)
```

```
[[ 0.26667261  0.6758651 ]
 [ 0.57494299  0.8291755 ]
 [ 0.46495934  0.47031982]
 [ 0.63454385  0.54332791]
 [ 0.47773781  0.19049624]
 [ 0.53271829  0.33254615]
 [ 0.40162347  0.15202088]
 [ 0.54419772  0.32422244]
 [ 0.09282703  0.59698184]
 [ 0.4161234   0.14584759]]
[[ 0.38069664  0.8724181   0.03751577]
 [ 0.25163811  0.79546849  0.7908924 ]]
```

In [211]:

```
for epoch in range(0,120000):

  eta=0.2
  #Weighted Sum

  z_h=np.dot(X_train2,w5)
  a_h=sigmoid(z_h)

  # second weighted sum
  z_o=np.dot(a_h,w6)
  a_o=sigmoid(z_o)
  #Error Calculate
  Ea=(0.5*(np.power((a_o-y_train2),2)))
  #BackPropagation dE/dw2
  #dE/da0
  dEda0=(a_o-y_train2)
  #da0/dz0
  da0dz0=sigmoid(a_o,deriv=True)
  #dz0/dw
  dz0dw=a_h
  #dE/dw2
  st=da0dz0*dEda0
  dEdw2=np.dot(dz0dw.T,st)
  #Weight update
  w6=w6-(eta*dEdw2)
  #hidden Layer dEdw1
  #dE/dah
  dEdah=np.dot(st,w6.T)
  #dzh/dw
  dzhdw=X_train2
  #dah/dzh
  dahdzh=sigmoid(a_h,deriv=True)
  fs=dEdah*dahdzh
  dEdw1=np.dot(dzhdw.T,fs)
  #Weight Update
  w5=w5-(eta*dEdw1)

print(w5)
print(w6)
```

```
[[ 7.55792312  7.07528084]
 [ 7.01090116  6.79450096]
 [ 7.19243764  7.42452306]
 [ 7.27062025  7.61075113]
 [ 7.49326649  8.10156459]
 [ 7.59049928  8.03307366]
```

```
 [ 7.7896999   8.28772748]
 [ 3.07221206  2.82021673]
 [ 3.0860287   2.59238659]
 [ 1.75922785  2.44719612]]
[[-7.62376916 -7.70471539 -7.92372095]
 [-7.92605155 -7.2985553  -7.35083632]]
```

In [ ]:

```
np.set_printoptions(suppress=True)
test_ah=sigmoid(np.dot(X_test2,w5))
prediction=sigmoid(np.dot(test_ah,w6))
print(prediction)
prediction.shape
```

In [ ]:

```
result=[]
for i in range(0,102):
    tp=[]
    for j in range(0,3):
        if(prediction[i][j]>0.40):
            tp.append(1)
        else:
            tp.append(0)
    #print(result)
    result.append(tp)
#target=np.reshape(result,(20,3))
result=np.array(result)
target=np.reshape(result,(102,3))
print(target)
```

In [215]:

```
#Testing Accuracy
cnt=0

for i in range(0,102):
    for j in range(0,3):
        if(target[i][j]==y_test2[i][j]):
            cnt=cnt+1

accuracy=(cnt/306.00)*100.00
print(accuracy,"%")
```

```
49.34640522875817 %
```

In [ ]:

```

```