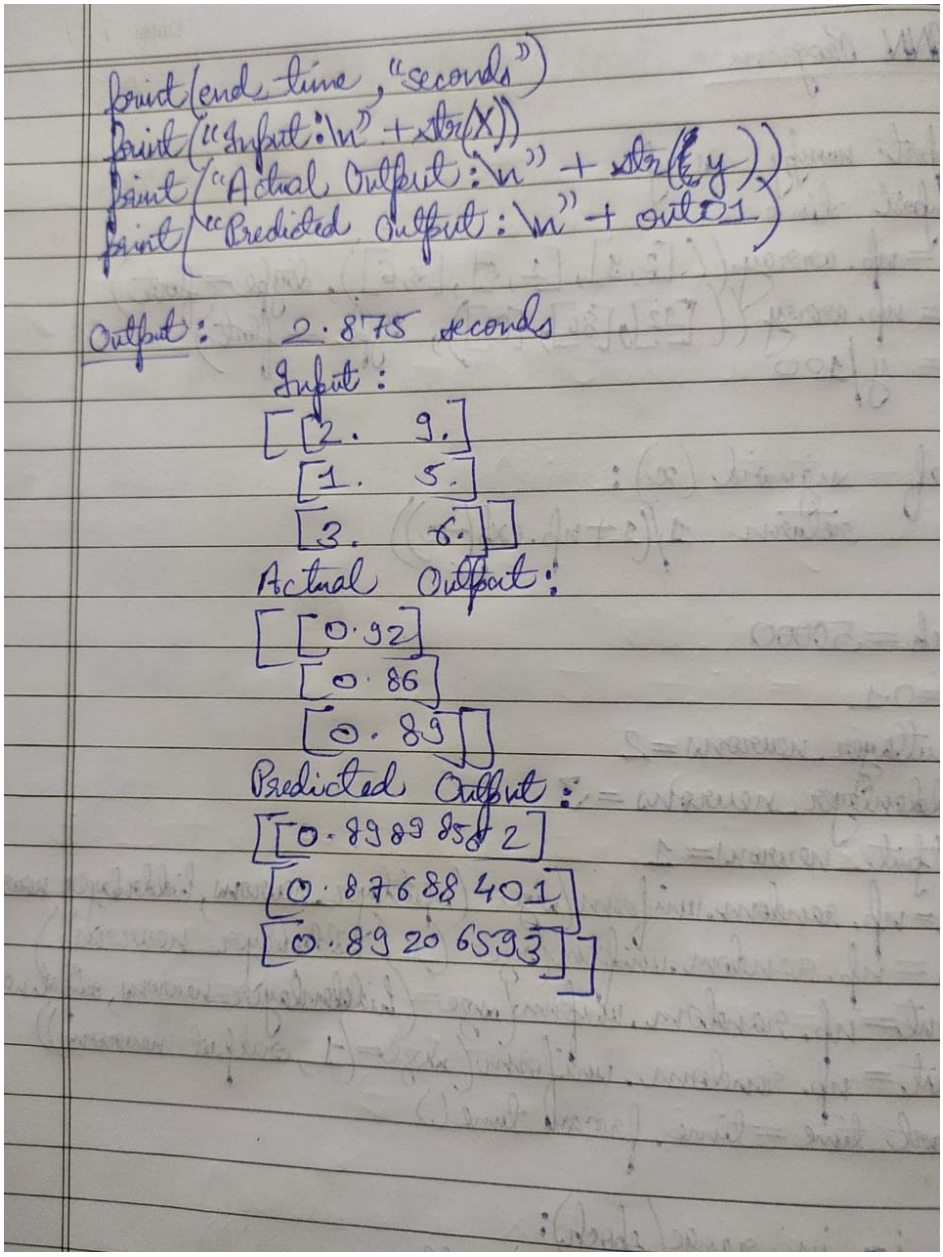
ANN PROGRAM

CODE

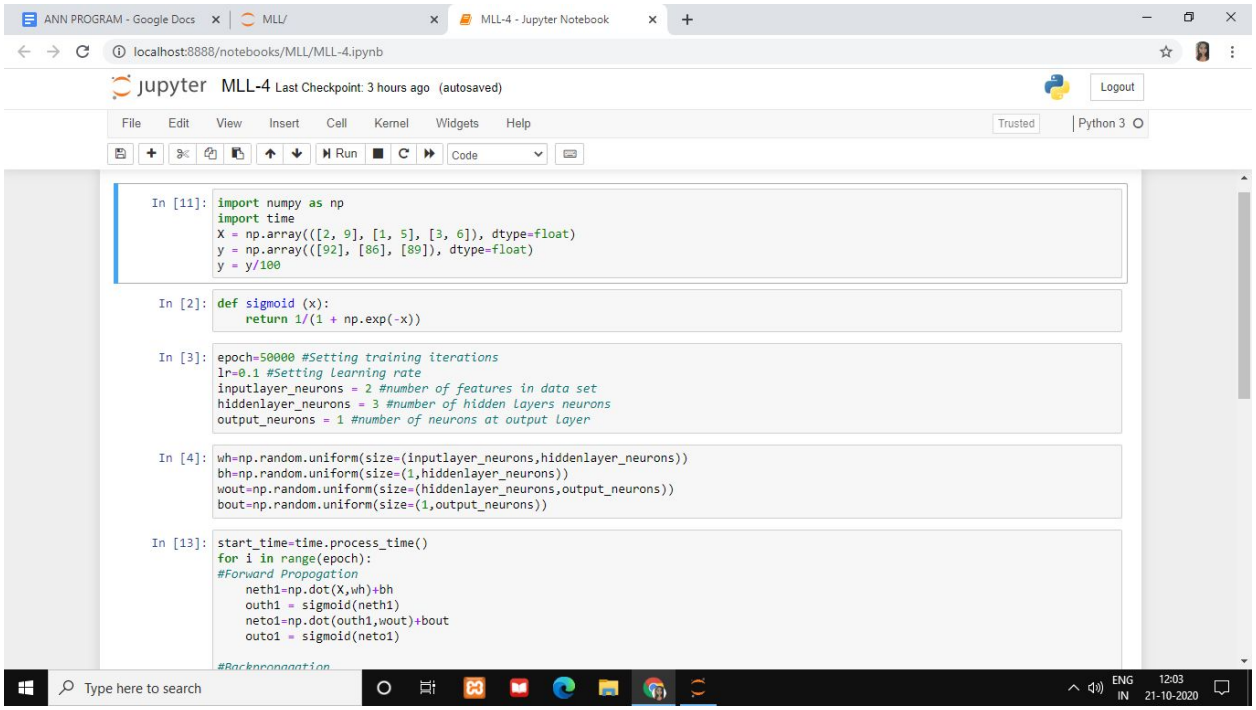
```
import numpy as np
import time
X = np.array(([2, 9], [1, 5], [3, 6]), dtype=float)
y = np.array(([92], [86], [89]), dtype=float)
y = y/100
def sigmoid (x):
    return 1/(1 + np.exp(-x))
epoch=50000 #Setting training iterations
lr=0.1 #Setting learning rate
inputlayer_neurons = 2 #number of features in data set
hiddenlayer_neurons = 3 #number of hidden layers neurons
output_neurons = 1 #number of neurons at output layer
wh=np.random.uniform(size=(inputlayer_neurons,hiddenlayer_neurons))
bh=np.random.uniform(size=(1,hiddenlayer_neurons))
wout=np.random.uniform(size=(hiddenlayer_neurons,output_neurons))
bout=np.random.uniform(size=(1,output_neurons))
start_time=time.process_time()
for i in range(epoch):
#Forward Propogation
    neth1=np.dot(X,wh)+bh
    outh1 = sigmoid(neth1)
    neto1=np.dot(outh1,wout)+bout
    outo1 = sigmoid(neto1)

#Backpropagation
    deltak=(y-outo1)*(outo1)*(1-outo1)

    deltaj = deltak.dot(wout.T)*(outh1)*(1-outh1)

    wout += outh1.T.dot(deltak) *lr
    #bout += np.sum(deltak, axis=0,keepdims=True) *lr
    wh += X.T.dot(deltaj) *lr
    #bh += np.sum(deltaj, axis=0,keepdims=True) *lr
end_time=time.process_time()-start_time
print(end_time,"seconds")
print("Input: \n" + str(X))
print("Actual Output: \n" + str(y))
print("Predicted Output: \n" ,outo1)
```

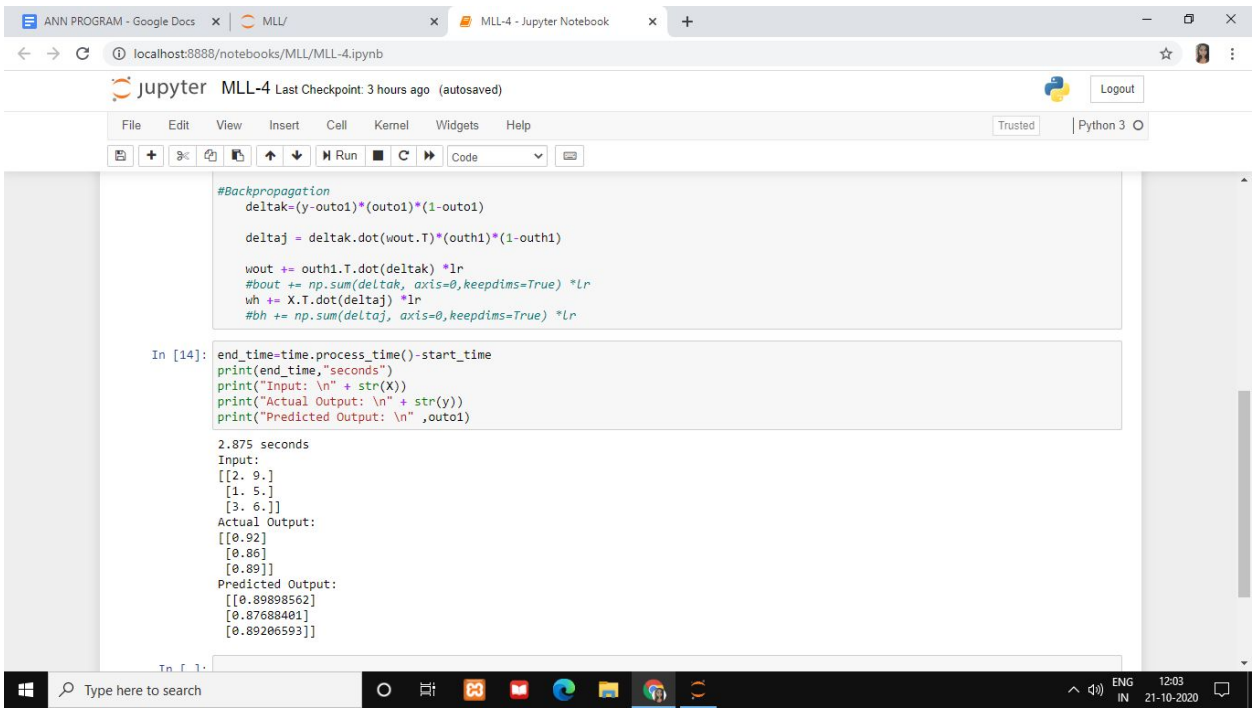
ANN Program

```
import numpy as np
import time
X = np.array(([2,9],[1,5],[3,6]), dtype=float)
y = np.array(([92],[86],[89]), dtype=float)
y = y/100

def sigmoid(x):
    return 1/(1+np.exp(-x))

epoch=50000
lr=0.1
inputlayer_neurons=2
hiddenlayer_neurons=3
output_neurons=1
wh=np.random.uniform(size=(inputlayer_neurons,hiddenlayer_neurons))
bh=np.random.uniform(size=(1,hiddenlayer_neurons))
wout=np.random.uniform(size=(hiddenlayer_neurons,output_neurons))
bout=np.random.uniform(size=(1,output_neurons))
start_time=time.process_time()

for i in range(epoch):
    hinp1=np.dot(X,wh)+bh
    outh1=sigmoid(hinp1)
    outo1=np.dot(outh1,wout)+bout
    out01=sigmoid(outo1)
    deltak=(y-out01)*(out01)*(1-outh1)
    deltaj=deltak.dot(wout.T)*(outh1)*(1-outh1)
    wout+=outh1.T.dot(deltak)*lr
    wh+=X.T.dot(deltaj)*lr

end_time=time.process_time()-start_time
```

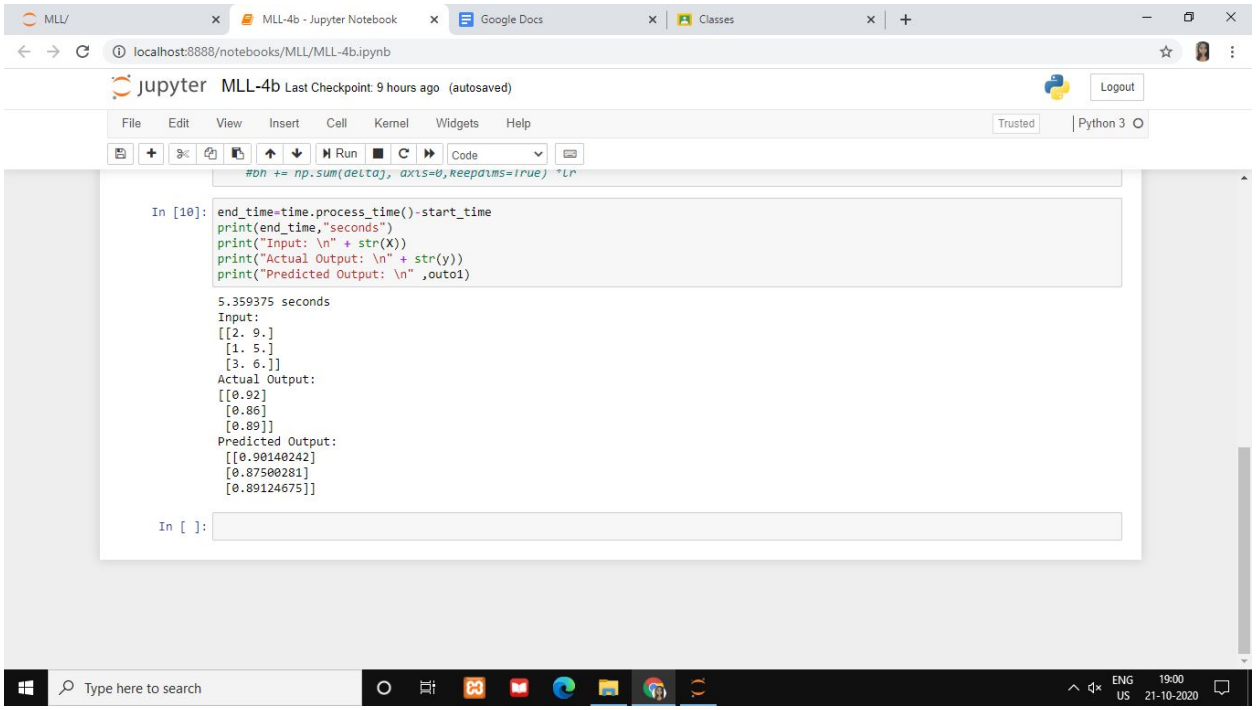

print(end_time, "seconds")
print("Input:\n" + str(X))
print("Actual Output:\n" + str(y))
print("Predicted Output:\n" + str(out01))
Output: 2.875 seconds
Input:
[[2. 9.]
[1. 5.]
[3. 6.]]
Actual Output:
[[0.92]
[0.86]
[0.89]]
Predicted Output:
[[0.89898582]
[0.87688401]
[0.89206593]]

OUTPUT

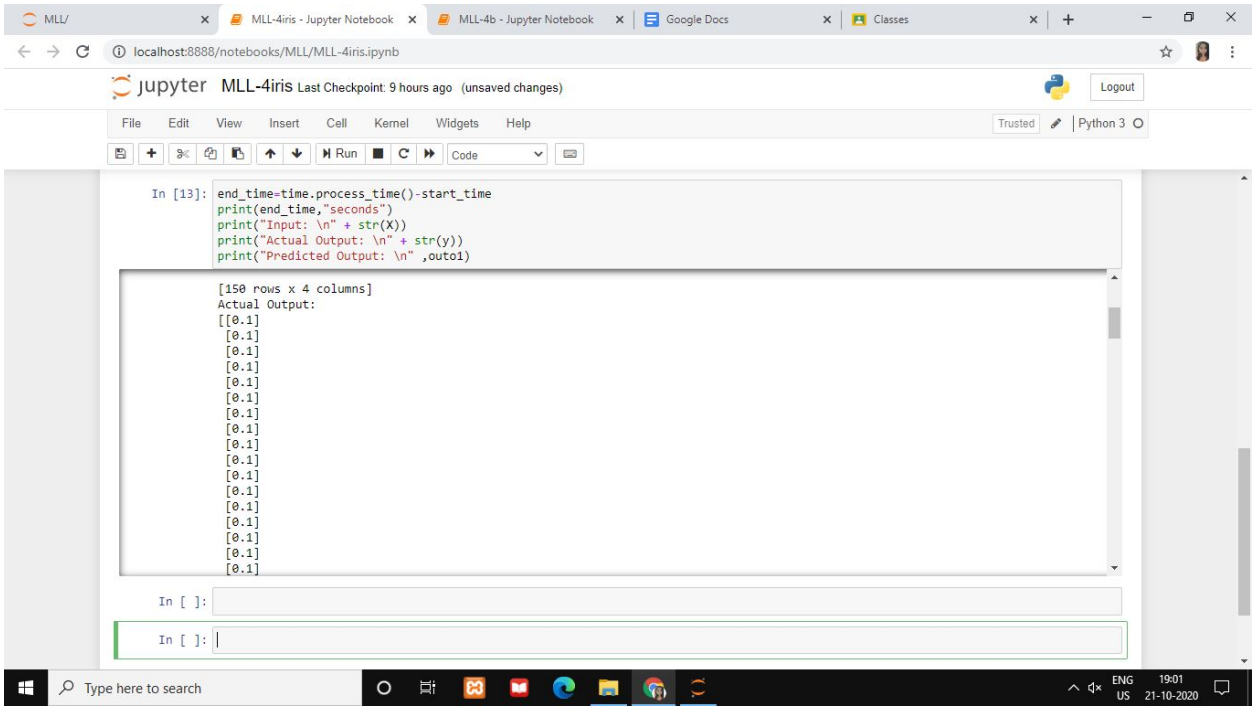


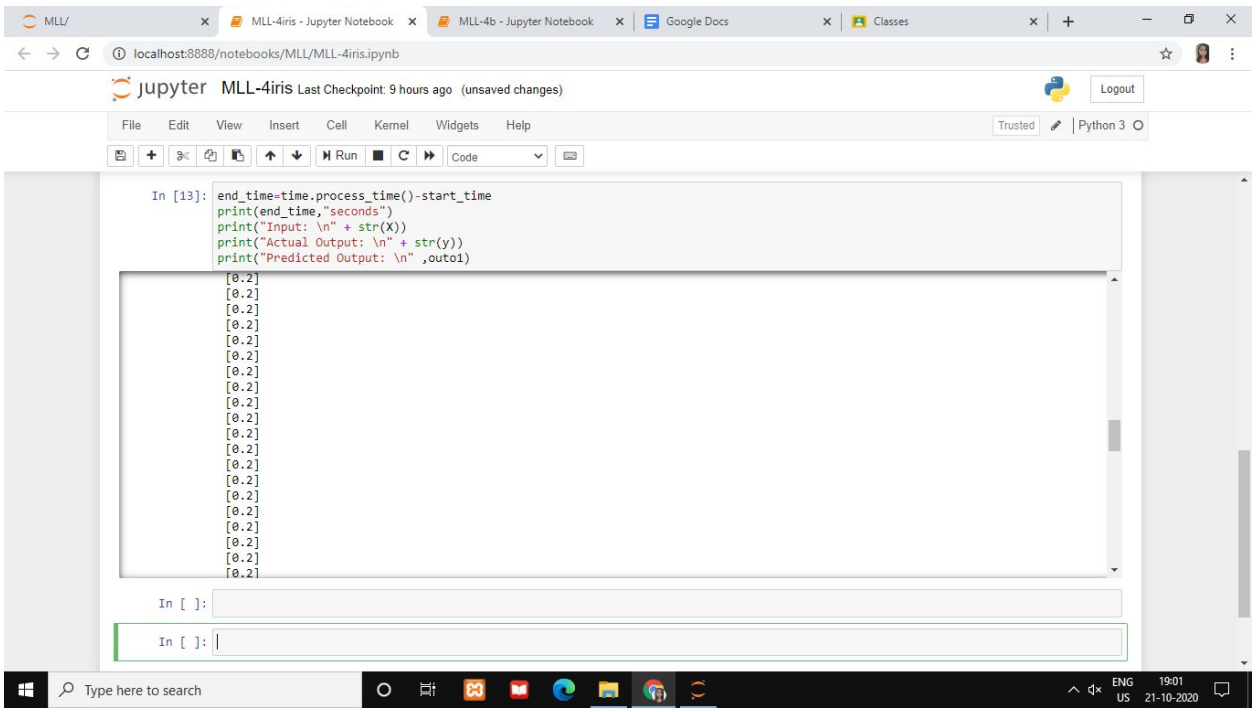


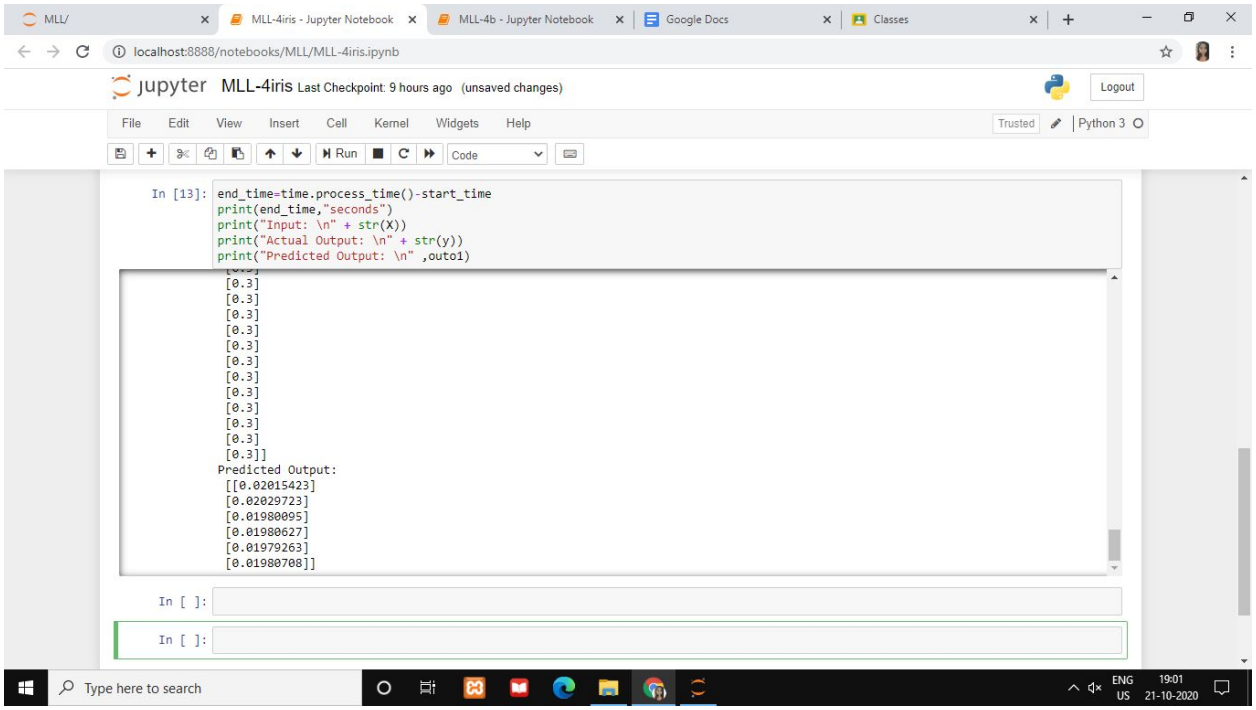
Output for 100000 epochs



Output for iris dataset


MLL/
MLL-4iris - Jupyter Notebook
MLL-4b - Jupyter Notebook
Google Docs
Classes
localhost:8888/notebooks/MLL/MLL-4iris.ipynb
jupyter MLL-4iris Last Checkpoint: 9 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code
In [13]:
end_time=time.process_time()-start_time
print(end_time,"seconds")
print("Input: \n" + str(X))
print("Actual Output: \n" + str(y))
print("Predicted Output: \n" ,outo1)
[150 rows x 4 columns]
Actual Output:
[[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
[0.1]
In [ ]:
In [ ]:
Type here to search
ENG
US
19:01
21-10-2020


MLL/
MLL-4iris - Jupyter Notebook
MLL-4b - Jupyter Notebook
Google Docs
Classes
localhost:8888/notebooks/MLL/MLL-4iris.ipynb
jupyter MLL-4iris Last Checkpoint: 9 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code
In [13]:
end_time=time.process_time()-start_time
print(end_time,"seconds")
print("Input: \n" + str(X))
print("Actual Output: \n" + str(y))
print("Predicted Output: \n" ,outo1)
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
[0.2]
In [ ]:
In [ ]:
Type here to search
ENG
US
19:01
21-10-2020


MLL/
MLL-4iris - Jupyter Notebook
MLL-4b - Jupyter Notebook
Google Docs
Classes
localhost:8888/notebooks/MLL/MLL-4iris.ipynb
jupyter MLL-4iris Last Checkpoint: 9 hours ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code
In [13]:
end_time=time.process_time()-start_time
print(end_time,"seconds")
print("Input: \n" + str(X))
print("Actual Output: \n" + str(y))
print("Predicted Output: \n" ,outo1)
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]
[0.3]]
Predicted Output:
[[0.02015423]
[0.02029723]
[0.01980095]
[0.01980627]
[0.01979263]
[0.01980708]]
In [ ]:
In [ ]:
Type here to search
ENG
US
19:01
21-10-2020