Rajitha Bhavani Kantheti

<rkanthe-C46686177>

ECE 8720

Spring 2020

Takehome #1

Balancing Data sets:

->Before Balancing the datasets we need to clean the data given.

That is,

1.Add target value for each input in AEP and Non-AEP data sets.

Target value for AEP is 0, Non-AEP is 1.

2.Convert it to .csv format for easier access.

->Fetching the data from jupyter notebook:

I used read_csv() defined in the pandas library for fetching the data into the notebook

->Divide the data into training and test datasets

Before we balance data we first divide them into training and testing data sets.

Here I used 80% of the data as training data and 20% of the data as testing data.

Used function test_train_split to split the data sets

->Balancing Datasets

Majority of the dataitems in the given data belong to Non-AEP class, because of which overfiitting may happen.

Balancing the data can be done using: Over-sampling and undersampling methods.

After dividing the data as training and testing there are approximately 60 dataitems in AEP data set in training data set. So, if we choose under-sampling for balancing the training set, we might end up with only 120 dataitems for training, which is insufficient. Hence over-sampling is choosen to balance the training dataitems.

Also choose over-sampling for balancing the test dataset.

Training set dataitems-4962

Testing set dataitems-168

Over-sampling is done using resample() from sklearn.utils

Case 1:

Network contains no bias, no momentum.

Input nuerons=27

Hidden nuerons=55 (2d+1)

Output neurons=2

Step1:

All the weights are randomized using random() function

After returning the network containing all these weights,

Step 2:

We enter the first epoch and forward propogate the network.

After computing the activation function we compute outputs. Then Backpropogation error is calculated and weights are updated using learning rate.

Learning rate=0.1

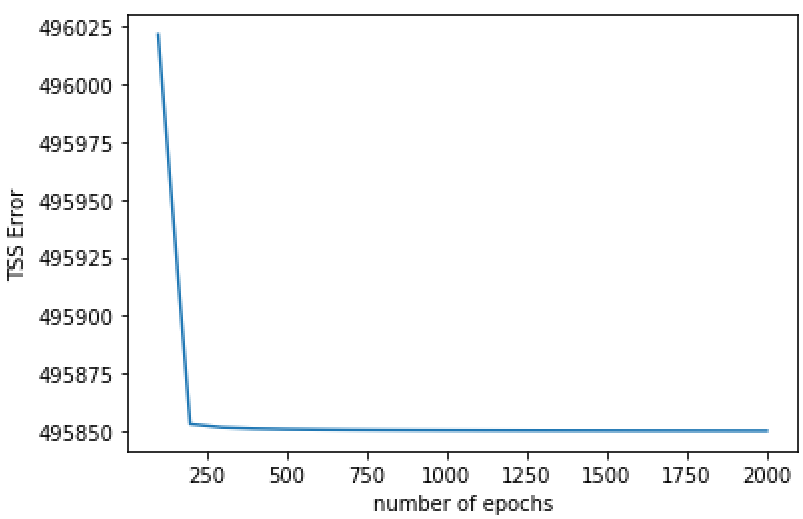
Epochs=2000

Output: TP=85, FN=83, FP=0, TN=0

Sensitivity=0.5059

E(TSS)=495850.145

Graph between TSS Error vs number of epochs



Case 2:

Network contains bias, no momentum.

Input nuerons=27

Hidden nuerons=55 (2d+1)

Output neurons=2

Learning rate=0.1

Epochs=1600

Bias Weights used in code:

- Here there are n+1 weights, one of them is the bias.
- Bias generated is added when calculating net weights to the activation function.
- It is a special weight which does not contain any input to multiply with.
- After calculating the back propogation error bias weight updation is important.

Advantage seen:

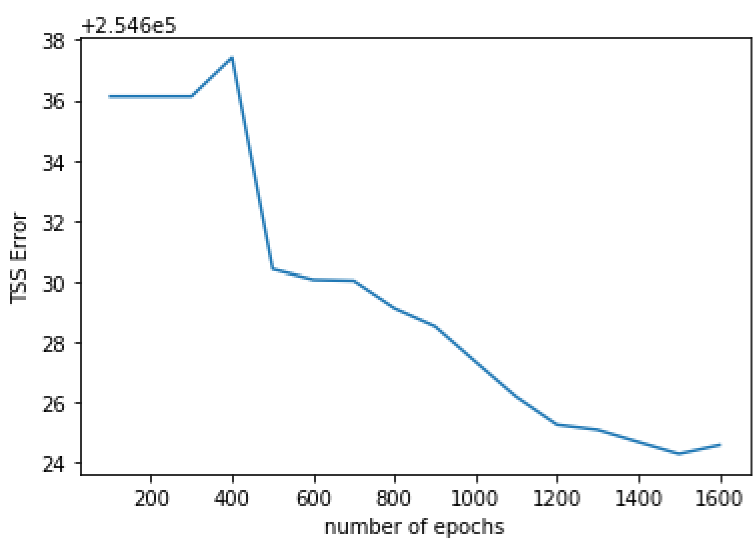
- E(TSS) error has dropped drastically
- Contolled learning rate
- Can perform like the case 1 network with learning rate of 0.04
- Reaches the possible minima
- Adjusts the output better

Output: TP=85, FN=83, FP=0, TN=0

Sensitivity=0.5059

E(TSS)= 254624.298

Graph between TSS Error vs number of epochs



Case 3:

Network contains no bias, but contains momentum.

Input nuerons=27

Hidden nuerons=55 (2d+1)

Output neurons=2

Learning rate=0.1

Momentum=0.5

Epochs=2000

momentum used in code:

- This case takes the initial weights of case 1.
- Momentum is included when we update the weights in the network

Advantage seen:

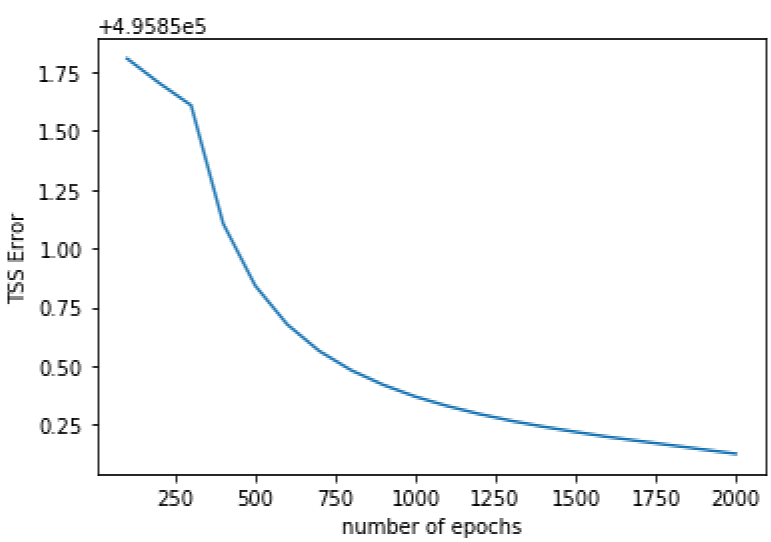
- Weights are adjusted smoothely
- Doesn't have sharp accelerations
- Helps reaching local minima

Output: TP=0, FN=0, FP=84, TN=84

Specificity=0.5000

E(TSS)= 495850.126

Graph between TSS Error vs number of epochs



Case 4:

Network contains bias and momentum.

Input nuerons=27

Hidden nuerons=55 (2d+1)

Output neurons=2

Learning rate=0.1

Momentum=0.5

Epochs=1600

Momentum, bias used in code:

- This case takes the initial weights of case 2.
- Momentum is included when we update the weights in the network
- Bias used near weights as an additional weight

Advantage seen:

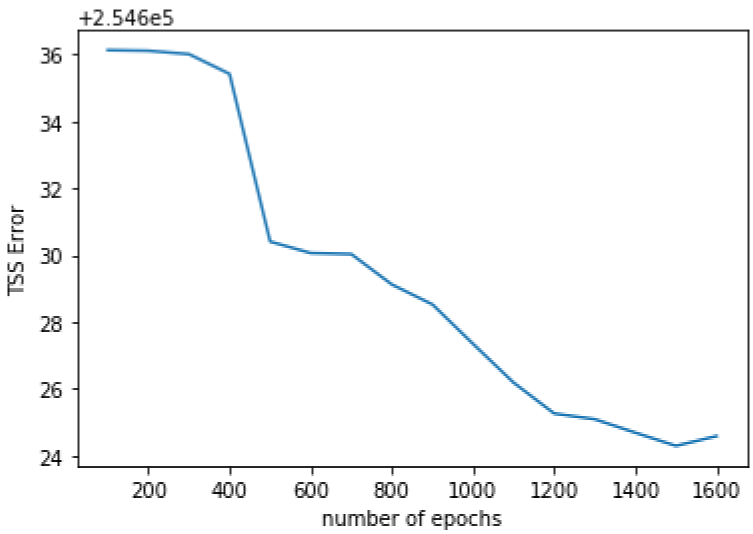
- Weights are adjusted smoothely and faster
- Faster and smoother graphs
- Helps reaching local minima
- Shows a guassian curve in the graph

Output: TP=0, FN=0, FP=84, TN=84

Specificity=0.5000

E(TSS)= 495850.126

Graph between TSS Error vs number of epochs



Problems Faced:

- Time for training is too long
- Takes lots of epochs to train the data
- Learning rates are bigger than thought would be
- Total sum of squares Error is really big
- Since I used jupyter notebook, the server always shuts down when I try to run all the four cases at the same time.

Overall Assessment:

If taken more epochs, I thinks I would have seen more better specificity and sensitivity measures.

Appendix

Programs:

Importing CSV file:

```
import numpy as np

import pandas as pd

 df_train = pd.read_csv('C:/Users/Rajitha Bhavani/Documents/EEGdata.csv')

target_count = df_train.target.value_counts()

print('Class 0:', target_count[0])

print('Class 1:', target_count[1])

print('Proportion:', round(target_count[0] / target_count[1], 2), ': 1')

 target_count.plot(kind='bar', title='Count (target)');
```

Dividing into training and testing:

```
from sklearn.model_selection import train_test_split

labels = df_train.columns[1:]


X = df_train[labels]

y = df_train['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=1)
```

Resampling:

```
from sklearn.utils import resample

pos_upsampled = resample(positive,

replace=True, # sample with replacement

n_samples=len(negative), # match number in majority class

random_state=27) # reproducible results

# combine majority and upsampled minority

upsampled = pd.concat([negative, pos_upsampled])
```

check new class counts

upsampled.target.value_counts()

upsampled.to_csv('C:/Users/Rajitha Bhavani/Documents/upsampled12.csv',index=False)

Case 3:

```
from math import exp
from random import seed
from random import random
import matplotlib.pyplot as plt
import numpy as np

# Initialize a network
def initialize_network(n_inputs, n_hidden, n_outputs):
    network = list()
    hidden_layer = [{'weights':[random() for i in range(n_inputs)]} for i in range(n_hidden)]
    network.append(hidden_layer)
    output_layer = [{'weights':[random() for i in range(n_hidden)]} for i in range(n_outputs)]
    network.append(output_layer)
    print(network)
    return network

# Calculate neuron activation for an input
def activate(weights, inputs):
    activation = 0
    for i in range(len(weights)):
        activation += weights[i] * inputs[i]
    return activation
```

```
# Transfer neuron activation
def transfer(activation):
    return 1.0 / (1.0 + exp(-activation))

# Forward propagate input to a network output
def forward_propagate(network, row):
    inputs = row
    for layer in network:
        new_inputs = []
        for neuron in layer:
            activation = activate(neuron['weights'], inputs)
            neuron['output'] = transfer(activation)
            new_inputs.append(neuron['output'])
        inputs = new_inputs
    return inputs

# Calculate the derivative of an neuron output
def transfer_derivative(output):
    return output * (1.0 - output)

# Backpropagate error and store in neurons
def backward_propagate_error(network, expected):
    for i in reversed(range(len(network))):
        layer = network[i]
        errors = list()
        if i != len(network)-1:
            for j in range(len(layer)):
```

```
            error = 0.0
            for neuron in network[i + 1]:
                error += (neuron['weights'][j] * neuron['delta'])
            errors.append(error)
    else:
        for j in range(len(layer)):
            neuron = layer[j]
            errors.append(expected[j] - neuron['output'])
    for j in range(len(layer)):
        neuron = layer[j]
        neuron['delta'] = errors[j] * transfer_derivative(neuron['output'])

# Update network weights with error
def update_weights(network, row, l_rate):
    for i in range(len(network)):
        inputs = row[1:]
        prevval=0
        if i != 0:
            inputs = [neuron['output'] for neuron in network[i - 1]]
        velocity=[None]*len(inputs)
        for neuron in network[i]:
            for j in range(len(inputs)):
                if j==1:
                    velocity[j]=l_rate * neuron['delta'] * inputs[j]
                else:
                    velocity[j]=l_rate * neuron['delta'] * inputs[j] + 0.5 * prevval
                prevval=velocity[j]
```

```
            neuron['weights'][j] += velocity[j]
        #neuron['weights'][-1] += l_rate * neuron['delta']

# Train a network for a fixed number of epochs
def train_network(network, train, l_rate, n_epoch, n_outputs):
    sum_errors=0
    errors=list()
    while sum_errors<1:
        for epoch in range(n_epoch):
            sum_error = 0
            for row in train:
                #print(row)
                outputs = forward_propagate(network, row)
                #print(outputs)
                expected = [0 for i in range(n_outputs)]
                expected[row[0]] = 1
                sum_error += sum([(expected[i]-outputs[i])**2 for i in range(len(expected))])/2
                backward_propagate_error(network, expected)
                update_weights(network, row, l_rate)
            if epoch!=0:
                sum_errors+=sum_error
            if epoch!=0 and epoch%100==0:
                errors.append(sum_errors*2)
                ep.append(epoch)
                print('>epoch=%d, lrate=%.3f, error=%.3f' % (epoch, l_rate, sum_errors*2))
                sum_errors=0
    return errors
```

```
def predict(network, row):
    outputs = forward_propagate(network, row)
    return outputs.index(max(outputs))

# Backpropagation Algorithm With Stochastic Gradient Descent
def back_propagation(train, test, l_rate, n_epoch, n_hidden):
    scores=list()
    n_inputs = 27
    n_outputs = len(set([row[0] for row in train]))
    network = initialize_network(n_inputs, n_hidden, n_outputs)
    sum_errors=train_network(network, train, l_rate, n_epoch, n_outputs)
    predictions = list()
    #for row in test:
    #   prediction = predict(network, row)
    #    predictions.append(prediction)
    #    accuracy=accuracy_metric(row[0],predictions)
    #    scores.append(accuracy)
    #return(scores)
    return sum_errors

def accuracy_metric(actual, predicted):
    correct = 0
    for i in range(actual):
        print(actual)
        print(predict[i])
        if actual == predicted:
```

```
            correct += 1
    return correct / float(actual) * 100.0

# Test training backprop algorithm
#dataset=open('C:/Users/Rajitha Bhavani/Documents/upsampled.txt','r')
#dataset=[[0,5.239742700000001,-6.3819862999999994],[1,9.716892,-9.001553900000001]];
#test=[[0,7.937372200000001,-7.651209599999999],[1,2.7600646,-2.6970943]];
ep=list()
#sum_errors=back_propagation(dataset,test,0.1,1000,55)
scores=list()
n_inputs = 27
n_outputs = len(set([row[0] for row in dataset]))
#network = initialize_network(n_inputs, 55, n_outputs)
sum_errors=train_network(network, dataset, 0.1, 1000, n_outputs)
TN=0
TP=0
FN=0
FP=0
for row in test:
    prediction = predict(network, row)
    if row[0]==0:
        if prediction==row[0]:
            TN+=1
        else:
            FN+=1
    if row[0]==1:
        if prediction==row[0]:
```

```
            TP+=1
        else:
            FP+=1
print(TN,FN,TP,FP)
if TP!=0 or FN!=0:
    SN=TP/TP+FN
    print(SN)
if TN!=0 or FP!=0:
    SP=TN/TN+FP
    print(SP)
plt.plot(ep,sum_errors)
plt.show()
#print('Scores: %s' % scores)
#print('Mean Accuracy: %.3f%%' % (sum(scores)/float(len(scores))))
```

Case 1:

```
from math import exp
from random import seed
from random import random
import matplotlib.pyplot as plt
import numpy as np

# Initialize a network
def initialize_network(n_inputs, n_hidden, n_outputs):
    network = list()
    hidden_layer = [{'weights':[random() for i in range(n_inputs)]} for i in range(n_hidden)]
    network.append(hidden_layer)
```

```
    output_layer = [{'weights':[random() for i in range(n_hidden)]} for i in range(n_outputs)]
    network.append(output_layer)
    print(network)
    return network

# Calculate neuron activation for an input
def activate(weights, inputs):
    activation = 0
    for i in range(len(weights)):
        activation += weights[i] * inputs[i]
    return activation

# Transfer neuron activation
def transfer(activation):
    return 1.0 / (1.0 + exp(-activation))

# Forward propagate input to a network output
def forward_propagate(network, row):
    inputs = row
    for layer in network:
        new_inputs = []
        for neuron in layer:
            activation = activate(neuron['weights'], inputs)
            neuron['output'] = transfer(activation)
            new_inputs.append(neuron['output'])
        inputs = new_inputs
    return inputs
```

```
# Calculate the derivative of an neuron output
def transfer_derivative(output):
    return output * (1.0 - output)

# Backpropagate error and store in neurons
def backward_propagate_error(network, expected):
    for i in reversed(range(len(network))):
        layer = network[i]
        errors = list()
        if i != len(network)-1:
            for j in range(len(layer)):
                error = 0.0
                for neuron in network[i + 1]:
                    error += (neuron['weights'][j] * neuron['delta'])
                errors.append(error)
        else:
            for j in range(len(layer)):
                neuron = layer[j]
                errors.append(expected[j] - neuron['output'])
        for j in range(len(layer)):
            neuron = layer[j]
            neuron['delta'] = errors[j] * transfer_derivative(neuron['output'])

# Update network weights with error
def update_weights(network, row, l_rate):
    for i in range(len(network)):
```

```
        inputs = row[1:]
        if i != 0:
            inputs = [neuron['output'] for neuron in network[i - 1]]
        for neuron in network[i]:
            for j in range(len(inputs)):
                neuron['weights'][j] += l_rate * neuron['delta'] * inputs[j]
            #neuron['weights'][-1] += l_rate * neuron['delta']

# Train a network for a fixed number of epochs
def train_network(network, train, l_rate, n_epoch, n_outputs):
    sum_errors=0
    errors=list()
    while sum_errors<1:
        for epoch in range(n_epoch):
            sum_error = 0
            for row in train:
                #print(row)
                outputs = forward_propagate(network, row)
                #print(outputs)
                expected = [0 for i in range(n_outputs)]
                expected[row[0]] = 1
                sum_error += sum([(expected[i]-outputs[i])**2 for i in range(len(expected))])/2
                backward_propagate_error(network, expected)
                update_weights(network, row, l_rate)
            if epoch!=0:
                sum_errors+=sum_error
            if epoch!=0 and epoch%100==0:
```

```
            errors.append(sum_errors*2)
            ep.append(epoch)
            print('>epoch=%d, lrate=%.3f, error=%.3f' % (epoch, l_rate, sum_errors*2))
            sum_errors=0
    return errors

def predict(network, row):
    outputs = forward_propagate(network, row)
    return outputs.index(max(outputs))

# Backpropagation Algorithm With Stochastic Gradient Descent
def back_propagation(train, test, l_rate, n_epoch, n_hidden):
    scores=list()
    n_inputs = 27
    n_outputs = len(set([row[0] for row in train]))
    network = initialize_network(n_inputs, n_hidden, n_outputs)
    sum_errors=train_network(network, train, l_rate, n_epoch, n_outputs)
    predictions = list()
    #for row in test:
    #   prediction = predict(network, row)
    #    predictions.append(prediction)
    #    accuracy=accuracy_metric(row[0],predictions)
    #    scores.append(accuracy)
    #return(scores)
    return sum_errors

def accuracy_metric(actual, predicted):
```

```
    correct = 0
    for i in range(actual):
        print(actual)
        print(predict[i])
        if actual == predicted:
            correct += 1
    return correct / float(actual) * 100.0

# Test training backprop algorithm
#dataset=open('C:/Users/Rajitha Bhavani/Documents/upsampled.txt','r')
#dataset=[[0,5.239742700000001,-6.381986299999994],[1,9.716892,-9.001553900000001]];
#test=[[0,7.937372200000001,-7.651209599999999],[1,2.7600646,-2.6970943]];
ep=list()
#sum_errors=back_propagation(dataset,test,0.1,1000,55)
scores=list()
n_inputs = 27
n_outputs = len(set([row[0] for row in dataset]))
network = initialize_network(n_inputs, 55, n_outputs)
sum_errors=train_network(network, dataset, 0.1, 4000, n_outputs)
TN=0
TP=0
FN=0
FP=0
for row in test:
    prediction = predict(network, row)
    if row[0]==0:
        if prediction==row[0]:
```

```
            TN+=1
        else:
            FN+=1
    if row[0]==1:
        if prediction==row[0]:
            TP+=1
        else:
            FP+=1
print(TN,FN,TP,FP)
if TP!=0 or FN!=0:
    SN=TP/TP+FN
    print(SN)
if TN!=0 or FP!=0:
    SP=TN/TN+FP
    print(SP)
plt.plot(ep,sum_errors)
plt.show()
#print('Scores: %s' % scores)
#print('Mean Accuracy: %.3f%%' % (sum(scores)/float(len(scores))))
```

Case 4:

```
from math import exp
from random import seed
from random import random
import matplotlib.pyplot as plt
import numpy as np

# Initialize a network
```

```
def initialize_network(n_inputs, n_hidden, n_outputs):
    network = list()
    hidden_layer = [{'weights':[random() for i in range(n_inputs+1)]} for i in range(n_hidden)]
    network.append(hidden_layer)
    output_layer = [{'weights':[random() for i in range(n_hidden+1)]} for i in range(n_outputs)]
    network.append(output_layer)
    print(network)
    return network

# Calculate neuron activation for an input
def activate(weights, inputs):
    activation = weights[-1]
    for i in range(len(weights)-1):
        activation += weights[i] * inputs[i]
    return activation

# Transfer neuron activation
def transfer(activation):
    return 1.0 / (1.0 + exp(-activation))

# Forward propagate input to a network output
def forward_propagate(network, row):
    inputs = row
    for layer in network:
        new_inputs = []
        for neuron in layer:
            activation = activate(neuron['weights'], inputs)
```

```
            neuron['output'] = transfer(activation)
            new_inputs.append(neuron['output'])
        inputs = new_inputs
    return inputs

# Calculate the derivative of an neuron output
def transfer_derivative(output):
    return output * (1.0 - output)

# Backpropagate error and store in neurons
def backward_propagate_error(network, expected):
    for i in reversed(range(len(network))):
        layer = network[i]
        errors = list()
        if i != len(network)-1:
            for j in range(len(layer)):
                error = 0.0
                for neuron in network[i + 1]:
                    error += (neuron['weights'][j] * neuron['delta'])
                errors.append(error)
        else:
            for j in range(len(layer)):
                neuron = layer[j]
                errors.append(expected[j] - neuron['output'])
        for j in range(len(layer)):
            neuron = layer[j]
            neuron['delta'] = errors[j] * transfer_derivative(neuron['output'])
```

```
# Update network weights with error
def update_weights(network, row, l_rate):
    for i in range(len(network)):
        inputs = row[1:]
        prevval=0
        if i != 0:
            inputs = [neuron['output'] for neuron in network[i - 1]]
        velocity=[None]*len(inputs)
        for neuron in network[i]:
            for j in range(len(inputs)):
                if j==1:
                    velocity[j]=l_rate * neuron['delta'] * inputs[j]
                else:
                    velocity[j]=l_rate * neuron['delta'] * inputs[j] + 0.5 * prevval
                prevval=velocity[j]
                neuron['weights'][j] += velocity[j]
            neuron['weights'][-1] += l_rate * neuron['delta']

# Train a network for a fixed number of epochs
def train_network(network, train, l_rate, n_epoch, n_outputs):
    sum_errors=0
    errors=list()
    while sum_errors<1:
        for epoch in range(n_epoch):
            sum_error = 0
            for row in train:
```

```
            #print(row)
            outputs = forward_propagate(network, row)
            #print(outputs)
            expected = [0 for i in range(n_outputs)]
            expected[row[0]] = 1
            sum_error += sum([(expected[i]-outputs[i])**2 for i in range(len(expected))])/2
            backward_propagate_error(network, expected)
            update_weights(network, row, l_rate)
        if epoch!=0:
            sum_errors+=sum_error
        if epoch!=0 and epoch%100==0:
            errors.append(sum_errors*2)
            ep.append(epoch)
            print('>epoch=%d, lrate=%.3f, error=%.3f' % (epoch, l_rate, sum_errors*2))
            sum_errors=0
    return errors

def predict(network, row):
    outputs = forward_propagate(network, row)
    return outputs.index(max(outputs))

# Backpropagation Algorithm With Stochastic Gradient Descent
def back_propagation(train, test, l_rate, n_epoch, n_hidden):
    scores=list()
    n_inputs = 27
    n_outputs = len(set([row[0] for row in train]))
    network = initialize_network(n_inputs, n_hidden, n_outputs)
```

```
    sum_errors=train_network(network, train, l_rate, n_epoch, n_outputs)
    predictions = list()
    #for row in test:
    #   prediction = predict(network, row)
    #    predictions.append(prediction)
    #    accuracy=accuracy_metric(row[0],predictions)
    #    scores.append(accuracy)
    #return(scores)
    return sum_errors

def accuracy_metric(actual, predicted):
    correct = 0
    for i in range(actual):
        print(actual)
        print(predict[i])
        if actual == predicted:
            correct += 1
    return correct / float(actual) * 100.0

# Test training backprop algorithm
#dataset=open('C:/Users/Rajitha Bhavani/Documents/upsampled.txt','r')
#dataset=[[0,5.239742700000001,-6.3819862999999994],[1,9.716892,-9.001553900000001]];
#test=[[0,7.937372200000001,-7.651209599999999],[1,2.7600646,-2.6970943]];
ep=list()
#sum_errors=back_propagation(dataset,test,0.1,1000,55)
scores=list()
n_inputs = 27
```

```
n_outputs = len(set([row[0] for row in dataset]))
#network = initialize_network(n_inputs, 55, n_outputs)
sum_errors=train_network(network, dataset, 0.1, 1000, n_outputs)
TN=0
TP=0
FN=0
FP=0
for row in test:
    prediction = predict(network, row)
    if row[0]==0:
        if prediction==row[0]:
            TN+=1
        else:
            FN+=1
    if row[0]==1:
        if prediction==row[0]:
            TP+=1
        else:
            FP+=1
print(TN,FN,TP,FP)
if TP!=0 or FN!=0:
    SN=TP/TP+FN
    print(SN)
if TN!=0 or FP!=0:
    SP=TN/TN+FP
    print(SP)
plt.plot(ep,sum_errors)
```

```
plt.show()
#print('Scores: %s' % scores)
#print('Mean Accuracy: %.3f%%' % (sum(scores)/float(len(scores))))
```

Case 2:

```
from math import exp
from random import seed
from random import random
import matplotlib.pyplot as plt
import numpy as np

# Initialize a network
def initialize_network(n_inputs, n_hidden, n_outputs):
    network = list()
    hidden_layer = [{'weights':[random() for i in range(n_inputs+1)]} for i in range(n_hidden)]
    network.append(hidden_layer)
    output_layer = [{'weights':[random() for i in range(n_hidden+1)]} for i in range(n_outputs)]
    network.append(output_layer)
    print(network)
    return network

# Calculate neuron activation for an input
def activate(weights, inputs):
    activation = weights[-1]
    for i in range(len(weights)-1):
        activation += weights[i] * inputs[i]
    return activation
```

```
# Transfer neuron activation
def transfer(activation):
    return 1.0 / (1.0 + exp(-activation))

# Forward propagate input to a network output
def forward_propagate(network, row):
    inputs = row
    for layer in network:
        new_inputs = []
        for neuron in layer:
            activation = activate(neuron['weights'], inputs)
            neuron['output'] = transfer(activation)
            new_inputs.append(neuron['output'])
        inputs = new_inputs
    return inputs

# Calculate the derivative of an neuron output
def transfer_derivative(output):
    return output * (1.0 - output)

# Backpropagate error and store in neurons
def backward_propagate_error(network, expected):
    for i in reversed(range(len(network))):
        layer = network[i]
        errors = list()
        if i != len(network)-1:
            for j in range(len(layer)):
```

```
            error = 0.0
            for neuron in network[i + 1]:
                error += (neuron['weights'][j] * neuron['delta'])
            errors.append(error)
        else:
            for j in range(len(layer)):
                neuron = layer[j]
                errors.append(expected[j] - neuron['output'])
        for j in range(len(layer)):
            neuron = layer[j]
            neuron['delta'] = errors[j] * transfer_derivative(neuron['output'])

# Update network weights with error
def update_weights(network, row, l_rate):
    for i in range(len(network)):
        inputs = row[1:]
        if i != 0:
            inputs = [neuron['output'] for neuron in network[i - 1]]
        for neuron in network[i]:
            for j in range(len(inputs)):
                neuron['weights'][j] += l_rate * neuron['delta'] * inputs[j]
            neuron['weights'][-1] += l_rate * neuron['delta']

# Train a network for a fixed number of epochs
def train_network(network, train, l_rate, n_epoch, n_outputs):
    sum_errors=0
    errors=list()
```

```
    while sum_errors<1:
        for epoch in range(n_epoch):
            sum_error = 0
            for row in train:
                #print(row)
                outputs = forward_propagate(network, row)
                #print(outputs)
                expected = [0 for i in range(n_outputs)]
                expected[row[0]] = 1
                sum_error += sum([(expected[i]-outputs[i])**2 for i in range(len(expected))])/2
                backward_propagate_error(network, expected)
                update_weights(network, row, l_rate)
            if epoch!=0:
                sum_errors+=sum_error
            if epoch!=0 and epoch%100==0:
                errors.append(sum_errors*2)
                ep.append(epoch)
                print('>epoch=%d, lrate=%.3f, error=%.3f' % (epoch, l_rate, sum_errors*2))
                sum_errors=0
    return errors

def predict(network, row):
    outputs = forward_propagate(network, row)
    return outputs.index(max(outputs))

# Backpropagation Algorithm With Stochastic Gradient Descent
def back_propagation(train, test, l_rate, n_epoch, n_hidden):
```

```
    scores=list()
    n_inputs = 27
    n_outputs = len(set([row[0] for row in train]))
    network = initialize_network(n_inputs, n_hidden, n_outputs)
    sum_errors=train_network(network, train, l_rate, n_epoch, n_outputs)
    predictions = list()
    #for row in test:
    #   prediction = predict(network, row)
    #    predictions.append(prediction)
    #    accuracy=accuracy_metric(row[0],predictions)
    #    scores.append(accuracy)
    #return(scores)
    return sum_errors

def accuracy_metric(actual, predicted):
    correct = 0
    for i in range(actual):
        print(actual)
        print(predict[i])
        if actual == predicted:
            correct += 1
    return correct / float(actual) * 100.0

# Test training backprop algorithm
#dataset=open('C:/Users/Rajitha Bhavani/Documents/upsampled.txt','r')
#dataset=[[0,5.239742700000001,-6.3819862999999994],[1,9.716892,-9.001553900000001]];
#test=[[0,7.937372200000001,-7.651209599999999],[1,2.7600646,-2.6970943]];
```

```
ep=list()

#sum_errors=back_propagation(dataset,test,0.1,1000,55)

scores=list()

n_inputs = 27

n_outputs = len(set([row[0] for row in dataset]))

network = initialize_network(n_inputs, 55, n_outputs)

sum_errors=train_network(network, dataset, 0.1, 4000, n_outputs)

TN=0

TP=0

FN=0

FP=0

for row in test:

   prediction = predict(network, row)

   if row[0]==0:

      if prediction==row[0]:

         TN+=1

      else:

         FN+=1

   if row[0]==1:

      if prediction==row[0]:

         TP+=1

      else:

         FP+=1

print(TN,FN,TP,FP)

if TP!=0 or FN!=0:

   SN=TP/TP+FN

   print(SN)
```

```
if TN!=0 or FP!=0:
    SP=TN/TN+FP
    print(SP)
plt.plot(ep,sum_errors)
plt.show()
#print('Scores: %s' % scores)
#print('Mean Accuracy: %.3f%%' % (sum(scores)/float(len(scores))))
```

<u>Weights from Case 1:</u>

[[{'weights': [0.041163081276754054, 0.6377906283355136, 0.4152200353796044,
0.8496307142028408, 0.31800933869593484, 0.3367615232463541, 0.9724807586102966,
0.33916253070657076, 0.34072253831961963, 0.1494637049137224, 0.01767745539059029,
0.9131258151612119, 0.6807107150691365, 0.3402942394805606, 0.9540410018891469,
0.99901458509961, 0.9821195581910707, 0.12697312236943403, 0.927735297438272,
0.023777579376491387, 0.1909016912584025, 0.6752140974356449, 0.4499854202652265,
0.7836457931397884, 0.6632256204208304, 0.3856408602032878, 0.7794219470339151]},
{'weights': [0.737421437491662, 0.9330930677695038, 0.36889767862834044,
0.6336014376889871, 0.01781395763857463, 0.05708335999077607, 0.14857134587887388,
0.9571664018728443, 0.3142792925446364, 0.1819685396264621, 0.4513776505561623,
0.04568763533961051, 0.1245137108959431, 0.8497365143284102, 0.6966731672498668,
0.17595580456475668, 0.7295014845716674, 0.013812919190690387,
0.40793300920136977, 0.7654972374869221, 0.23276022994978218, 0.31025840399175464,
0.29143955105481856, 0.5001076254089749, 0.9489418511625463, 0.8285389563462908,
0.6961188115859935]}, {'weights': [0.17117062667098926, 0.5603442608536228,
0.146780417370737, 0.008685099312668698, 0.3879490142816291, 0.6786973868818807,
0.8197988186877918, 0.9963324387992746, 0.3475719776535775, 0.5589585908008226,
0.00853302282384183, 0.8808430862641461, 0.23284119780496004, 0.17241061439389427,
0.7786719211512703, 0.22380095612176687, 0.521425726755379, 0.3424843823649153,
0.1111898220393801, 0.3696179038927613, 0.020548679727064867, 0.8284824331980596,
0.4138038522099413, 0.3933264007321041, 0.9967797287663973, 0.03270793022042651,
0.8787578289397984]}, {'weights': [0.13644510852350356, 0.7511224886573714,
0.23761396749001318, 0.11450876369650964, 0.037579195776157426, 0.381006410452031,
0.8262970594540042, 0.06932953472650638, 0.7317214301017869, 0.21148155268525548,
0.17122920678871634, 0.3905203991685774, 0.32333719145928186, 0.9083961660076539,
0.3205710115348266, 0.06320020807895, 0.21410122676833043, 0.7515753871896764,
0.05241523235287926, 0.2710611795893737, 0.2583067948162654, 0.5657244332708494,

0.6202848829807525, 0.2770314198970343, 0.5792434740137856, 0.15044984291896413, 0.8740121294827596]}, {'weights': [0.09408476086655138, 0.29439479863616413, 0.5369217834129922, 0.3970075007641716, 0.37651531442027675, 0.23029017093163584, 0.6634142392817978, 0.8827775979791854, 0.7570173510888868, 0.2579578574045645, 0.36336906785140377, 0.5670601189063806, 0.2181259450873002, 0.14632642797247108, 0.9841351976923925, 0.861469454601801 4, 0.5754854931623444, 0.3126737267189722, 0.33523331672018186, 0.07987048334509372, 0.672930472779507, 0.7772594368325201, 0.38203802332568815, 0.9170305535287571, 0.6864672236738087, 0.8350595877728159, 0.00585704933264941 1]}, {'weights': [0.31712029843640555, 0.659620222501721, 0.9231468343597554, 0.5598384474129132, 0.5874116170956892, 0.05231580344483821 6, 0.22452106719213605, 0.9778191016681664, 0.0631202815566105 4, 0.3605980853126102, 0.4203576601823116, 0.5596463274010675, 0.5336310523765294, 0.552654896329415, 0.18275727359725003, 0.36982345436249386, 0.759446360614904, 0.27759025311031427, 0.4088920789533127, 0.3178107674934686, 0.8799720850666596, 0.8367622614376264, 0.07098448562354887, 0.1464063067056519, 0.320465540648579, 0.587121183251065, 0.8262408952829133]}, {'weights': [0.3486983879936395, 0.866567072538052, 0.5148680182824582, 0.7809693889897352, 0.8429064724553098, 0.18040406525040187, 0.5536655874405051, 0.30622588330154354, 0.5130338547334838, 0.01580444392382363, 0.1510586003815425, 0.6673172933191878, 0.7399664022871181, 0.4169112982241169, 0.940914722908228 5, 0.9428087684814316, 0.02708918644154523 2, 0.45995521581877186, 0.15621297098559273, 0.7775096341004581, 0.30763487381749255, 0.3395795226994709, 0.9362189419363786, 0.9498419536933961, 0.9013239032139067, 0.9108063761345342, 0.17666210034036123]}, {'weights': [0.6168566214294344, 0.2295441198951228, 0.9488891322986445, 0.505548542187176, 0.3640826994672267, 0.11102289231453943, 0.03597153000445463, 0.9315947179163715, 0.9168080174184339, 0.16178039857680648, 0.3710673229599609, 0.01744438950443283 3, 0.23646939356711083, 0.0600084243798954, 0.9844368542582447, 0.00606403092641227 8, 0.5341341470211424, 0.03799273736296638, 0.27634534239143604, 0.5071522943086342, 0.441507254135476, 0.02688182269915984, 0.5265345007687485, 0.5171057981036714, 0.050687804306853046, 0.00817505400245188 7, 0.13014485292143474]}, {'weights': [0.7313703153460455, 0.5710763264749391, 0.3344774653711493, 0.5939912432200006, 0.5354738347133813, 0.5488462461783499, 0.9162593472788438, 0.8694836990310876, 0.06814479750124292, 0.07597319543372816, 0.8354993966825043, 0.4023942897058471, 0.5354567614301349, 0.06585079447905884, 0.6355617405897329, 0.24792232772856648, 0.9248596843343274, 0.36141288933768934, 0.6630348185391979, 0.8339840723257733, 0.5401767266873462, 0.5634266404590046, 0.9242848389321582, 0.7057311543222763, 0.7445421509450417, 0.53336533205446, 0.02026149353989759]}, {'weights': [0.3683653326137891, 0.5013358556854228, 0.1468842848477998, 0.18028945705980637, 0.33364535122043915, 0.24361630725702277, 0.5707290006425348, 0.9041502012216761, 0.1473704056169266, 0.5644537842683397, 0.4365835032725016, 0.7778007820222985, 0.2580801878696195, 0.811849671264069, 0.17062012657443737, 0.8397786188925132, 0.5921651143593797, 0.8711677696146476, 0.743231230842268, 0.6412037846639668, 0.18628556706471044, 0.38968721896997804, 0.0016849906208704635, 0.5567536625003895, 0.8202884248600099, 0.9548367748062159, 0.34160178303044364]}, {'weights': [0.5867828808558864, 0.5496397225458919, 0.3184956032214111, 0.7485494614734766, 0.4282578264320934, 0.9942105147289837, 0.35883653620515976, 0.4174197223732027, 0.35440099188601293, 0.6167219688961115, 0.2810254703949857, 0.1849716239024748, 0.8456144316467578, 0.26149995486897093, 0.7316353736808832, 0.6762743348675202, 0.018715112001257417, 0.5634083288868874, 0.23182970132819936, 0.4360832444661519, 0.006229539272359141, 0.8938782883468789, 0.9856058442689533, 0.7283568845478846, 0.5254492175867619, 0.978270425479032, 0.728036327257439]}, {'weights': [0.2132182039548164, 0.3233391638960269, 0.08898885330759398, 0.8176225018916926, 0.01948433768750324, 0.7917281752380895, 0.3782184937909797, 0.7369857088330728, 0.24524881762693362, 0.38491634349843873, 0.6036122997663227, 0.7441130410174317, 0.7969568781855041, 0.6062544415226185, 0.40755768164193185, 0.6060467297402484, 0.48903841075888665, 0.26614118412157717, 0.49383949170116503, 0.19277056302534246, 0.0591825467446232, 0.8015232089060744, 0.21612375922370952, 0.4215559494276373, 0.28753189806428536, 0.8135018679708642, 0.18414311730413158]}, {'weights': [0.36576644194199637, 0.6042537142713093, 0.010035643589808352, 0.050757282944500326, 0.5546185871843438, 0.143309174082901, 0.6414962560150097, 0.9877296025443884, 0.5465721206557824, 0.3984034735992743, 0.20575601466058513, 0.9383288233989562, 0.6873750699155526, 0.14827877666874756, 0.14603043411950012, 0.4764561130816656, 0.7676320193725003, 0.033419691480554126, 0.04683982211005766, 0.600740966014803, 0.15021557155304266, 0.6044932736907557, 0.2954420824740518, 0.2983267176763521, 0.6109003117744928, 0.1859003048054495, 0.5217745160254793]}, {'weights': [0.12212151506609603, 0.7826013131919212, 0.1843832250434142, 0.39322967005393583, 0.5005040083652093, 0.9041479813811168, 0.8250107744916526, 0.2304294836088715, 0.9644850467943334, 0.4797609038980527, 0.32952532714756866, 0.9849884116890177, 0.5878439113216886, 0.017359048347206696, 0.11506425432514034, 0.9385038490469174, 0.9343264287230141, 0.7856835285148871, 0.6950268040519426, 0.36052644955200397, 0.8159149105183544, 0.2761470021548883, 0.010445885358409401, 0.9782351102991957, 0.6757531207411758, 0.4767956902732209, 0.10743149835563215]}, {'weights': [0.9107978292213675, 0.6739925438249641, 0.6047173871694327, 0.9870057128065712, 0.4755075390239566, 0.7579914479899728, 0.8117512098626064, 0.6591937704116957, 0.09340319742615666, 0.024965160283260013, 0.5492804753075515, 0.20473397656762204, 0.504942884590601, 0.4760251894765092, 0.20859174674842285, 0.27117678149508595, 0.48751321203666165, 0.08657738926812619, 0.5055416231582459, 0.8820708666844962, 7.585736040138791e-05, 0.5423450035485373, 0.49225919342003466, 0.42059782839987614, 0.9739062828035326, 0.2193036614485775, 0.2362759674128505]}, {'weights': [0.24548673910772723, 0.8623967021173636, 0.4820607799570584, 0.8723256530172202, 0.19781052206838612, 0.7143596895065354, 0.5576384562975123, 0.5482693564341865, 0.5412717253979459, 0.7496876406442702, 0.6524966660037461,

0.7298853743418214, 0.9132209230749558, 0.9708769451946865, 0.794470974962918,
0.993366278265789, 0.6750600789269042, 0.5731194881192604, 0.08972934480471217,
0.19245574341526306, 0.8602374617336731, 0.20637366456260964, 0.3353659111178955,
0.7621803201485979, 0.8765188676522119, 0.2507396427275951, 0.768035998550011]},
{'weights': [0.9059785052017223, 0.8550713498291498, 0.9337107036128719,
0.49425351489754377, 0.5983748135147877, 0.581160104646514 7, 0.5274038348076294,
0.06374514586068714, 0.5286622741259339, 0.34732591475106855, 0.958848390216747,
0.7827783713846987, 0.13427910472784355, 0.9997007855788399, 0.32972329751453766,
0.1782065595695016, 0.09990723746079899, 0.7357097927345396, 0.8550779927358964,
0.6957817809893589, 0.09558083691181096, 0.9369576249741344, 0.786832573607903,
0.5896748089630109, 0.4306112631038441, 0.3968965527317544, 0.3874643987250812]},
{'weights': [0.006728714230239574, 0.9886453692094215, 0.988769231794373,
0.5312458931572462, 0.2025274850342491, 0.18250297419322226, 0.7471767643609323,
0.7873356951055474, 0.5906103732356136, 0.6285952266633751, 0.7187700097195995,
0.699091699846886, 0.11676731338762858, 0.8892512400576249, 0.13078788585454437,
0.26225371677185205, 0.7094589109343252, 0.34995042226369855, 0.996540408739643,
0.29146587028018656, 0.017512003252882935, 0.48800060154126534,
0.28675095377641013, 0.48802834805210227, 0.7029291599822302, 0.32664299248703954,
0.7560702452996254]}, {'weights': [0.40218560050654295, 0.09031469014745153,
0.27073433003409786, 0.5712870019366115, 0.19701550322178973, 0.3755470920132622,
0.046184267508497134, 0.17354770949534348, 0.6844940827387683, 0.071665914104316,
0.4010643508579804, 0.30043077679495767, 0.06733199552329672, 0.10549514830463702,
0.37193164565375436, 0.39015415235989215, 0.38835139648392003, 0.681727154232445,
0.46219347241652053, 0.4018120576116275, 0.4631864498134646, 0.62321292057731,
0.7361684041352596, 0.24379126562929465, 0.7510726512690359, 0.18215490904546716,
0.12316683584028787]}, {'weights': [0.8673982589607152, 0.29291069845983964,
0.9137138006908627, 0.2684599567019649, 0.10145319551855725, 0.7995920890246297,
0.3449602839510756, 0.41291471480210884, 0.5669276698874708, 0.2315813852343479,
0.6320404113049597, 0.2575754944075449, 0.23429073290533287, 0.8819563741753323,
0.005081542240227943, 0.2447116268235544, 0.9312791863042413, 0.24582525963118707,
0.8044674073895616, 0.015958643282403262, 0.42180712646830265,
0.47360328141341956, 0.7804626778034205, 0.8015899516757236, 0.7275519835677103,
0.5120285804842124, 0.4827622625617193]}, {'weights': [0.8427255147276052,
0.23350849608098012, 0.6808683204666389, 0.02423048672234074, 0.6413931434041695,
0.010039436005229163, 0.435898412128641, 0.6053539222191816, 0.4504036697305792,
0.05843211215888222, 0.38574492335237387, 0.4394645804754066, 0.5722535645835972,
0.6252605350643288, 0.13837530299697332, 0.07640097726420247, 0.15731611425835168,
0.7175930625487135, 0.46763382843844936, 0.14938767965983024, 0.8794835703688467,
0.2835961408804457, 0.7878707693548389, 0.14009900559041588, 0.8674102406249617,
0.4310671286808868, 0.8741408288556239]}, {'weights': [0.12717385035393203,
0.43499258814834263, 0.716100910984818, 0.13180328661071616, 0.6937984290299992,

0.6753560209040952, 0.19529498986189786, 0.6710486174688579, 0.26455791734408607, 0.1764513855007075, 0.829630108421913, 0.140974060709268, 0.2658840800874487, 0.4782685395648325, 0.12122903321972311, 0.6558842521867869, 0.886012194699477, 0.4351073694981227, 0.5802979344493138, 0.5626662172581437, 0.9183539809636956, 0.474974998177246, 0.6603952023011508, 0.694127670913984, 0.23996155178847012, 0.1261564943294019, 0.6687142646398104]}, {'weights': [0.4513066039174932, 0.8850996742213096, 0.7080592011116498, 0.8767580888963209, 0.43572766805906826, 0.38449798070578356, 0.23875273353255344, 0.8547637270621686, 0.9338239092925766, 0.5445769094146039, 0.3702615195479748, 0.3032450600236988, 0.7548360469327994, 0.07895975049628079, 0.6532805998604109, 0.22163605668490183, 0.821484825431 6215, 0.828682264616332, 0.9426818986287008, 0.2883993180164154, 0.613081902546444, 0.7587617661846738, 0.24567725070351198, 0.86184877255385, 0.7498375016738665, 0.09933617647837578, 0.3101404684535607]}, {'weights': [0.5684707887604267, 0.9590724445626572, 0.34199442630578825, 0.34436493809611013, 0.4322597230904789, 0.717034476391079, 0.5333040302974821, 0.5352966458880474, 0.6434851788160479, 0.6301337325980206, 0.22883175373612596, 0.2107946879915985, 0.6744022759191833, 0.15196967595503308, 0.07513136406029874, 0.5398054348755889, 0.2537264775875955, 0.9538097908000099, 0.18475097561991505, 0.667992343887654, 0.07959320160316097, 0.015861419074299832, 0.8352162959710634, 0.4543266951803405, 0.6500722617888518, 0.09484127052986313, 0.6558687046232602]}, {'weights': [0.43177963943996556, 0.6714894133719638, 0.8375413658140107, 0.6690704453194274, 0.6142204876910048, 0.4030622416749716, 0.737334525482309, 0.7030329799739452, 0.8620369620013177, 0.03399381669134116, 0.7983621049721278, 0.8033479065128074, 0.4940837395039491, 0.02719885053103488, 0.6725746979982543, 0.29121217341183414, 0.13280216761223673, 0.9365751476098538, 0.639733881257824, 0.48348388988964497, 0.29618950408424816, 0.4174933056517378, 0.05748924068075445, 0.23128964012076214, 0.7767986160359431, 0.32522939384759797, 0.8187402690722237]}, {'weights': [0.4414380145994422, 0.3032712248599574, 0.9812789750629921, 0.6178617467880693, 0.6191943290872428, 0.5269000091584766, 0.7375584183827673, 0.7018728850858327, 0.9862970630651716, 0.5191774981185175, 0.5637767581539952, 0.611349935384245, 0.4561373844317892, 0.05297698038329701, 0.09511709107988087, 0.661951988543667, 0.9731173920732354, 0.1938873440761596, 0.30323601283822543, 0.5422094086704934, 0.9581400335691422, 0.6862003728707472, 0.5150195950590776, 0.841786487426633, 0.42632097992919593, 0.5255225699875032, 0.975216435527314]}, {'weights': [0.2856486306852162, 0.5253856021943345, 0.9676413566040992, 0.010284182173986745, 0.5471688324407683, 0.6252547953039419, 0.7113662413580758, 0.4453704709097833, 0.6473791171359662, 0.724253081053699, 0.5479705934308584, 0.89360426712902, 0.5807458352640983, 0.623260044724789, 0.310638103654569, 0.6031052489663276, 0.6176584636502703, 0.7619086222757253, 0.298592294314853, 0.7687730334529224, 0.4297241971461414, 0.8209627690501566, 0.7611500064385581, 0.8691944431538546, 0.25859543580776745, 0.9721207772272688, 0.46508609018401637]}, {'weights': [0.06633077700471379,

0.8382750841697703, 0.4692901359234498, 0.3896392442654649, 0.8998791140805872,
0.7943975752149779, 0.106197099608625, 0.4813209538214611, 0.6355273545219708,
0.2249303941111509, 0.8731232730087728, 0.6183225338480138, 0.4615253346077537,
0.3644666405110355, 0.18488441291361324, 0.7508410687082012, 0.44765414465578024,
0.3672456574973032, 0.3340123427834193, 0.15701317293132921, 0.1726039599651341,
0.35827269507593296, 0.9326098889929402, 0.8893480693007301, 0.004300362927250556,
0.3633028389732855, 0.04774760376792109]}, {'weights': [0.7766805613092541,
0.6613726334891079, 0.5806565915752664, 0.20137619009280283, 0.6782786816193904,
0.061286391154789954, 0.1088293448530524, 0.015653890678903548,
0.5231146613472923, 0.105792058426732, 0.6504222813502639, 0.2797036170992635,
0.7575588931250081, 0.030059042249319434, 0.7412196470041775, 0.24432545226534053,
0.44732917536607053, 0.030341593141274048, 0.9504156572877127, 0.8927888723707702,
0.4662585253619791, 0.47979937343230883, 0.9267671247078758, 0.047008919956617934,
0.5741740208164136, 0.018386583018367375, 0.1581897621454652]}, {'weights':
[0.8537422106237135, 0.17496740042296077, 0.9279508846577147, 0.05184195727797647,
0.2864947270632703, 0.9664507702330737, 0.5407884025857427, 0.22546605069632553,
0.0563925625325723, 0.10873150054657321, 0.6312707069700607, 0.28465080156807976,
0.3246657058612443, 0.2894100432657083, 0.6796784070026314, 0.7875811415612183,
0.7759952479555555, 0.9237035784994037, 0.5197886423787359, 0.036006349506055746,
0.8824072724482378, 0.3646260914355407, 0.7664185665644365, 0.41958300672396587,
0.2419783294732215, 0.277437782815533, 0.5931816849423253]}, {'weights':
[0.39031884826221386, 0.3829500996654127, 0.9281988261119519, 0.7518157758083268,
0.24370996706508774, 0.4391259480116362, 0.09781237937760823, 0.43556704184665695,
0.27811950153599097, 0.493186696492668, 0.982166034772336, 0.36436082884406007,
0.3788699259392453, 0.5807202824032424, 0.682463513499749, 0.4341463140445888,
0.8197982910081936, 0.315145745239491, 0.9640322449444482, 0.5390925247513937,
0.5441257494543109, 0.803396978543788, 0.8378943279143508, 0.18134635422655998,
0.29544228642955384, 0.6503137776342195, 0.2446258526457944]}, {'weights':
[0.5064917890015824, 0.8012774995863631, 0.9086429660304812, 0.8326707889671898,
0.6455598102839033, 0.854093496963155, 0.3328250343431608, 0.7272544304957178,
0.7513492705132074, 0.5145894522926281, 0.3242106014012278, 0.3078807852485964,
0.7076921743882868, 0.5643655319919457, 0.5858016167527855, 0.5803763545489415,
0.5907945176570003, 0.7473213124679723, 0.4895235046450992, 0.741903095482034,
0.8420374211489715, 0.6565420452444434, 0.057302483002145244, 0.64837378703713,
0.2960521006826331, 0.9391724934772085, 0.45540044543517566]}, {'weights':
[0.28243407851190594, 0.14111671450881258, 0.1292590152172456, 0.5469042547719645,
0.5654120924633379, 0.1523081612096071, 0.07360738486701368, 0.7112400553075171,
0.8302457516203794, 0.7229680991004735, 0.11203311400755123, 0.06289084560319214,
0.35009584204835076, 0.6895479731451736, 0.8101882994341946, 0.5303354419335623,
0.8245547979001517, 0.5657546095401041, 0.0948035005953145, 0.12371815645365003,
0.1191829058325935, 0.25443506044681574, 0.8662101687796215, 0.20885071116010556,

0.3556070182059562, 0.009555416879350354, 0.2562283577955021]}, {'weights':
[0.6880873877047827, 0.3821837835316244, 0.048750099924059764, 0.769988791623472,
0.11413121384666725, 0.2377653077113846, 0.4387901602998273, 0.5815289344947407,
0.5725729383748869, 0.07303894341264572, 0.6020574497234884, 0.7220900819941499,
0.9141049909178545, 0.1833097996497326, 0.9947839593175242, 0.6128107823334163,
0.9777059215332524, 0.06009029822719403, 0.017283512527329292,
0.10935431203197732, 0.6191414158301823, 0.9377320723864294, 0.6826386543912872,
0.9249854204981979, 0.9005959086124578, 0.7818498205529999, 0.6057215318628555]},
{'weights': [0.28623080260533673, 0.7207962900271628, 0.4227620723721188,
0.5997832770636186, 0.4433664381104536, 0.5925482348058639, 0.6848100298711735,
0.35354641483569804, 0.45415375041217476, 0.5760493446749534, 0.7785372375263152,
0.9152887258449369, 0.47826177958590066, 0.39232535598003593, 0.7603664303763358,
0.26232016640746014, 0.5801670950269202, 0.4931730066970744, 0.8560680598543602,
0.8187649755761043, 0.3050091563482593, 0.7730832457560263, 0.1687074587488071,
0.8931141672393955, 0.3617469317322377, 0.29199122359111873, 0.3851941463921029]},
{'weights': [0.8780351466288921, 0.9050929568533355, 0.13441661441748787,
0.20219014674356617, 0.10277196258541466, 0.38443290582203293, 0.8038289282058291,
0.19098779057489024, 0.6732479881952681, 0.45394048245365715, 0.2154830537175192,
0.8878001531643186, 0.8112819468942287, 0.326849856177715, 0.6199148014524263,
0.12106983430476714, 0.39437576189235113, 0.668397063168841, 0.7530486769739438,
0.35797211271904017, 0.48188352606427776, 0.5910890430683788, 0.6043443413390427,
0.7293428007712965, 0.5397721412103954, 0.14068358062151798, 0.5582424462829845]},
{'weights': [0.3151164572752141, 0.6272961846402176, 0.3010642254649981,
0.3335213752911912, 0.13394908461162947, 0.4699395535524368, 0.9091392466639763,
0.5853933572733357, 0.16216651738521026, 0.16648678340268452, 0.8362864369980316,
0.9945587010323927, 0.3182895157726223, 0.541640264278903, 0.19309354151272673,
0.06477929821349293, 0.3957455535677702, 0.26220078471987196, 0.02176207201557878,
0.5078565918958039, 0.8649839570246339, 0.1760430661288328, 0.9360991425270251,
0.052059030722879096, 0.5969223097108465, 0.6707176949756406, 0.3809795120481483]},
{'weights': [0.6458102217157262, 0.8626696696755408, 0.4189810229513058,
0.22768020896453067, 0.8062472262463447, 0.5031934369254364, 0.2784545258870157,
0.35423884368844893, 0.7557627704782508, 0.7258712717249837, 0.9440837844128448,
0.7113766267816626, 0.9092366260734421, 0.9006343605833379, 0.16131933815921107,
0.42385532856798, 0.31880175624215035, 0.26339352018763795, 0.5046525849413451,
0.6022221636447664, 0.1494920841366676, 0.9149155566408532, 0.8312801614603554,
0.8279060017075447, 0.2094523614045254, 0.8437648928108589, 0.5361570544592377]},
{'weights': [0.3688352722313325, 0.23107582320179942, 0.8267522807390492,
0.6648642716801262, 0.0983362236747668, 0.6653466947060483, 0.028813475276472267,
0.5432311686525776, 0.44690045256397437, 0.1703067335595384, 0.7156288483913142,
0.7832903665110602, 0.5823578374709676, 0.03320985709826019, 0.3533969114196045,
0.040467315437267004, 0.7174485173571393, 0.1044081210780321, 0.7144408343933661,

0.2768242701850988, 0.6438511901668431, 0.2870660120208358, 0.08297950130279541, 0.9552588067177323, 0.5182056484221578, 0.7160528944071535, 0.3184433117032689]}, {'weights': [0.038630796700272474, 0.2441049398442292, 0.7271806603235058, 0.8096581141325773, 0.6164579849857097, 0.761244825341343, 0.6427602082942907, 0.8556881678959161, 0.23926910301255366, 0.14941004920243794, 0.3898065576211195, 0.3389406142156929, 0.09764380875814394, 0.43157131600181065, 0.25331185717322346, 0.10759086123377226, 0.7931776455150399, 0.32832309730267695, 0.4573839833597897, 0.4144785824710624, 0.7713584153814349, 0.04175321729913384, 0.30013596808264, 0.535195733469503, 0.6301329073209941, 0.6071387654822796, 0.24642994142542662]}, {'weights': [0.13695401210846025, 0.7321370901546714, 0.16738134794706405, 0.9238272792330424, 0.8654006855552963, 0.19445366098359596, 0.9148289871534765, 0.6794687502791682, 0.2401982428140208, 0.1566482549704219, 0.2975173133634039, 0.3291307809733195, 0.19921335286199437, 0.8316390947445007, 0.23981234151474862, 0.3684858640896672, 0.5366108486708763, 0.9820867002855073, 0.5914361961362536, 0.788496672558255, 0.6071728793736277, 0.3875965760710671, 0.42895493792219763, 0.15606719188707996, 0.5753646238489409, 0.5880817396423894, 0.5105570636985357]}, {'weights': [0.6583250643036104, 0.7792540594815525, 0.3681177254629402, 0.4933031782589804, 0.48507267013374644, 0.0608646823515141, 0.280639466499469, 0.5792755892515173, 0.8116801197488688, 0.8132316546652859, 0.005576493357166923, 0.31804252304872416, 0.5069259757792042, 0.7143945893793793, 0.6011708831431294, 0.6684450915038288, 0.6343455610354474, 0.5263964943856626, 0.948946101082887, 0.27983058166586927, 0.2686341855395148, 0.012741727913416767, 0.39250196953854255, 0.049377611163728186, 0.19388380353640866, 0.676210465601455, 0.24661698852594305]}, {'weights': [0.21306969054362823, 0.3905364098920584, 0.530925164113823, 0.6058853256823367, 0.4838216471332554, 0.7560655773367475, 0.6351474974689384, 0.6957174252459833, 0.33902154769177073, 0.4818093421805565, 0.9010667488692222, 0.1325074762223445, 0.4958603737561531, 0.7575001281904054, 0.12329612445644766, 0.8276175075709762, 0.01941820191624055, 0.9391729735887946, 0.6900391449574219, 0.0587877973558284, 0.9371383348714178, 0.2579677443176237, 0.8438373420358579, 0.5817632924546752, 0.38644678407697797, 0.4424560388829918, 0.8186861784967253]}, {'weights': [0.8814828868648008, 0.04721558468714815, 0.9112884342281542, 0.3347162809496773, 0.15223810210105948, 0.44449361681164123, 0.25084567115294154, 0.32230291128072075, 0.18414923670821726, 0.15538368348482823, 0.07441555210643924, 0.3742876541685799, 0.32431540448023743, 0.49432000828265354, 0.1998893030347244, 0.23286930524689708, 0.39586405252390755, 0.6720386673468178, 0.48925986409958677, 0.2142376665453265, 0.33075375541646, 0.43839492483094455, 0.4257861581549788, 0.7248965187718992, 0.10007123253083783, 0.14625421091626056, 0.6796141379062234]}, {'weights': [0.05027068413746183, 0.6880940406890296, 0.9272538457376894, 0.08031069389633128, 0.24135542723342074, 0.8889027807176543, 0.6696734293360623, 0.369762523763167, 0.8237332585963205, 0.4455740941748443, 0.578868680056529, 0.16716553384013577, 0.01604390541421097,

0.8503267500796462, 0.9449990618594225, 0.10599241071089915, 0.4170573607468595,
0.597340873591642, 0.7972019555620565, 0.5941246047802528, 0.42898061798052334,
0.4438372006049647, 0.6197609704447009, 0.07376613651660446, 0.5127006824624988,
0.4658027212128173, 0.0418529900568978]}, {'weights': [0.7197804768669627,
0.4329526313028088, 0.042427874791070685, 0.18832211693182077, 0.3016650857777209,
0.48541887446580556, 0.014708790396253613, 0.5883722100711395,
0.20465783931262138, 0.8862428896659801, 0.2151047183530611, 0.46945381961849675,
0.4954657742003603, 0.854663327042195, 0.7995307032274397, 0.14160125558213232,
0.692013018575533, 0.6406207650502332, 0.5185508380288549, 0.14255895749808523,
0.5144470452383997, 0.9173215631426889, 0.19758312314711768, 0.5876615462198232,
0.3468091179519237, 0.06780927607633402, 0.36324328900344205]}, {'weights':
[0.7541777556779443, 0.3994263680203758, 0.8580880903559847, 0.0654741100456302,
0.7934169459779841, 0.9358100108519106, 0.10859432764729371, 0.8500690433809912,
0.4946682977233974, 0.06025430953650279, 0.38097655199886027, 0.37607030572817457,
0.2736303273455736, 0.8808505804903873, 0.3507893355379119, 0.6653855375354656,
0.12798651996475086, 0.857254236067469, 0.7643417059567107, 0.3895832682871857,
0.7073166166507812, 0.4902062443547307, 0.815318879445972, 0.7723459080906269,
0.5997220706761123, 0.1217745592913344, 0.4824761307577683]}, {'weights':
[0.9280053721265988, 0.9841208326689047, 0.8972181048014227, 0.6249342985644118,
0.9840964940830164, 0.40685516424687196, 0.9263102108726509, 0.044542721741282154,
0.3826657564823611, 0.47022832250289415, 0.9702100674845641, 0.18230484135070646,
0.1421798312891004, 0.8857056041095442, 0.5058833655115736, 0.030433007325024408,
0.26901190073951997, 0.6146932955776826, 0.7677299539845724, 0.6466271342279208,
0.04726880884114293, 0.4942467747996414, 0.11324741382185688, 0.9273218603026857,
0.5114619580363544, 0.041315388197145086, 0.410645098301726]}, {'weights':
[0.9761717084751906, 0.7420867773039006, 0.9968596005722017, 0.29524321492120575,
0.3971347420732939, 0.4109542634815886, 0.9753530399415166, 0.3216380281668366,
0.26278584670639105, 0.30073777775400645, 0.030574274450941186,
0.6717942974940703, 0.01745605546356488, 0.1814744487701735, 0.42688882322305666,
0.06530716861466668, 0.20420557562752273, 0.8143234380857004,
0.030560457112067008, 0.30148080464979965, 0.08172445955139807,
0.023258348970606857, 0.5165363456761977, 0.7264614882114213, 0.20559147701687952,
0.10072810230008189, 0.4966714363334698]}, {'weights': [0.08996749552052796,
0.4663888441749392, 0.20020185453015016, 0.6770953439035988, 0.3273403435200207,
0.7880940067899088, 0.24965524210687717, 0.9406287847926228, 0.04061504449785547,
0.27042108092791506, 0.5861806247413575, 0.12908809787202924, 0.4903359140614221,
0.40793877698369674, 0.6836210289332083, 0.8599992749500668, 0.4090491090006102,
0.6950811697542063, 0.07689222217519354, 0.08573097751589576, 0.6402871970868986,
0.94895707986637, 0.6947453323635069, 0.4330020968028342, 0.6455338099302552,
0.7668873457367413, 0.14617699572510645]}, {'weights': [0.4096786916453413,
0.6656969599621398, 0.9374339131600876, 0.9882401089545033, 0.49128265552701655,

0.401630593063534, 0.19310828796771873, 0.565406839261603, 0.7840170927869713,
0.9311646540098165, 0.732316445768893, 0.3231261314595131, 0.515889883690095,
0.737113080148742, 0.16574903369159166, 0.779036177333477, 0.4318672753150715,
0.9892997924370546, 0.694057941717255, 0.18964629215315776, 0.4362938764615002,
0.66019879012912, 0.07397676367494732, 0.34794663993902186, 0.48931288779154536,
0.20108441955485, 0.3516752942597581]}, {'weights': [0.1384277865133796,
0.3892441150595397, 0.4169486469223107, 0.521132206123003, 0.41751278839455364,
0.8309940453470597, 0.8819314509789594, 0.03598140504671832, 0.15480719160264245,
0.2789645055267307, 0.29561470182031624, 0.4265353477382994, 0.15356605783692512,
0.635090734383404, 0.2981246862144783, 0.8406948542999397, 0.09873147969571938,
0.6649332833335672, 0.0863869727045985, 0.22763433431148594, 0.909791459666955,
0.11575826563010949, 0.22288715684923133, 0.3894090492011969, 0.8218433920862759,
0.9905235715285049, 0.06644133861099011]}, {'weights': [0.44514396066343676,
0.43767616554467026, 0.1745339899676812, 0.8848279561634492, 0.04054745846326746,
0.9026052772620697, 0.9904574190150924, 0.377382575896249, 0.7710211292598249,
0.46066547193200413, 0.5660822666618274, 0.18354622015091626, 0.6817630739785743,
0.8206587493366523, 0.39998190627053243, 0.48499337048212, 0.1589249222481145,
0.5726440258603486, 0.9204334181029666, 0.7203955912978246, 0.8117366349957263,
0.14679709499040672, 0.7331719616728596, 0.9796872317296599, 0.26111892564421724,
0.6743254971985871, 0.21404000298438375]}, {'weights': [0.2719396240582347,
0.3954715136176328, 0.3402171621765403, 0.8545469524197729, 0.4337526241214461,
0.09682027178401054, 0.971955699510031, 0.8967895830610195, 0.2087398111894624,
0.6029417916357394, 0.11358765471809618, 0.9600114397876086, 0.9456487768569417,
0.9367950358290735, 0.08123045308322163, 0.3050759291351772, 0.8123470325121926,
0.6449516745784907, 0.1729887880609684, 0.4324061386943999, 0.5212592505326262,
0.558816216182173, 0.7693924398208791, 0.14172386116142677, 0.47532758407263076,
0.5841049588408338, 0.394926457156456]}, {'weights': [0.9284122356875107,
0.5046166833282318, 0.666529905422336, 0.8182522647140563, 0.2953279320159651,
0.23614143467031845, 0.9040481697096102, 0.2355297401851526, 0.5431562971972097,
0.7666828987483827, 0.2021873946034538, 0.6478290068823624, 0.41495518601221937,
0.6931405915913664, 0.33882762422254586, 0.997426960242676, 0.4813234555476048,
0.09934852788088377, 0.07287116733801258, 0.17711034935538883, 0.702720390697394,
0.7399187908118042, 0.3083971420668198, 0.2142713278913706, 0.41819507806092504,
0.98183830346838, 0.09142761132016519]}], [{'weights': [0.8483765253645064,
0.0968234409240033, 0.5570275710790507, 0.7190864605506324, 0.35692717996583523,
0.7981869960380368, 0.6287928985132941, 0.7880069806695035, 0.9239322077569796,
0.4013969613702456, 0.5189337559808131, 0.13372745474740377, 0.6017928107582742,
0.6513577283726567, 0.41057041311169473, 0.425065267556182, 0.6232792133489825,
0.5539230091434847, 0.5930839729032581, 0.8323250849504331, 0.23357694065414325,
0.4711035555354959, 0.27035451019937407, 0.5136751426263175, 0.1759617480929604,
0.13812138064946833, 0.6077385553209473, 0.3426742153953646, 0.7138722946572981,

0.7547567125117293, 0.6660870429841882, 0.8403829264164875, 0.03246457290509985, 0.4000805267672911, 0.9439718099959997, 0.04360027491674445, 0.33650834306490773, 0.8142462525074684, 0.21853842892148467, 0.31659478752964276, 0.06125254294660143, 0.7538840612786523, 0.4378713211896277, 0.5597793710968451, 0.48169113920699236, 0.7875075307057637, 0.44977047000600223, 0.8347003908605379, 0.5606978753662055, 0.5589888021014227, 0.5035878620692553, 0.7718194729984862, 0.5457930380596466, 0.630897897091534, 0.9474067003441327]}, {'weights': [0.3556252709470422, 0.6331967632394491, 0.00988560620965473, 0.9326516573056195, 0.31122361827783007, 0.341854145916339, 0.7033182099406077, 0.886698500112806, 0.6207651784637652, 0.34893187903854694, 0.2338360364423805, 0.7358282510974824, 0.5957952051955011, 0.015552299525308011, 0.5111639208112162, 0.5280892503513048, 0.9869160102989283, 0.922694452913546, 0.9449612009500165, 0.2137420035349088, 0.19740124151602223, 0.9674120597402109, 0.6370349631060558, 0.513673341729648, 0.5107578005762183, 0.8367689738737725, 0.9247469843685785, 0.610149897894479, 0.7181235425420855, 0.278472726818479, 0.46982516265668295, 0.22176420867730795, 0.779756156388348, 0.42988455612748455, 0.9643875389188336, 0.8055812381090709, 0.7146968142030129, 0.9818390153327534, 0.07620679812709252, 0.7074895937629926, 0.3930844782098313, 0.1516984563394833, 0.4486601207893286, 0.011368366125282914, 0.22302364868292524, 0.7078486833385466, 0.19604531624319954, 0.9788750829131311, 0.7492740626548929, 0.93171566434956, 0.22642191850647042, 0.35089223186636487, 0.37656700086670314, 0.0646574877648813, 0.3097969012008718]}]]

<u>Result From Case 1:</u>

>epoch=100, lrate=0.100, error=496021.707

>epoch=200, lrate=0.100, error=495853.103

>epoch=300, lrate=0.100, error=495851.608

>epoch=400, lrate=0.100, error=495851.105

>epoch=500, lrate=0.100, error=495850.841

>epoch=600, lrate=0.100, error=495850.676

>epoch=700, lrate=0.100, error=495850.563

>epoch=800, lrate=0.100, error=495850.481

>epoch=900, lrate=0.100, error=495850.419

>epoch=1000, lrate=0.100, error=495850.369

>epoch=1100, lrate=0.100, error=495850.329

>epoch=1200, lrate=0.100, error=495850.295

>epoch=1300, lrate=0.100, error=495850.266

>epoch=1400, lrate=0.100, error=495850.241

>epoch=1500, lrate=0.100, error=495850.219

>epoch=1600, lrate=0.100, error=495850.198

>epoch=1700, lrate=0.100, error=495850.180

>epoch=1800, lrate=0.100, error=495850.162

>epoch=1900, lrate=0.100, error=495850.144

>epoch=2000, lrate=0.100, error=495850.126

Weights From Case 2:

[[{'weights': [0.712129350102695, 0.18222896613725081, 0.15180959788620207, 0.5551304209805042, 0.9741723945285157, 0.7018220244666719, 0.42395227858487416, 0.07519816734360318, 0.8311876710980036, 0.0692716386836506, 0.26401715296176576, 0.268855869073224, 0.950013325869599, 0.1998334599957199, 0.2160065803560407, 0.9601611450262671, 0.017151410794021715, 0.3143131154186526, 0.6356380099950615, 0.33850902835994146, 0.21874285624883372, 0.33356482659947573, 0.385856291545279, 0.05519115666652774, 0.22379102545184482, 0.5001742594717306, 0.3437503307138857, 0.8439993411342014]}, {'weights': [0.1492105318123229, 0.8079065155706857, 0.4747066181263859, 0.9165594788316784, 0.7969236869996312, 0.34132150505042547, 0.5746677444498021, 0.6583989856512333, 0.09232505094911636, 0.8375983718539503, 0.8037565184590039, 0.48083646036498695, 0.6941428734545818, 0.7646935852585821, 0.3729937765023318, 0.46871965979075325, 0.005361944009864894, 0.3904166105254162, 0.49186490533665794, 0.2929599117347289, 0.9196911449886779, 0.8254929134728102, 0.8672341446987283, 0.14027618991105928, 0.43874671021592115, 0.5114135924252341, 0.7698565311396993, 0.8747222125315471]}, {'weights': [0.865036521673338, 0.8712578082535752, 0.4887627020960674, 0.6117523499401915, 0.4646548800922692, 0.12284840378226003, 0.29735045957635375, 0.21499430442847323, 0.5403097775387812, 0.42807461799719015, 0.7448634280721445, 0.3133704674746385, 0.8270454205862061, 0.9135943272128128, 0.8192210296059225, 0.9941128834110139, 0.47630727310478593, 0.4276108203579235, 0.07536860424573111, 0.07489396463734521, 0.5414339414852639, 0.2922737554358995, 0.8763272173006443, 0.32858124593961757, 0.379278177093217, 0.11005979035505176, 0.9218954284357302, 0.5674077693281032]}, {'weights': [0.7688915518480438, 0.22122030139990856, 0.10315865384853395, 0.8078050734011645, 0.8432302155720174, 0.11024521546759825, 0.6580275521736446, 0.1395779076729271, 0.18319440875414117, 0.9510633340095971, 0.5693606964863518, 0.5993050070635415, 0.04797209936396751, 0.8473257473293264, 0.4225012001769082, 0.15511277756330533,

0.4941722996286414, 0.7054186163002144, 0.7986077700825145, 0.6646829615131749,
0.07121435361758743, 0.14902634024270722, 0.6910569745855498,
0.0022708673990624595, 0.16112109594521695, 0.5792986389207974,
0.6262407976460322, 0.3810898116561 9027]}, {'weights': [0.15996918788327275,
0.9780709566270503, 0.08833943965790003, 0.07291461732679327, 0.8632064189293503,
0.0820070262859941, 0.20290125752550014, 0.11089530410047699, 0.5090547143238445,
0.3451706769549687, 0.1415913299179321, 0.9336871347615898, 0.2757598624111698,
0.623104130201256, 0.7655917327477914, 0.0902378861678248, 0.8631930748502251,
0.34922619906720187, 0.5632008321941409, 0.3171421068906731, 0.8865850916308048,
0.13873492578484148, 0.3206124442440009, 0.8583755349164552, 0.9146348975323153,
0.7283812989871464, 0.9347503514319891, 0.631349222807817]}, {'weights':
[0.17247765399883597, 0.10297545589803037, 0.7465940032518099, 0.5681571697025936,
0.449406927479042, 0.7827742157496853, 0.8592181798731621, 0.13216245208922273,
0.39503253114416503, 0.9973015415928899, 0.9760010277997724, 0.469725857718968,
0.9401754415493886, 0.9594012391189916, 0.2587655074374794, 0.05164342534839616 6,
0.8274390204434056, 0.05007651521055012, 0.0640793843293318, 0.5930083289132793,
0.865842408950083, 0.693693825896983, 0.7608540842449704, 0.6498131921271249,
0.7439536381055804, 0.5445074754090506, 0.0623144037807644 7, 0.2466893923824358]},
{'weights': [0.875571421037859, 0.13501200262352298, 0.8447043693006034,
0.9259949327953435, 0.6824489068520994, 0.967577695170903, 0.3133124321011411,
0.41497718156672425, 0.687997904770168, 0.7630283582222298, 0.5025489438718448,
0.1940742008483522 5, 0.1835735757022945, 0.13219440551249317, 0.46023082616314026,
0.13626935155855524, 0.8901123419074344, 0.5398521001387715, 0.2753713210516382,
0.6117881019466545, 0.8216025045366045, 0.6845269029258719, 0.25130510371979253,
0.9144851368331286, 0.8508303903592945, 0.764130063671319, 0.7529356178591825,
0.1425997397226889]}, {'weights': [0.9184606597164391, 0.8707035974774543,
0.4548497971526462, 0.9775368742197792, 0.03305828892768836, 0.2963833091636321,
0.2747782957620334, 0.42038943063817413, 0.7038818720128122, 0.8102156308387678,
0.2104348400106112, 0.9360011651522075, 0.3520935334330665, 0.2512907103206099,
0.4745531211199123, 0.6093566397900221, 0.9837180416708121, 0.316271242031681 1,
0.8093085589904414, 0.7933655385270294, 0.03689346233355584, 0.3501012728216134,
0.6147303979125069, 0.6190273900510315, 0.971390155617561, 0.3998130045024969,
0.9945148308775498, 0.515166649175514]}, {'weights': [0.47233576046196935,
0.34315669806379423, 0.4556251557986407, 0.34902719874925603, 0.9360206471910056,
0.506069378691329, 0.20358043124246183, 0.991400407602694, 0.3219369642314126,
0.8482572491118963, 0.3409829513214555, 0.11410440781981857, 0.49508905422905913,
0.34227981978453026, 0.41328966648287335, 0.2637658574108537, 0.6321752893982681,
0.9202119270491432, 0.07181194088916809, 0.42901951313205844, 0.2222537881585036,
0.44703335718706927, 0.2413362565915238, 0.6395140324018815, 0.28889385629571596,
0.08999843763409043, 0.33430567070890704, 0.7691969380608039]}, {'weights':
[0.7929659026824509, 0.7444793064976009, 0.5528524852290998, 0.5873119202142849,

0.9736121749317718, 0.4021136652502417, 0.35132729042756494, 0.558060223459146, 0.89011352341417, 0.7622731862239202, 0.6124910737742201, 0.6665766495313533, 0.8509827667282954, 0.11460409669892313, 0.9829419596371641, 0.06479028999058867, 0.40222708950643626, 0.946696931205132, 0.7842461148462898, 0.8147715418027649, 0.18248952978305777, 0.2195032663672979, 0.9581747703305643, 0.7852709727154733, 0.32342216086035946, 0.2047123748613663, 0.6968733367506309, 0.5926182586421376]}, {'weights': [0.060760143040294756, 0.7607319431400594, 0.5556521260828298, 0.614295602127259, 0.3874007951428473, 0.689911572167232, 0.23732051242544083, 0.6960820367125949, 0.03451736935984795, 0.8807605531440585, 0.3989392518723538, 0.5808527770943835, 0.6448067371013391, 0.6397225115912976, 0.43592547411749916, 0.9210359938798975, 0.18460871831937353, 0.5489660869408407, 0.43863566143518895, 0.748907948937681, 0.7891583614817809, 0.9962165462376023, 0.5267078494494942, 0.33438929949040985, 0.35815364582308273, 0.3485861739318461, 0.8792831987394947, 0.06883314924539263]}, {'weights': [0.9047856072342907, 0.793128422756965, 0.23377808921257726, 0.5128195212166428, 0.14206679869803118, 0.530622231478266, 0.5096631037334705, 0.428409552712368, 0.2283629090788417, 0.4598103386315301, 0.7226535845489006, 0.2400946247452591, 0.1187691845956842, 0.7349460593690552, 0.9064188694386625, 0.8383520155831895, 0.12056797425115162, 0.9587378825765599, 0.8441175091628816, 0.7563905186105513, 0.6007293388761593, 0.5121049792219785, 0.39226075170840036, 0.45843491565604566, 0.8476391832587213, 0.6482241943061327, 0.7991968306237587, 0.4635175215654055]}, {'weights': [0.8406128854410067, 0.5596069499082078, 0.5181752391151219, 0.9566876574523647, 0.8553420282398857, 0.681925770878647, 0.9812626440852055, 0.7167884942892949, 0.04234156776574016, 0.12838203420371097, 0.9638924685831772, 0.96870763202778, 0.4021861006542661, 0.4923658360279165, 0.8481773151139682, 0.7369143160077485, 0.4389957928340139, 0.7020740955117238, 0.19773900423516455, 0.752260013615579, 0.25018261733911484, 0.563848084328801, 0.7633174097237817, 0.5446487370940494, 0.2692009812434363, 0.25072469337928527, 0.6587438695267039, 0.11297758650822265]}, {'weights': [0.8595122249062159, 0.9567548821477774, 0.9531233066150241, 0.08606977182713327, 0.3684652366428959, 0.4821287076938938, 0.7017643002940328, 0.709662534895365, 0.40252026569906585, 0.47536282396773477, 0.2061668415208855, 0.7728850663479945, 0.48575242733869695, 0.5744075291908162, 0.17331129539013268, 0.7984288541678215, 0.2737645535666875, 0.7480481607829529, 0.613321285541242, 0.9039072150159163, 0.597702742745245, 0.2860892794234654, 0.35430487410842515, 0.31029360593351607, 0.3882851791534926, 0.14634913379394787, 0.7754826344312641, 0.21322392042498173]}, {'weights': [0.44267565589554125, 0.056070742558628206, 0.08009368944608519, 0.3054624244232339, 0.8808831964076401, 0.11123756348600278, 0.12624370554414954, 0.3659815918280408, 0.2668716766858241, 0.9126900763408079, 0.9192160915961112, 0.638304557463782, 0.7310088253421644, 0.24330117315757716, 0.8593797345114758, 0.7583248041659398, 0.9360986171800155, 0.7344738746457726, 0.9447962299472967, 0.9078732879773842, 0.39286037786133743, 0.59121309547035, 0.828445100748332, 0.7704683751609235, 0.02630988272235002, 0.313275472407173, 0.0229387847120619, 0.9060262358548906]}, {'weights': [0.06757512472712723, 0.8394623930609215, 0.3956925790999856, 0.010884011134707827, 0.2886952603315449, 0.048883857229625294, 0.7137081858708685, 0.9975795635444457, 0.7710251362568598, 0.943433139603897, 0.9636773313682575, 0.7929948663808836, 0.6455039781537819, 0.560772212549249, 0.17878317597696358, 0.5438052967951404, 0.8089138888791761, 0.8409016238816754, 0.5200801883364974, 0.28191245845684176, 0.11665558985192814, 0.2965782396779145, 0.08654484227588788, 0.3604718118456358, 0.050395144580270834, 0.9260435210127578, 0.7227146348269022, 0.44382688233072465]}, {'weights': [0.517507013827786, 0.17347702752808758, 0.7994243358515786, 0.3224445187371162, 0.5518944947119375, 0.27888381653096395, 0.9426222254154999, 0.3909472909454511, 0.32261822507023463, 0.09993179282925646, 0.5014440729905869, 0.4592938870739923, 0.5659689981241072, 0.685903005443739, 0.8731483992877972, 0.5673394888285218, 0.04851128676261984, 0.5012667683669235, 0.6514095591385293, 0.7342822626909765, 0.9789576384782059, 0.18967607747393433, 0.2617978603221047, 0.17969102395193381, 0.2516468589185086, 0.96278591406409, 0.5158315784053705, 0.7234449773355976]}, {'weights': [0.26639526507530054, 0.4507853977111632, 0.7902768921521187, 0.45203817660629997, 0.9053891854513032, 0.02398368690334507, 0.9083674553968388, 0.4705306582703882, 0.37616455962335493, 0.022486336887500902, 0.28102645472291743, 0.6264453604999786, 0.9108746013539053, 0.32971922166645584, 0.6979715299605493, 0.026895406455730075, 0.2799825287521188, 0.8783073223132981, 0.8996980331476688, 0.07173502690646594, 0.32967758081585585, 0.5510356698998657, 0.7440014406230344, 0.7711424133652622, 0.15633513916477626, 0.4730455415964666, 0.8435984453616465, 0.9244542911608817]}, {'weights': [0.45287590920500675, 0.4376899654219364, 0.3404202074136249, 0.6215600679671539, 0.9166692448908718, 0.42316959603827253, 0.14908872351173053, 0.9583935406135683, 0.11329985804218123, 0.38702051126924564, 0.2671022117419033, 0.5393816522749684, 0.8036533344207496, 0.37740216945525007, 0.27061565316839375, 0.8778425650777675, 0.33812945419381346, 0.1701036644240328, 0.11713938316214956, 0.9923898603137201, 0.9655484594235438, 0.9158064496194098, 0.13431377724757965, 0.5234779468362772, 0.7157663048954467, 0.08771720861944665, 0.639850719273932, 0.21334939272813003]}, {'weights': [0.8451691689840162, 0.6599981137651071, 0.6990376409627441, 0.18862215029594276, 0.3487667793744206, 0.3123230968828934, 0.31759394322537804, 0.6058612959015676, 0.5835669414831121, 0.9189093108637204, 0.5792687961661066, 0.2355091498035381, 0.938367742034121, 0.4039335156987527, 0.8596372959045323, 0.17341991275950464, 0.9905699707732811, 0.06184778768334909, 0.8822467570380134, 0.06927697996735027, 0.09472324437477664, 0.008819638198594215, 0.9063189538380189, 0.7850054976529018, 0.3478989558259654, 0.6826320666771182, 0.27721888208871615, 0.15456833162860062]}, {'weights': [0.866255986080925, 0.3263555276317296, 0.7319732259058843, 0.3233417884723605, 0.7512582799604228, 0.5813637147316415, 0.6085596513980093, 0.5352760161335438, 0.7820676633727488, 0.3893244276321033, 0.8389294588339296, 0.09837271491306054, 0.936994866855699, 0.6611027694993387, 0.08029044956618525, 0.03969081540242059, 0.31230125104860584, 0.532239309793069, 0.8224873571132211, 0.5942707452039397, 0.49783423811830585, 0.37469494613128485, 0.615284631164153, 0.12466821523585947, 0.46377184297680685, 0.8414810861276403, 0.9064433349035528, 0.5063781016999671]}, {'weights': [0.3897198485258001, 0.8687960856527455, 0.656468044684061, 0.019395866169569898, 0.3261853652822658, 0.4996903082379818, 0.17588945018549962, 0.9077856937481803, 0.6249784957825403, 0.880239161538966, 0.3281403868975845, 0.2763608020620829, 0.07426137106921216, 0.2891897060055133, 0.8086346888418908, 0.9940706882278485, 0.023241656814416922, 0.3735190211187388, 0.28937005942810046, 0.9931841536668468, 0.6379810019830154, 0.20853169247882353, 0.0876662894439606, 0.2437256041994208, 0.4374896347440915, 0.37292495391152813, 0.14348179813682171, 0.5052730245507299]}, {'weights': [0.7962070359843705, 0.07678433855768763, 0.6978311342529886, 0.014722832498590521, 0.838033903987749, 0.5070445384078629, 0.5339083998737014, 0.2547141254110157, 0.8841565085907414, 0.7515455840241643, 0.1575502045941397, 0.25312273944955554, 0.16685514535710866, 0.3998760950414162, 0.49454610217993633, 0.740399518340784, 0.656859083758799, 0.1898702275375883, 0.016697632806482843, 0.05542452214244076, 0.6615479026358914, 0.23717212355410622, 0.7917621454619839, 0.04363558369111631, 0.6597824600542685, 0.23755771239816548, 0.7365579996501889, 0.4058310503233655]}, {'weights': [0.7217497545608502, 0.7130920803801067, 0.7713779309919105, 0.5883777585176949, 0.4562219410116989, 0.06230573003124118, 0.2263419461089775, 0.2337151152253537, 0.5779226552475724, 0.6619835407984508, 0.6638821259534677, 0.3103565590733738, 0.5879928790982195, 0.43926458934598256, 0.6348269568970513, 0.9106645759017148, 0.46216347761301535, 0.819171466646464, 0.05618147408516727, 0.46881162137796606, 0.9139313360643067, 0.3770788709845049, 0.1054303200714638, 0.29727025753070424, 0.6142527616764979, 0.19858376899991603, 0.7722336825605151, 0.6988677018547501]}, {'weights': [0.8504488742264149, 0.27028539669305385, 0.1712372098624111, 0.8596228776962621, 0.6400909469244558, 0.02133108622550095, 0.5192361547970944, 0.7819816132990649, 0.49103586178434167, 0.4850611373157573, 0.2534131784879269, 0.45091622857943237, 0.38487322675332114, 0.9631638254753302, 0.9489407735966439, 0.2783320888611396, 0.4506449150082128, 0.29440031760393526, 0.8876367969603772, 0.6247330670358127, 0.09048806319290359, 0.7312625620226366, 0.38216075670817773, 0.9912244095283099, 0.05629703172216416, 0.9271778079560967, 0.9016280293023465, 0.09813612835251606]}, {'weights': [0.04361767124186211, 0.28140304388265514, 0.5941778226273731, 0.360793673219441, 0.8659727148749342, 0.30071580493773165, 0.9018059288930332, 0.8780329986346506, 0.6070671529429287, 0.04173063413600964, 0.6713795455558986, 0.6610695343830238, 0.910526982371181, 0.7598718539554611, 0.875583621257666, 0.17855230526354382, 0.7838438985806594, 0.5409581176565085, 0.5410115362976228, 0.1987895513021427, 0.2558174807258059, 0.29534907849764247, 0.5241329710262629, 0.9758049506613643, 0.4688125893054398, 0.5325520095959265, 0.9588693700398844, 0.20590214093617476]}, {'weights': [0.5780612048245174, 0.399554004982295, 0.7591204339787118, 0.5431789378094838, 0.9009696337564322, 0.7335805140059267, 0.6748974714304772, 0.823946938101928, 0.9953434277362959, 0.31446367478512216, 0.5101999140569015, 0.8377431300407153, 0.6709423724398148, 0.47678460361369124, 0.8951187775782766, 0.3601646669344205, 0.41360041151994487, 0.4809810188329988, 0.542268432180787, 0.8752099026690063, 0.6547392433708807, 0.17561161815056314, 0.26101387313722524, 0.5990825167916489, 0.4401288946346247, 0.827200807196543, 0.7849235156273349, 0.2671023960437935]}, {'weights': [0.10428880508544103, 0.17467584291569072, 0.3668188389271084, 0.8127818421601187, 0.9947355324019974, 0.4935932297948832, 0.9292327419647135, 0.06079175668046499, 0.3287090441729179, 0.7339987353217997, 0.13679483820982918, 0.3086140816356133, 0.6199077205737422, 0.6604286677829028, 0.4154146841977058, 0.30163717348118346, 0.4689206483058903, 0.5286361347703857, 0.14374201527162633, 0.055764155305289664, 0.0368615387884057, 0.5470233179416611, 0.6546093124617129, 0.310360462246581, 0.28749718705826466, 0.288090633447426, 0.10435036588632174, 0.1660805726232416]}, {'weights': [0.9429643493479543, 0.3368581256630446, 0.2245683426073506, 0.5309343164882826, 0.3650393423119769, 0.18785631876185083, 0.8329147980008924, 0.04379656044131974, 0.26662704371768753, 0.09492979917862343, 0.05023406513619588, 0.19080014376736032, 0.6637525596489136, 0.23426741515530913, 0.10673157628629337, 0.48690728656324056, 0.6770736302150587, 0.10632970386781448, 0.43046335439156236, 0.8201060969915125, 0.999787289361441, 0.08778542494452135, 0.43210843017648204, 0.37888733830576116, 0.7985205511790873, 0.3343904541511331, 0.3133257917568485, 0.8452952075351562]}, {'weights': [0.998617120345985, 0.22219734888203357, 0.9225862176345317, 0.5214852822206324, 0.1322442544062239, 0.1808192204770781, 0.8814635819220291, 0.6767976534225807, 0.8381426765030997, 0.9798736164758107, 0.9905550069338991, 0.7774425726752819, 0.36489632341076694, 0.062286308806412505, 0.6624236103686301, 0.15230794150625515, 0.742830407166109, 0.516779254964882, 0.974637591002623, 0.11974339014251767, 0.13351446668394196, 0.2727299254862988, 0.8613580911467468, 0.6323642104935604, 0.5761819408938287, 0.9982543353842563, 0.9652700772191304, 0.16890650441640465]}, {'weights': [0.7961802670894392, 0.12124761849624477, 0.10266614466929935, 0.49475226171507625, 0.9477057836034195, 0.9490348578618137, 0.6373694177299472, 0.5450112019877633, 0.28552370136964866, 0.9040073854267959, 0.6467930266719433, 0.006071812574002244, 0.2942547088341788, 0.4534951322399827, 0.647790828816514, 0.59651164665457, 0.47673806586875667, 0.013273415723376814, 0.27630404588862534, 0.8456301929444886, 0.4766950873641622, 0.11200817551681264, 0.06414080051765003, 0.9431483088090941, 0.4899341892276792, 0.8114280061995292, 0.5397988951765131, 0.9120403737440551]}, {'weights': [0.34386774644125506, 0.2871045538632152, 0.9776723618239014, 0.1636244030170645, 0.51905395468635, 0.03273761644046391, 0.14240830905807977, 0.47795637141257763, 0.43322912051663776, 0.5481957194199318, 0.25009597670712236, 0.8480432148569454, 0.1443034712235527, 0.5899323228081904, 0.47400421477383214, 0.9447765496337479, 0.9336136085498294, 0.75785198407906, 0.30711365463437845, 0.4884720498317835, 0.6444707086172233, 0.9433673860330308, 0.4026500532256162, 0.620638570941336, 0.8507865123750029, 0.30890485222608755, 0.6023812538751728, 0.5318124700105235]}, {'weights': [0.2376195102591282, 0.47793263978881606, 0.36711429864630307, 0.1165087059637333, 0.0924976530094479, 0.6550990200099829, 0.9163182146533495, 0.1487094465813158, 0.5786061273830622, 0.3528946631590623 7, 0.4514967711497564, 0.9214647299738362, 0.7017334880979593, 0.59251415814352, 0.8458804652407731, 0.9424623268151734, 0.8075992045585079, 0.49375672944457605, 0.9748736266779731, 0.3282961739170366, 0.1852028987905019, 0.24733229180851435, 0.08712744497989955, 0.47369399730536743, 0.16969514054982093, 0.3714040966497005 3, 0.9692222614800109, 0.680303127561847]}, {'weights': [0.9174964088395975, 0.8324061417645258, 0.345192544362356, 0.7728851127723297, 0.5582062363885969, 0.04953403346825769, 0.34654944778398056, 0.5813026211019509, 0.5002342258793763, 0.8701008411428084, 0.9857499222951852, 0.5985011744704795, 0.12967624865358585, 0.6649142349021131, 0.7045288266534177, 0.7872237633960419, 0.7988301099252811, 0.31172540722309916, 0.2811451126998940 6, 0.578877732675778, 0.529235654264953, 0.2685396188605781, 0.4926110504725849, 0.7805760845042754, 0.45055836651043535, 0.5575336355959796, 0.4888017429184127, 0.14696235097310106]}, {'weights': [0.8794533751640771, 0.8951666229673169, 0.253428104527467, 0.699637693080555, 0.4549659664769873, 0.579867483334062, 0.8830203547070693, 0.1915480248781699, 0.3732304786162701, 0.16815376650512015, 0.9680852256347269, 0.9282628151029327, 0.26021361319440395, 0.029992703525853703, 0.9735919189471007, 0.3361991351112028, 0.0894457950788626 3, 0.5737141284280928, 0.9675357130972189, 0.8429555056712449, 0.20968568464374082, 0.28034183504654764, 0.8599863268332247, 0.21966623121312878, 0.20161033813537643, 0.9086684042394432, 0.4611790106072923, 0.09087223922170129]}, {'weights': [0.1732342786354537, 0.1388158597201251, 0.250046257345248, 0.8036041863478091, 0.9851794871861499, 0.013268885181220158, 0.056378042938620254, 0.4067794025751641, 0.18330915942431525, 0.18248270259816646, 0.9071018475287506, 0.8419418950077739, 0.7505249694045425, 0.802390360719135, 0.12898386481239288, 0.00035610396281693 38, 0.8250517167395878, 0.33973486342396797, 0.6184851838497648, 0.952870843859405, 0.7566553115326263, 0.015953505369546428, 0.1268577207885221, 0.45054328519072173, 0.32377262648885585, 0.5939460402308622, 0.9578015422087514, 0.1664831512083792]}, {'weights': [0.6297322439243519, 0.5546066808615778, 0.3566830700163086, 0.22265782508920673, 0.9178733047801746, 0.34556594756150394, 0.4248978566660191, 0.4961918541848558, 0.023844748614618716, 0.3521013854115478, 0.719521492465639, 0.5018229096766345, 0.5897617718757389, 0.6556611819887174, 0.30863895919959206, 0.5668717510616524, 0.34828176962984625, 0.854523193529247, 0.9329441687791047, 0.35060981278108116, 0.8308788924508047, 0.13922123032785727, 0.0881833242358736, 0.9980586236994087, 0.23372748350871797, 0.45102679300696014, 0.20174040765361378, 0.00801466973848363 7]}, {'weights': [0.9387415655883301, 0.13572954997306574, 0.2526953090116595, 0.2242719826253019, 0.6292832818628219, 0.8063637857150175,

0.7621693348262788, 0.09677754184347431, 0.46980124511527244, 0.11692650793056103,
0.054285443956050505, 0.8550446718306466, 0.5259174565216201, 0.43885825840728965,
0.6652482773613317, 0.04301727099402697, 0.33196665031650463, 0.26164251175593767,
0.3391847283389471, 0.5341005832355654, 0.07052197487684941, 0.965845837041909,
0.46086413866563314, 0.6659098990681439, 0.7702714984951359, 0.6216395592238287,
0.6770986759490547, 0.3518072768721926]}, {'weights': [0.2787936478611237,
0.22434845547795423, 0.2273331008212235, 0.3084064065272536, 0.9522428330648017,
0.7395859143808605, 0.8123283667104959, 0.9942979857201241, 0.9624992020767295,
0.013272029528880402, 0.22090646718296614, 0.7013185489121542,
0.17858539856894462, 0.5859581903536248, 0.10567771769560186, 0.2785574594599588,
0.3334519612656207, 0.3986981202234441, 0.3085978504993454, 0.24079906453555677,
0.4346335897376846, 0.972378608682181, 0.29094884503683005, 0.7433143208569725,
0.49414830773597207, 0.745149133436226, 0.5084506172158643, 0.6706886424376985]},
{'weights': [0.5804764267104813, 0.13353957475563172, 0.5567926393282674,
0.29897380080545444, 0.9661791328469325, 0.2387840625500819, 0.6529160384105968,
0.8145085548871257, 0.15400831116799474, 0.26399494084208786, 0.8403250067372093,
0.4808414669776987, 0.20475496957222905, 0.6877993121073249, 0.9035862028647655,
0.31102116711825034, 0.18447928379368572, 0.03899137167171851,
0.05212735393119472, 0.9135174687453667, 0.5133837389539054, 0.6917106080466954,
0.07059337593043757, 0.11448058914466352, 0.5164711756164725, 0.9734095992290502,
0.9719121662899781, 0.2796540036012283]}, {'weights': [0.6943770426526603,
0.12171758340833427, 0.36589057630940947, 0.11107415437903179, 0.5286913528749283,
0.3738270176484936, 0.3436244958980079, 0.09572675636733174, 0.6712957147272425,
0.14914042814149053, 0.6977138279673583, 0.5210260620170113, 0.46329824639034367,
0.6915553297783669, 0.02746476178782742, 0.2449335289934974, 0.36202798207081,
0.5967137070910132, 0.03598993468676437, 0.5970203393720902, 0.5913581721929968,
0.39126152678523207, 0.7708320987182189, 0.5629751249864828, 0.8787483216289342,
0.6348058055891703, 0.30750371827469025, 0.8919582272967554]}, {'weights':
[0.7054885819932896, 0.36779911773917173, 0.4199261746227031, 0.37142549302011085,
0.8551213525863796, 0.15220405350174282, 0.830791704100078, 0.8157814432312341,
0.8710328297170854, 0.7123389132110055, 0.6580667604432362, 0.42807151368533747,
0.6849817478056058, 0.6910854499193251, 0.7755451789341705, 0.4797147348844699,
0.7999746781683783, 0.4829260736378709, 0.25948442192458077, 0.8641330716031165,
0.8795446119528098, 0.5581219065699968, 0.2869321817704741, 0.6873643592094131,
0.764121939720809, 0.2857281629384554, 0.5121003212533339, 0.11334125181227994]},
{'weights': [0.2745426846302824, 0.5031281795187204, 0.49810046486137405,
0.8805267329471249, 0.901360414032059, 0.6776345427917597, 0.1407009020770732,
0.3836751658065013, 0.5712567377450798, 0.8305282134707871, 0.8145496865887534,
0.7680090198388281, 0.17539407908568805, 0.3507375743488236, 0.9412850247116363,
0.27418820801012433, 0.5915918859547984, 0.12197443093108473, 0.603250951769331,
0.16766793287365567, 0.7239362520611546, 0.6541423033030279, 0.28838753475000867,

0.6599685580392811, 0.37141876143125685, 0.8969184780384116, 0.40896291824493014, 0.6776322151079148]}, {'weights': [0.518191272183076, 0.12380398610004228, 0.4654956956278332, 0.04008774769084067, 0.7303366025557768, 0.33935077431906024, 0.14117144218028, 0.4980756058272868, 0.7613904508949685, 0.501704278984695, 0.20495420804795017, 0.07210840256809037, 0.9835761448959597, 0.7417953862234956, 0.8599051655820834, 0.10239337038558427, 0.947929024256261 7, 0.746974700204720 1, 0.6376857117315758, 0.8741401503894888, 0.8044420429282705, 0.0757042691661729, 0.03470856366623987, 0.5568189981238407, 0.9883589972694363, 0.7723703723281198, 0.3403746992972062, 0.2766680414599396]}, {'weights': [0.6554921668221166, 0.7385056913449698, 0.1799166254678204, 0.3485350307138556, 0.12284626735062953, 0.1411303842525352, 0.7084441396475228, 0.004906243976162128, 0.818591295189276, 0.5196792164931985, 0.2513859613368108, 0.8881434023566631, 0.5648549668284634, 0.848885047021622 7, 0.12328219031346888, 0.49997369342167985, 0.4499895547853291, 0.7131312393919015, 0.10911342856800021, 0.8794490555722793, 0.8258142669683902, 0.8468907148024759, 0.7056632782853903, 0.1150239903095096, 0.016422627189700778, 0.4685629384278266, 0.4528836887855572, 0.24582032706884804]}, {'weights': [0.47313123640962285, 0.40283925703991763, 0.6460934025637604, 0.7671768715269802, 0.9915227275433296, 0.9735612134025283, 0.9799291760785508, 0.14622445867199307, 0.21783038719132386, 0.2998452553643327, 0.19026604410956538, 0.23771357767017576, 0.4404291616855539, 0.05807217429383649, 0.6437270870275942, 0.4546792591068238 3, 0.8213365243734383, 0.13924499299070114, 0.06628188766360454, 0.03965339355039965, 0.15855338617680437, 0.6643125246376981, 0.5500049122256725, 0.9899035608113095, 0.611617501381482, 0.10556862477218998, 0.324186640466548, 0.5554624401366587]}, {'weights': [0.9444977825574898, 0.1565505554153096, 0.7696970636901979, 0.675805278013684, 0.5593457629503308, 0.15342717632803493, 0.7413658950514294, 0.5345065414797507, 0.7493760849822171, 0.6927621416672085, 0.30240510457569936, 0.9626621223600551, 0.0512532341386166, 0.06206369806635592, 0.5559376001118553, 0.15791152575733713, 0.864405861147798, 0.22226082117923807, 0.9631455969619324, 0.8709246217351757, 0.22445189090763884, 0.05452566285356919, 0.8112506310529639, 0.895069424688075, 0.48047366249689305, 0.7641367262283564, 0.3058092484415278, 0.9938919673164615]}, {'weights': [0.3461289473292154, 0.8498430443983092, 0.1792805778617842, 0.9657800519497469, 0.5917761489706722, 0.06710206920769457, 0.6507377342150414, 0.5981597806286336, 0.8246849239816794, 0.13048783430898503, 0.9921155545812934, 0.636827509797971, 0.6942434023873171, 0.43353030818859073, 0.6060249644248813, 0.33316023391427385, 0.13319101289937663, 0.9640784337533739, 0.6729526032851745, 0.20401560265008134, 0.06557288746417855, 0.20199852208422509, 0.5150980188440604, 0.831194037626116, 0.620756729957114, 0.08961242334258712, 0.7558634000657132, 0.19380593085381714]}, {'weights': [0.05555863002302852, 0.5682689360178164, 0.6450466577755674, 0.9535326089284751, 0.5376638577998111, 0.7164795105103251, 0.7686558223479848, 0.09545187351746731, 0.5828791301073256, 0.6197778346360929, 0.11447879825867724, 0.1953092856253349, 0.3753914991468952, 0.7619110006326967, 0.4691213131913964, 0.2568564899046891, 0.926604934662403, 0.3442072056005416З, 0.412929193006077, 0.139673852826112, 0.27025069196749274, 0.26851074304680567, 0.6772333708436878, 0.5668080461327476, 0.2777881533200699, 0.5106206671771392, 0.7726523491487521, 0.06957593483180624]}, {'weights': [0.4248770199602725, 0.22535282849338578, 0.49318142824489075, 0.020662095011070525, 0.053861186385005144, 0.987538871391521, 0.7903662776637933, 0.7925313563441796, 0.004133566211552808, 0.5935017267468412, 0.18029744294177696, 0.8324838888549556, 0.143634034970352, 0.6202052287719588, 0.767138029221324, 0.38528538166752824, 0.3717121925513346, 0.7551542703846112, 0.9770881103732005, 0.3643078062437711, 0.1548763436980557, 0.03332455454115979, 0.2406527392069896, 0.3314930418324631, 0.266914816175123, 0.12257309221679447, 0.0098281234175005, 0.6297234483503097]}, {'weights': [0.6838111709543255, 0.09953595133640136, 0.26801002589894385, 0.36470242167304157, 0.1113152087710042, 0.9834760621573665, 0.5633456025088234, 0.8957409869374884, 0.39362877536050944, 0.885079264178212, 0.39445527532594415, 0.23643240253563702, 0.7444215995144112, 0.03268195366529569, 0.6739038645195216, 0.2175868453083475, 0.7935691795401452, 0.12680679041435639, 0.6034586041798358, 0.9400470686889532, 0.11051978376421401, 0.48783846726038715, 0.8369996325464057, 0.9215817119066012, 0.6818674648253077, 0.9447407719543945, 0.5924824213867561, 0.29844388202873573]}, {'weights': [0.13406161819667184, 0.156966553946552, 0.1734900058018929, 0.02904796527736131, 0.5404866737354, 0.1179270375551944, 0.06778630151698894, 0.859925586362638, 0.9895725541572156, 0.14401158654618318, 0.5415089677726133, 0.9933217144972339, 0.2906308592984014, 0.6257159760317477, 0.08448451283929148, 0.34730489009443855, 0.4606217164450114, 0.16777201763219252, 0.682046379340317, 0.20264709746964582, 0.02951291620303942, 0.2990725576839579, 0.9320462090936821, 0.3761647745606316, 0.703174964783545, 0.7196226358906864, 0.11644633822131989, 0.8239636267191585]}, {'weights': [0.7776145557949244, 0.3666607266836559, 0.9373694959865886, 0.7594250438930015, 0.30155510236015215, 0.17124761970903069, 0.6933428113660233, 0.37943052203496097, 0.9377194627917725, 0.53860940616847, 0.3093125336751029, 0.5547222987119653, 0.6995819447443116, 0.06084359384477911, 0.11874880384222497, 0.7005279007071891, 0.1316635842227153, 0.6809362180778674, 0.7488106204857693, 0.21074409608674827, 0.22904202237199522, 0.7687531513672907, 0.711943147495707, 0.28570653101786847, 0.5042820812882576, 0.5904729013245026, 0.9827323647254878, 0.595552209328293]}, {'weights': [0.541060637808909, 0.3167735091859297, 0.4842847647581785, 0.51113963570904, 0.8606091618342195, 0.1465569412860105, 0.5858789996623499, 0.0788299064735215, 0.9955281583535194, 0.8132372293758371, 0.06425317512064066, 0.670656318341262, 0.7616856142303498, 0.7435276365453102, 0.17091842762092813, 0.34665167991524426, 0.619892428117034, 0.4213218261004731, 0.5186345392444448, 0.1498818399967876, 0.4580259085523414, 0.45460952527855214, 0.8640156115032296, 0.751256726359514, 0.8612839087438934, 0.05619317988746764, 0.923286421158918, 0.6076049404962364]}, {'weights': [0.8834091105513154, 0.8650377740021424,

0.21915655458055794, 0.2946844156896199, 0.4449902654403758, 0.7638923195628025,
0.8381796697568399, 0.23500838720415407, 0.2673475666451036, 0.7262375446240689,
0.4680081044824842, 0.5511434463769299, 0.019673448441758112, 0.06003644446385914,
0.38340776928782294, 0.17613229867622837, 0.8483447569963256,
0.002754378580752337, 0.7037978277438668, 0.76465615168712, 0.1847122734564236,
0.5653919198305727, 0.4398512374999163, 0.7943939911239584, 0.36102050086711446,
0.800168182136872, 0.490511457013575, 0.31305731473221565]}], [{'weights':
[0.7928960687378618, 0.5387169094952106, 0.0036056949776246405, 0.6952187343577845,
0.2408710755404655, 0.9373718803160926, 0.7854316730530574, 0.03888835437687266,
0.23435552461410347, 0.32604063575203246, 0.4506220829081904, 0.8422577507308062,
0.8204962173514435, 0.6654021515222028, 0.3119060709874507, 0.4461988398861019,
0.3592026669668801, 0.7947528689892197, 0.6460765916738638, 0.5380002244849432,
0.7538781099734888, 0.3764715931182695, 0.8864920036382038, 0.032973738634587746,
0.7438741889495395, 0.30829029124394125, 0.9208361931837009, 0.5828683427284386,
0.6812197567825712, 0.0503963516854099, 0.1847762760049928, 0.9422854257004719,
0.7148231488104732, 0.042643840583741266, 0.31269642158755306,
0.06495345933815644, 0.03278299557223607, 0.2780605525445672, 0.7135761374786239,
0.4691172160046614, 0.7571878878346461, 0.9256166194508382, 0.3126030503798869,
0.11786526158810906, 0.25032330067166, 0.16952975978705398, 0.5347037571489665,
0.004461187221555196, 0.6039991031404671, 0.4657394477707102, 0.6823854712323356,
0.3734677207736168, 0.4825779147048609, 0.6350272399267904, 0.012089358386122684,
0.6447652663486719]}, {'weights': [0.5583044568277448, 0.02355067312434833,
0.3692764875101764, 0.21009500883225363, 0.08343709577265546, 0.5742942956164095,
0.2483638510741325, 0.29678732454827617, 0.04420776885684918, 0.768087447449396,
0.06025178286445365, 0.17580492716298612, 0.8356592111072626, 0.3958793095870037,
0.22455881071177586, 0.2722999637206476, 0.49217941553626254, 0.8284540043460361,
0.26962622670392333, 0.9651904354103742, 0.5805375291165594, 0.7184026441752345,
0.767561153415456, 0.6024918786521695, 0.9830014036067797, 0.7953273609850972,
0.14163812366918827, 0.3783510686687913, 0.4490397745210112, 0.507576851879427,
0.168874349605461, 0.3138906251679552, 0.7505372797945442, 0.7331691688499503,
0.5193960999179479, 0.9201409687373167, 0.00925903030067321, 0.022872529314722723,
0.33194255329698596, 0.1861557693794441, 0.1569280111993241, 0.6396553945703235,
0.9906757468053211, 0.4173811781039696, 0.855464757481413, 0.4828207871806477,
0.5761058380648197, 0.8348882986463898, 0.8586826288644763, 0.11755062913617165,
0.7575229518504043, 0.8542378618947982, 0.0032080720277110064,
0.15730898761177825, 0.7227732347031696, 0.6827178003766373]}]]

<u>Results from Case 2:</u>

>epoch=100, lrate=0.100, error=495977.842

>epoch=200, lrate=0.100, error=439222.687

```
>epoch=300, lrate=0.100, error=254636.124
>epoch=400, lrate=0.100, error=254637.414
>epoch=500, lrate=0.100, error=254630.407
>epoch=600, lrate=0.100, error=254630.064
>epoch=700, lrate=0.100, error=254630.033
>epoch=800, lrate=0.100, error=254629.124
>epoch=900, lrate=0.100, error=254628.528
>epoch=1000, lrate=0.100, error=254627.349
>epoch=1100, lrate=0.100, error=254626.189
>epoch=1200, lrate=0.100, error=254625.265
>epoch=1300, lrate=0.100, error=254625.095
>epoch=1400, lrate=0.100, error=254624.692
>epoch=1500, lrate=0.100, error=254624.298
>epoch=1600, lrate=0.100, error=254624.590
```