Single Neuron Simulation

By: Vishnu Nampoothiri

Create a Perceptron Logic

In [179…

```
import numpy as np
import matplotlib.pyplot as plt
class Perceptron():
    def __init__(self, input_size, learning_rate=0.1):
        self.bias = 1
        self.learning_rate = learning_rate
        # Assume that the initial weights are 0 with +1 bias
        self.weights = np.zeros(input_size+1)
        self.errors = []

    # Prediction Function: weighted_sum
    def predict(self,x):
        x = np.insert(x,0,1)
        weighted_sum = np.dot(x,self.weights)
        if weighted_sum >=0:
            return 1
        else:
            return 0

    # Traing the neuron over multiple epochs
    def train(self,X,y,epochs=1000):
        for _ in range(epochs):
            total_error = 0
            for xi, target in zip(X,y):
                xi = np.insert(xi,0,1)
                prediction = self.predict(xi[1:])
                error = target - prediction
                total_error = total_error + abs(error)
                # Update the old weights
                self.weights = self.weights + self.learning_rate*(target-prediction
            self.errors.append(total_error)

    def __str__(self):
        return f'Weights: {self.weights}'
```

Create a function to create binary inputs

In [262…

```
def generate_binary_inputs(size):
    binary_inputs = []
    for i in range(2**size):
        # Convert to binary and pad with leading zeros
        binary_str = bin(i)[2:].zfill(size)
```

```
        # Convert each character to an integer
        binary_list = list(map(int,binary_str))
        binary_inputs.append(binary_list)
    return binary_inputs
```

Writing a Function for Easy Plotting of Errors

In [274…
```
def plot_error(errors, title):
    plt.figure()
    plt.plot(errors)
    plt.title(f'Error Progression During Training: {title}')
    plt.xlabel('Epochs')
    plt.ylabel('Total Error')
    plt.show()
```

In [275…
```
all_errors = []
input_size = 10
binary_inputs = generate_binary_inputs(input_size)
#print(binary_inputs)
```

Task A: Check if Input is an Even Number

Test To Check Even/Odd Values

In [276…
```
def test_even_odd(perceptron, odd=False):
    for i in range(input_size):
        binary_input = list(map(int, bin(i)[2:].zfill(input_size)))
        #decimal_value = int("".join(map(str,i)),2)
        prediction = perceptron.predict(binary_input)
        expected = i % 2 if odd else 1 - i % 2
        print(f"Input: {binary_input}, Prediction: {prediction}, Expected: {expecte
```

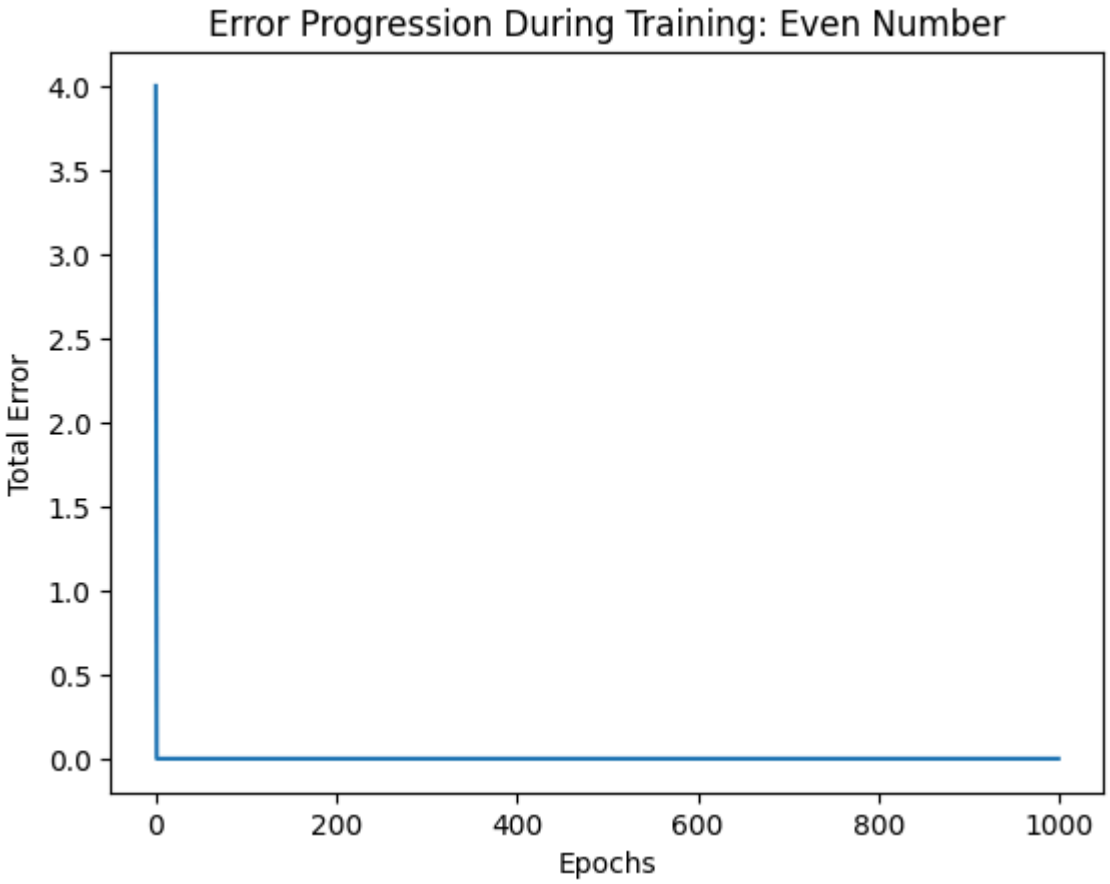
Task Even Values Function

In [277…
```
def task_even_number():
    global all_errors
    perceptron = Perceptron(input_size)
    target = []
    for i in binary_inputs:
        decimal_value = int("".join(map(str,i)),2)
        if decimal_value % 2 == 0:
            target.append(1)
        else:
            target.append(0)
    perceptron.train(binary_inputs,target)
    print(f'Weights: {perceptron.weights}')
    test_even_odd(perceptron)
    all_errors.append(perceptron.errors)
    plot_error(perceptron.errors, "Even Number")
```

Run the Even Task: Output Weights and a Graph of Total Error

In [278…
```
# Run the task
task_even_number()
```

```
Weights: [ 0.   0.   0.   0.   0.   0.   0.   0.   0.1  0.  -0.2]
Input: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 0, 0, 1], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 0, 1, 0], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 0, 1, 1], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 1, 0, 0], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 1, 0, 1], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 1, 1, 0], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 1, 1, 1], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 1, 0, 0, 0], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 1, 0, 0, 1], Prediction: 0, Expected: 0
```



Task B: Check if Input is an Odd Number

Task Odd Values Function

In [151…
```
def task_odd_number():
    global all_errors
    perceptron = Perceptron(input_size)
    target = []
    for i in binary_inputs:
        decimal_value = int("".join(map(str,i)),2)
        if decimal_value % 2 != 0:
            target.append(1)
```

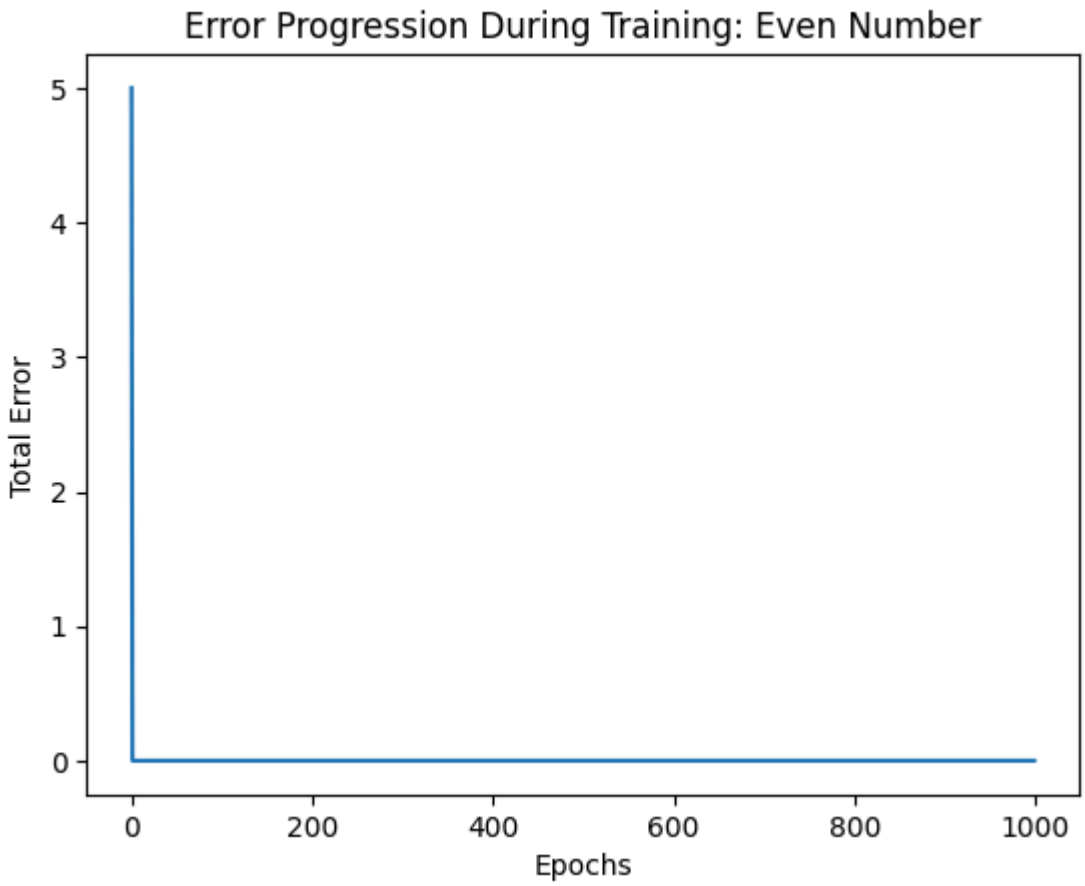
```
        else:
            target.append(0)
    perceptron.train(binary_inputs,target)
    print(f'Weights: {perceptron.weights}')
    test_even_odd(perceptron, odd=True)
    all_errors.append(perceptron.errors)
    plot_error(perceptron.errors, "Even Number")
```

Run the Odd Task: Output Weights and a Graph of Total Error

In [152…

```
# Run the task
task_odd_number()
```

```
Weights: [-0.1  0.   0.   0.   0.   0.   0.   0.  -0.1  0.   0.2]
Input: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 0, 0, 1], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 0, 1, 0], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 0, 1, 1], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 1, 0, 0], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 1, 0, 1], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 0, 1, 1, 0], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 0, 1, 1, 1], Prediction: 1, Expected: 1
Input: [0, 0, 0, 0, 0, 0, 1, 0, 0, 0], Prediction: 0, Expected: 0
Input: [0, 0, 0, 0, 0, 0, 1, 0, 0, 1], Prediction: 1, Expected: 1
```



Task C: Check if the Input in Decimal is Larger Than or Equal to 512

Test To Check Value Greater than or Equal to 512

In [204…

```
def test_larger_than(perceptron, threshold):
    for i in range(threshold-2, threshold + 10):
        binary_input = list(map(int, bin(i)[2:].zfill(input_size)))
        prediction = perceptron.predict(binary_input)
        if binary_input[-10]==1:
            expected = 1
        else:
            expected = 0
        print(f"Input: {binary_input}, Prediction: {prediction}, Expected: {expecte
```

Task Larger or Equal to Function

In [205…

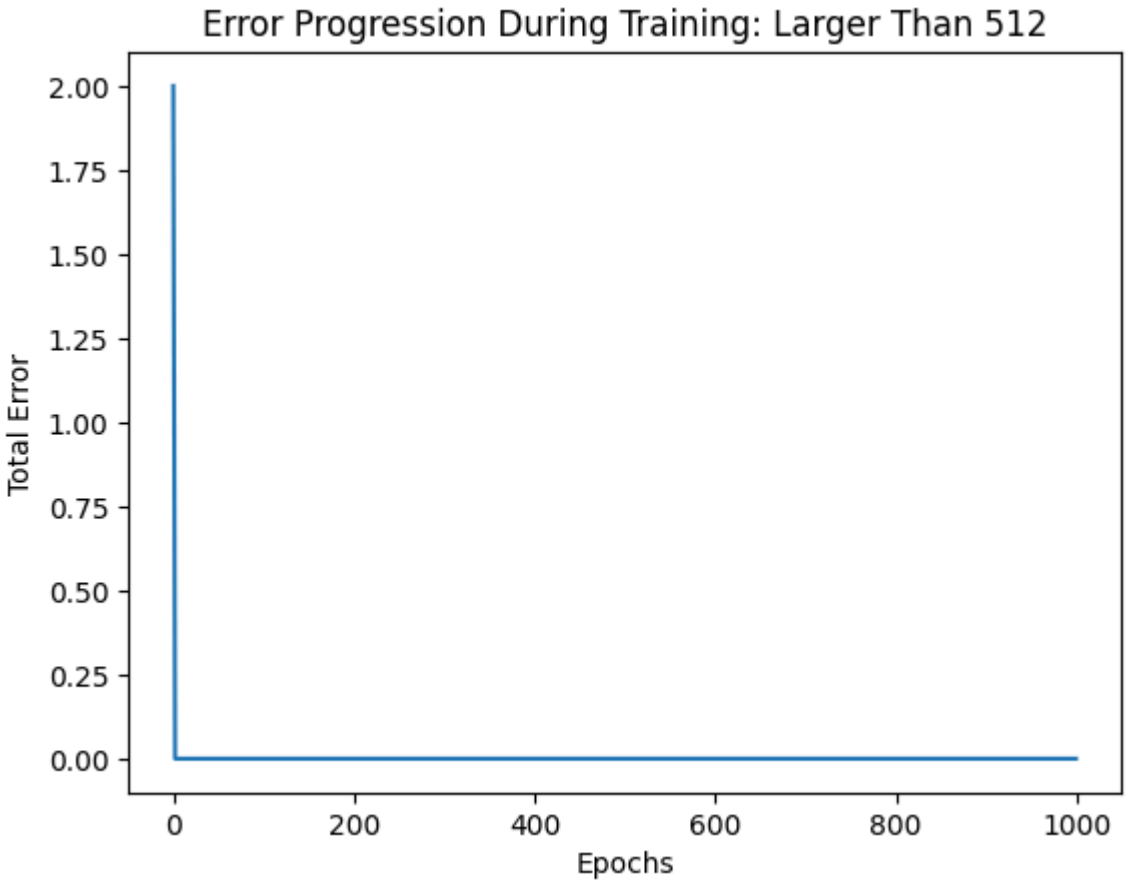
```
def task_larger_than_512():
    perceptron = Perceptron(input_size)
    targets = []
    for i in binary_inputs:
        decimal_value = int("".join(map(str, i)), 2)
        if decimal_value >= 512:
            targets.append(1)
        else:
            targets.append(0)
    perceptron.train(binary_inputs, targets)
    print(f'Weights: {perceptron.weights}')
    test_larger_than(perceptron, threshold=512)
    all_errors.append(perceptron.errors)
    plot_error(perceptron.errors, "Larger Than 512")
```

Run the Greater than or Equal To Task: Output Weights and a Graph of Total Error

In [206…

```
task_larger_than_512()
```

```
Weights: [-0.1  0.1  0.   0.   0.   0.   0.   0.   0.   0.   0. ]
Input: [0, 1, 1, 1, 1, 1, 1, 1, 1, 0], Prediction: 0, Expected: 0
Input: [0, 1, 1, 1, 1, 1, 1, 1, 1, 1], Prediction: 0, Expected: 0
Input: [1, 0, 0, 0, 0, 0, 0, 0, 0, 0], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 0, 0, 1], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 0, 1, 0], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 0, 1, 1], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 1, 0, 0], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 1, 0, 1], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 1, 1, 0], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 1, 1, 1], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 1, 0, 0, 0], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 1, 0, 0, 1], Prediction: 1, Expected: 1
```



Task D: Check if the Input in Decimal is Smaller than 512

Test To Check Value Less than 512

In [212…

```
def test_smaller_than(perceptron, threshold):
    for i in range(threshold - 10, threshold+2):
        binary_input = list(map(int, bin(i)[2:].zfill(input_size)))
        prediction = perceptron.predict(binary_input)
        if binary_input[-10]!=1:
            expected = 1
        else:
            expected = 0
        print(f"Input: {binary_input}, Prediction: {prediction}, Expected: {expecte
```

Task Smaller than Function

In [213…

```
def task_smaller_than_512():
    perceptron = Perceptron(input_size)
    targets = []
    for i in binary_inputs:
        decimal_value = int("".join(map(str, i)), 2)
        if decimal_value < 512:
            targets.append(1)
        else:
```

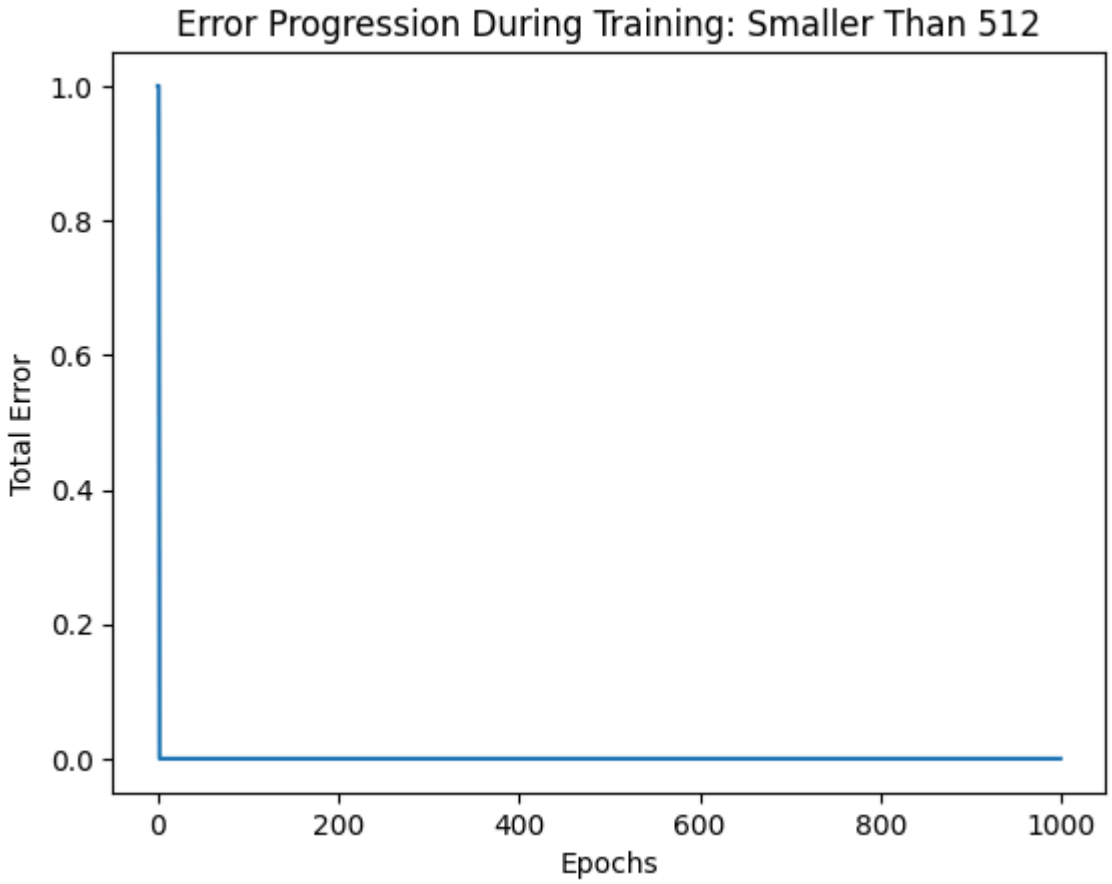
```
            targets.append(0)
    perceptron.train(binary_inputs, targets)
    print(f'Weights: {perceptron.weights}')
    test_smaller_than(perceptron, threshold=512)
    all_errors.append(perceptron.errors)
    plot_error(perceptron.errors, "Smaller Than 512")
```

Run the Greater than or Equal To Task: Output Weights and a Graph of Total Error

In [214…

```
task_smaller_than_512()
```

```
Weights: [ 0.  -0.1  0.   0.   0.   0.   0.   0.   0.   0.   0. ]
Input: [0, 1, 1, 1, 1, 1, 0, 1, 1, 0], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 0, 1, 1, 1], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 0, 0, 0], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 0, 0, 1], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 0, 1, 0], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 0, 1, 1], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 1, 0, 0], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 1, 0, 1], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 1, 1, 0], Prediction: 1, Expected: 1
Input: [0, 1, 1, 1, 1, 1, 1, 1, 1, 1], Prediction: 1, Expected: 1
Input: [1, 0, 0, 0, 0, 0, 0, 0, 0, 0], Prediction: 0, Expected: 0
Input: [1, 0, 0, 0, 0, 0, 0, 0, 0, 1], Prediction: 0, Expected: 0
```



In [ ]:

Task E: Check if the Input in Decimal Equal to 872

Test To Equal to 872

In [244…

```
def test_equal_to(perceptron, value):
    for i in range(value-2, value + 3):
        binary_input = list(map(int, bin(i)[2:].zfill(input_size)))
        prediction = perceptron.predict(binary_input)
        if (binary_input[-4] == 1 and binary_input[-6] == 1 and binary_input[-7] ==
            expected = 1
        else:
            expected = 0
        print(f"Input: {binary_input}, Prediction: {prediction}, Expected: {expecte
```

Task Equal to than Function

In [245…

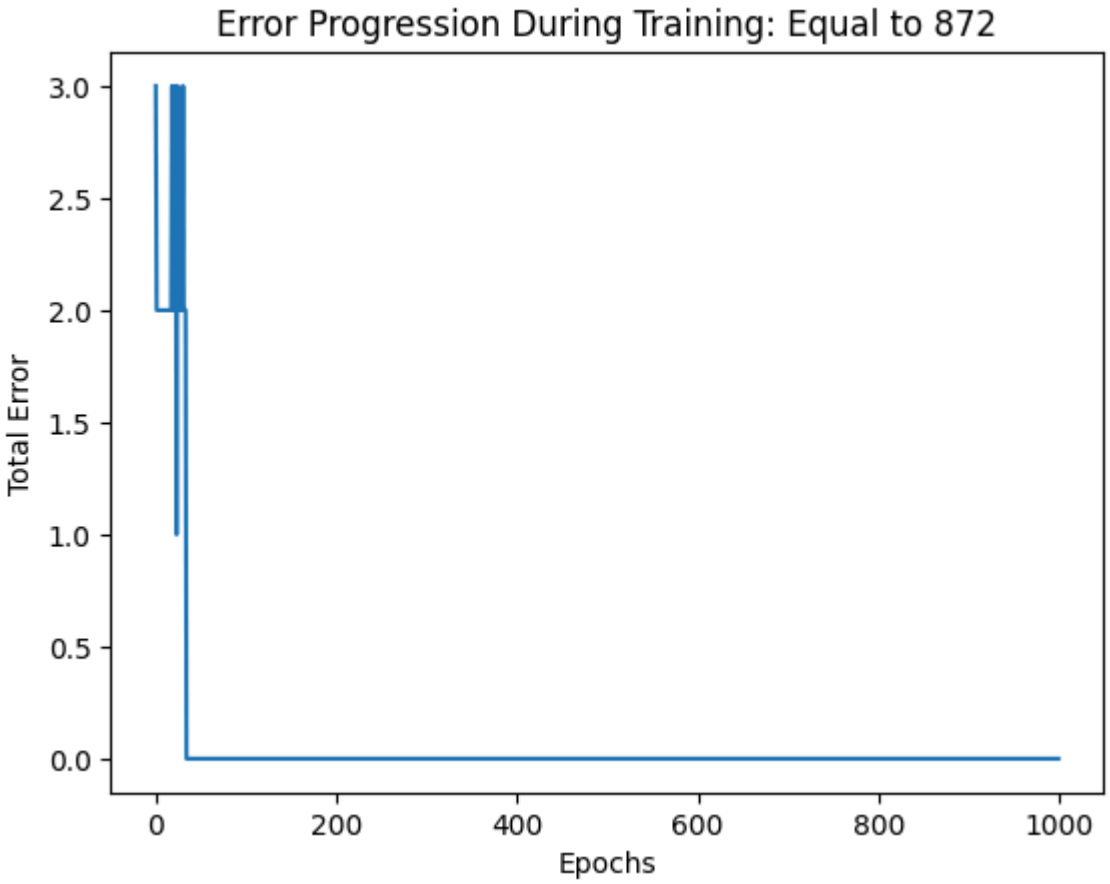
```
def task_equal_872():
    perceptron = Perceptron(input_size)
    targets = []
    for i in binary_inputs:
        decimal_value = int("".join(map(str, i)), 2)
        if decimal_value == 872:
            targets.append(1)
        else:
            targets.append(0)
    perceptron.train(binary_inputs, targets)
    print(f'Weights: {perceptron.weights}')
    test_equal_to(perceptron, value=872)
    all_errors.append(perceptron.errors)
    plot_error(perceptron.errors, "Equal to 872")
```

Run the Equal To Task: Output Weights and a Graph of Total Error

In [246…

```
task_equal_872()
```

```
Weights: [-0.4  0.1  0.1 -0.5  0.1  0.1 -0.7  0.1 -0.7 -0.7 -0.7]
Input: [1, 1, 0, 1, 1, 0, 0, 1, 1, 0], Prediction: 0, Expected: 0
Input: [1, 1, 0, 1, 1, 0, 0, 1, 1, 1], Prediction: 0, Expected: 0
Input: [1, 1, 0, 1, 1, 0, 1, 0, 0, 0], Prediction: 1, Expected: 1
Input: [1, 1, 0, 1, 1, 0, 1, 0, 0, 1], Prediction: 0, Expected: 0
Input: [1, 1, 0, 1, 1, 0, 1, 0, 1, 0], Prediction: 0, Expected: 0
```



Resources

https://www.sharpsightlabs.com/blog/python-perceptron-from-scratch/

https://medium.com/@becaye-balde/perceptron-building-it-from-scratch-in-python-15716806ef64

https://pyimagesearch.com/2021/05/06/implementing-the-perceptron-neural-network-with-python/

https://boccignone.di.unimi.it/IN_2019_files/MacKay_SingleNeuronClassifier.pdf