import numpy as np

class Perceptron:

def \_\_init\_\_(self, input\_size, learning\_rate=0.1):

self.weights = np.zeros(input\_size)

self.learning\_rate = learning\_rate

def predict(self, inputs):

return 1 if np.dot(inputs, self.weights) >= 0 else 0

def train(self, inputs, targets, epochs=100):

for \_ in range(epochs):

for input\_vector, target in zip(inputs, targets):

self.weights += self.learning\_rate \* (target - self.predict(input\_vector)) \* input\_vector

# Convert digit to ASCII binary form

def number\_to\_binary(number):

return [int(b) for b in format(ord(str(number)), '07b')]

# Input: Numbers 0-9 and output: Even -> 1, Odd -> 0

inputs = np.array([number\_to\_binary(i) for i in range(10)])

outputs = np.array([1 if i % 2 == 0 else 0 for i in range(10)])

# Train Perceptron

perceptron = Perceptron(7)

perceptron.train(inputs, outputs)

# Test Perceptron

for i in range(10):

binary\_input = number\_to\_binary(i)

prediction = perceptron.predict(binary\_input)

print(f"Input: {i} - Predicted: {'Even' if prediction == 1 else 'Odd'}")