pratical-3-ascii

May 9, 2024

[1]:

```
import numpy as np

class Perceptron:
    def __init__(self, input_size):
        self.weights = np.zeros(input_size + 1)  # Additional weight for bias
        self.learning_rate = 0.1

    def activate(self, inputs):
        # Activation function (Step function)
        return 1 if np.dot(inputs, self.weights[1:]) + self.weights[0] >= 0␣
 ↪else 0

    def train(self, training_inputs, labels, epochs):
        for _ in range(epochs):
            for inputs, label in zip(training_inputs, labels):
                prediction = self.activate(inputs)
                self.weights[1:] += self.learning_rate * (label - prediction) *␣
 ↪inputs
                self.weights[0] += self.learning_rate * (label - prediction)

def preprocess_input(num):
    # Convert ASCII representation of digits to binary
    return [int(bit) for bit in format(num, '08b')]  # 8-bit binary␣
 ↪representation

def is_even(num):
    return num % 2 == 0

def is_odd(num):
    return not is_even(num)

# Training data
training_data = []
labels = []

for i in range(48, 58):  # ASCII values for digits 0 to 9
    binary_representation = preprocess_input(i)
    training_data.append(binary_representation)
    labels.append(1 if is_even(i) else 0)  # Label 1 for even, 0 for odd

# Create and train the perceptron
perceptron = Perceptron(input_size=8)  # 8 bits for ASCII representation
perceptron.train(np.array(training_data), np.array(labels), epochs=100)

# Test the perceptron
test_numbers = [49, 50, 51, 52, 53, 54, 55, 56, 57, 48]  # ASCII values for␣
 ↪digits 1 to 9 and 0
for num in test_numbers:
    input_data = preprocess_input(num)
    prediction = perceptron.activate(input_data)
    print(f"Number: {chr(num)}, Prediction: {'Even' if prediction == 1 else␣
 ↪'Odd'}")
```

```
Number: 1, Prediction: Odd
Number: 2, Prediction: Even
Number: 3, Prediction: Odd
Number: 4, Prediction: Even
Number: 5, Prediction: Odd
Number: 6, Prediction: Even
Number: 7, Prediction: Odd
Number: 8, Prediction: Even
Number: 9, Prediction: Odd
Number: 0, Prediction: Even
```

```
[ ]:
```