notes_01

October 20, 2024

1 Neuron with 3 inputs

[4]:
```
inputs = [1, 2, 3]
weights = [0.2, 0.8, -0.5]
bias = 2

output = (inputs[0]*weights[0] + inputs[1]*weights[1] + inputs[2]*weights[2] +␣
 ↪bias)

print(f"Output: {output}")
```

```
Output: 2.3
```

2 Neuron with 4 inputs

[5]:
```
inputs = [1.0, 2.0, 3.0, 2.5]
weights = [0.2, 0.8, -0.5, 1.0]
bias = 2

output = (inputs[0]*weights[0] + inputs[1]*weights[1] + inputs[2]*weights[2] +␣
 ↪inputs[3]*weights[3] + bias)

print(output)
```

```
Output: 4.8
```

3 Layer of 3 neurons with 4 inputs

[1]:
```
num_neurons = 3
num_inputs = 4
inputs = [1.0, 2.0, 3.0, 2.5]
weights = [[0.2, 0.8, -0.5, 1.0],
           [0.5, -0.91, 0.26, -0.5],
           [-0.26, -0.27, 0.17, 0.87]]
biases = [2, 3, 0.5]
```

```
outputs = []
for i in range(num_neurons):
    output = 0
    for j in range(num_inputs):
        output += inputs[j]*weights[i][j]
    output += biases[i]
    outputs.append(output)

print(outputs)
```

```
[4.8, 1.21, 2.385]
```

4 Layer of 3 neurons with 4 inputs

[2]:
```
num_neurons = 3
num_inputs = 4
inputs = [1.0, 2.0, 3.0, 2.5]
weights = [[0.2, 0.8, -0.5, 1.0],
           [0.5, -0.91, 0.26, -0.5],
           [-0.26, -0.27, 0.17, 0.87]]
biases = [2, 3, 0.5]

outputs = []
for neuron_weights, neuron_bias in zip(weights, biases):
    neuron_output = 0
    for input, weight in zip(inputs, neuron_weights):
        neuron_output += input*weight
    neuron_output += neuron_bias
    outputs.append(neuron_output)

print(outputs)
```

```
[4.8, 1.21, 2.385]
```

5 Single Neuron using Numpy

[6]:
```
import numpy as np

inputs = [1.0, 2.0, 3.0, 2.5]
weights = [0.2, 0.8, -0.5, 1.0]
bias = 2

output = np.dot(inputs, weights) + bias
print(output)
```

```
4.8
```

6 Layer of Neurons Using Numpy

[7]:
```
import numpy as np

inputs = [1.0, 2.0, 3.0, 2.5]
weights = np.array([
    [0.2, 0.8, -0.5, 1],
    [0.5, -0.91, 0.26, -0.5],
    [-0.26, -0.27, 0.17, 0.87]
])
biases = [2.0, 3.0, 0.5]

layer_outputs = np.dot(weights, inputs) + biases
# must be dot(weights, inputs), not dot(inputs, weights)
# this takes the dot each row of the weights by the column of inputs (remember␣
 ↪the second term is transposed)
print(layer_outputs)

layer_outputs_2 = np.dot(inputs, weights.T) + biases
# this takes each input and multiplies by the weight. also correct.
print(layer_outputs_2)
```

```
[4.8   1.21  2.385]
[4.8   1.21  2.385]
```

7 Layer of Neurons and Batch of Data Using Numpy

Batch of data is simply a set of inputs.

[9]:
```
import numpy as np

inputs = [ # Batch of inputs
    [1.0, 2.0, 3.0, 2.5],
    [2.0, 5.0, -1.0, 2.0],
    [-1.5, 2.7, 3.3, -0.8]
]
weights = np.array([
    [0.2, 0.8, -0.5, 1],
    [0.5, -0.91, 0.26, -0.5],
    [-0.26, -0.27, 0.17, 0.87]
])
biases = [2.0, 3.0, 0.5]

outputs = np.dot(inputs, weights.T) + biases
# For every row of inputs, compute the dot of input set and weights
print(outputs)
```

```
[[ 4.8    1.21   2.385]
 [ 8.9   -1.81   0.2  ]
 [ 1.41   1.051  0.026]]
```