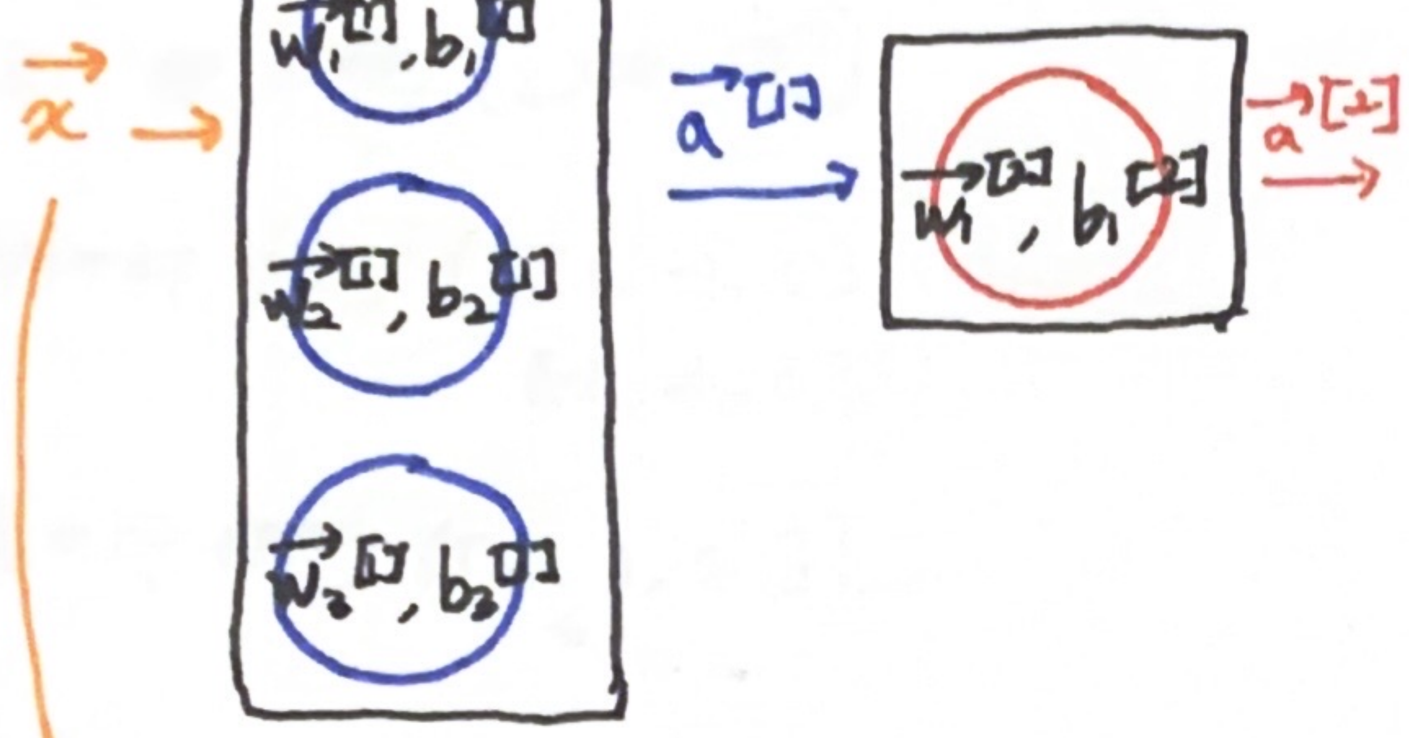
<Inference in code> (Python)

ex) coffee roasting model



* $\vec{w}_1^{[1]} \xrightarrow{\text{code}}$ W1_1 $= \begin{bmatrix} 1 \\ 2 \end{bmatrix}$: $b_1^{[1]} = -1$

$\vec{w}_2^{[1]} \longrightarrow$ W1_2 $= \begin{bmatrix} -3 \\ 4 \end{bmatrix}$: $b_2^{[1]} = 1$

$\vec{w}_3^{[1]} \longrightarrow$ W1_3 $= \begin{bmatrix} 5 \\ -6 \end{bmatrix}$: $b_3^{[1]} = 2$

$\vec{w}_1^{[2]} \longrightarrow$ W2_1

* $a_1^{[1]} = g(\vec{w}_1^{[1]} \cdot \vec{x} + b_1^{[1]})$

$a_2^{[1]} = g(\vec{w}_2^{[1]} \cdot \vec{x} + b_2^{[1]})$

$a_3^{[1]} = g(\vec{w}_3^{[1]} \cdot \vec{x} + b_3^{[1]})$

$a_1^{[2]} = g(\vec{w}_1^{[2]} \cdot \vec{a}^{[1]} + b_1^{[2]})$

$a^{[0]} = \begin{bmatrix} -2 \\ 4 \end{bmatrix}$ (input)

```
x = np.array([200, 17])
```

```
# 1st unit (layer1)
W1_1 = np.array([1,2])
b1_1 = np.array([-1])
Z1_1 = np.dot(W1_1, x) + b1_1
a1_1 = sigmoid(Z1_1)
```

```
# 2nd unit (layer1)
W1_2 = np.array([-3, 4])
b1_2 = np.array([1])
Z1_2 = np.dot(W1_2, x) + b1_2
a1_2 = sigmoid(Z1_2)
```

```
# 3rd unit (layer1)
W1_3 = np.array([5, -6])
b1_3 = np.array([2])
Z1_3 = np.dot(W1_3, x) + b1_3
a1_3 = sigmoid(Z1_3)
```

```
a1 = np.array([a1_1, a1_2, a1_3])
```

```
# 1st unit (layer2)
W2_1 = np.array([-7, 8])
b2_1 = np.array([3])
Z2_1 = np.dot(W2_1, a1) + b2_1
a2_1 = sigmoid(Z2_1)
```

⇓

```
W = np.array([
        [1, -3, 5],
        [2, 4, -6]])
b = np.array([-1, 1, 2])
a_in = np.array([-2, 4])
```

```
def dense(a_in, W, b, g):
  units = W.shape(1)
  a_out = np.zeros(units)
  for j in range(units):
      w = W[:, j]
      z = np.dot(w, a_in) + b[j]
      a_out[j] = g(z)
  return a_out
```

```
def sequential(x):
    a1 = dense(x, W1, b1)
    a2 = dense(a1, W2, b2)
    a3 = dense(a2, W3, b3)
    a4 = dense(a3, W4, b4)
    f_x = a4
    return f_X
```