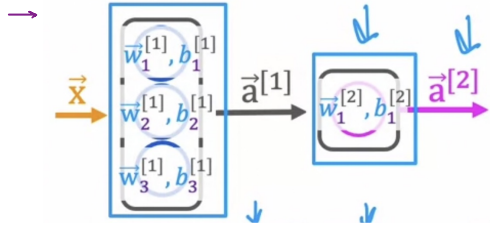
Issue :-

Understanding the logic behind what's happening during Forward prop. instead of just memorizing Tensorflow code.

Relevant point :-

- Hard code for each layer and its hidden neurons.




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

```
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)
```

```
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)
```

layer 1

```
w2_1 = np.array([-7, 8, 9])
b2_1 = np.array([3])
z2_1 = np.dot(w2_1, x) + b2_1
a2_1 = sigmoid(z2_1)
```

layer 2

- Create a Function —

→ we'll stack $\vec{w}$ and create a matrix.

$\vec{w}_1^{[1]} = \begin{bmatrix}1\\2\end{bmatrix}$ $\vec{w}_2^{[1]} = \begin{bmatrix}-3\\4\end{bmatrix}$ $\vec{w}_3^{[1]} = \begin{bmatrix}5\\-6\end{bmatrix}$

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

similarly,

$b_1^{[l]} = -1$ $b_2^{[l]} = 1$ $b_3^{[l]} = 2$

```
b = np.array([-1, 1, 2])
```

Function :-

prev. layer output → a_in = np.array([-2,4])

```
def dense(a_in,W,b):
    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
```

range (3) = 0, 1, 2 times loop

shape = (2, 3) ⇒ shape[1] = 3 ⇒ 3 neurons
row column

final output is initialized to (0, 0, 0) [np.zeroes (3)]

If j == 0 :

w = W[:, 0] ⇒ $w = \begin{bmatrix}1\\2\end{bmatrix}$

z = np. dot (w, a_in) + b[0]

a_out[j] = g(z) [sigmoid]

The above loop goes on for 1 and 2 also.
Create another function for sequential.

```
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
```

'W' refers to matrix
'w' refers to vectors, scalars

```
def dense(a_in,W,b):
  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
```