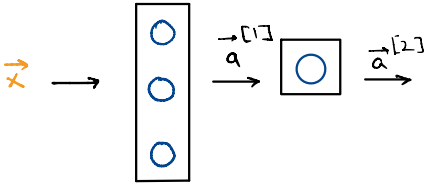
Tensorflow helps us avoid the manual forward propagation.



		y
200	17	1
120	5	0
425	20	0
212	18	1

```
layer_1 = Dense(units = 3, activation = "sigmoid")
layer_2 = Dense(units = 1, activation = "sigmoid")
```

creating required layers

instead of manual forward prop, we use tensorflow

```
model = sequential([layer_1, layer_2])
```

← forms a neural network by sequencially stringing the layers together

```
x = np.array([[200,17],[120,5],[425,20],[218,18]])
y = np.array([1,0,0,1])
```
← output stored as a 1-D array

To train the network using x and y :-

↳ some parameters for compile function

```
model.compile(...)
model.fit(x,y)
```

Predict values :-

In order to predict target value for a new x array (called x_new), we call :-

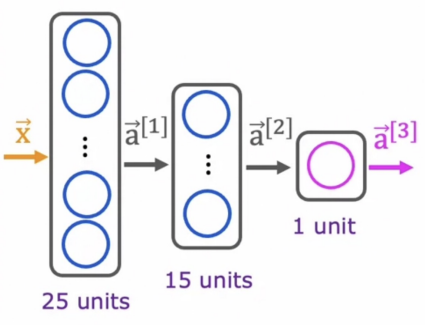
```
model.predict(x_new)
```

Originally, we don't explicitly assign the layers and then create the neural network.

Conventionally,

```
model = Sequential([
            Dense(units=3, activation="sigmoid"),
            Dense(units=1, activation="sigmoid")])
```

Showcasing this for digit classification question




```
model = Sequential ([
        Dense(units = 25, activation = "sigmoid",
        Dense(units = 15, activation = "sigmoid",
        Dense(units = 1, activation = "sigmoid")])

model.compile(...)
x = np.array([[0..., 245, ..., 17],
              [0..., 200, ..., 184]])

y = np.array([1,0])

model.fit(x,y)

model.predict(x_new)
```