**NEURAL NETWORK TRAINING**

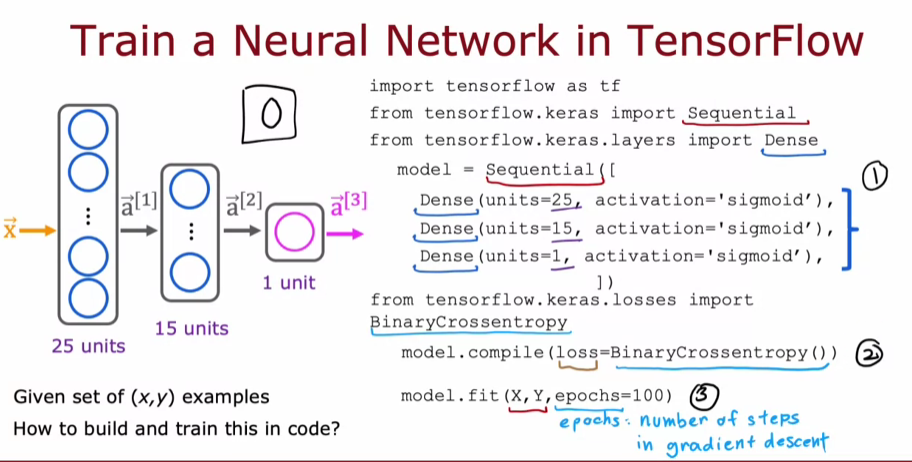
**TensorFlow IMPLEMENTATION**

**Training a Neural Network**

* **The process begins with defining the neural network architecture, which includes input layers, hidden layers, and output layers. For example, a network for handwritten digit recognition might have an input layer for images, two hidden layers, and one output unit.**
* **The next step is to compile the model by specifying a loss function, such as binary crossentropy, which helps measure how well the model is performing.**

**Fitting the Model**

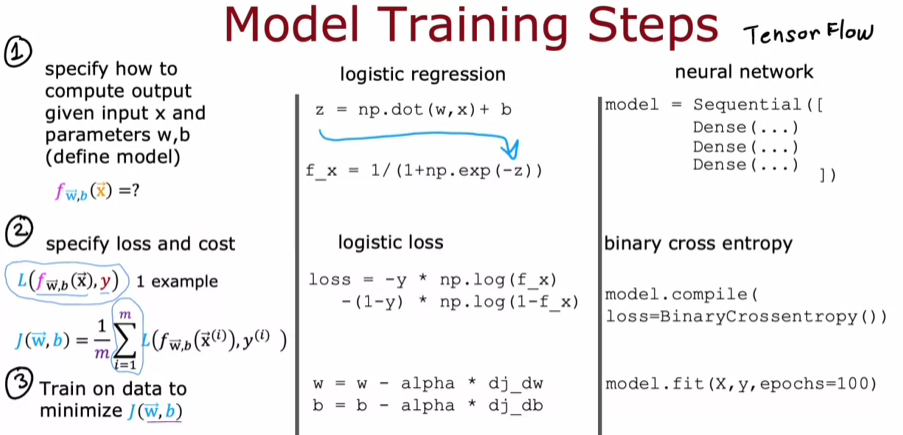
* **After compiling, the model is trained using the fit function, which adjusts the model parameters based on the training data (X, Y) and the specified loss function.**
* **The number of epochs, or iterations of the training process, is also defined, which determines how long the training will run.**

****

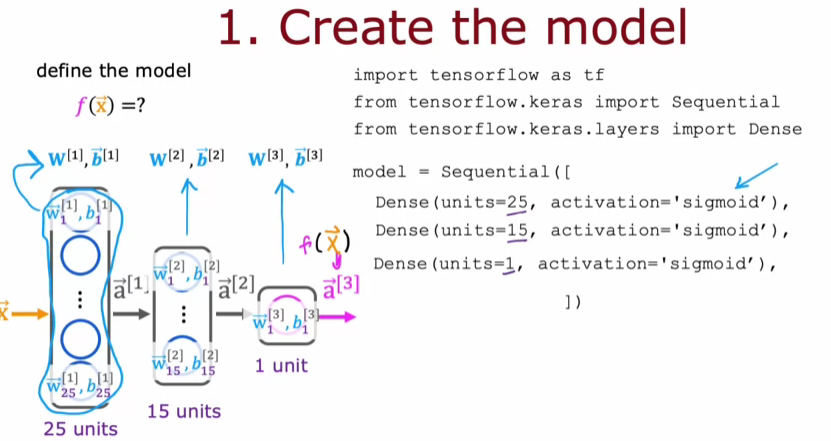
**TRAINING DETAILS**

**Steps to Train a Neural Network**

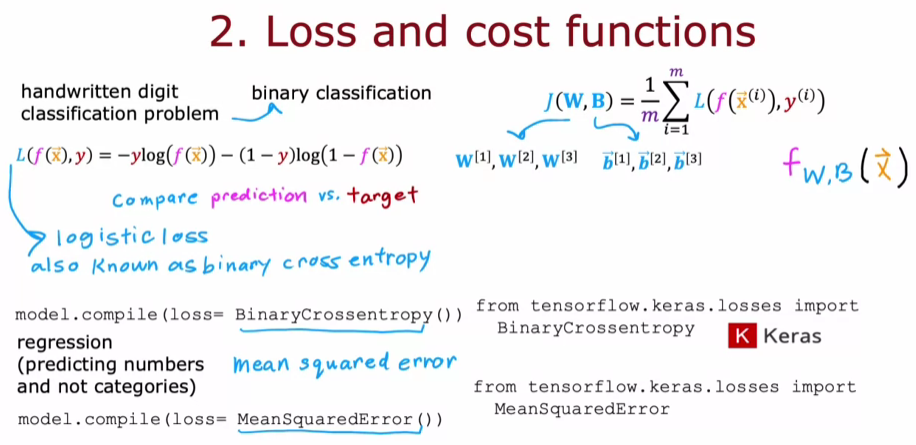
* **Step 1: Specify the Output Function**Define how to compute the output given the input features and model parameters. This involves setting up the architecture of the neural network, including the number of layers and activation functions.
* **Step 2: Define the Loss and Cost Functions**Specify the loss function, such as binary cross-entropy for classification tasks, which measures how well the model performs on individual training examples. The cost function is the average of the loss across all training examples.
* **Step 3: Minimize the Cost Function**Use an optimization algorithm like gradient descent to minimize the cost function by updating the model parameters iteratively. TensorFlow automates this process through backpropagation and the fit function.

****

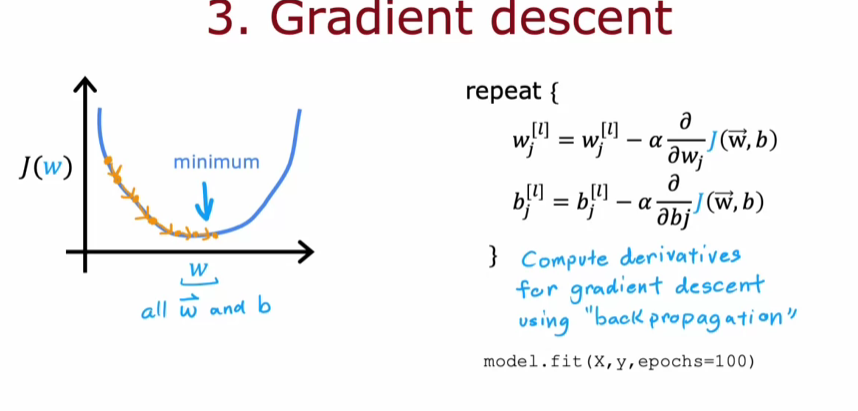
**STEP 1: CREATE THE MODEL**

****

**STEP 2: LOSS AND COST FUNCTIONS**

****

**STEP 3: GRADIENT DESCENT**

****