tensors.

Initialization

```
init = tf.global_variables_initializer()
with tf.Session() as sess:
    sess.run(init)
```

The `tf.global_variables_initializer()` call returns an operation that will initialize all TensorFlow variables from the graph. You call the operation using a session to initialize all the variables as shown above. Using the `tf.Variable` class allows us to change the weights and bias, but an initial value needs to be chosen.

Initializing the weights with random numbers from a normal distribution is good practice. Randomizing the weights helps the model from becoming stuck in the same place every time you train it. You'll learn more about this in the next lesson, when you study gradient descent.

Similarly, choosing weights from a normal distribution prevents any one weight from overwhelming other weights. You'll use the `tf.truncated_normal()` function to generate random numbers from a normal distribution.

tf.truncated_normal()

```
n_features = 120
n_labels = 5
weights = tf.Variable(tf.truncated_normal((n_features, n_labels)))
```

The `tf.truncated_normal()` function returns a tensor with random values from a normal distribution whose magnitude is no more than 2 standard deviations from the mean.

Since the weights are already helping prevent the model from getting stuck, you don't need to randomize the bias. Let's use the simplest solution, setting the bias to 0.