Linear functions in TensorFlow

The most common operation in neural networks is calculating the linear combination of inputs, weights, and biases. As a reminder, we can write the output of the linear operation as

$$\mathbf{y} = \mathbf{x}\mathbf{W} + \mathbf{b}$$

Here, $\mathbf{W}$ is a matrix of the weights connecting two layers. The output $\mathbf{y}$, the input $\mathbf{x}$, and the biases $\mathbf{b}$ are all vectors.

Weights and Bias in TensorFlow

The goal of training a neural network is to modify weights and biases to best predict the labels. In order to use weights and bias, you'll need a Tensor that can be modified. This leaves out `tf.placeholder()` and `tf.constant()`, since those Tensors can't be modified. This is where `tf.Variable` class comes in.

tf.Variable()

```
x = tf.Variable(5)
```

The `tf.Variable` class creates a tensor with an initial value that can be modified, much like a normal Python variable. This tensor stores its state in the session, so you must initialize the state of the tensor manually. You'll use the `tf.global_variables_initializer()` function to initialize the state of all the Variable tensors.

Initialization

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