```
>>> c = tf.matmul(a, b)
>>> c.eval()
array([[ 3.,  3.,  3.,  3.],
       [ 3.,  3.,  3.,  3.]], dtype=float32)
```

You can check that this answer matches the mathematical definition of matrix multiplication we provided earlier.

Tensor Types

You may have noticed the `dtype` notation in the preceding examples. Tensors in TensorFlow come in a variety of types such as `tf.float32`, `tf.float64`, `tf.int32`, `tf.int64`. It's possible to to create tensors of specified types by setting `dtype` in tensor construction functions. Furthermore, given a tensor, it's possible to change its type using casting functions such as `tf.to_double()`, `tf.to_float()`, `tf.to_int32()`, `tf.to_int64()`, and others (Example 2-15).

Example 2-15. Creating tensors of different types

```
>>> a = tf.ones((2,2), dtype=tf.int32)
>>> a.eval()
array([[0, 0],
       [0, 0]], dtype=int32)
>>> b = tf.to_float(a)
>>> b.eval()
array([[ 0.,  0.],
       [ 0.,  0.]], dtype=float32)
```

Tensor Shape Manipulations

Within TensorFlow, tensors are just collections of numbers written in memory. The different shapes are views into the underlying set of numbers that provide different ways of interacting with the set of numbers. At different times, it can be useful to view the same set of numbers as forming tensors with different shapes. `tf.reshape()` allows tensors to be converted into tensors with different shapes (Example 2-16).

Example 2-16. Manipulating tensor shapes

```
>>> a = tf.ones(8)
>>> a.eval()
array([ 1.,  1.,  1.,  1.,  1.,  1.,  1.,  1.], dtype=float32)
>>> b = tf.reshape(a, (4, 2))
>>> b.eval()
array([[ 1.,  1.],
       [ 1.,  1.],
       [ 1.,  1.],
       [ 1.,  1.]], dtype=float32)
```