TensorFlow Introduction for beginners

Presented by
Youngnam Kim, Nayeong Kim
2018.04.24

This presentation is available here

goo.gl/Lbi2Bs

Computational Graph

All tensorflow programs consist of

I. Building the computational graph

II. Running the computational graph

If you don’t obey this rule, then you will meet a terrible situation

Computational Graph

Computational graph: a series of TensorFlow operations arranged into a graph. The graph is composed of two types of objects

- **Operations**: The nodes of the graph, describe calculations that consume and produce tensors.
- **Tensors:** The edges in the graph, represent the values that will flow through the graph

Computational Graph

Example code1 - simple computational graph

```
import tensorflow as tf

a = tf.constant(3.0, dtype=tf.float32)
b = tf.constant(4.0)
total = a + b
print(a)
print(b)
print(total)
```

Result

```
Tensor("Const:0", shape=(), dtype=float32)
Tensor("Const_1:0", shape=(), dtype=float32)
Tensor("add:0", shape=(), dtype=float32)
```

tf.Tensor does not return a value!!

Computational Graph

Session

I. **To run tensorflow objects - tensors & operations**

II. To manipulate tensorflow runtime information

Computational Graph

Example code2 - session to run 1

```
import tensorflow as tf

a = tf.constant(3.0, dtype=tf.float32)
b = tf.constant(4.0)
total = a + b

with tf.Session() as sess:
        a_value = sess.run(a)
        b_value = sess.run(b)
        total_value = sess.run(total)

        print a_value
        print b_value
        print total_value
```

Computational Graph

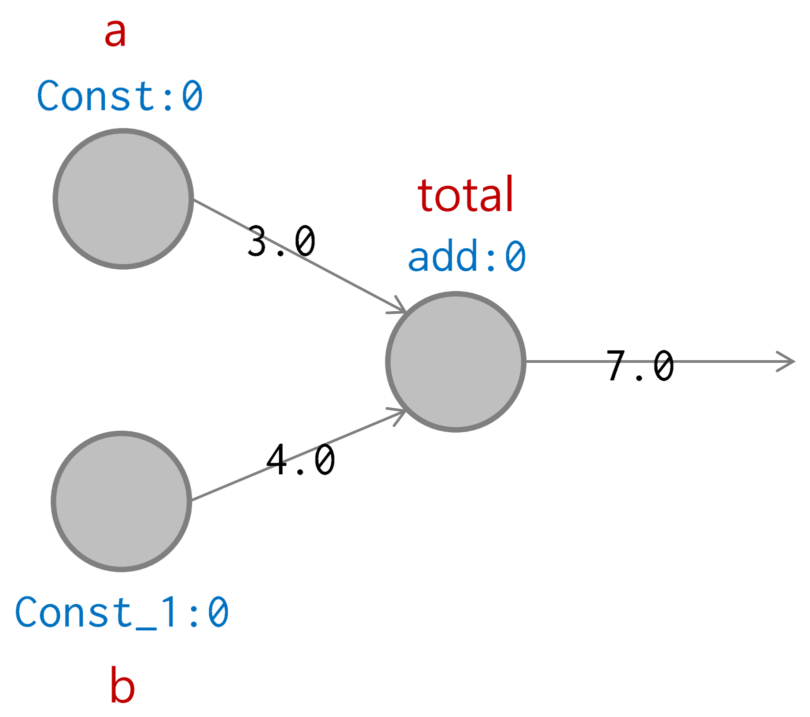
Example code2

Result

```
<some messages>
3.0
4.0
7.0
```

Computational Graph

Computational graph on example 1,2



operation name
python variable name

Computational Graph

Example code3 - session to run 2

```
import tensorflow as tf

a = tf.constant(3.0, dtype=tf.float32)
b = tf.constant(4.0)
total = a + b

with tf.Session() as sess:
        print(sess.run({‘ab’: (a,b), ‘total’: total}))
        print(sess.run([a,b,total]))
        print(sess.run([[a],[b],[total]]))
```

Computational Graph

Example code3 - session to run 2

Result

```
<some messages>
{'total': 7.0, 'ab': (3.0, 4.0)}
[3.0, 4.0, 7.0]
[[3.0], [4.0], [7.0]]
```

tf.Session.run returns value of the same structure as input

Components in TensorFlow

Basic components of computational graph

- **Variable:** a mutable variable (parameter)
- **Placeholder:** a tensor taking external inputs (input data)
- **Constant:** a non-mutable variable

Components in TensorFlow

Operations

- **Initialization:**
  - Initialize at a time - tf.global_variables_initializer()
  - Initialize locally - tf.Variable.initializer
- **Assignment:**
  - Assign specific value - tf.assign
- **Training:**
  - Optimizer
- **other operations:**
  - matrix multiplication
  - squeeze & expand dimension
  - argmax
  - functions which are similar to numpy

Components in TensorFlow

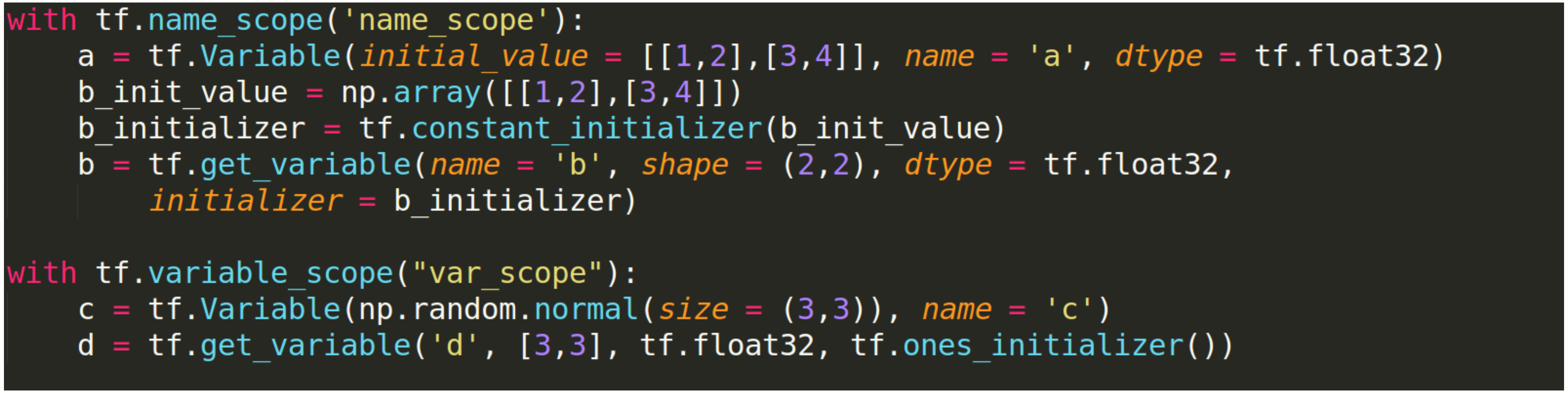
How to declare variables

- **tf.Variable** constructor
  - cannot be called by its operation name
  - stratified by **tf.name_scope**

- **tf.get_variable** function
  - can be called by its operation name
  - stratified by **tf.variable_scope**
  - the strongly recommended way

Components in TensorFlow

Example code4 - How to declare variables



```python
with tf.name_scope('name_scope'):
    a = tf.Variable(initial_value = [[1,2],[3,4]], name = 'a', dtype = tf.float32)
    b_init_value = np.array([[1,2],[3,4]])
    b_initializer = tf.constant_initializer(b_init_value)
    b = tf.get_variable(name = 'b', shape = (2,2), dtype = tf.float32,
        initializer = b_initializer)

with tf.variable_scope("var_scope"):
    c = tf.Variable(np.random.normal(size = (3,3)), name = 'c')
    d = tf.get_variable('d', [3,3], tf.float32, tf.ones_initializer())
```

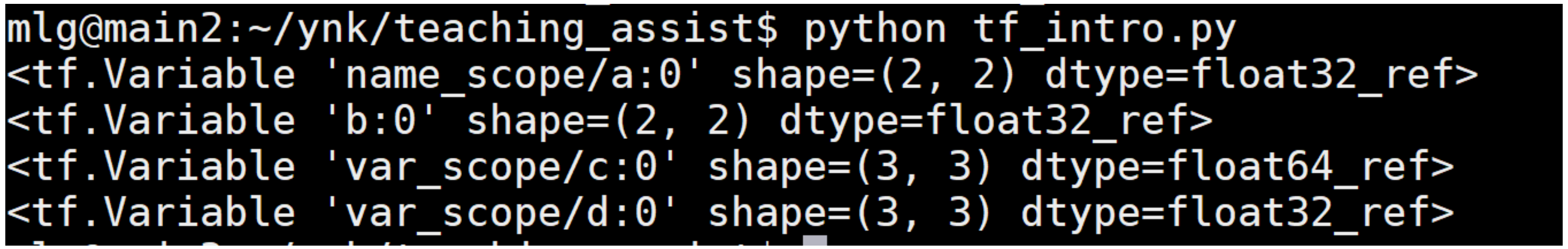
Try by yourself

1.print all variables

2.print values of variables

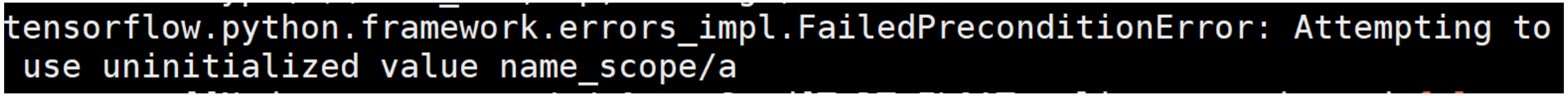
Components in TensorFlow

Results - After print



```
mlg@main2:~/ynk/teaching_assist$ python tf_intro.py
<tf.Variable 'name_scope/a:0' shape=(2, 2) dtype=float32_ref>
<tf.Variable 'b:0' shape=(2, 2) dtype=float32_ref>
<tf.Variable 'var_scope/c:0' shape=(3, 3) dtype=float64_ref>
<tf.Variable 'var_scope/d:0' shape=(3, 3) dtype=float32_ref>
```

Results - After run



```
tensorflow.python.framework.errors_impl.FailedPreconditionError: Attempting to
 use uninitialized value name_scope/a
```

Components in TensorFlow

Example code5 - How to initialize variables at a time

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

Try again to print values of variables

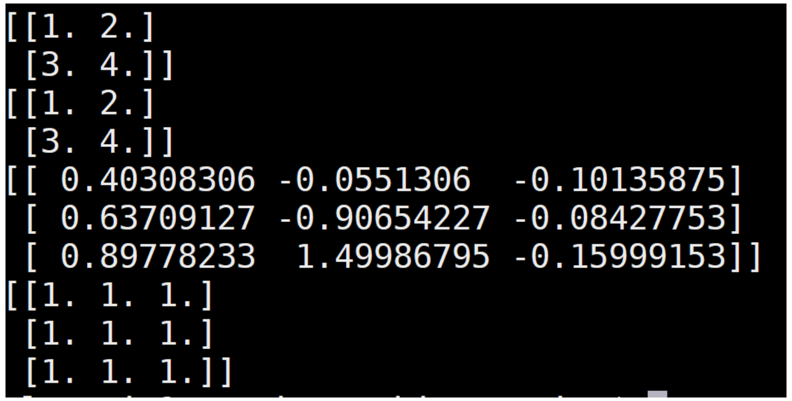
Example code6 - How to initialize variables locally

```
with tf.Session() as sess:
    sess.run(c.initializer)
```

Try again to print values of variables

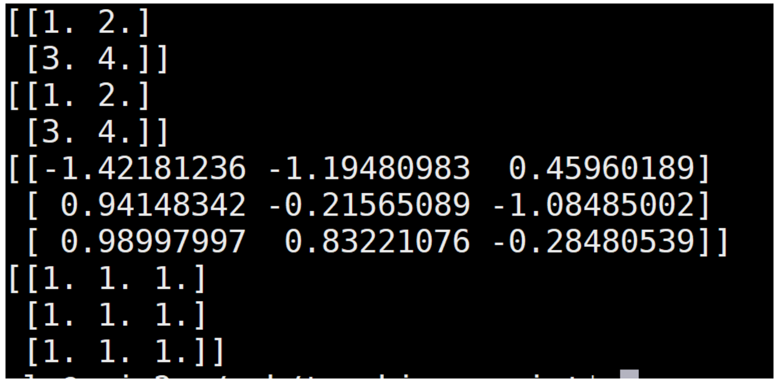
Components in TensorFlow

Example code5 - Result



```
[[1. 2.]
 [3. 4.]]
[[1. 2.]
 [3. 4.]]
[[ 0.40308306 -0.0551306   -0.10135875]
 [ 0.63709127 -0.90654227 -0.08427753]
 [ 0.89778233  1.49986795 -0.15999153]]
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

Example code6 - Result



```
[[1. 2.]
 [3. 4.]]
[[1. 2.]
 [3. 4.]]
[[-1.42181236 -1.19480983  0.45960189]
 [ 0.94148342 -0.21565089 -1.08485002]
 [ 0.98997997  0.83221076 -0.28480539]]
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

Components in TensorFlow

Example code7 - call variable by name

```
called = tf.get_variable('name_scope/a')

with tf.Session() as sess:
    print sess.run(called)
```

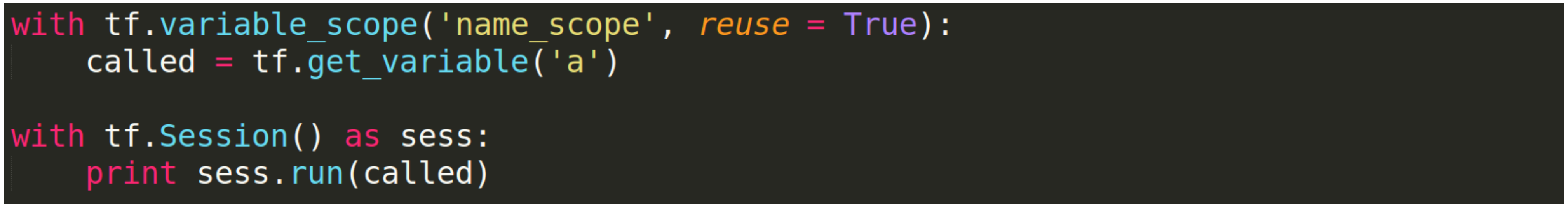
Results

```
ValueError: Shape of a new variable (name_scope/a) must be fully defined
, but instead was <unknown>.
```

don't be recognized by an existing variable

Components in TensorFlow

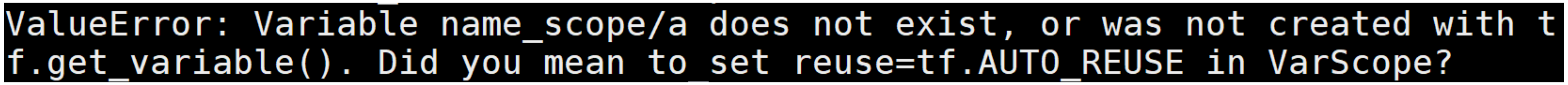
Example code7 - call variable by name



```python
with tf.variable_scope('name_scope', reuse = True):
    called = tf.get_variable('a')

with tf.Session() as sess:
    print sess.run(called)
```

Results

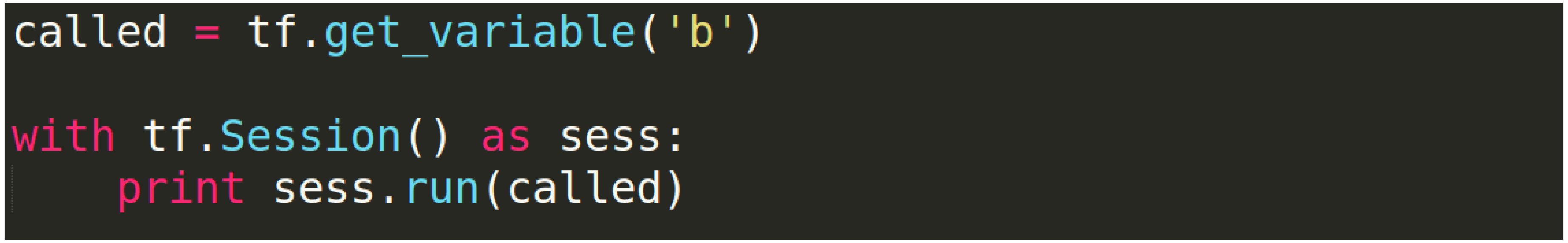


```
ValueError: Variable name_scope/a does not exist, or was not created with t
f.get_variable(). Did you mean to set reuse=tf.AUTO_REUSE in VarScope?
```

don’t be recognized by an existing variable

Components in TensorFlow

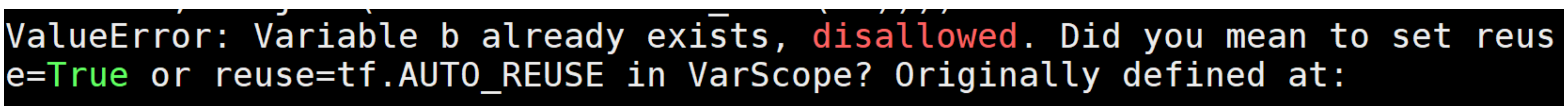
Example code7 - call variable by name



```
called = tf.get_variable('b')

with tf.Session() as sess:
    print sess.run(called)
```

Results

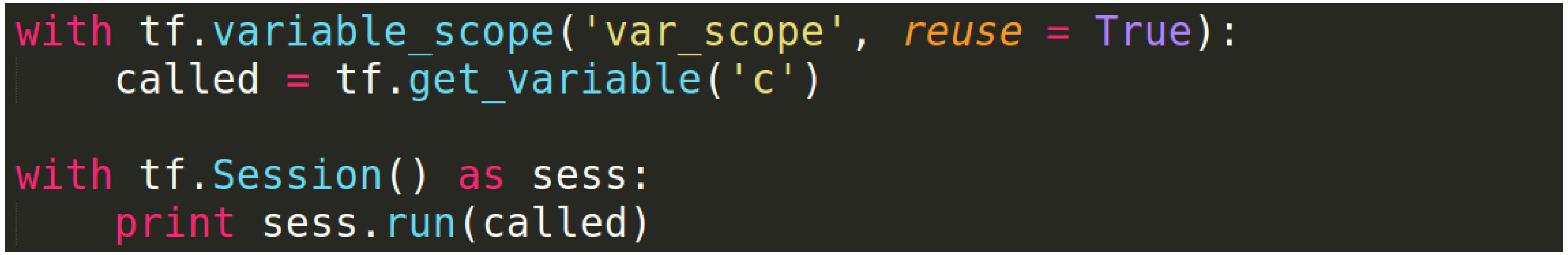


```
ValueError: Variable b already exists, disallowed. Did you mean to set reus
e=True or reuse=tf.AUTO_REUSE in VarScope? Originally defined at:
```

can be recognized but denied in default

Components in TensorFlow

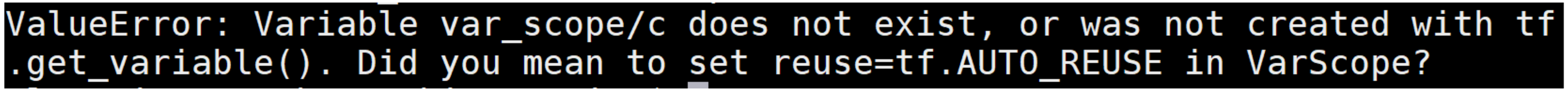
Example code7 - call variable by name



```
with tf.variable_scope('var_scope', reuse = True):
    called = tf.get_variable('c')

with tf.Session() as sess:
    print sess.run(called)
```

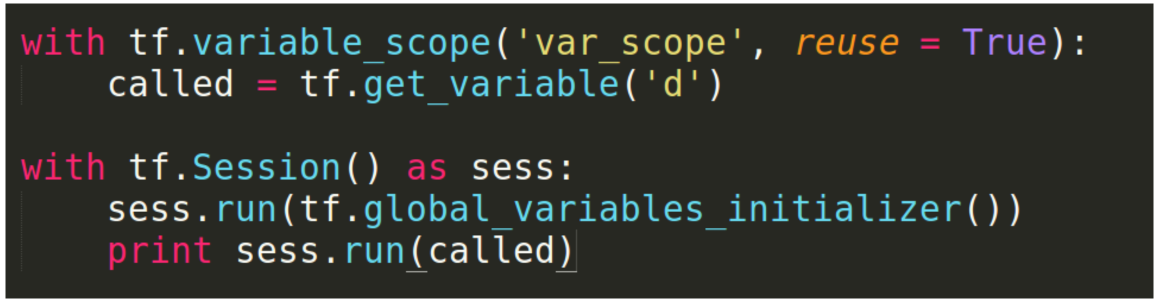
Results



```
ValueError: Variable var_scope/c does not exist, or was not created with tf
.get_variable(). Did you mean to set reuse=tf.AUTO_REUSE in VarScope?
```

Components in TensorFlow

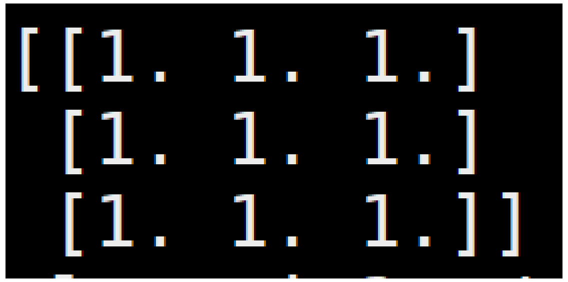
Example code7 - call variable by name



```
with tf.variable_scope('var_scope', reuse = True):
    called = tf.get_variable('d')

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    print sess.run(called)
```

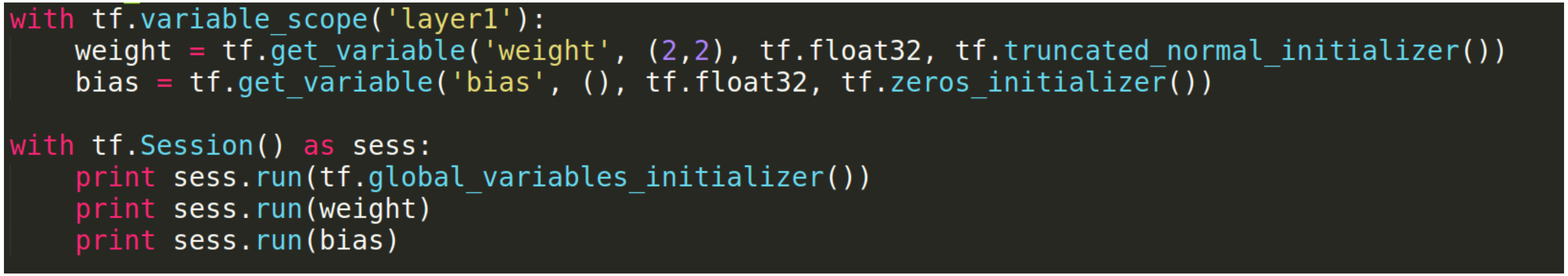
Results



```
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

Components in TensorFlow

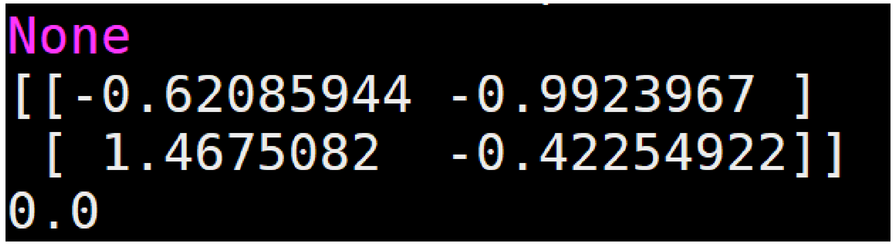
A strongly recommend way of declaring variables



```
with tf.variable_scope('layer1'):
    weight = tf.get_variable('weight', (2,2), tf.float32, tf.truncated_normal_initializer())
    bias = tf.get_variable('bias', (), tf.float32, tf.zeros_initializer())

with tf.Session() as sess:
    print sess.run(tf.global_variables_initializer())
    print sess.run(weight)
    print sess.run(bias)
```

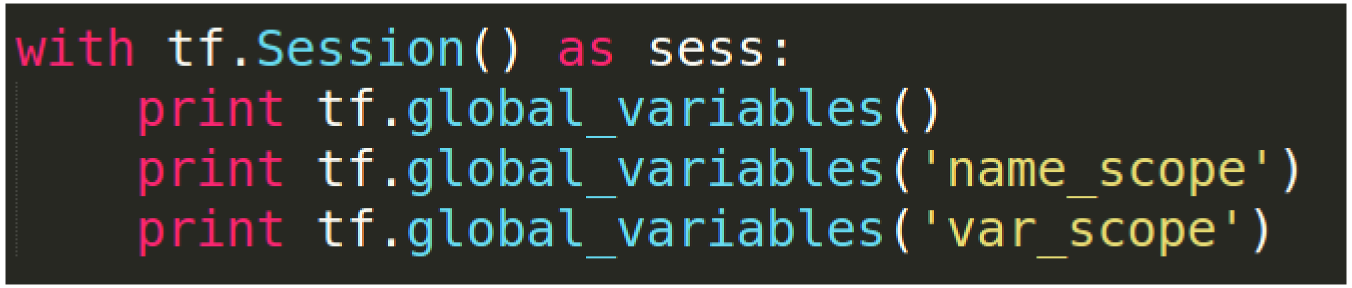
Results



```
None
[[-0.62085944 -0.9923967 ]
 [ 1.4675082  -0.42254922]]
0.0
```

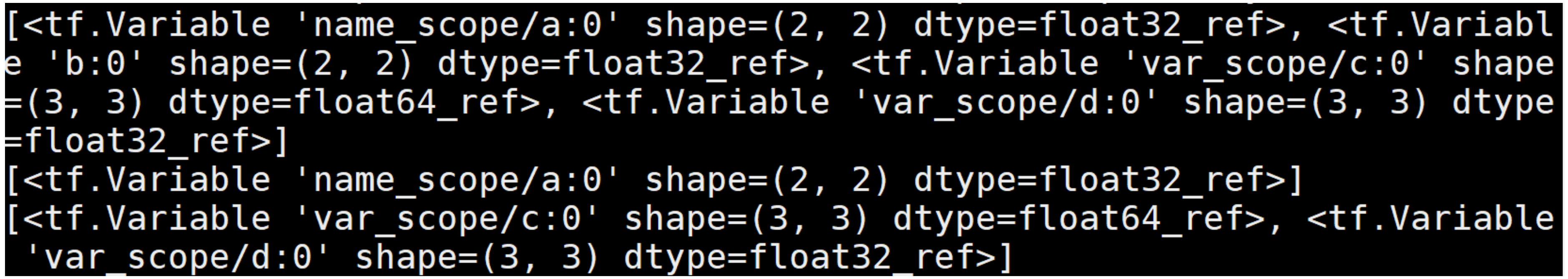
Components in TensorFlow

Example code8 - call variables by scope



```
with tf.Session() as sess:
    print tf.global_variables()
    print tf.global_variables('name_scope')
    print tf.global_variables('var_scope')
```

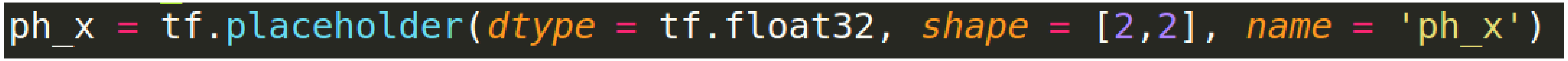
Results



```
[<tf.Variable 'name_scope/a:0' shape=(2, 2) dtype=float32_ref>, <tf.Variabl
e 'b:0' shape=(2, 2) dtype=float32_ref>, <tf.Variable 'var_scope/c:0' shape
=(3, 3) dtype=float64_ref>, <tf.Variable 'var_scope/d:0' shape=(3, 3) dtype
=float32_ref>]
[<tf.Variable 'name_scope/a:0' shape=(2, 2) dtype=float32_ref>]
[<tf.Variable 'var_scope/c:0' shape=(3, 3) dtype=float64_ref>, <tf.Variable
 'var_scope/d:0' shape=(3, 3) dtype=float32_ref>]
```

Components in TensorFlow

Example code9 - how to declare placeholder



```
ph_x = tf.placeholder(dtype = tf.float32, shape = [2,2], name = 'ph_x')
```

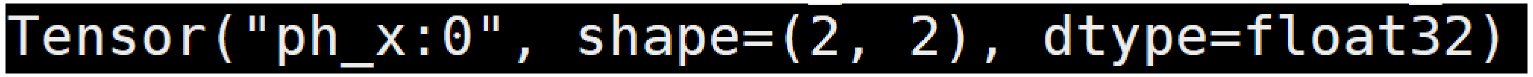
Try by yourself

1.print it

2.run and print it

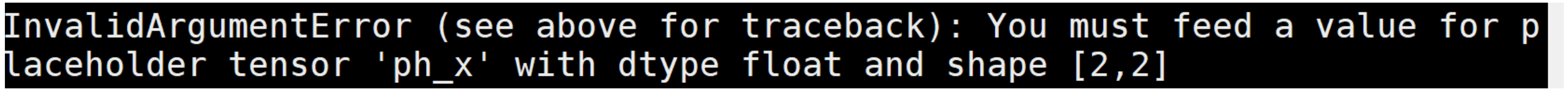
Components in TensorFlow

Result1



```
Tensor("ph_x:0", shape=(2, 2), dtype=float32)
```

Result2



```
InvalidArgumentError (see above for traceback): You must feed a value for p
laceholder tensor 'ph_x' with dtype float and shape [2,2]
```

Components in TensorFlow

Example code9 - how to feed placeholder

```
with tf.Session() as sess:
    print(sess.run(ph_x, feed_dict = {ph_x : [[1,2],[3,4]]}))
```

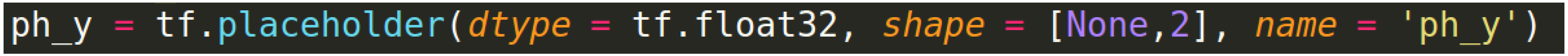
Result

```
[[1. 2.]
 [3. 4.]]
```

Try by yourself
1.feed another value with the same shape
2.feed another value with a different shape

Components in TensorFlow

Example code10 - how to feed placeholder



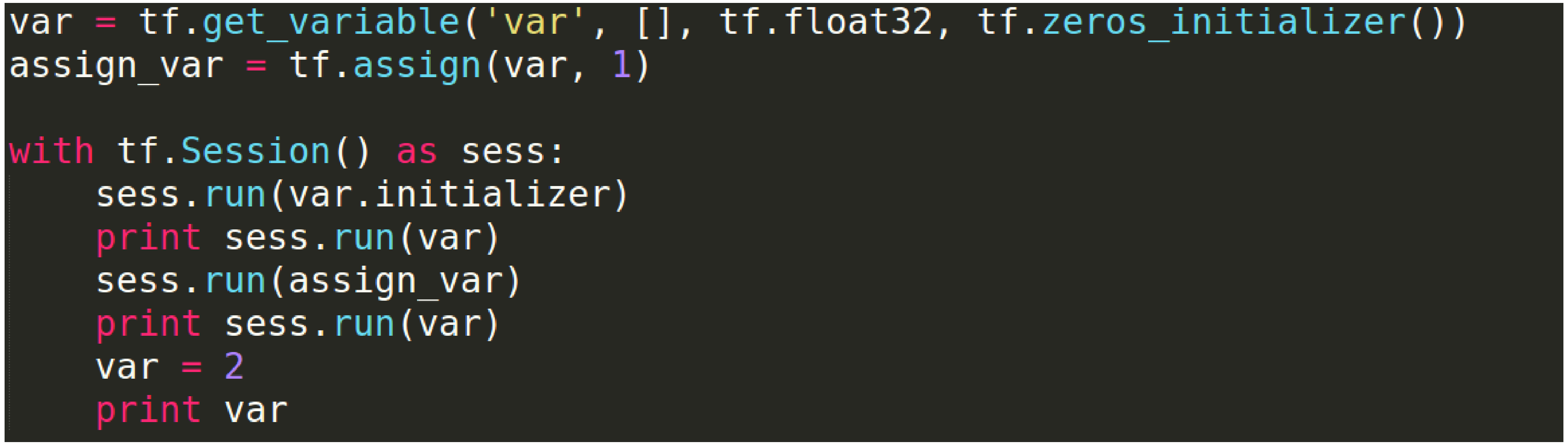
```
ph_y = tf.placeholder(dtype = tf.float32, shape = [None,2], name = 'ph_y')
```

Try by yourself

1. feed another value with the shape (2,2)
2. feed another value with the shape (3,2)

Components in TensorFlow

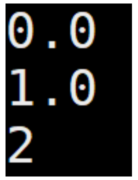
Example code11 - how to assign a specific value to a variable



```python
var = tf.get_variable('var', [], tf.float32, tf.zeros_initializer())
assign_var = tf.assign(var, 1)

with tf.Session() as sess:
    sess.run(var.initializer)
    print sess.run(var)
    sess.run(assign_var)
    print sess.run(var)
    var = 2
    print var
```

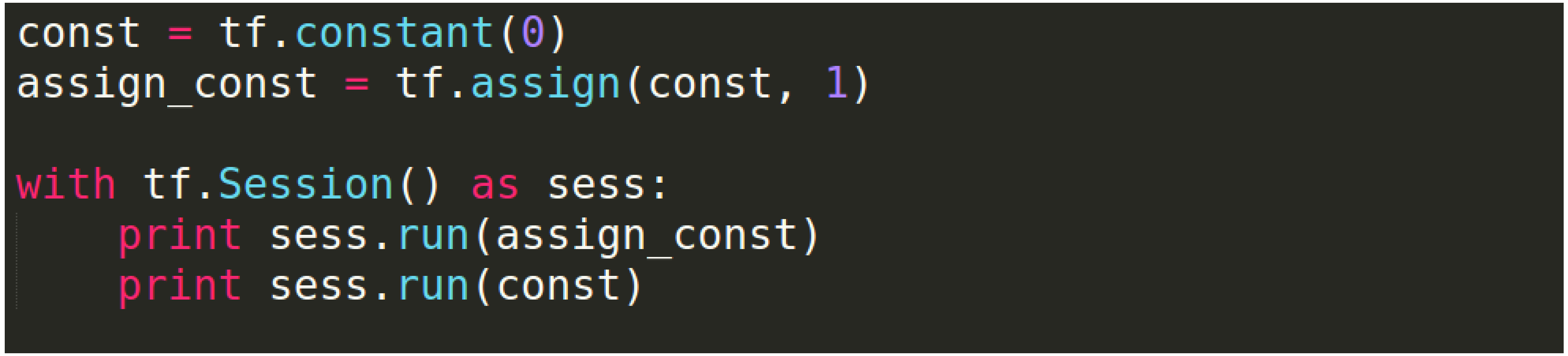
Result



```
0.0
1.0
2
```

Components in TensorFlow

Example code12 - constant and mutability



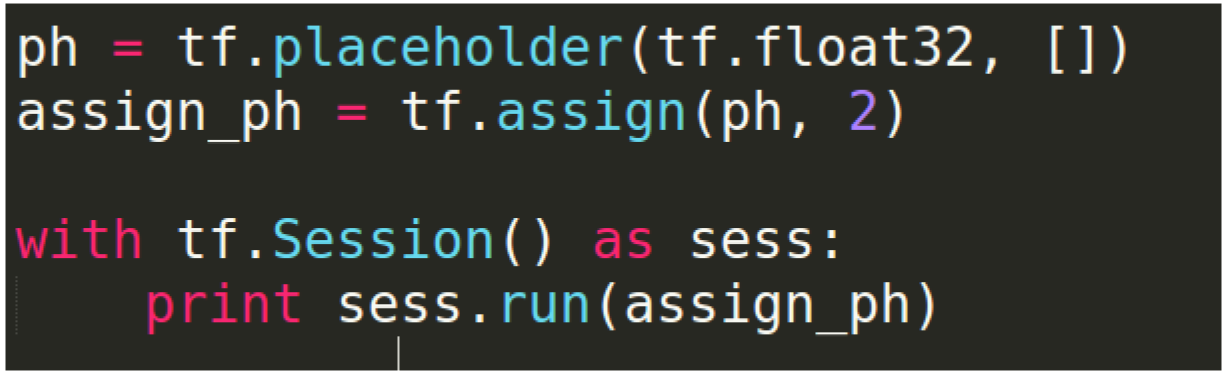
```
const = tf.constant(0)
assign_const = tf.assign(const, 1)

with tf.Session() as sess:
    print sess.run(assign_const)
    print sess.run(const)
```

Try and see the results

Components in TensorFlow

Example code12 - constant and mutability



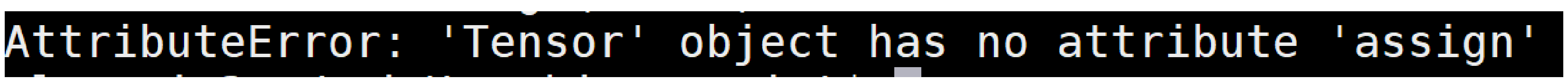
```
ph = tf.placeholder(tf.float32, [])
assign_ph = tf.assign(ph, 2)

with tf.Session() as sess:
    print sess.run(assign_ph)
```

Try and see the results

Components in TensorFlow

Results



```
AttributeError: 'Tensor' object has no attribute 'assign'
```

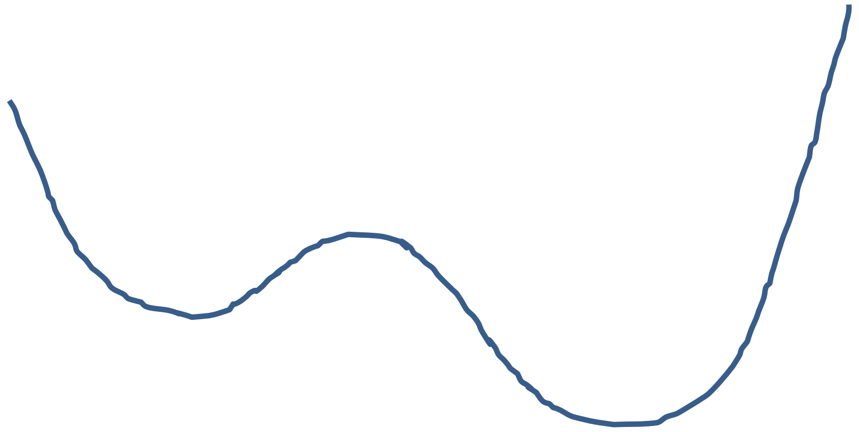
- **Mutability:** whether you can read but cannot write
  - python tuple is not a mutable type data structure
  - python list is mutable

Components in TensorFlow

training

$$L(x) = 3x^4 - 4x^3 - 12x^2 + 3$$

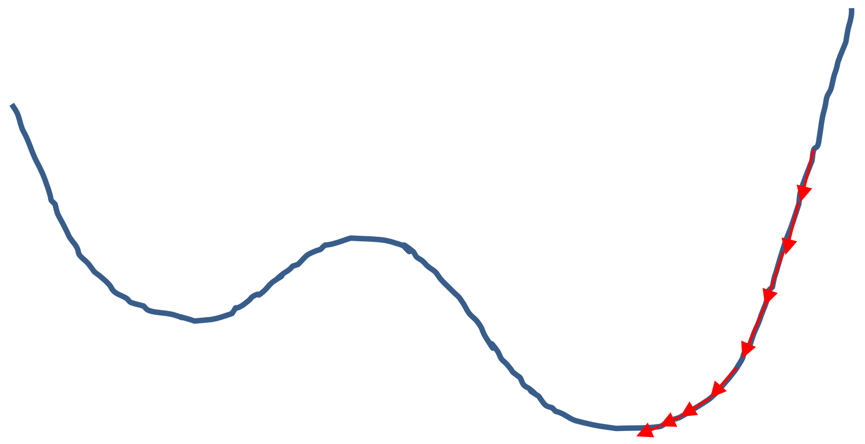
- How can we find minimal point?



Components in TensorFlow

Gradient based optimization - a single dimension

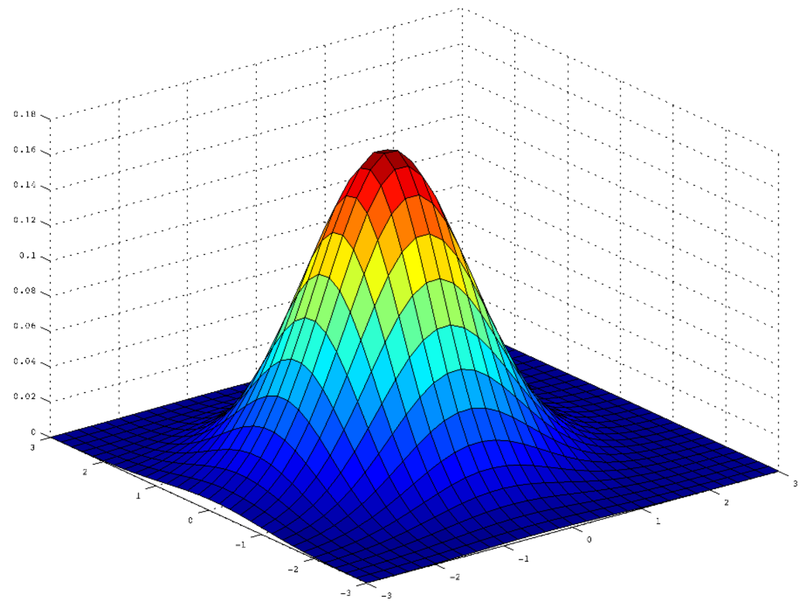
$$x_{i+1} = x_i - \alpha \mathrm{L}'(x_i)$$

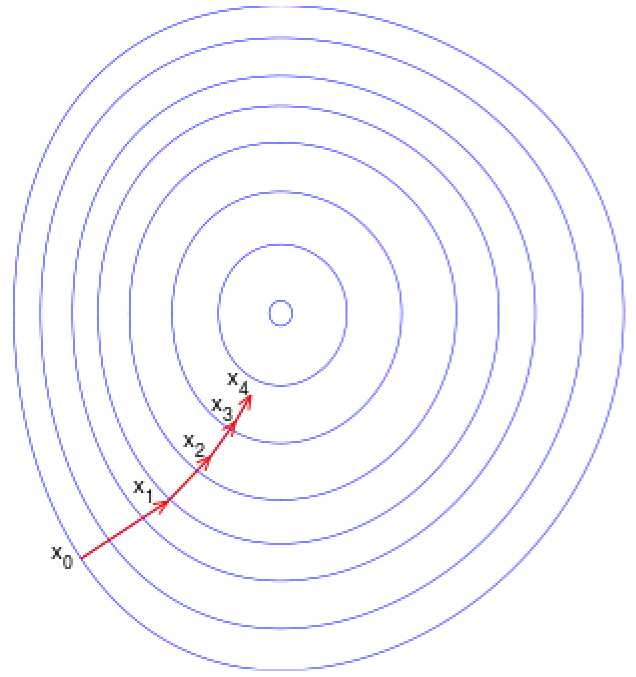


Components in TensorFlow

Gradient based optimization - multiple dimension

$$x_{i+1} = x_i + \alpha \nabla \mathrm{L}(x_i)$$





Components in TensorFlow

Gradient based optimization - multiple dimension

$$x = [x_1, x_2, \dots, x_d]$$

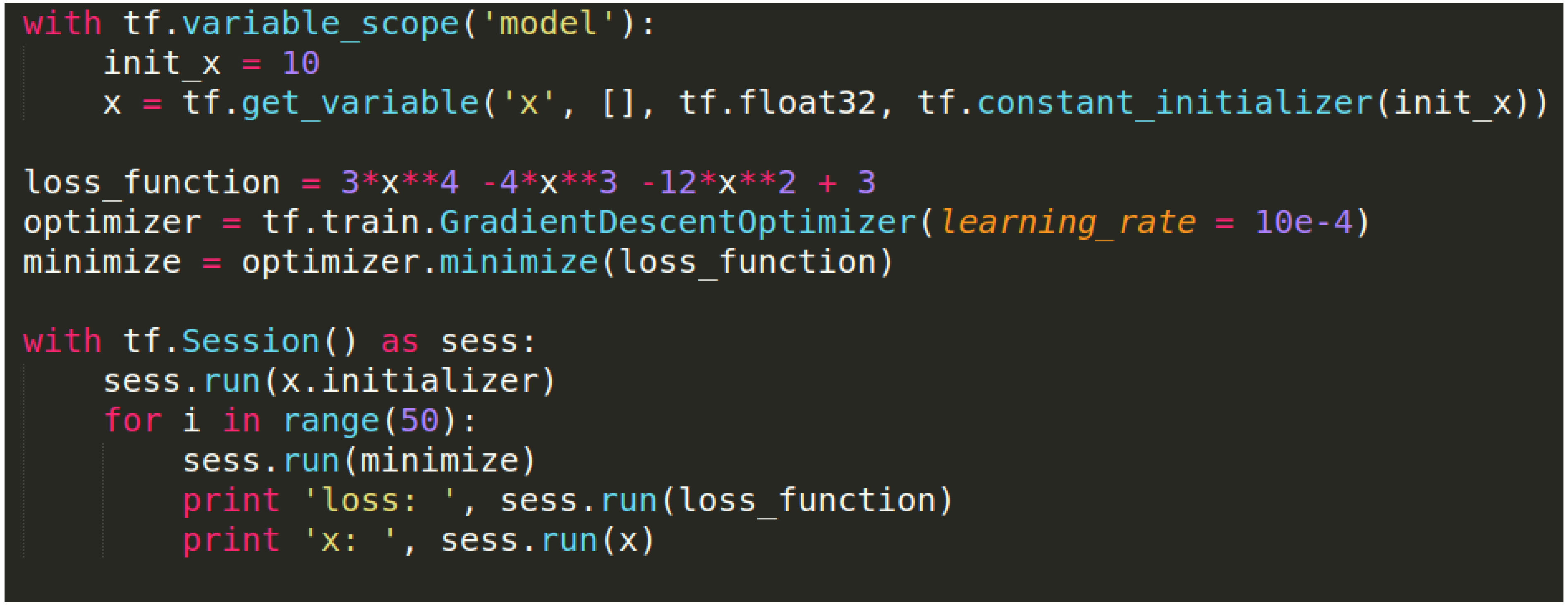
$$\nabla \mathrm{L}(x) = [\frac{\partial L}{\partial x_1}, \frac{\partial L}{\partial x_2}, \dots, \frac{\partial L}{\partial x_d}]$$

gradient's direction is the direction
in which the function increases with the greatest rate

why?

Components in TensorFlow

Example code13 - training



```
with tf.variable_scope('model'):
    init_x = 10
    x = tf.get_variable('x', [], tf.float32, tf.constant_initializer(init_x))

loss_function = 3*x**4 -4*x**3 -12*x**2 + 3
optimizer = tf.train.GradientDescentOptimizer(learning_rate = 10e-4)
minimize = optimizer.minimize(loss_function)

with tf.Session() as sess:
    sess.run(x.initializer)
    for i in range(50):
        sess.run(minimize)
        print 'loss: ', sess.run(loss_function)
        print 'x: ', sess.run(x)
```

Components in TensorFlow

Example code13 - training

Try by yourself
1. set init_x = 0 see the result
2. set init_x = 3 see the result
3. set init_x = -2 see the result
4. set init_x = -10 see the result

and think about why these results happen

Components in TensorFlow

Most Important!!

All tensorflow programs consist of

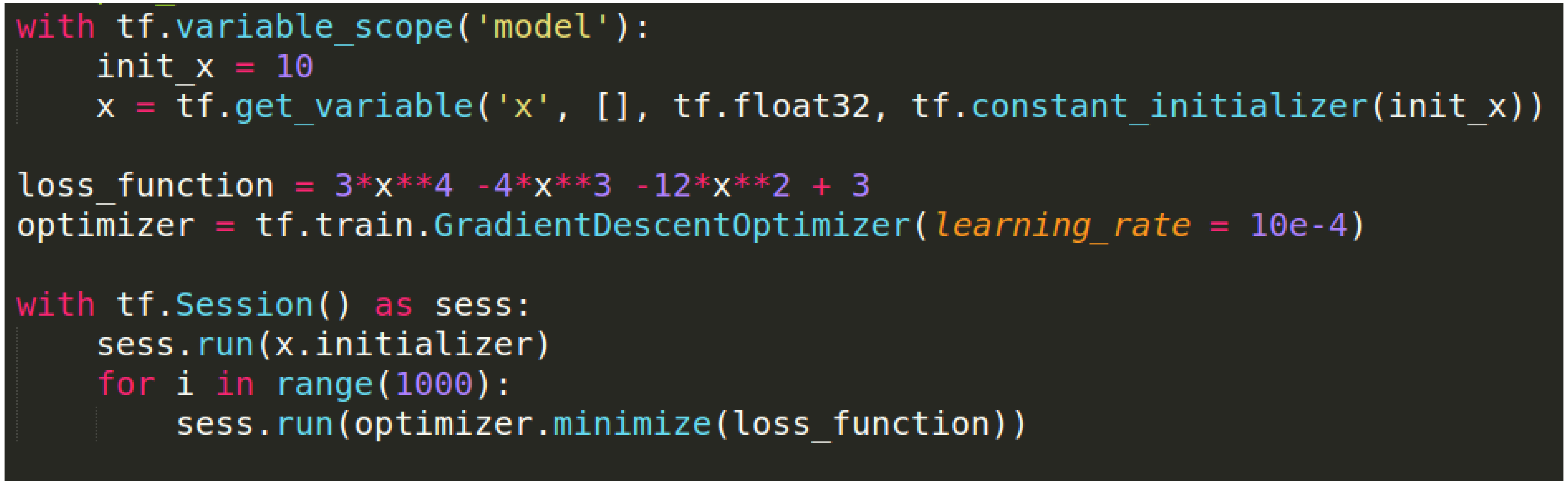
I. Building the computational graph

II. Running the computational graph

If you don’t obey this rule, then you will meet a terrible situation

Components in TensorFlow

Just see what happens if you don't comply with the rule



```python
with tf.variable_scope('model'):
    init_x = 10
    x = tf.get_variable('x', [], tf.float32, tf.constant_initializer(init_x))

loss_function = 3*x**4 -4*x**3 -12*x**2 + 3
optimizer = tf.train.GradientDescentOptimizer(learning_rate = 10e-4)

with tf.Session() as sess:
    sess.run(x.initializer)
    for i in range(1000):
        sess.run(optimizer.minimize(loss_function))
```

This program do not separate build and run

Components in TensorFlow

Next steps in TensorFlow

I. TensorFlow obejcts for Deep Learning

II. How to sharing variable in a different model

III. How to save and restore variables during training

IV. TensorBoard - visualize your model and learning

Components in TensorFlow

$$maximize\ L(x) = e^{-(x-\mu)^T(x-\mu)}$$

$$where\ \mu = (1,2,3,4)$$

Hint)

1. set μ to be a placeholder
2. use random initializer for x
3. shape is [4,1] column vector
4. search tf.matmul, tf.exp, tf.transpose in google