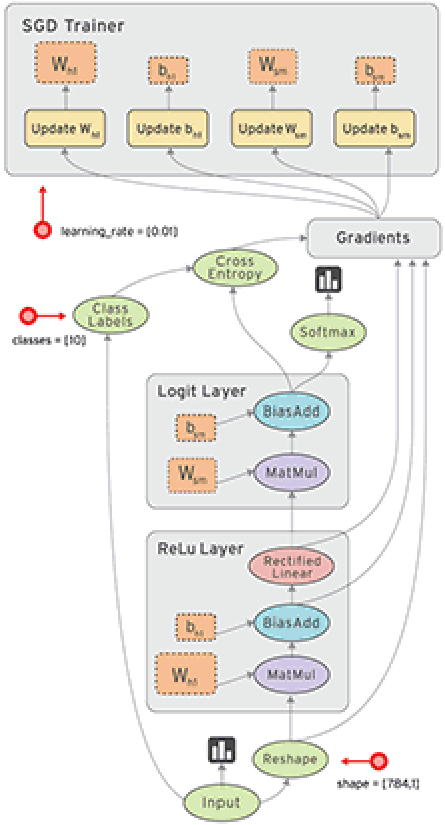
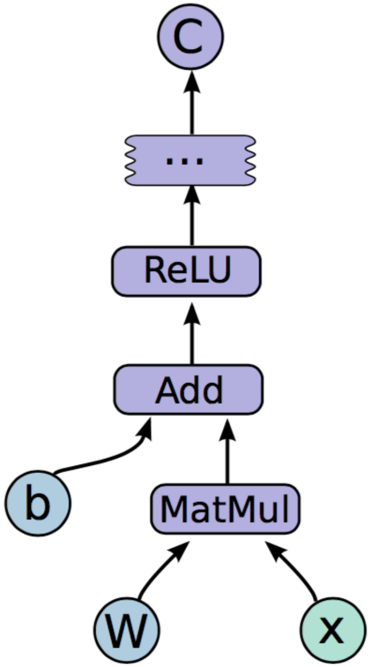
Deep Learning

Practice7

Tensorflow Basic

- Dataflow Graphs

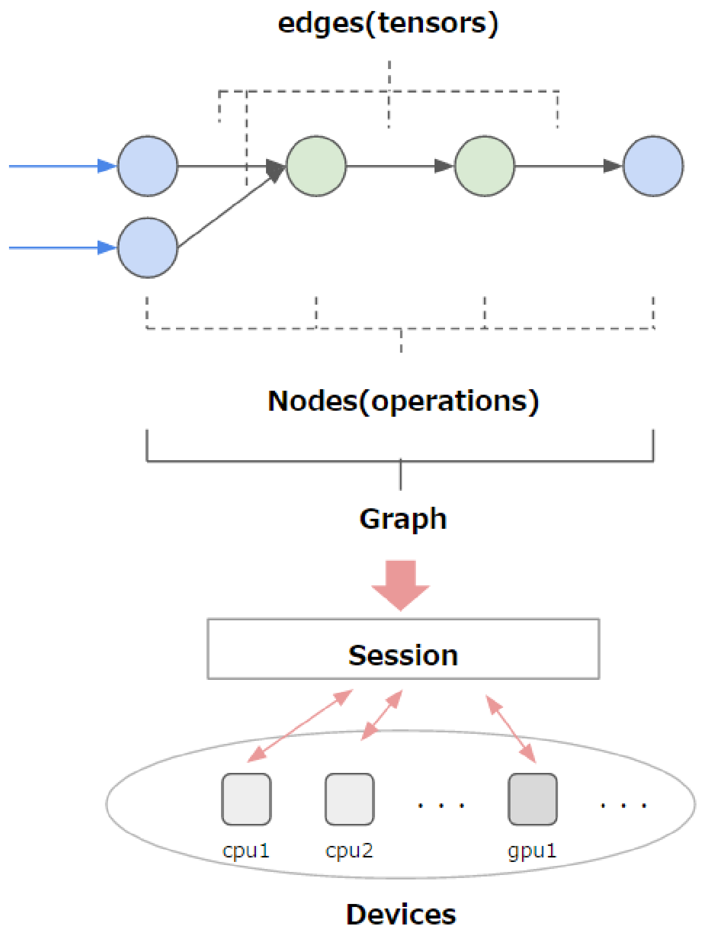


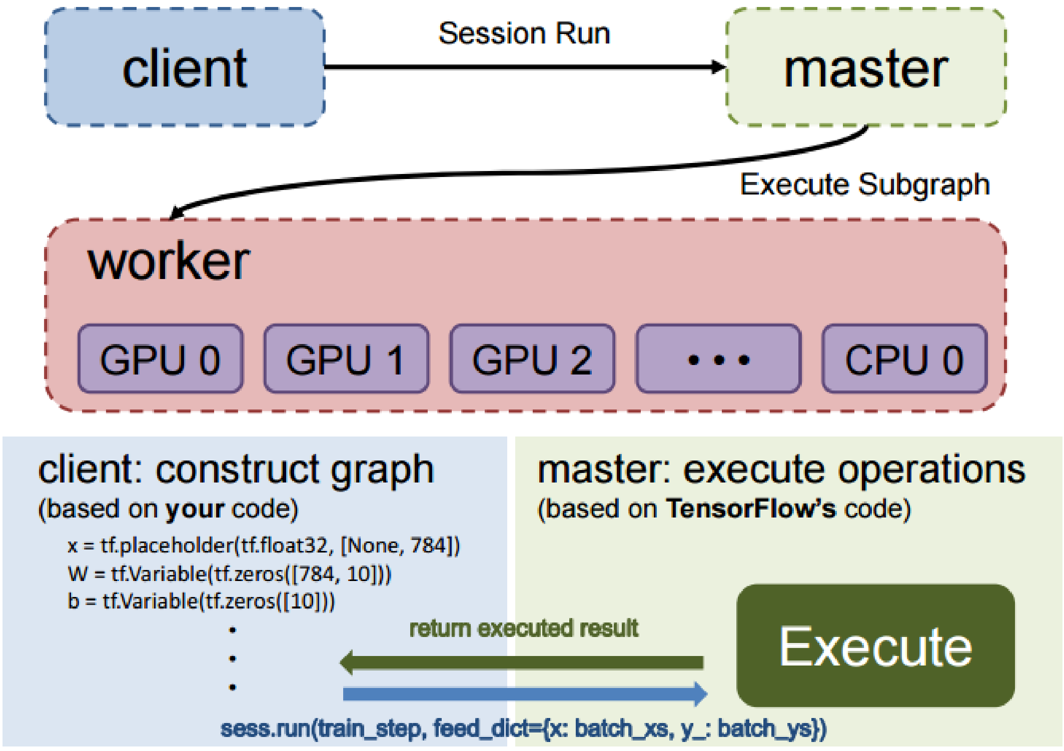


```
import tensorflow as tf

b = tf.Variable(tf.zeros([100]))
W = tf.Variable(tf.random_uniform([784,100],-1,1))
x = tf.placeholder(name="x")
relu = tf.nn.relu(tf.matmul(W, x) + b)
C = [...]


s = tf.Session()
for step in xrange(0, 10):
  input = ...construct 100-D input array ...
  result = s.run(C, feed_dict={x: input})
  print step, result
```

Tensorflow Basic

- Tensorflow





client: construct graph
(based on **your** code)

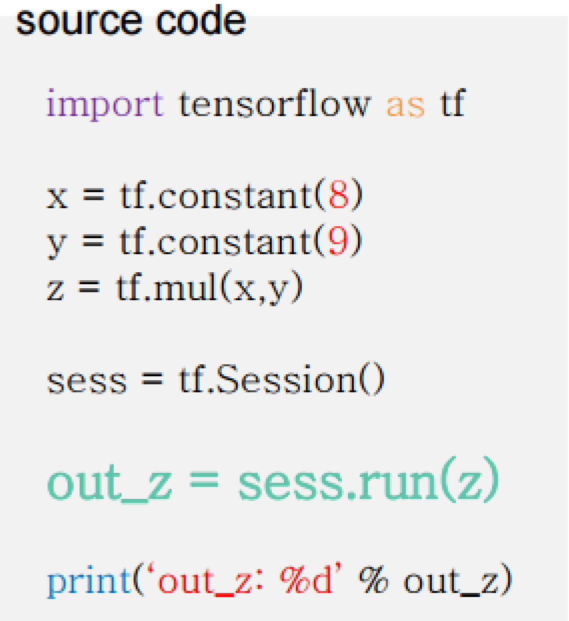
```
x = tf.placeholder(tf.float32, [None, 784])
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))
        •
        •
        •
```

master: execute operations
(based on **TensorFlow's** code)

```
sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
```

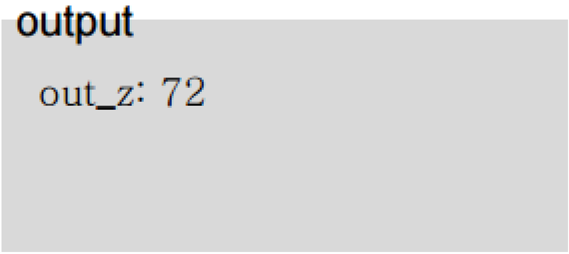
Tensorflow Basic

- Tensorflow



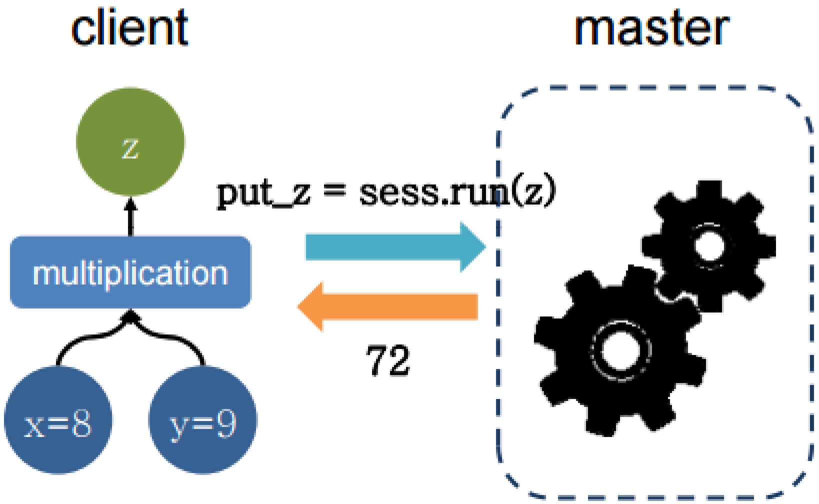
source code

```
import tensorflow as tf

x = tf.constant(8)
y = tf.constant(9)
z = tf.mul(x,y)

sess = tf.Session()

out_z = sess.run(z)

print('out_z: %d' % out_z)
```



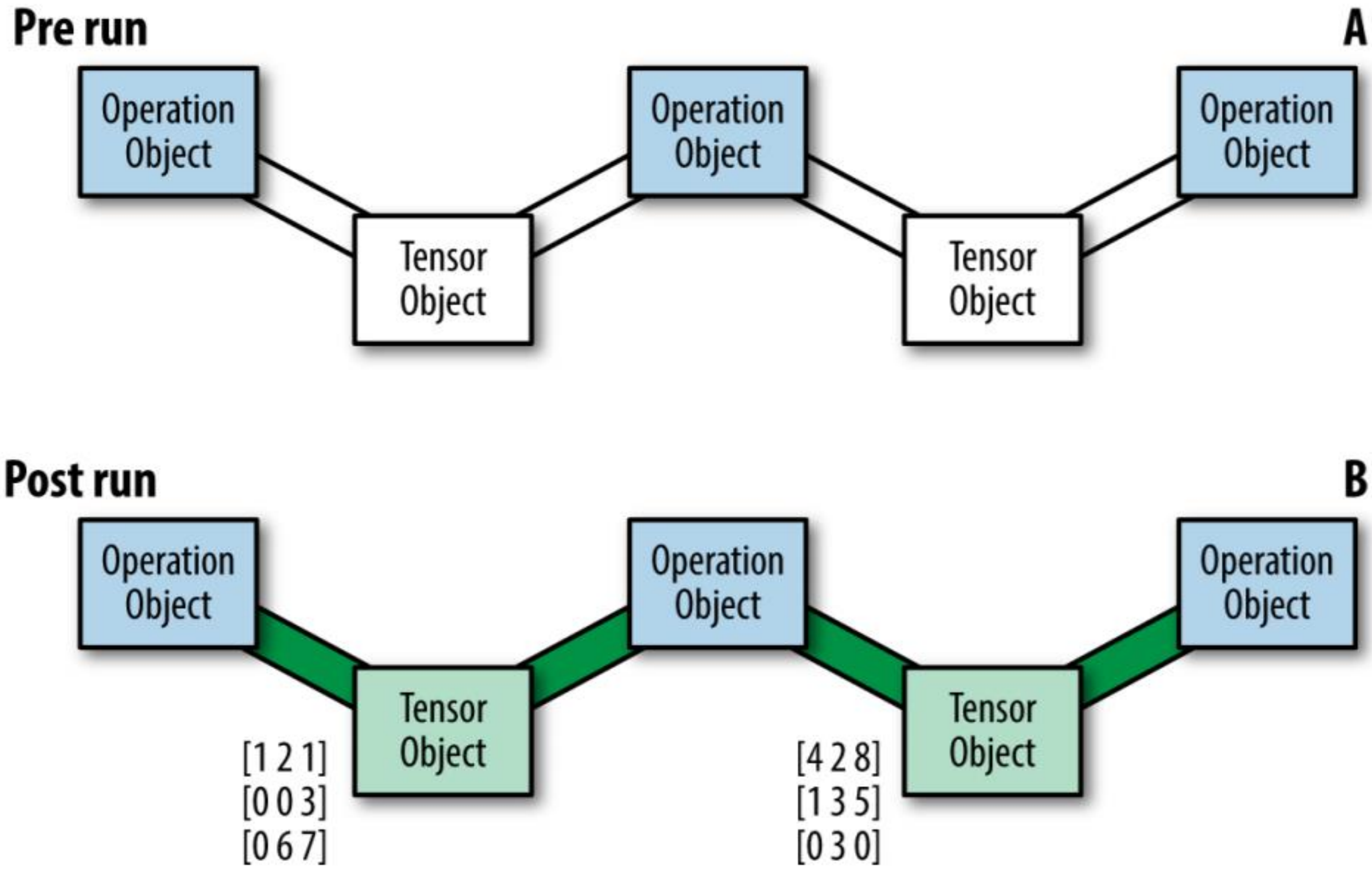
output

```
out_z: 72
```



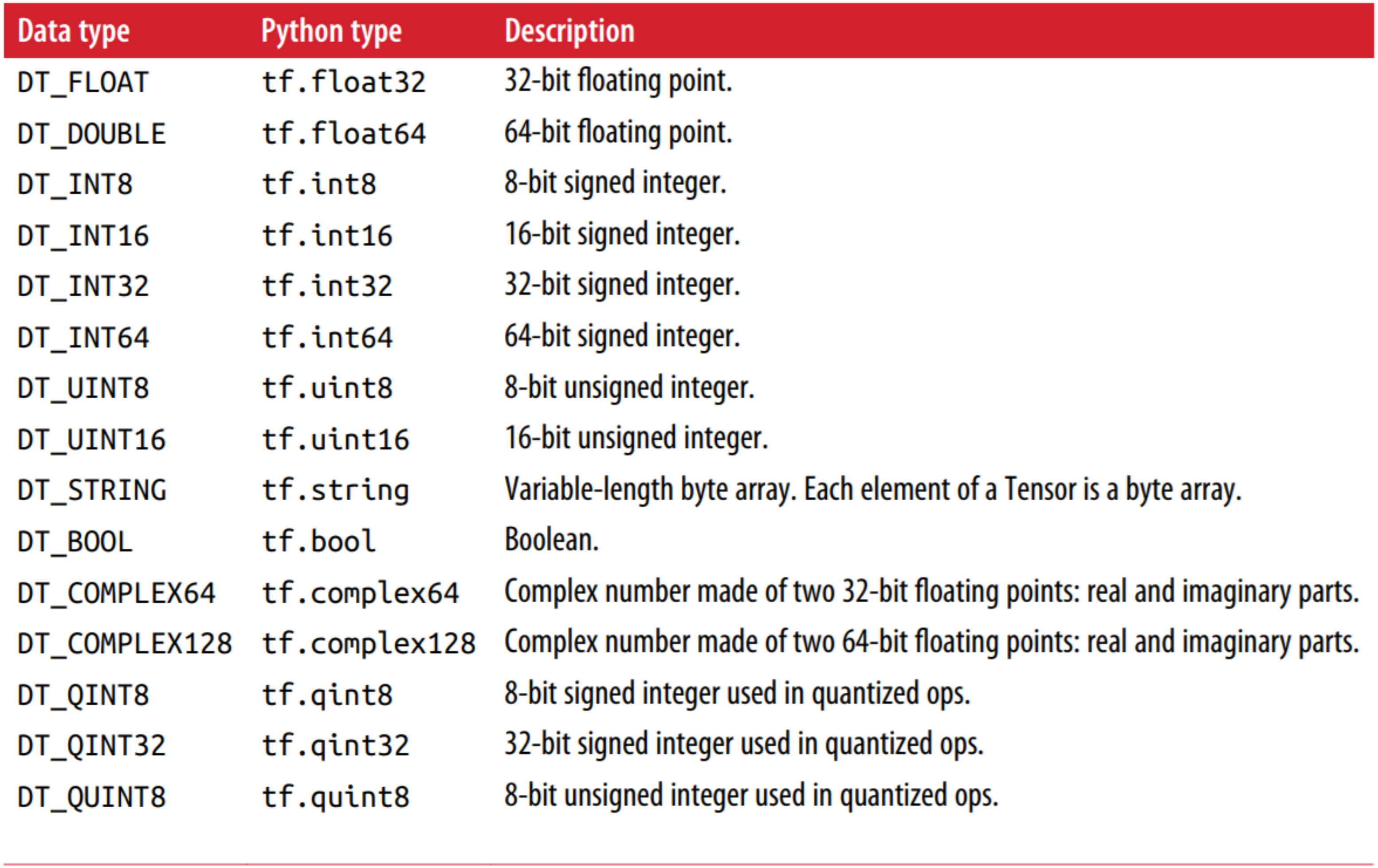
Tensorflow Basic

- Tensorflow



Tensorflow Basic

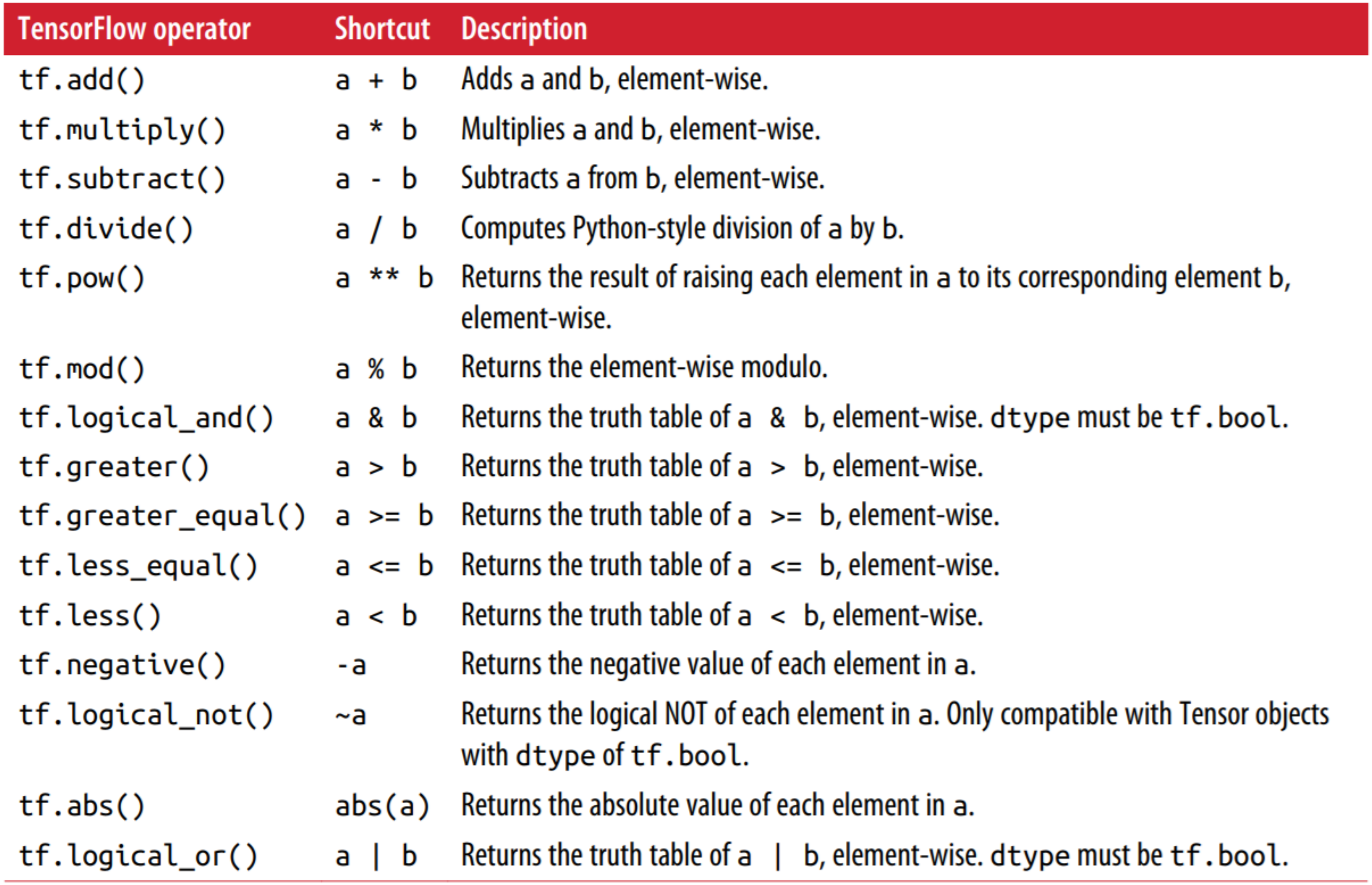
- Data Type



Data type	Python type	Description
DT_FLOAT	tf.float32	32-bit floating point.
DT_DOUBLE	tf.float64	64-bit floating point.
DT_INT8	tf.int8	8-bit signed integer.
DT_INT16	tf.int16	16-bit signed integer.
DT_INT32	tf.int32	32-bit signed integer.
DT_INT64	tf.int64	64-bit signed integer.
DT_UINT8	tf.uint8	8-bit unsigned integer.
DT_UINT16	tf.uint16	16-bit unsigned integer.
DT_STRING	tf.string	Variable-length byte array. Each element of a Tensor is a byte array.
DT_BOOL	tf.bool	Boolean.
DT_COMPLEX64	tf.complex64	Complex number made of two 32-bit floating points: real and imaginary parts.
DT_COMPLEX128	tf.complex128	Complex number made of two 64-bit floating points: real and imaginary parts.
DT_QINT8	tf.qint8	8-bit signed integer used in quantized ops.
DT_QINT32	tf.qint32	32-bit signed integer used in quantized ops.
DT_QUINT8	tf.quint8	8-bit unsigned integer used in quantized ops.

Tensorflow Basic

- Tensorflow operator



TensorFlow operator	Shortcut	Description
`tf.add()`	`a + b`	Adds a and b, element-wise.
`tf.multiply()`	`a * b`	Multiplies a and b, element-wise.
`tf.subtract()`	`a - b`	Subtracts a from b, element-wise.
`tf.divide()`	`a / b`	Computes Python-style division of a by b.
`tf.pow()`	`a ** b`	Returns the result of raising each element in a to its corresponding element b, element-wise.
`tf.mod()`	`a % b`	Returns the element-wise modulo.
`tf.logical_and()`	`a & b`	Returns the truth table of `a & b`, element-wise. `dtype` must be `tf.bool`.
`tf.greater()`	`a > b`	Returns the truth table of `a > b`, element-wise.
`tf.greater_equal()`	`a >= b`	Returns the truth table of `a >= b`, element-wise.
`tf.less_equal()`	`a <= b`	Returns the truth table of `a <= b`, element-wise.
`tf.less()`	`a < b`	Returns the truth table of `a < b`, element-wise.
`tf.negative()`	`-a`	Returns the negative value of each element in a.
`tf.logical_not()`	`~a`	Returns the logical NOT of each element in a. Only compatible with Tensor objects with `dtype` of `tf.bool`.
`tf.abs()`	`abs(a)`	Returns the absolute value of each element in a.
`tf.logical_or()`	`a | b`	Returns the truth table of `a | b`, element-wise. `dtype` must be `tf.bool`.

Tensorflow Basic

- Tensorflow operator

함수	설명
tf.add	덧셈
tf.subtract	뺄셈
tf.multiply	곱셈
tf.div	나눗셈의 몫(Python 2 스타일)
tf.truediv	나눗셈의 몫(Python 3 스타일)
tf.mod	나눗셈의 나머지
tf.abs	절대값을 리턴합니다.
tf.negative	음수를 리턴합니다.
tf.sign	부호를 리턴합니다.(역주: 음수는 -1, 양수는 1, 0 일땐 0을 리턴합니다)

함수	설명
tf.reciprocal	역수를 리턴합니다.(역주: 3의 역수는 1/3 입니다)
tf.square	제곱을 계산합니다.
tf.round	반올림 값을 리턴합니다.
tf.sqrt	제곱근을 계산합니다.
tf.pow	거듭제곱 값을 계산합니다.
tf.exp	지수 값을 계산합니다.
tf.log	로그 값을 계산합니다.
tf.maximum	최대값을 리턴합니다.
tf.minimum	최소값을 리턴합니다.
tf.cos	코사인 함수 값을 계산합니다.
tf.sin	사인 함수 값을 계산합니다.

Tensorflow Basic

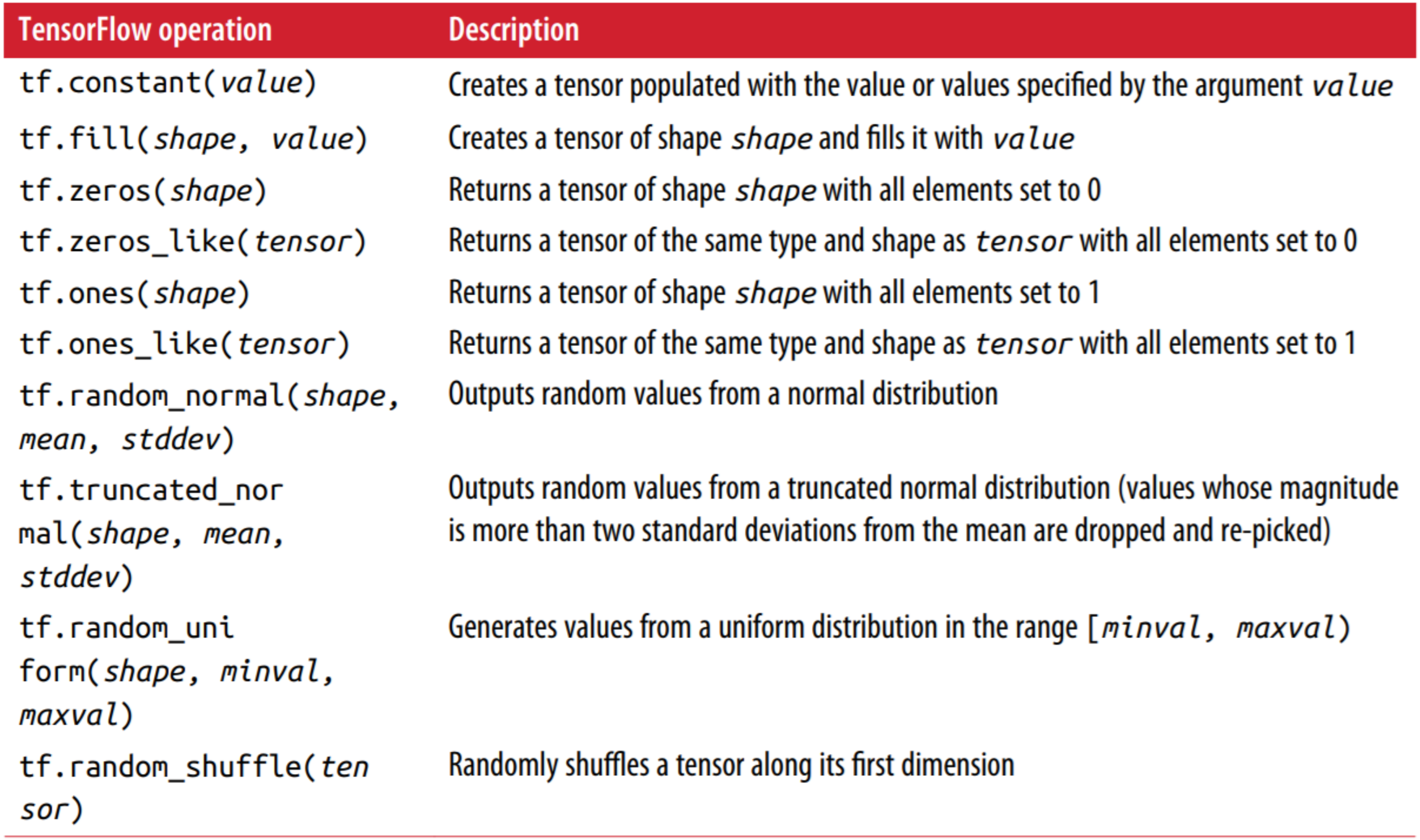
- Tensorflow operator & kernel

함수	설명
tf.diag	대각행렬을 리턴합니다.
tf.transpose	전치행렬을 리턴합니다.
tf.matmul	두 텐서를 행렬곱셈하여 결과 텐서를 리턴합니다.
tf.matrix_determinant	정방행렬의 행렬식 값을 리턴합니다.
tf.matrix_inverse	정방행렬의 역행렬을 리턴합니다.

연산 카테고리	연산 예
Maths	Add. Sub, Mul, Div, Exp, Log , Greater, Less ,Equal
Array	Concat, Slice, Split, Constant, Rank, Shape, Shuffle
Matrix	MatMul, MatrixInverse, MatrixDeterminant
Neuronal Network	SoftMax, Sigmoid, ReLU, Convolution2D, MaxPool
Checkpointing	Save, Restore
Queues and syncronizations	Enqueue, Dequeue, MutexAcquire, MutexRelease
Flow control	Merge, Switch, Enter, Leave, NextIteration

Tensorflow Basic

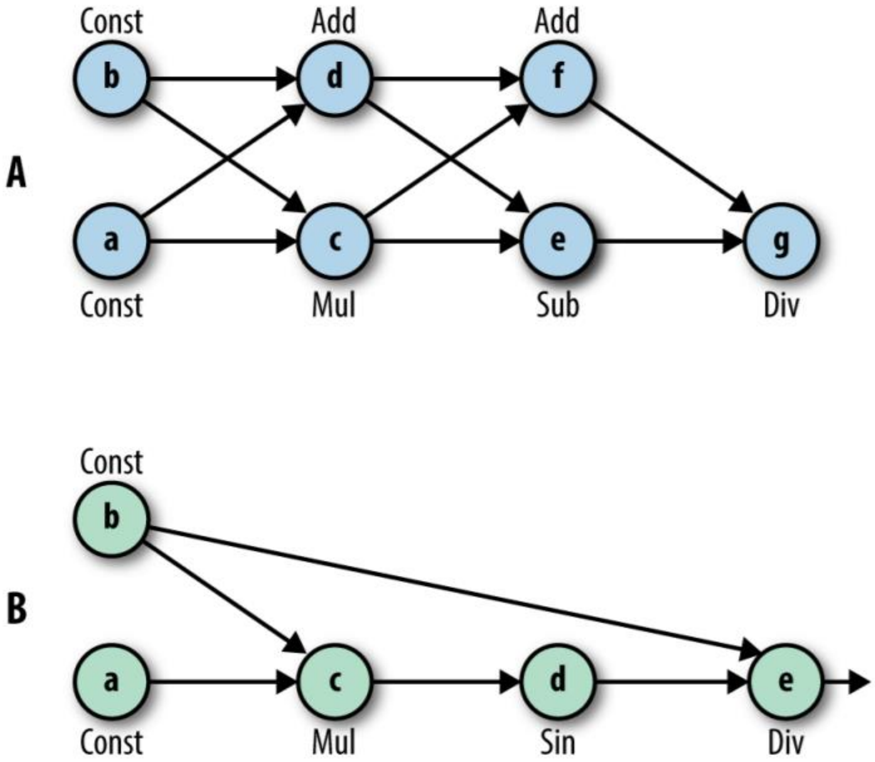
- Tensorflow operation



TensorFlow operation	Description
`tf.constant(value)`	Creates a tensor populated with the value or values specified by the argument *`value`*
`tf.fill(shape, value)`	Creates a tensor of shape *`shape`* and fills it with *`value`*
`tf.zeros(shape)`	Returns a tensor of shape *`shape`* with all elements set to 0
`tf.zeros_like(tensor)`	Returns a tensor of the same type and shape as *`tensor`* with all elements set to 0
`tf.ones(shape)`	Returns a tensor of shape *`shape`* with all elements set to 1
`tf.ones_like(tensor)`	Returns a tensor of the same type and shape as *`tensor`* with all elements set to 1
`tf.random_normal(shape, mean, stddev)`	Outputs random values from a normal distribution
`tf.truncated_normal(shape, mean, stddev)`	Outputs random values from a truncated normal distribution (values whose magnitude is more than two standard deviations from the mean are dropped and re-picked)
`tf.random_uniform(shape, minval, maxval)`	Generates values from a uniform distribution in the range `[minval, maxval)`
`tf.random_shuffle(tensor)`	Randomly shuffles a tensor along its first dimension

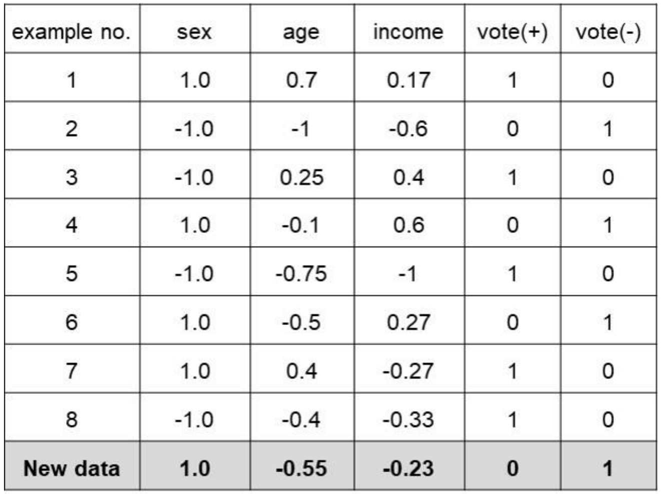
실습 1. Tensorflow Basic

- Make a Graph

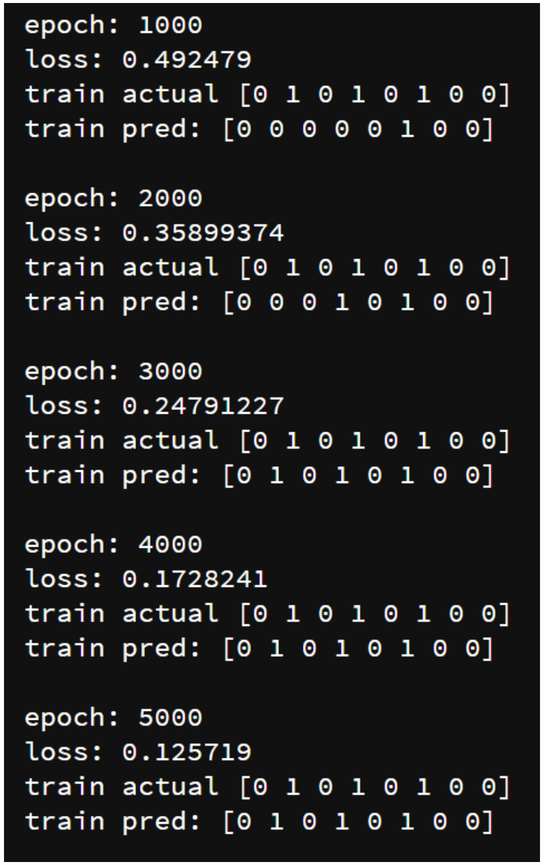


실습 2. Basic Neural Networks (tensorflow)

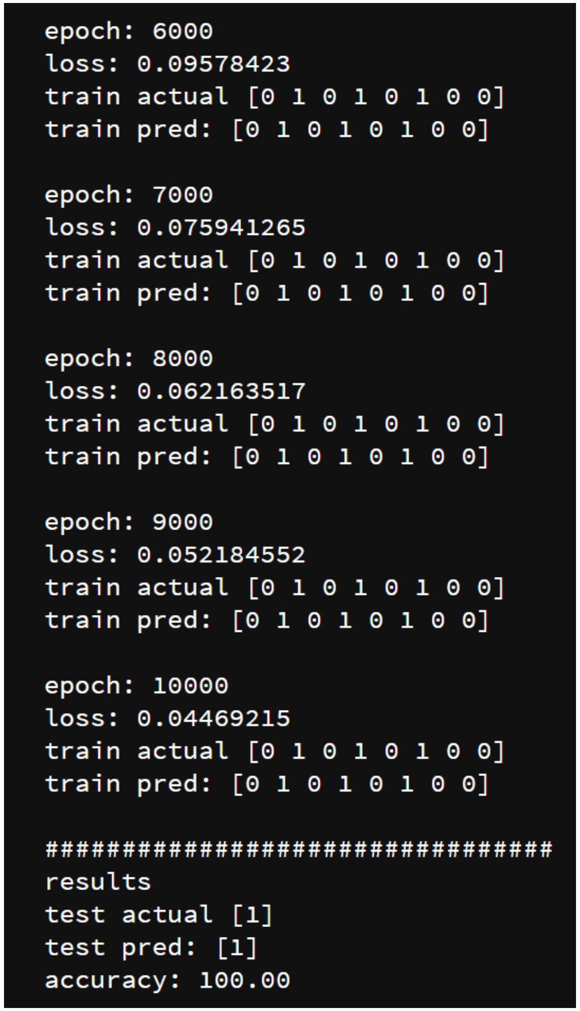
- Make a sample data
  - Dataset
  - Define model
  - Build a graph
  - Training
  - evaluation



example no.	sex	age	income	vote(+)	vote(-)
1	1.0	0.7	0.17	1	0
2	-1.0	-1	-0.6	0	1
3	-1.0	0.25	0.4	1	0
4	1.0	-0.1	0.6	0	1
5	-1.0	-0.75	-1	1	0
6	1.0	-0.5	0.27	0	1
7	1.0	0.4	-0.27	1	0
8	-1.0	-0.4	-0.33	1	0
New data	**1.0**	**-0.55**	**-0.23**	**0**	**1**



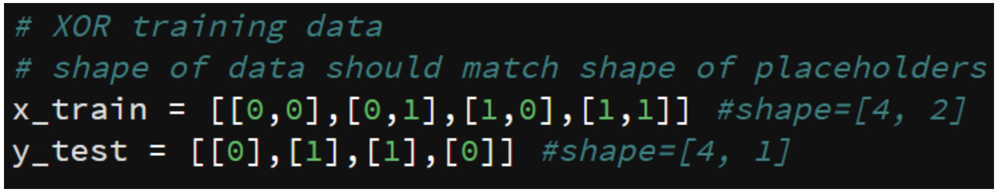
```
epoch: 1000
loss: 0.492479
train actual [0 1 0 1 0 1 0 0]
train pred: [0 0 0 0 0 1 0 0]

epoch: 2000
loss: 0.35899374
train actual [0 1 0 1 0 1 0 0]
train pred: [0 0 0 1 0 1 0 0]

epoch: 3000
loss: 0.24791227
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

epoch: 4000
loss: 0.1728241
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

epoch: 5000
loss: 0.125719
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]
```



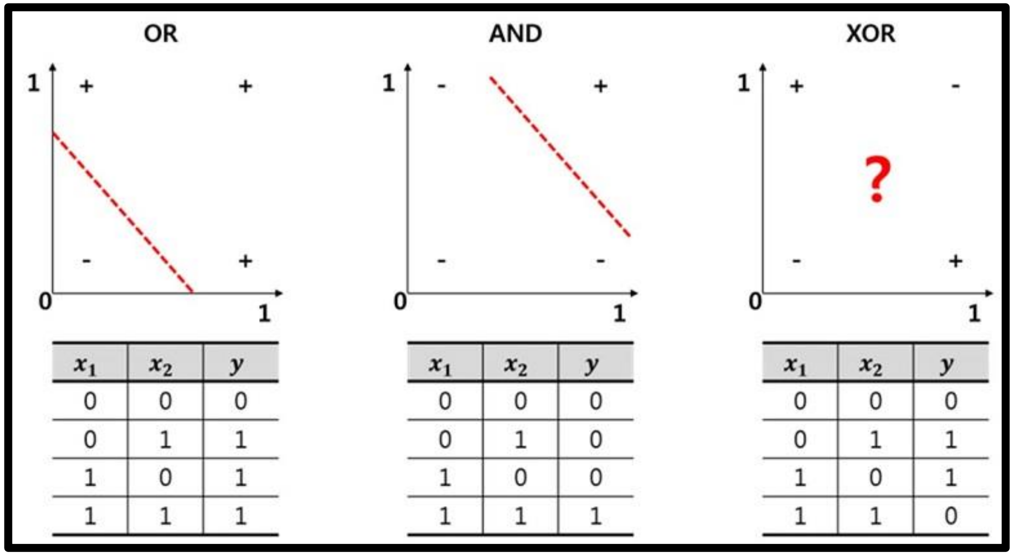
```
epoch: 6000
loss: 0.09578423
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

epoch: 7000
loss: 0.075941265
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

epoch: 8000
loss: 0.062163517
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

epoch: 9000
loss: 0.052184552
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

epoch: 10000
loss: 0.04469215
train actual [0 1 0 1 0 1 0 0]
train pred: [0 1 0 1 0 1 0 0]

#################################
results
test actual [1]
test pred: [1]
accuracy: 100.00
```

실 습 3. Solving a XOR Problem (tensorflow)

- XOR Problem
  - Multi-layer Neural Nets



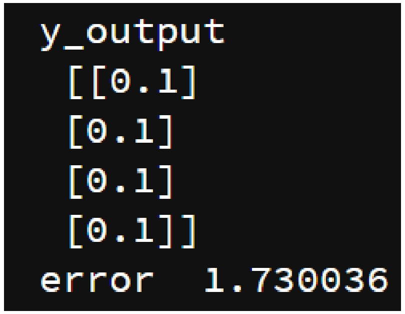
```
# XOR training data
# shape of data should match shape of placeholders
x_train = [[0,0],[0,1],[1,0],[1,1]] #shape=[4, 2]
y_test = [[0],[1],[1],[0]] #shape=[4, 1]
```



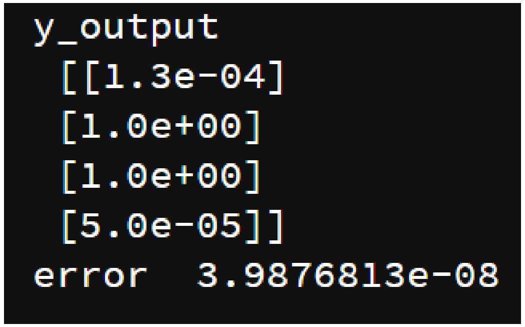
x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



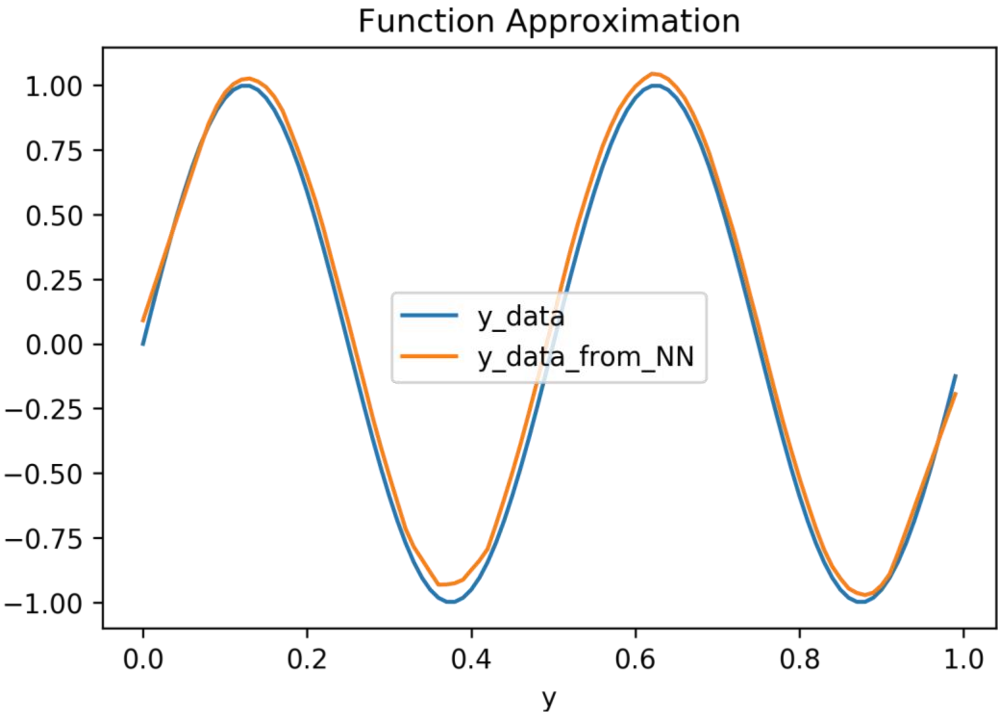
```
y_output
 [[0.1]
 [0.1]
 [0.1]
 [0.1]]
error  1.730036
```



```
y_output
 [[1.3e-04]
 [1.0e+00]
 [1.0e+00]
 [5.0e-05]]
error  3.9876813e-08
```

실습 4. Function Approximation

- Universal Approximation Theorem
  - In the mathematical theory of artificial neural networks, the universal approximation theorem states[1] that a feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions on compact subsets of Rn, under mild assumptions on the activation function.



실습 5. Custom Neural Networks

- Make a Custom NN for iris data or other dataset
- Change the number of hidden layer and neuron