## HU Extension Assignment 10 E63 Big Data Analytics

### Handed out: 11/03/2017 Due by 9:30AM EST on Saturday, 11/11/2017

You are welcome to implement TensorFlow problems in this problem set in any of supported languages.

**Problem 1.** Install newest release of TensorFlow 1.4 on the operating system of your choice. Use installation instructions on <https://www.tensorflow.org> site and instructions on <https://github.com/tensorflow/tensorflow>. If you know what you are doing install TensorFlow for GPU. Otherwise, install TensorFlow for CPU. Use attach Jupyter notebook: 0\_test\_install.ipynb to demonstrate that TensorFlow is properly installed. Please document all installation steps including the version of Python you are using. (15%)

**bin** **python -V**

Python 2.7.10

**conda create -n tensorflow python=2.7**

Fetching package metadata ...........

Solving package specifications: .

Package plan for installation in environment /Users/smukherjee5/anaconda2/envs/tensorflow:

The following NEW packages will be INSTALLED:

    libcxx:     4.0.1-h579ed51\_0

    libcxxabi:  4.0.1-hebd6815\_0

    libedit:    3.1-hb4e282d\_0

    ncurses:    6.0-hd04f020\_2

….

**conda install -c conda-forge matplotlib**

Fetching package metadata .............

Solving package specifications: .

Package plan for installation in environment /Users/smukherjee5/anaconda2:

The following packages will be UPDATED:

    matplotlib: 2.0.2-py27h2e09848\_1  --> 2.1.0-py27\_1     conda-forge

The following packages will be SUPERSEDED by a higher-priority channel:

    conda:      4.3.30-py27h407ed3a\_0 --> 4.3.29-py27\_0    conda-forge

    conda-env:  2.6.0-h36134e3\_0      --> 2.6.0-0          conda-forge

    freetype:   2.8-h143eb01\_0        --> 2.7-2            conda-forge

    libpng:     1.6.32-hce72d48\_2     --> 1.6.28-2         conda-forge

    pillow:     4.2.1-py27h2cf1d5f\_0  --> 4.2.1-py27\_0     conda-forge

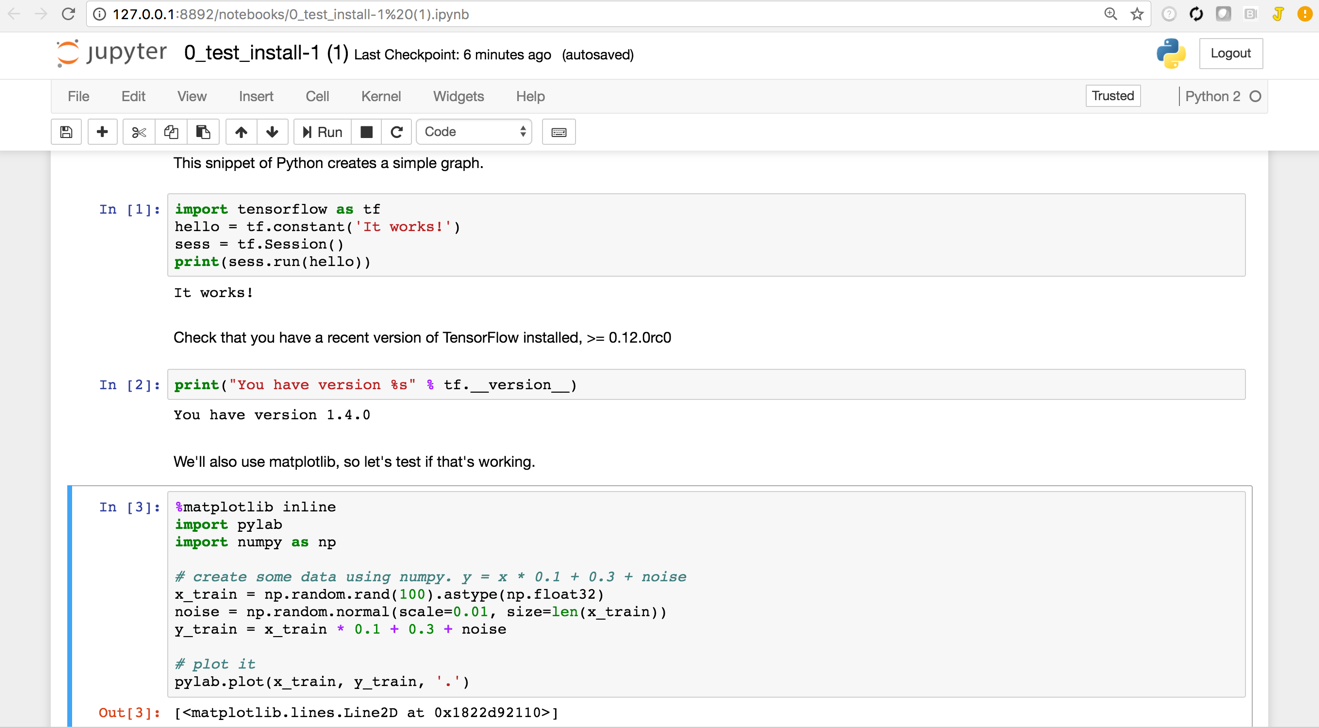
    qt:         5.6.2-h9975529\_14     --> 5.6.2-h9e3eb04\_4 conda-forge

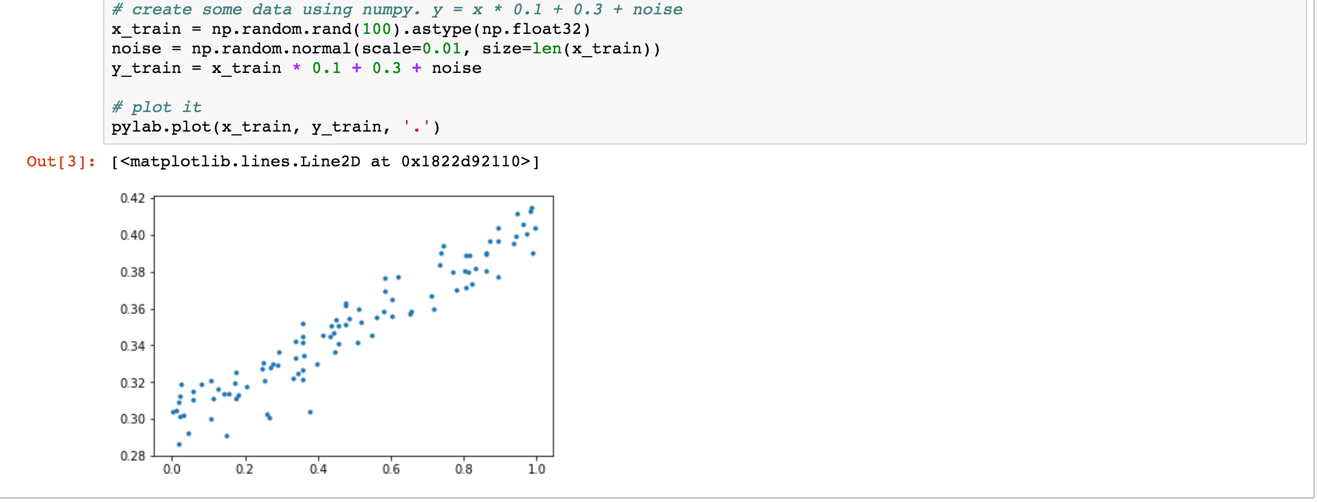
Proceed ([y]/n)? y

conda-env-2.6. 100% |##############################################################################################################################################################################################| Time: 0:00:00 667.52 kB/s

libpng-1.6.28- 100% |########

….





**Problem 2.** Construct a simple neural network (a network of logistic units) which will implement (X1 XOR X2) AND X3 function. Choose weights (-s) of all dendritic inputs and bias inputs. Demonstrate that your network works by presenting the truth table. Present your network by a simple graph. You can produce the graph in any way convenient including pen and paper**.** (25%)

XOR

X 1 X 2 Output

0 0 0

1 0 1

0 1 1

1 1 0

AND

Output X3 Output2

0 0 0

1 1 1

1 0 0

0 1 0

(X1 XOR X2) AND X3

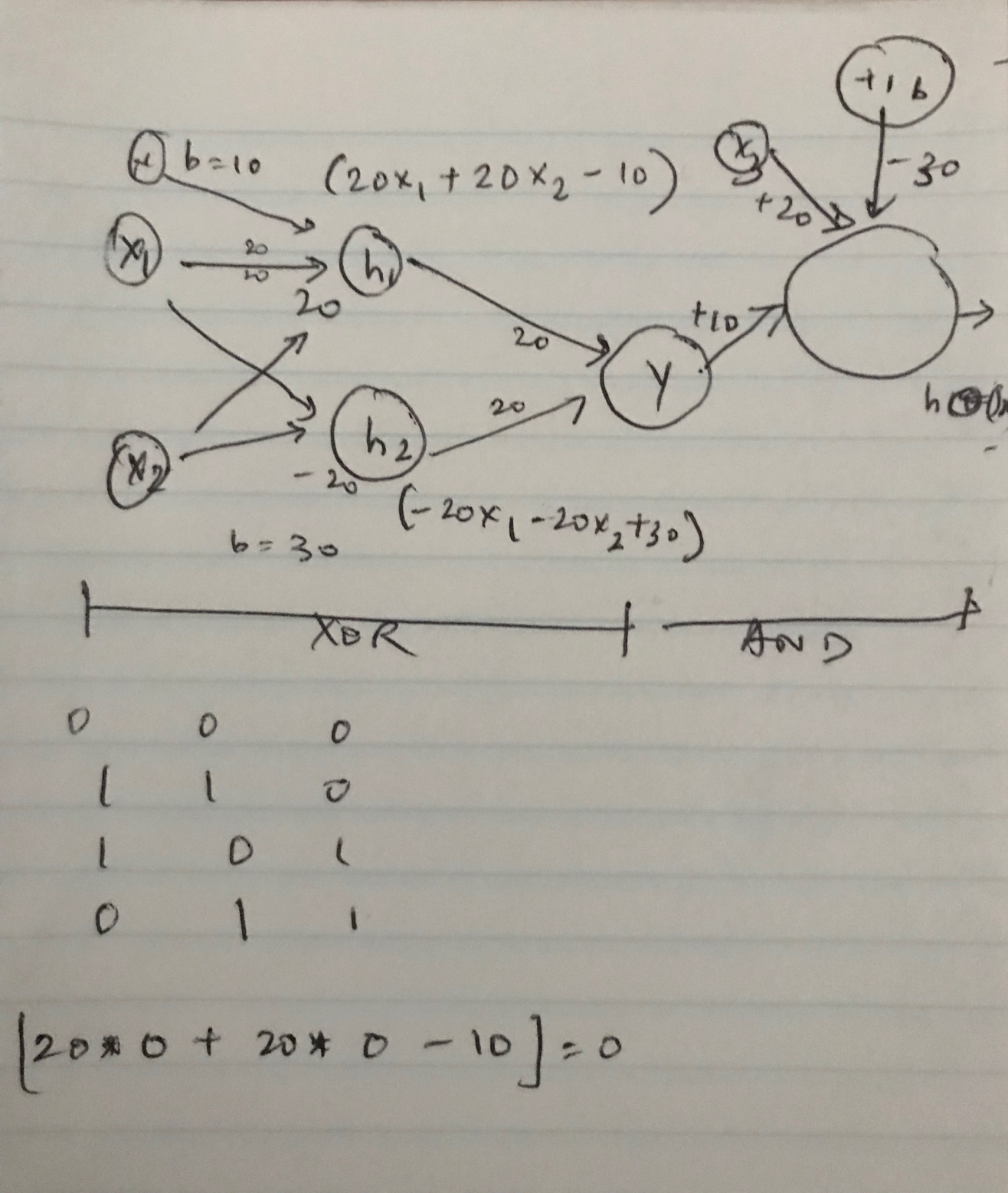
X 1 X 2 X3 Output2

0 0 0 0

1 0 1 1

0 1 0 0

1 1 1 0



**Problem 3.** Determine the value of number e = 2.7183… to 6 decimal places using Taylor expansion. Export the TensorBoard graph of your process. Perform similar calculation using expression for e as . Again export the TensorBoard graph of you process. Provide working code for both approaches.

(25%)

**import tensorflow as tf**

**def fact(x):**

**# x will be a numpy array with the contents of the placeholder below**

**return numpy.math.factorial(x)**

**with tf.Graph().as\_default() as g:**

**counter = tf.Variable(1.000000,'counter')**

**next\_value = tf.Variable(1.000000,'next\_value')**

**f0=tf.exp(tf.lgamma(counter-1))**

**f2 = tf.divide(1,f0)**

**f3 = tf.add(next\_value,f2)**

**inc = tf.assign\_add(counter, 1, name='increment')**

**init = tf.global\_variables\_initializer()**

**with tf.Session() as sess:**

**sess.run(init)**

**for step in range(10):**

**sess.run(inc)**

**print(1+sess.run(f3))**

**#if x > 2.718281:**

**# print x**

**# break**

**file\_writer = tf.summary.FileWriter("exp01", sess.graph)**

**file\_writer.add\_graph(sess.graph)**

**file\_writer.close()**

**sess.close()**

3.0

2.5

2.16666662693

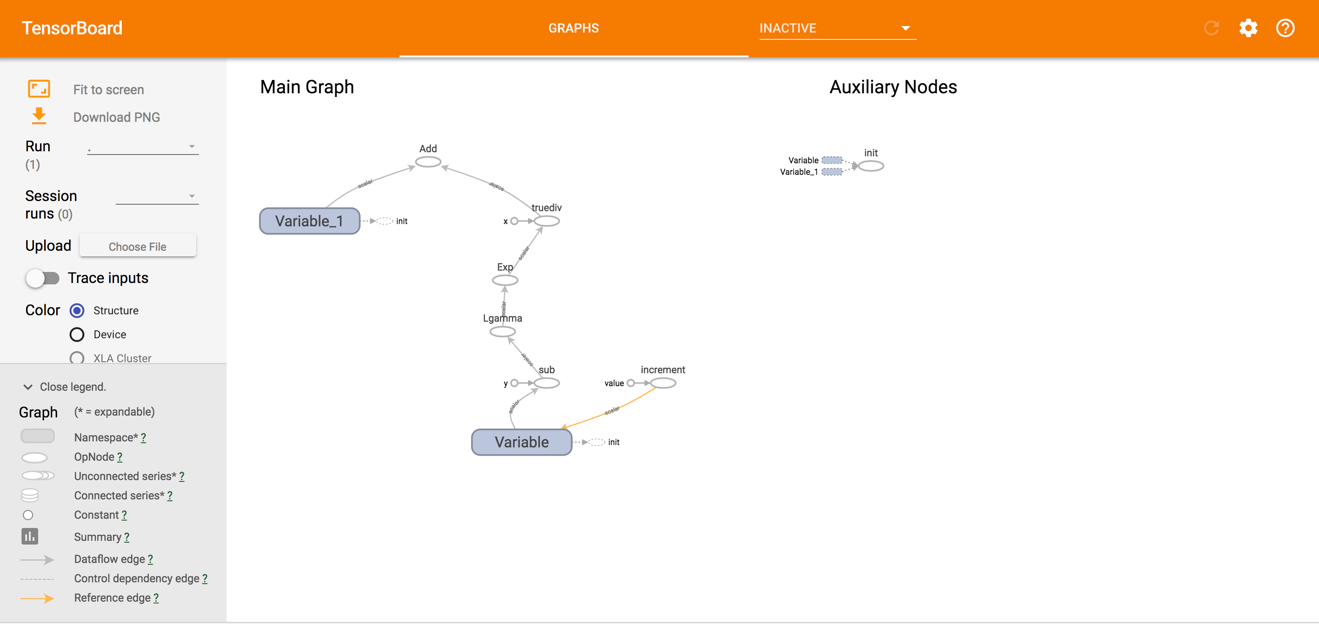
2.04166662693

2.00833332539

2.00138890743

2.00019836426

2.00002479553

****

**Problem 4.** When I tried running code on page 63 om my notes for lecture 10, the resulting TensorBoard graph was not entirely identical with the graph on page 64. Please fix the code on page 63 in order to produce the graph identical to the graph on page 64.

(15%)

**tf.reset\_default\_graph()**

**graph = tf.Graph()**

**import tensorflow as tf**

**with graph.as\_default():**

**with tf.name\_scope("Scope\_A"):**

**a = tf.add(1, 2, name="A\_add")**

**b = tf.multiply(a, 3, name="A\_mul")**

**with tf.name\_scope("Scope\_B"):**

**c = tf.add(4, 5, name="B\_add")**

**d = tf.multiply(c, 6, name="B\_mul")**

**#e = tf.add(b, d, name="output")**

**#output = tf.scalar\_mul(b,d)**

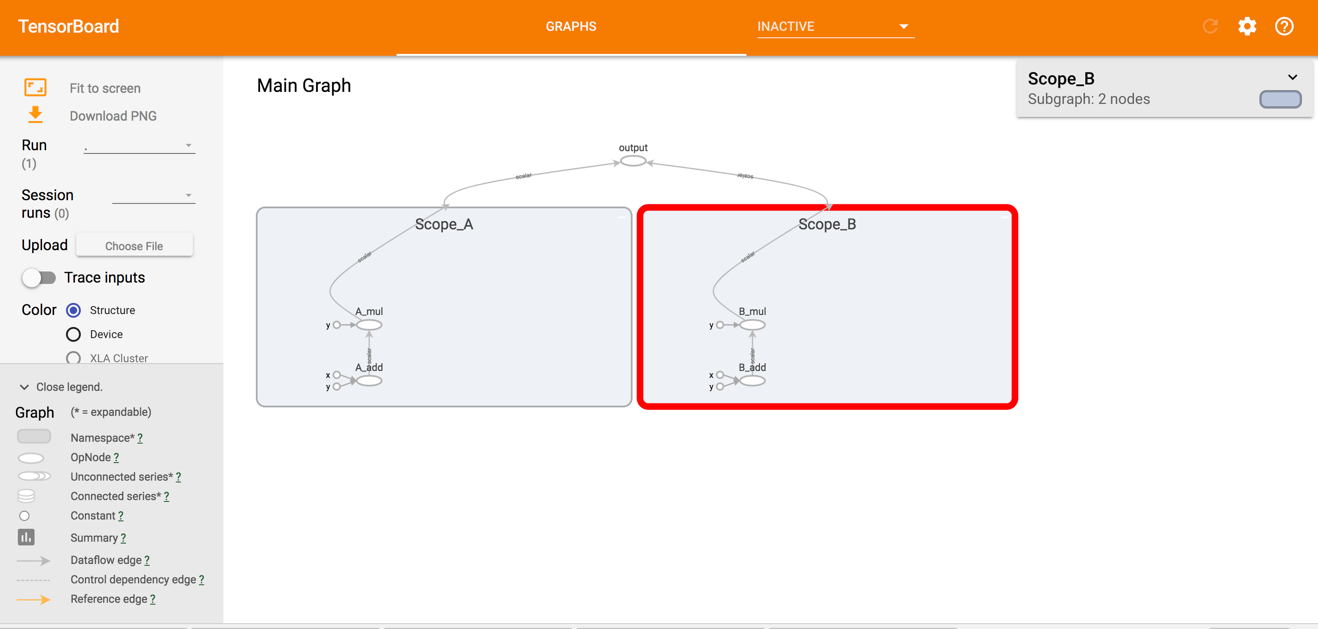
**result = tf.multiply(b,d,name="output")**

**filewriter = tf.summary.FileWriter("name\_scope\_12", graph=graph)**

**writer.close()**

In the terminal

**tensorboard --logdir name\_scope\_12**

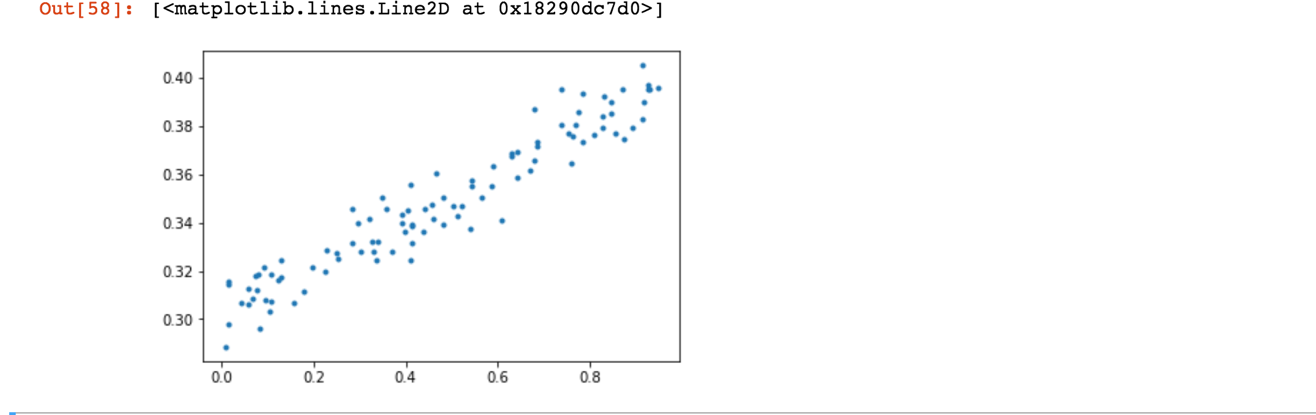


**Problem 5.** Please examine attached Jupyter notebook 2\_linear\_regression.ipynb. As you are running its cells, the notebook will complain about non-existent API calls. This notebook was written in an earlier version of TensorFlow API and some calls changed their names. Fix all code by replacing older calls with calls in TF 1.5. Uncomment all optional (print) lines. Provide a copy of this notebook with all intermediate results and the image of TensorFlow graph as captured by the TensorBoard.

**(20%)**

Output:

pylab.plot(x\_train, y\_train, '.')



Output:

W = tf.Variable(tf.random\_normal([1]), name='weights')

b = tf.Variable(tf.random\_normal([1]), name='bias')

# Uncomment the following lines to see what W and b are.

print(W)

print(b)

# Create a placeholder we'll use later to feed x's into the graph for training and eval.

# shape=[None] means we can put in any number of examples.

# This is used for minibatch training, and to evaluate a lot of examples at once.

x = tf.placeholder(shape=[None], dtype=tf.float32, name='x')

# Uncomment this line to see what x is

print(x)

<tf.Variable 'weights:0' shape=(1,) dtype=float32\_ref>

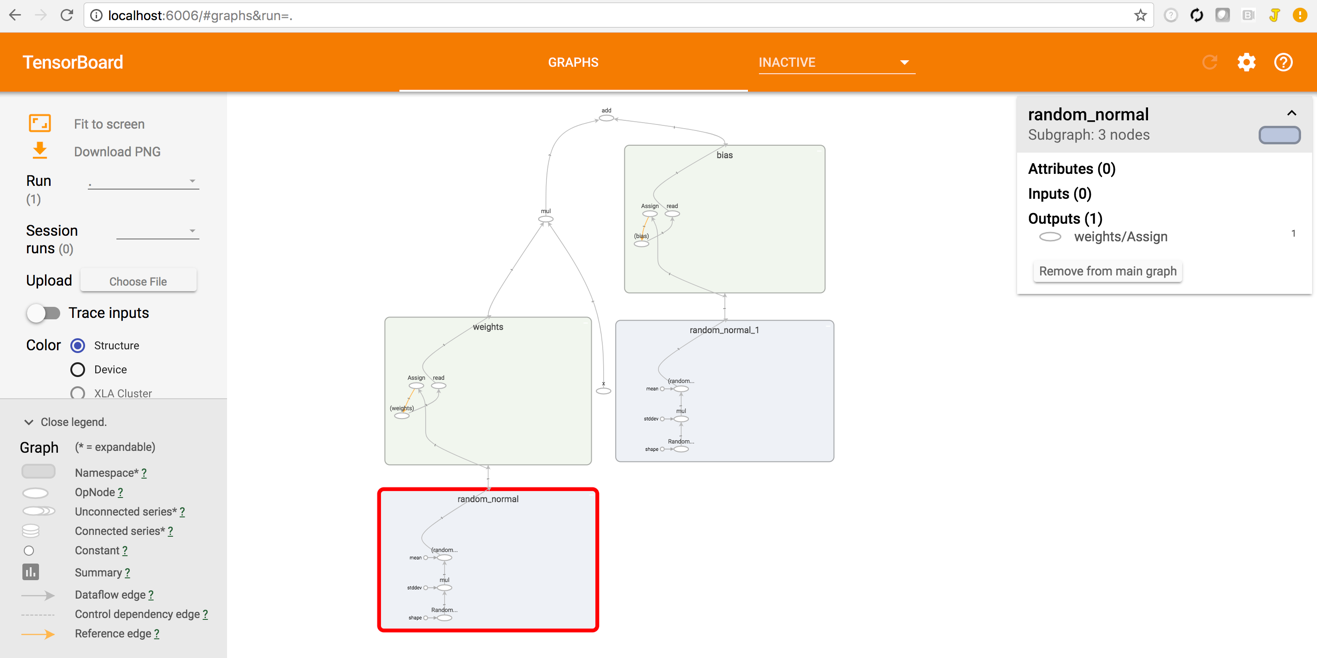
<tf.Variable 'bias:0' shape=(1,) dtype=float32\_ref>

Tensor("x:0", shape=(?,), dtype=float32)

**Changed**

**sw = tf.train.SummaryWriter('summaries/', graph=tf.get\_default\_graph())**

**sw = tf.summary.FileWriter('summaries/', graph=tf.get\_default\_graph())**



Output

print (y\_label)

Tensor("y\_label:0", shape=(?,), dtype=float32)

print("loss:", loss)

print("optimizer:", optimizer)

print("train:", train)

loss: Tensor("Mean:0", shape=(), dtype=float32)

optimizer: <tensorflow.python.training.gradient\_descent.GradientDescentOptimizer object at 0x181fd9a290>

train: name: "GradientDescent"

op: "NoOp"

input: "^GradientDescent/update\_weights/ApplyGradientDescent"

input: "^GradientDescent/update\_bias/ApplyGradientDescent"

Changed:

**init = tf.initialize\_all\_variables()**

**init = tf.global\_variables\_initializer()**

init = tf.global\_variables\_initializer()

print(init)

name: "init"

op: "NoOp"

input: "^weights/Assign"

input: "^bias/Assign"

print(sess.run([W, b]))

[array([ 0.88158017], dtype=float32), array([-1.58914769], dtype=float32)]

sess.run(y, feed\_dict={x: x\_in})

array([ 1.05559289], dtype=float32)

def eval\_loss():

return sess.run(loss, feed\_dict={x: x\_eval, y\_label: y\_eval})

eval\_loss()

2.3174069

Changed:

**tf.scalar\_summary('loss', loss)**

**summary\_op = tf.merge\_all\_summaries()**

**tf.summary.scalar('loss', loss)**

**summary\_op = tf.summary.merge\_all()**

Output

# Perform training.

for step in range(201):

# Run the training op; feed the training data into the graph

summary\_str, \_ = sess.run([summary\_op, train], feed\_dict={x: x\_train, y\_label: y\_train})

sw.add\_summary(summary\_str, step)

# Uncomment the following two lines to watch training happen real time.

if step % 20 == 0:

print(step, sess.run([W, b]))

0 [array([ 0.3591727], dtype=float32), array([ 0.73669899], dtype=float32)]

20 [array([ 0.10021377], dtype=float32), array([ 0.30003777], dtype=float32)]

40 [array([ 0.09564207], dtype=float32), array([ 0.30253857], dtype=float32)]

60 [array([ 0.094291], dtype=float32), array([ 0.30327761], dtype=float32)]

80 [array([ 0.09389172], dtype=float32), array([ 0.30349603], dtype=float32)]

100 [array([ 0.09377371], dtype=float32), array([ 0.30356058], dtype=float32)]

120 [array([ 0.09373882], dtype=float32), array([ 0.30357966], dtype=float32)]

140 [array([ 0.09372854], dtype=float32), array([ 0.30358529], dtype=float32)]

160 [array([ 0.09372547], dtype=float32), array([ 0.30358696], dtype=float32)]

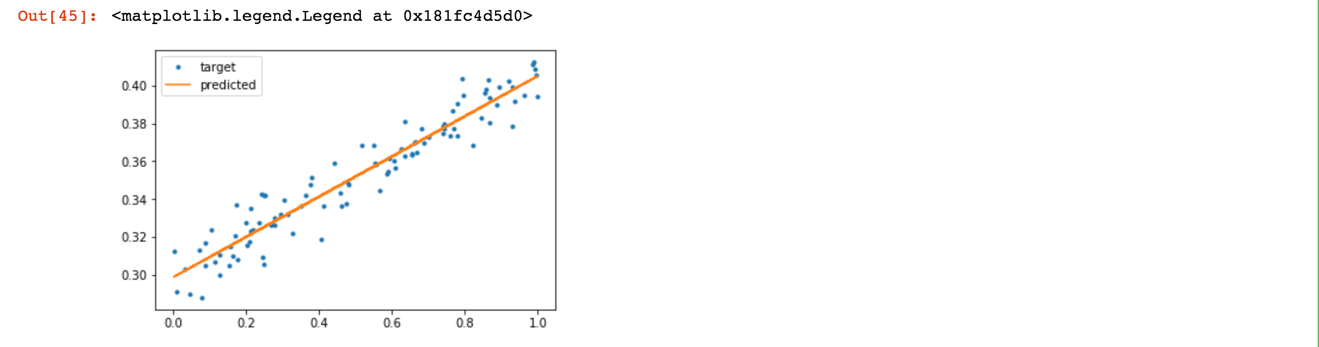
180 [array([ 0.0937246], dtype=float32), array([ 0.30358744], dtype=float32)]

200 [array([ 0.09372433], dtype=float32), array([ 0.30358759], dtype=float32)]

pylab.plot(x\_train, y\_train, '.', label="target")

pylab.plot(x\_train, sess.run(y, feed\_dict={x: x\_train, y\_label: y\_train}), label="predicted")

pylab.legend()



eval\_loss()

0.00012203625

predict(3)

array([ 0.58476055], dtype=float32)

# Prediction after variables reinitialized

predict(3)

array([-3.8621068], dtype=float32)

saver.restore(sess, './my\_checkpoint.ckpt')

INFO:tensorflow:Restoring parameters from ./my\_checkpoint.ckpt

# Predictions after variables restored

predict(3)

array([ 0.61810893], dtype=float32)

Please, describe every step of your work and present all intermediate and final results in a Word document. Please, copy past text version of all essential command and snippets of results into the Word document with explanations of the purpose of those commands. We cannot retype text that is in JPG images. Please, always submit a separate copy of the original, working scripts and/or class files you used. Sometimes we need to run your code and retyping is too costly. Please include in your MS Word document only relevant portions of the console output or output files. Sometime either console output or the result file is too long and including it into the MS Word document makes that document too hard to read. PLEASE DO NOT EMBED files into your MS Word document. For issues and comments visit the class Discussion Board.