

Using TensorFlow

Note that this assumes that you've installed all required packages/libraries in the directory in which you'll activate your virtual environment! At a minimum this includes:

- numpy
- scipy
- scikit-learn
- matplotlib
- pandas

1. In a shell (cmd Prompt on PC's or terminal app on Mac's), go to the directory in which you'll activate your virtual environment, e.g. the directory in which you've saved all your jupyter notebooks

a. To set up a virtual environment for Windows 10 go to the webpage:

<https://virtualenv.pypa.io/en/stable/userguide/> Once you've set-up your virtual environment you should have installed everything else for TensorFlow within that virtual environment, i.e. with that virtual environment activated. If you did not and want to run in a virtual environment (there are advantages to this) go back to the installation instructions.

If you want to make your life easier you'll want to create the equivalent of an "alias" on your PC to activate your virtual environment. To do this you need to go to your home directory and create a sub-directory or subfolder named "bin". You need to add the path to this directory to your environment variables, i.e. add the path

`%USERPROFILE%\bin`

In this "bin" directory you need to create an activate.bat file. In that file put the complete path to the script that activates your virtual environment. For example, my complete path is:

`C:\Users\mhamner\AppData\Local\Programs\Python\Python35\Lib\venv\Scripts\activate`

That is all you need in the activate.bat file. One problem I had is that on a PC the easiest way to make a file is to use notepad. However, notepad saves files as .txt

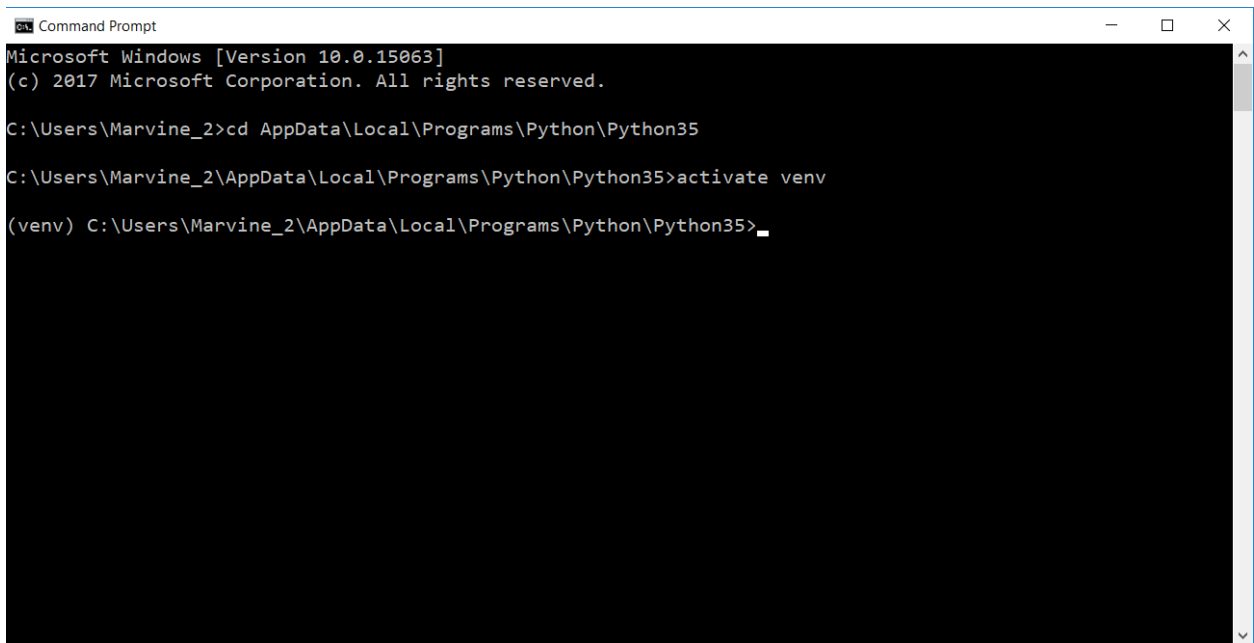
files. So, I got an activate.bat.txt file. I just used a command prompt window, went to the “bin” directory, then used the command “move” to move the .bat.txt file to .bat, i.e. at the prompt I entered:

```
mhamner>move activate.bat.txt activate.bat
```

Now I can just enter “activate” from wherever I might be and it executes my virtual environment that I use TensorFlow and Jupyter notebook in. P.S. you can create as many .bat files as you want to create shortcut commands like this. For example, since I am used to entering “ls” to list out directories and files in a shell I created a ls.bat file in the “bin” directory that only has the command “dir” in it. Now I can enter “ls” to list things out or “dir” whichever I might remember at the time.

- b. To set up a virtual environment on a Mac go to the Installing TensorFlow on macOS webpage: https://www.tensorflow.org/install/install_mac and follow the instructions for the Virtualenv installation.
2. Next, activate venv (or whatever you’ve named your virtual environment. I just named mine “venv”.) in a command window on PC’s and in the terminal app on Mac’s (figures below). The command on PC’s is “activate venv” and on Mac’s is “source activate venv”. If you forget and try “activate venv” on a Mac you’ll get the error message that you need to source to activate as shown in the figure for a Mac below. If the virtual environment is properly activated you will see what you’ve named your virtual environment in parenthesis before the path in the shell.

Screenshot for PC cmd Prompt shell:



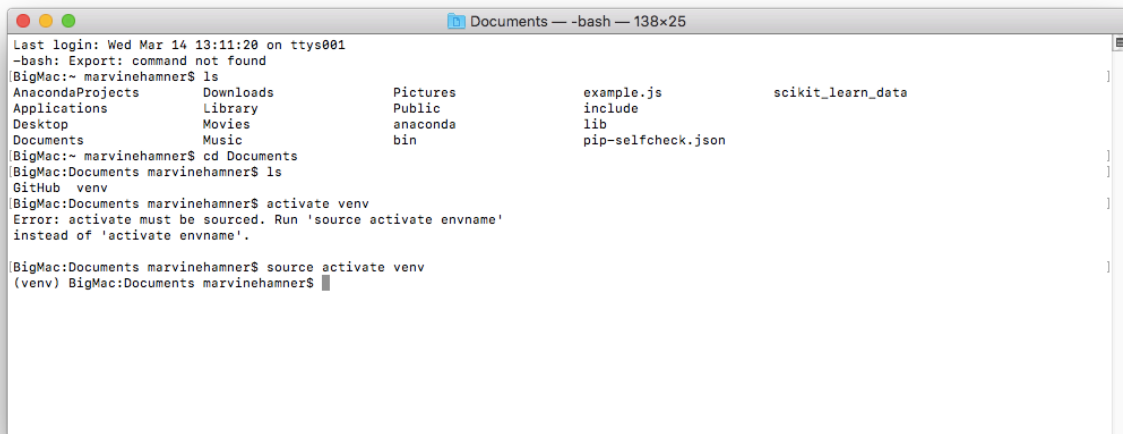
```
Command Prompt
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Marvine_2>cd AppData\Local\Programs\Python\Python35

C:\Users\Marvine_2\AppData\Local\Programs\Python\Python35>activate venv

(venv) C:\Users\Marvine_2\AppData\Local\Programs\Python\Python35>_
```

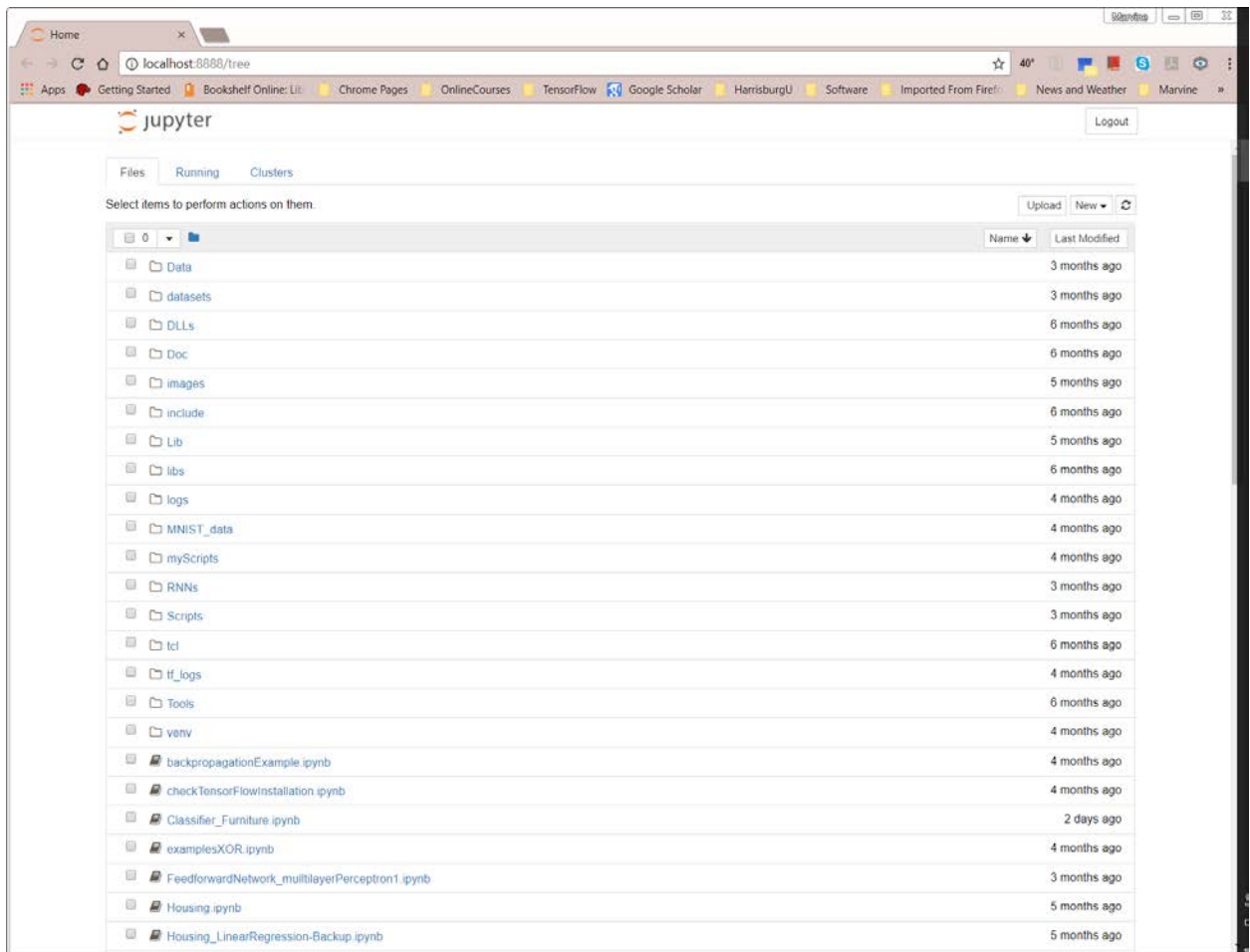
Screenshot for Mac terminal app:



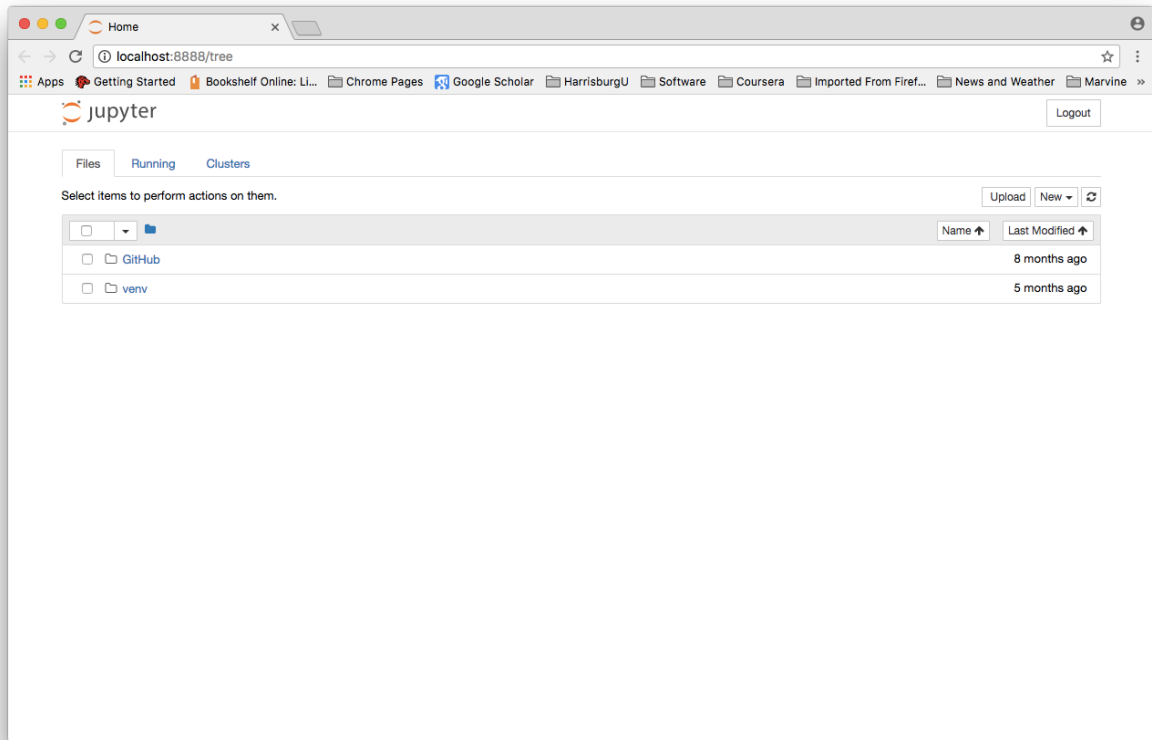
```
Documents — -bash — 138x25
Last login: Wed Mar 14 13:11:20 on ttys001
-bash: Export: command not found
BigMac:~ marvinehamner$ ls
AnacondaProjects  Downloads          Pictures           example.js         scikit_learn_data
Applications      Library           Public            include
Desktop          Movies            anaconda          lib
Documents        Music             bin              pip-selfcheck.json
BigMac:~ marvinehamner$ cd Documents
BigMac:Documents marvinehamner$ ls
GitHub  venv
BigMac:Documents marvinehamner$ activate venv
Error: activate must be sourced. Run 'source activate envname'
instead of 'activate envname'.
BigMac:Documents marvinehamner$ source activate venv
(venv) BigMac:Documents marvinehamner$
```

3. Key in “jupyter-notebook” (or if you haven’t already installed jupyter go to jupyter.org and follow the instructions for installing jupyter notebook) which will open a Jupyter notebook “Home” page in a browser window (figures below). If you do not want it to open in the browser window you already have open – open a new browser window.

Screenshot for PC's using the Chrome browser:



Screenshot for Mac's using the Chrome browser:



4. Once you've opened jupyter-notebook you should see everything in that directory or folder including all the jupyter notebooks you have saved in that directory (figure above). Note that now that you are working in a browser both PC's and Mac's browser window appear the same. So, I'll only include information for both PC's and Mac's when I need to show them separately again. That is, I'm only showing the one figure for the Chrome browser window which is the same on both PC's and Mac's. Click on the notebook you want to run and it will open in a new tab in that browser window (figure below).

```
In [1]: # To support both python 2 and python 3
from __future__ import division, print_function, unicode_literals
import os

#make sure you are in the correct working directory
os.chdir('C:\\Users\\Marvine_2\\AppData\\Local\\Programs\\Python\\Python35\\Lib\\site-packages')

# to make this notebook's output stable across runs
def reset_graph(seed=42):
    tf.reset_default_graph()
    tf.set_random_seed(seed)
    np.random.seed(seed)

import numpy as np

# To plot better figures
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
plt.rcParams['axes.labelsize'] = 14
plt.rcParams['xtick.labelsize'] = 12
plt.rcParams['ytick.labelsize'] = 12

logs_path = '/tmp/tensorflow_logs/example/'

# where to save
PROJECT_ROOT_DIR = "."
CHAPTER_ID = "chapters"

def save_fig(fig_id, tight_layout=True):
    path = os.path.join(PROJECT_ROOT_DIR, "images", CHAPTER_ID, fig_id + ".png")
    print("Saving figure", fig_id)
    if tight_layout:
        plt.tight_layout()
    plt.savefig(path, format='png', dpi=300)

import tensorflow as tf

reset_graph()

In [2]: node1 = tf.constant(3.0, name="firstconst")
node2 = tf.constant(4.0, name="secondconst")
node3 = tf.add(node1, node2, name="sum")
node4 = tf.divide(node3, node1, name="sumdiv")

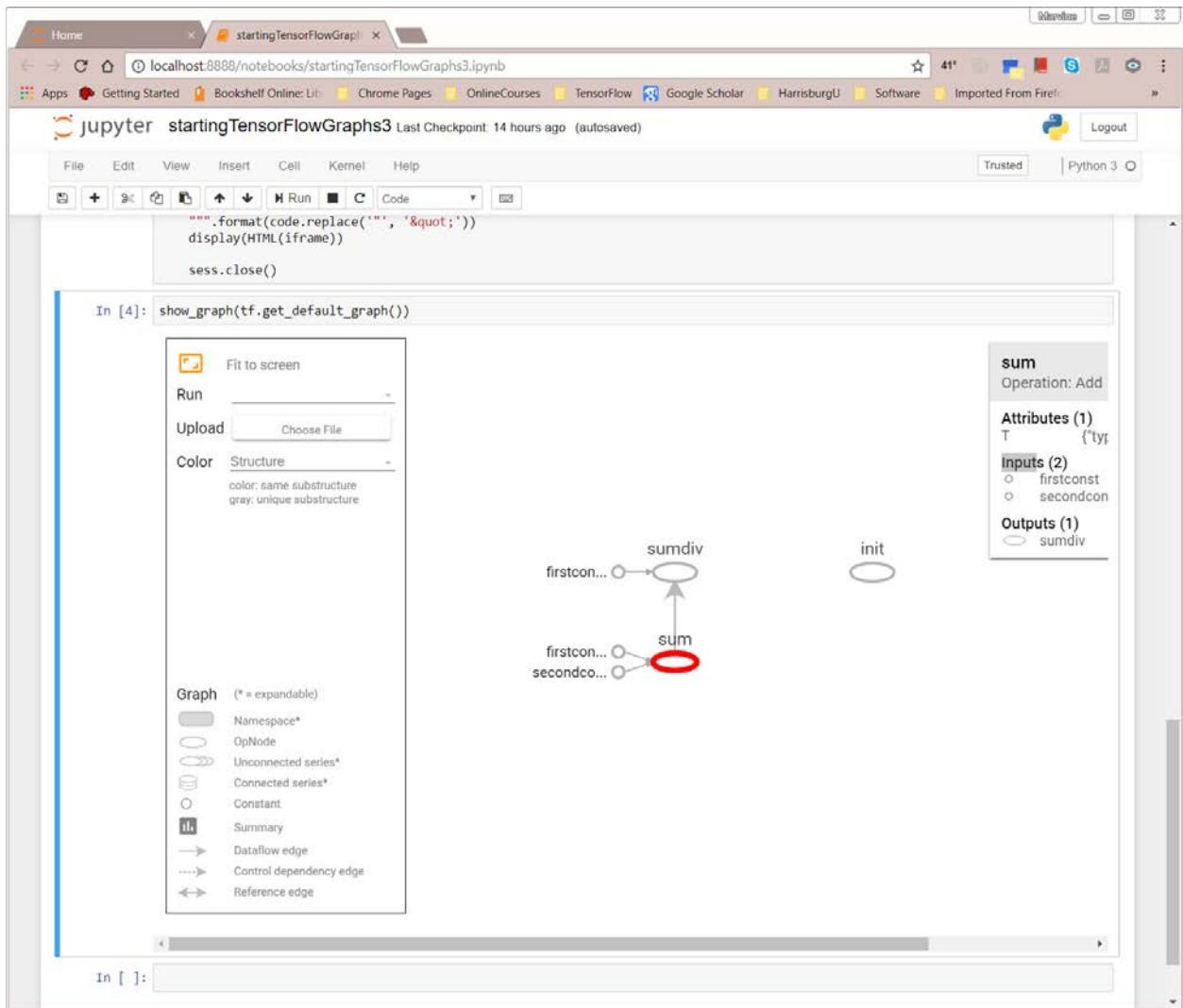
sess = tf.Session()
```

Out[2]: [3.0, 4.0, 7.0, 2.3333333]

5. The code, “startingTensorFlowGraphics” is for you to use to test your TensorFlow installation. You should run all blocks in this notebook in the order they are in for it to run correctly. I have had trouble getting Jupyter notebooks to function properly when I’ve skipped blocks or run blocks out of order. If you run all the blocks in order you should get

Out[2]: [3.0, 4.0, 7.0, 2.3333333]

as the output for the second block. Note that in the code this notebook sets up four nodes; Node1 named firstconst equals 3.0, Node2 named secondconst equals 4.0, Node3 is the sum of Nodes 1 and 2, and Node4 is Node3 divided by Node 1. You should also get the graph as shown in the figure below. I’ve highlighted the sum node for Nodes 1 and 2 which brings up the box that explains the inputs and outputs for that node.



If you have TensorFlow set-up properly this should be what you get.

What about something a bit more complicated? Let's look at the "classic" example of classifying hand-written digits using the MNIST dataset. We cover this example in detail in class. So, this document only highlights using the corresponding Jupyter notebook and getting the graphical output in Tensorboard.

6. This code is stored in the notebook "mnistClassifier_noRegularization". When you click on that notebook on the Jupyter notebooks Home page you will get a new tab that looks like the figure below. Note that all the code in this notebook is in one block.

```
In [1]: from __future__ import print_function

import os
os.chdir('C:\\Users\\Marvine_2\\AppData\\Local\\Programs\\Python\\Python35\\Lib\\site-packages')

import tensorflow as tf

# Import MNIST data
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("/tmp/data/", one_hot=True)

# Parameters
learning_rate = 0.01
training_epochs = 25
batch_size = 100
display_epoch = 1
logs_path = '/tmp/tensorflow_logs/mnist_example/'

# tf Graph Input
# mnist data image of shape 28*28=784
x = tf.placeholder(tf.float32, [None, 784], name='InputData')
# 0-9 digits recognition => 10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

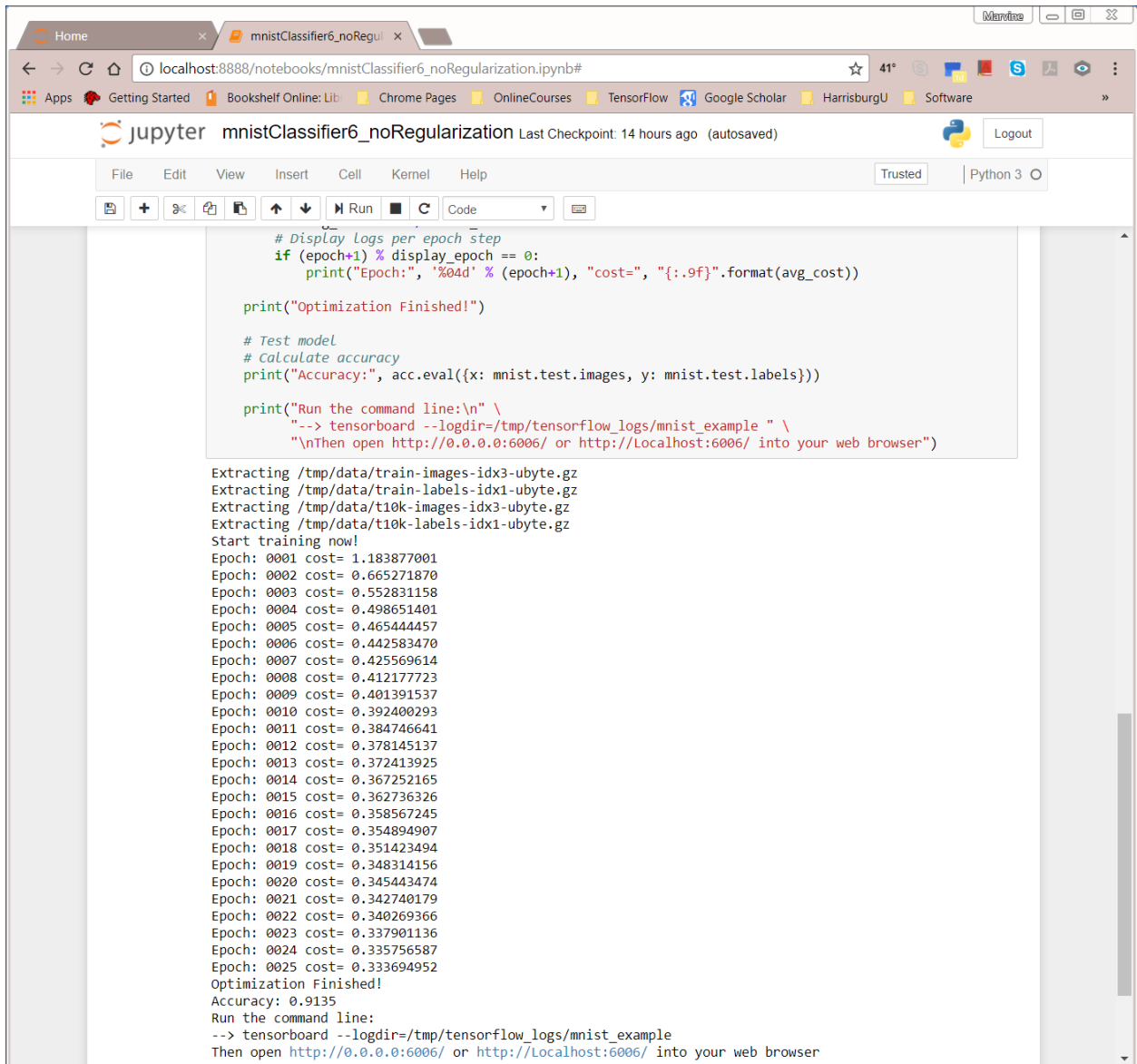
# Set model weights
W = tf.Variable(tf.zeros([784, 10]), name='Weights')
b = tf.Variable(tf.zeros([10]), name='Bias')

# Construct model and encapsulating all ops into scopes, making
# Tensorboard's Graph visualization more convenient
with tf.name_scope('Model'):
    # Model
    pred = tf.nn.softmax(tf.matmul(x, W) + b) # Softmax
with tf.name_scope('Loss'):
    # Minimize error using cross entropy
    cost = tf.reduce_mean(-tf.reduce_sum(y*tf.log(pred), reduction_indices=1))
with tf.name_scope('SGD'):
    # Gradient Descent
    optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(cost)
with tf.name_scope('Accuracy'):
    # Accuracy
    acc = tf.equal(tf.argmax(pred, 1), tf.argmax(y, 1))
    acc = tf.reduce_mean(tf.cast(acc, tf.float32))

# Initialize the variables (i.e. assign their default value)
init = tf.global_variables_initializer()
```

7. When I've run this code I've gotten a couple different types of errors. For example, I have set-up environment variables and paths to run TensorFlow on the GPU. I updated my rtools and started getting error messages. One of the proposed solutions is to reorder the paths/variables. I did that by moving the new path to rtools down. When I did that I was able to run this code again without error. But, I am not satisfied that this is a real solution. The other proposed solution is that the GPU is running out of memory. This has happened to me before but not with the MNIST dataset, only with much larger datasets. Perhaps by changing the order of the paths things start in a different order freeing up memory. But, I

really think that is a “stretch” to believe. The message is that you may have to play with things a bit to run different codes. When the code runs you should get the output in the figure below.



```
# Display Logs per epoch step
if (epoch+1) % display_epoch == 0:
    print("Epoch:", '%04d' % (epoch+1), "cost=", "{:.9f}".format(avg_cost))

print("Optimization Finished!")

# Test model
# Calculate accuracy
print("Accuracy:", acc.eval({x: mnist.test.images, y: mnist.test.labels}))

print("Run the command line:\n" \
      "--> tensorboard --logdir=/tmp/tensorflow_logs/mnist_example " \
      "\nThen open http://0.0.0.0:6006/ or http://localhost:6006/ into your web browser")

Extracting /tmp/data/train-images-idx3-ubyte.gz
Extracting /tmp/data/train-labels-idx1-ubyte.gz
Extracting /tmp/data/t10k-images-idx3-ubyte.gz
Extracting /tmp/data/t10k-labels-idx1-ubyte.gz
Start training now!
Epoch: 0001 cost= 1.183877001
Epoch: 0002 cost= 0.665271870
Epoch: 0003 cost= 0.552831158
Epoch: 0004 cost= 0.498651401
Epoch: 0005 cost= 0.465444457
Epoch: 0006 cost= 0.442583470
Epoch: 0007 cost= 0.425569614
Epoch: 0008 cost= 0.412177723
Epoch: 0009 cost= 0.401391537
Epoch: 0010 cost= 0.392400293
Epoch: 0011 cost= 0.384746641
Epoch: 0012 cost= 0.378145137
Epoch: 0013 cost= 0.372413925
Epoch: 0014 cost= 0.367252165
Epoch: 0015 cost= 0.362736326
Epoch: 0016 cost= 0.358567245
Epoch: 0017 cost= 0.354894907
Epoch: 0018 cost= 0.351423494
Epoch: 0019 cost= 0.348314156
Epoch: 0020 cost= 0.345443474
Epoch: 0021 cost= 0.342740179
Epoch: 0022 cost= 0.340269366
Epoch: 0023 cost= 0.337901136
Epoch: 0024 cost= 0.335756587
Epoch: 0025 cost= 0.333694952
Optimization Finished!
Accuracy: 0.9135
Run the command line:
--> tensorboard --logdir=/tmp/tensorflow_logs/mnist_example
Then open http://0.0.0.0:6006/ or http://localhost:6006/ into your web browser
```

8. Notice the last few lines in this window. If you have Tensorboard installed, the way I run Tensorboard is to open a second cmd Prompt shell in Windows or a second terminal on my Mac. Go to the appropriate directory or folder. And, if you are using one, don't forget to activate your virtual environment. Then, enter the line

“tensorboard --logdir=/tmp/tensorflow_logs/mnist_example”

in the new shell (figure below for PC's and use the same commands as shown above for Mac's). Note that you'll continue to get a message about finding a graph, etc. until you open the new browser window and enter the URL.

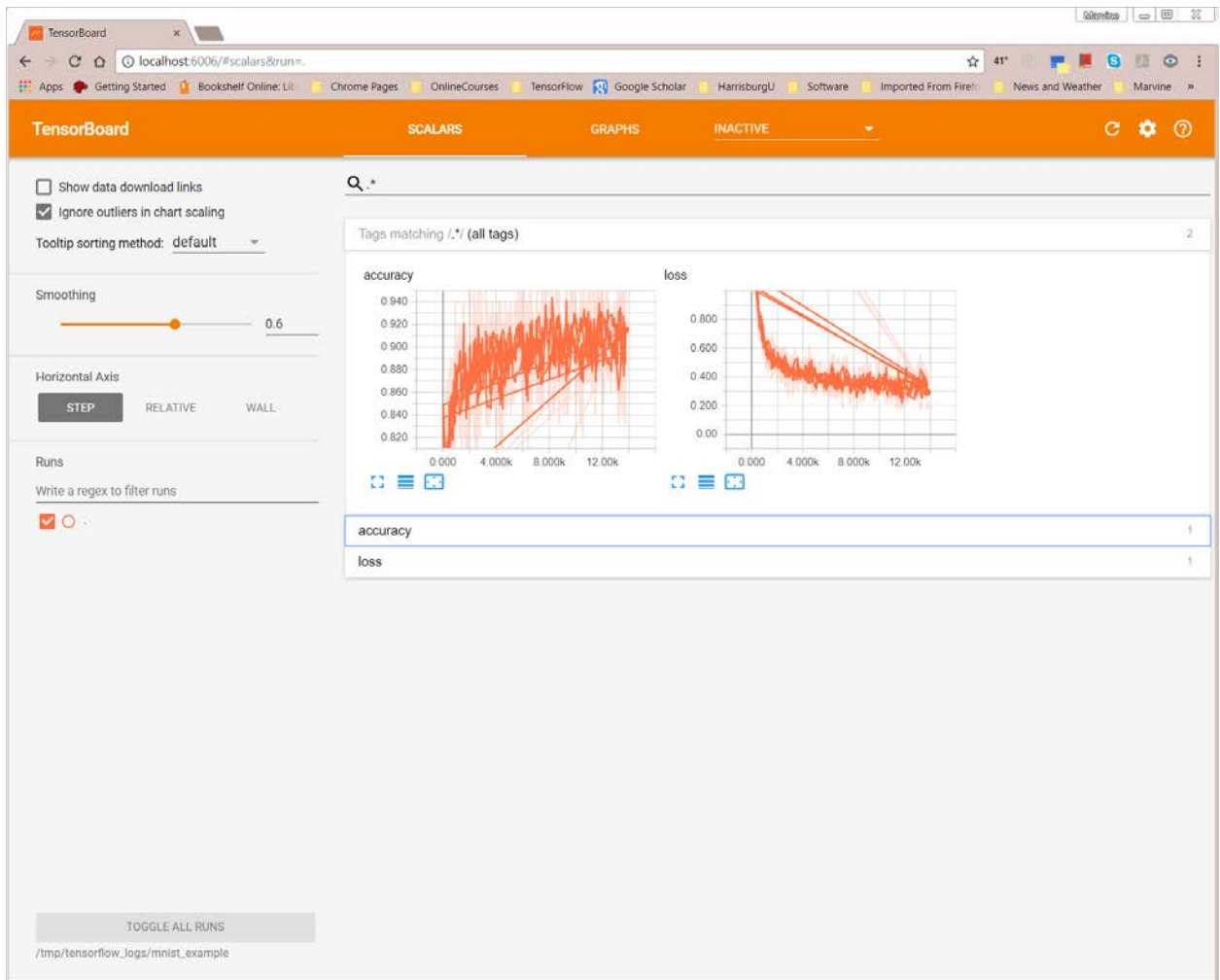
```
Command Prompt - tensorboard --logdir=tmp/tensorflow_logs/mnist_example
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

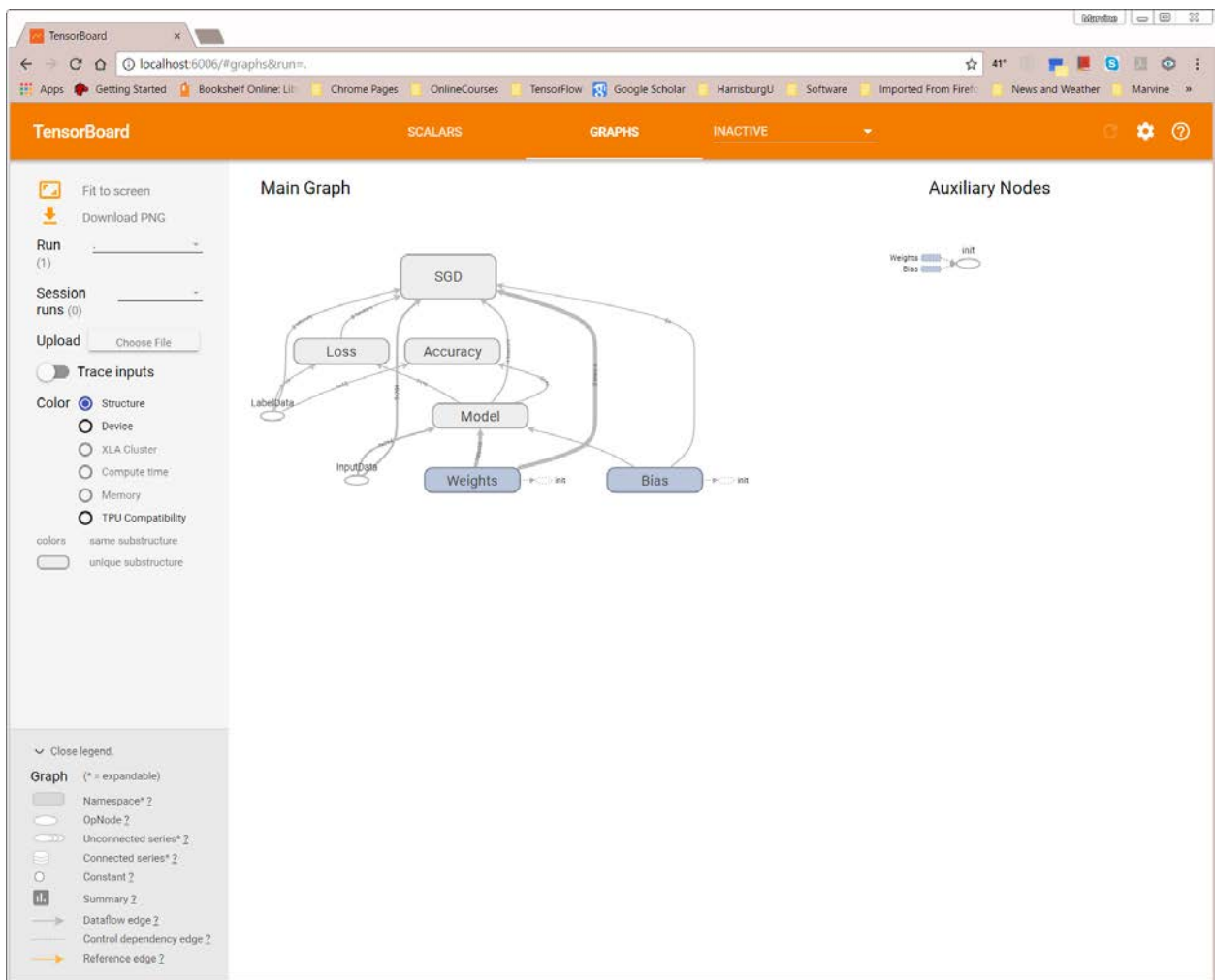
C:\Users\Marvine_2>cd AppData\Local\Programs\Python\Python35

C:\Users\Marvine_2\AppData\Local\Programs\Python\Python35>activate venv

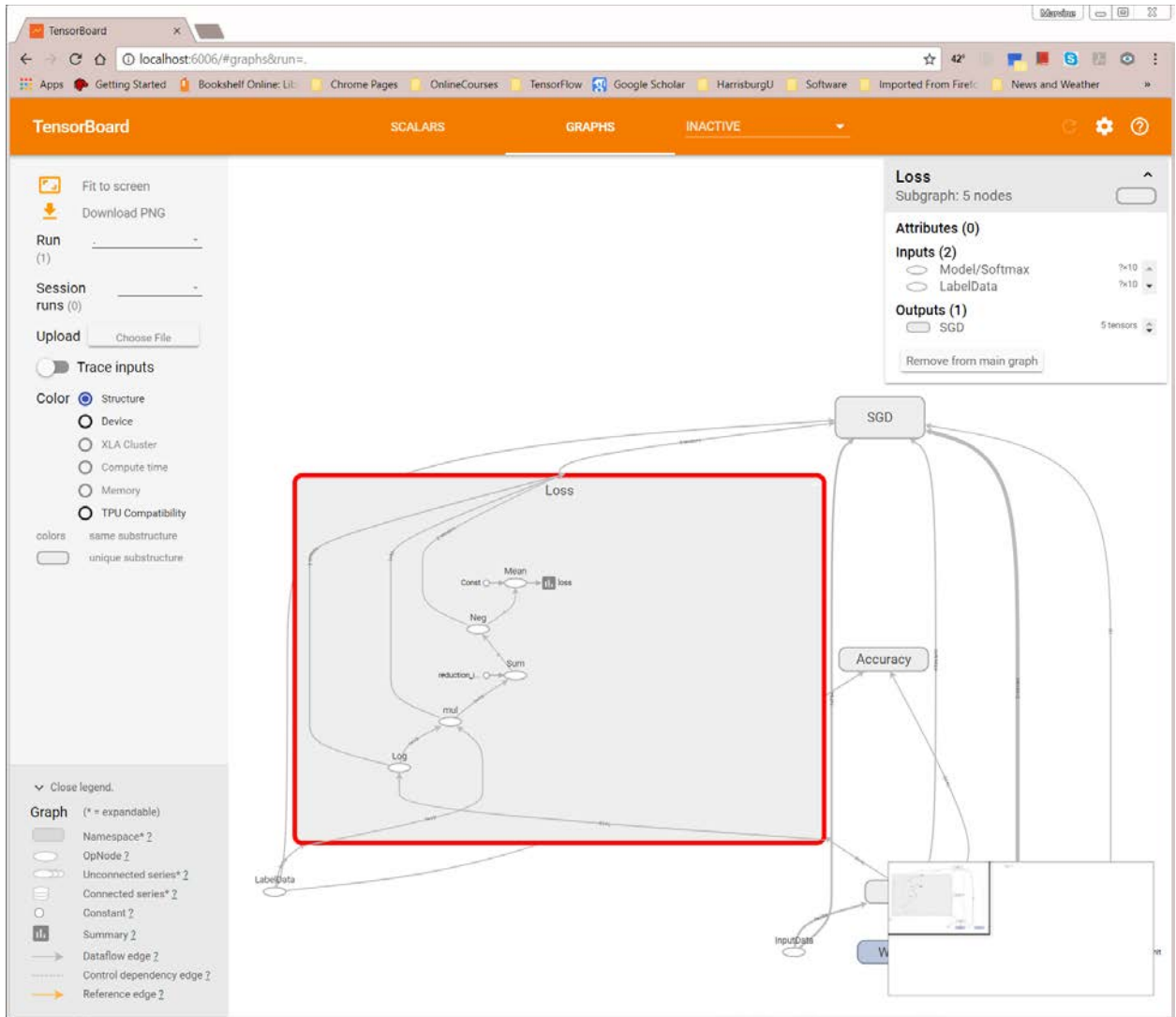
(venv) C:\Users\Marvine_2\AppData\Local\Programs\Python\Python35>tensorboard --logdir=tmp/tensorflow_logs/mnist_example
TensorBoard 0.4.0rc2 at http://Homer:6006 (Press CTRL+C to quit)
W0314 12:41:16.299290 Reloader tf_logging.py:86] Found more than one graph event per run, or there was a metagraph containing a graph_def, as
well as one or more graph events. Overwriting the graph with the newest event.
W0314 12:41:16.299290 Reloader tf_logging.py:86] Found more than one metagraph event per run. Overwriting the metagraph with the newest event.
W0314 12:41:22.083155 Reloader tf_logging.py:86] Found more than one graph event per run, or there was a metagraph containing a graph_def, as
well as one or more graph events. Overwriting the graph with the newest event.
W0314 12:41:22.083155 Reloader tf_logging.py:86] Found more than one metagraph event per run. Overwriting the metagraph with the newest event.
W0314 12:41:27.659443 Reloader tf_logging.py:86] Found more than one graph event per run, or there was a metagraph containing a graph_def, as
well as one or more graph events. Overwriting the graph with the newest event.
W0314 12:41:27.659942 Reloader tf_logging.py:86] Found more than one metagraph event per run. Overwriting the metagraph with the newest event.
W0314 12:41:27.678728 Reloader tf_logging.py:86] Found more than one graph event per run, or there was a metagraph containing a graph_def, as
well as one or more graph events. Overwriting the graph with the newest event.
W0314 12:41:27.678728 Reloader tf_logging.py:86] Found more than one metagraph event per run. Overwriting the metagraph with the newest event.
W0314 12:41:27.725602 Reloader tf_logging.py:86] Found more than one graph event per run, or there was a metagraph containing a graph_def, as
```

9. Open a second browser window and enter <http://0.0.0.0:6006/> or <http://localhost:6006/> in that browser and go to that webpage. This will display the graphical output from TensorFlow via Tensorboard in that window. For the MNIST Classifier you should get the following screens for the “Scalars” or chart figures and the “Graphs” for the actual network.

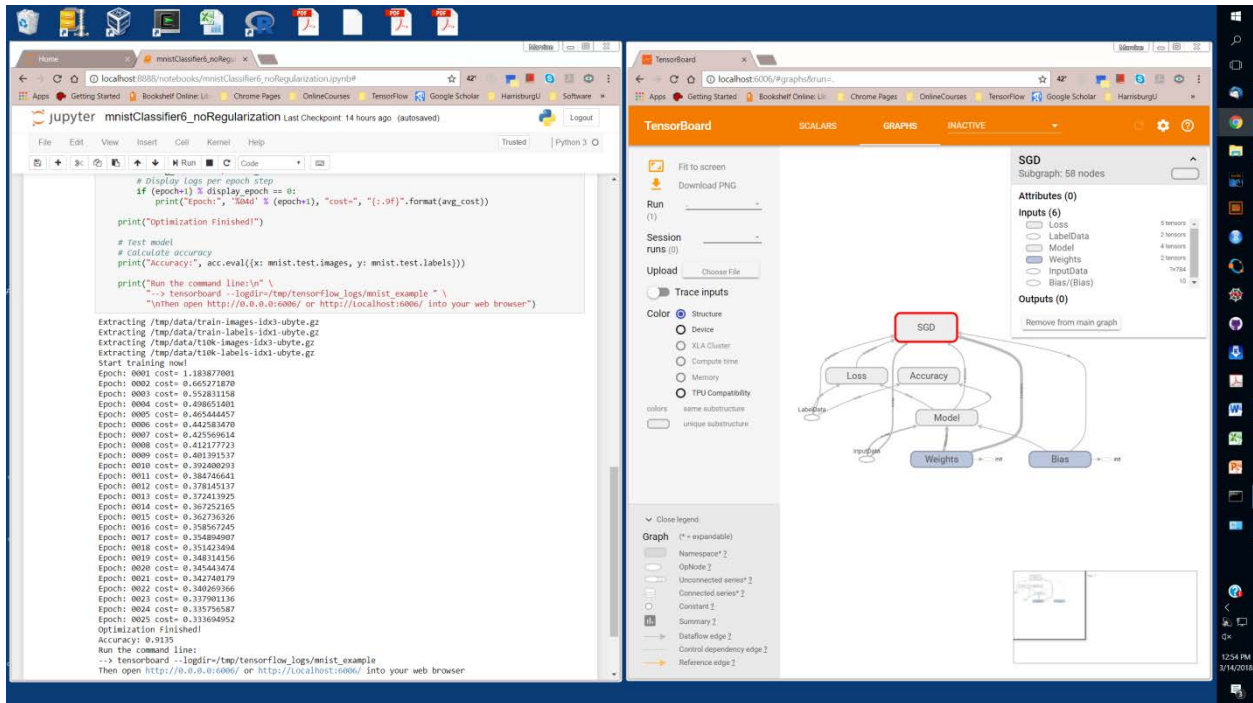




In this graph you can see the InputData, and LabelData from the MNIST data files. You can inspect the Weights and Bias for the network. And, you can look at the SGD (stochastic gradient descent), Loss or Accuracy nodes. For example, if you look at the Loss node you'll see that there are 2 inputs, the Model/Softmax and the LabelData. That is, TensorFlow has set-up the nodes and established the flow within the network for you (figure below).



10. Overall, when I run TensorFlow with Tensorboard I set-up my windows so I can see the code beside the graphical output (figure below).



And that's really all there is to getting started using TensorFlow and Tensorboard.

The last thing to do is to save your work. There are several ways of doing this. I usually use the "File" pull-down menu to "Save and Checkpoint" code when I've got it running. That way I can return to a previous checkpoint if I change a code and it stops working.

You can also use the "File" pull-down menu to "Close and Halt" a code in Jupyter notebook. Once you've done that it will close the tab that code was on. Then, just "Logout" on Jupyter notebook's Home page.

Once you've logged out of Jupyter notebook you may have to use "Ctrl-C" to quit that function in your shell (cmd Prompt on PC's or terminal app on Mac's). Then, deactivate your virtual environments using "deactivate venv" on PC's or "source deactivate venv" on Mac's and close the shells. Now you should have everything closed out properly and won't have any issues running anything else.

If you find typos in this document please let me know!

Marvine