Assignment 11
CSCI E 63 - Big Data Analytics

Problem 1:
Consider provided Jupyter notebook `Summaries and NameScopes.ipynb`. Add one more output summary. For example, calculate the rolling mean of all one dimensional tensors passed as arguments of `run_graph` function. Provide working notebook and images of your graphs and calculated summaries. In the Word document presented as your solution provide snippets of additional or modified code.

Answer:

→ Calculate the rolling mean of all one dimensional tensors passed as arguments of run_graph function

Code added(Notebook separately attached):

→ In the 'variables' scope, create a new variable to keep track of mean of all input values over time. We are going to add the mean for every input to this variable.

```
# Variable that keeps track of mean of all input values over time
total_input_mean = tf.Variable(0.0, dtype=tf.float32, name="total_input_mean")
```

→ In the 'output layer' of the 'transformation' scope, create a variable input_mean. We are going to calculate the mean of the inputs in this variable and add it to the variable created above.

```
input_mean = tf.reduce_mean(a, name = "input_mean")
```

→ In the 'update' scope, create a new variable to perform the assign_add operation. This operation adds the input_mean to the total_input_mean and assigns the updated total to the new variable.

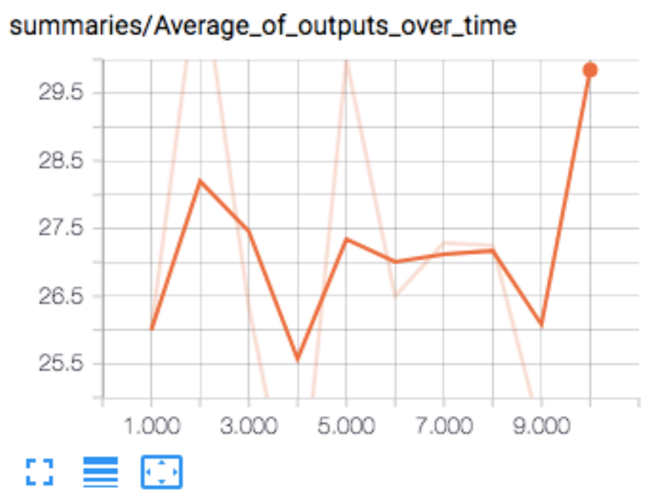
```
# Increments the total_input_mean variable by the latest input_mean
update_input_mean = total_input_mean.assign_add(input_mean)
```

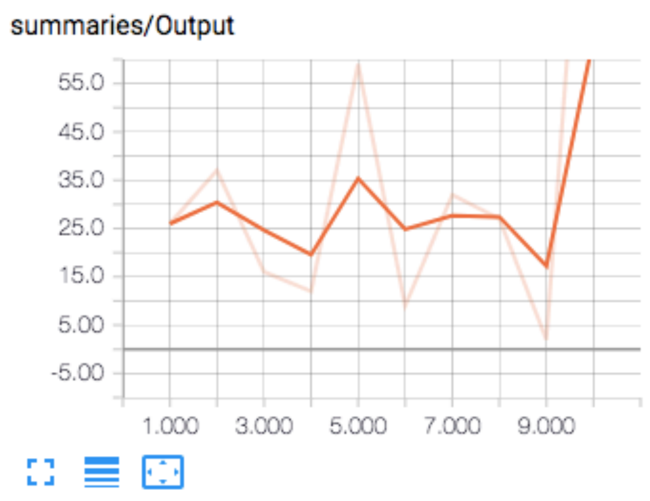
→ Add a new output summary for the rolling mean of the input arguments

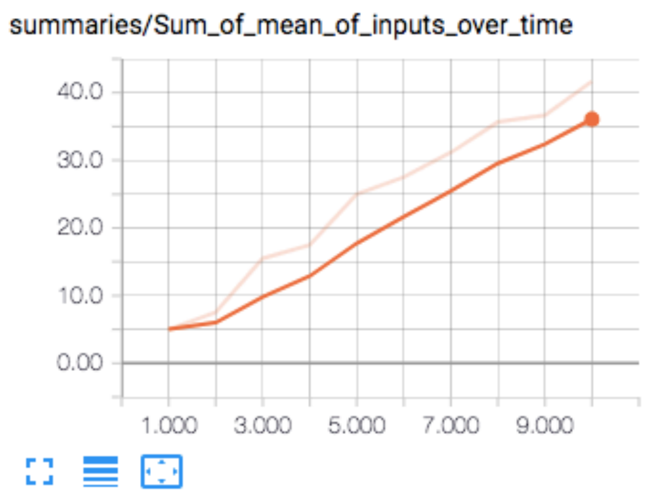
```
# Creates summaries for input node
tf.summary.scalar('Sum_of_mean_of_inputs_over_time',update_input_mean)
```

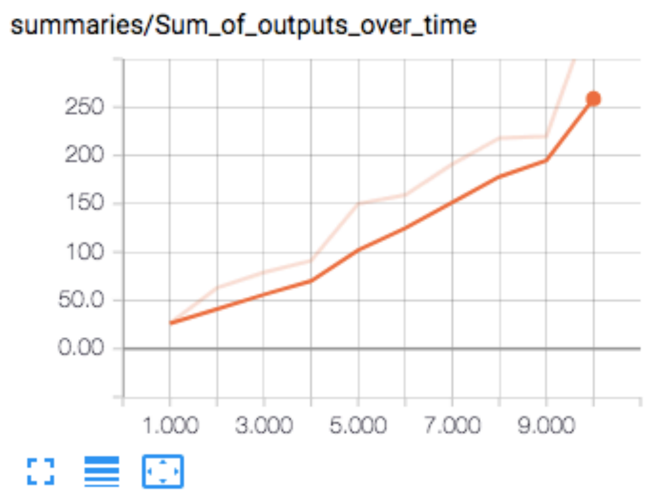
Assignment 11
CSCI E 63 - Big Data Analytics

→ Tensorboard - Graph

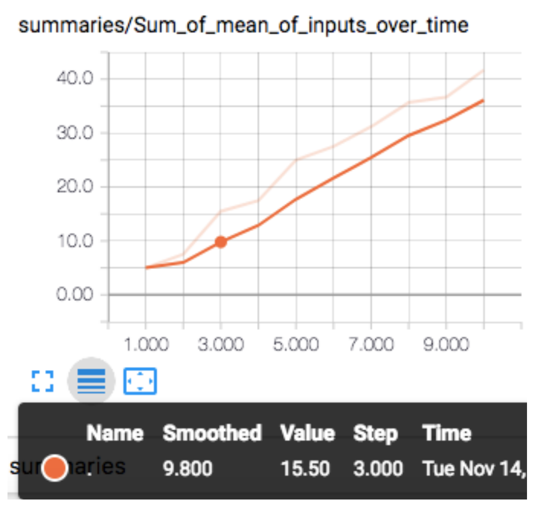




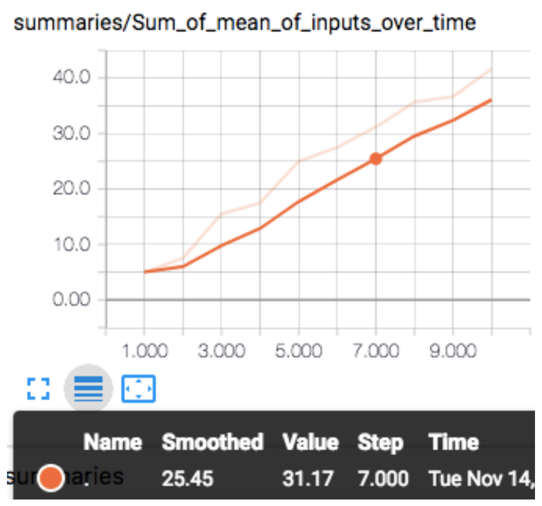




→ Tensorboard - Validating results at 3 inputs and 7 inputs



	Name	Smoothed	Value	Step	Time
●	sur●aries	9.800	15.50	3.000	Tue Nov 14,



	Name	Smoothed	Value	Step	Time
●	sur●aries	25.45	31.17	7.000	Tue Nov 14,

Karan A. Bhandarkar

Assignment 11
CSCI E 63 - Big Data Analytics

Problem 2:

Consider the attached file logistic_regression_mnist.py. Search through TensorFlow API documentation and the Internet and describe for us what is the meaning and purpose of functions used in step 5 and step 6. Demonstrate that you can run the code successfully. Fetch for us the TensorBoard Graph. Vary parameter `batch_size` through values: 8, 64, 128, 256 and report and plot changes in the execution time and accuracy. Keep other parameters the same as in the original program. Similarly, vary parameter `learning_rate` through values 0.001, 0.005, 0.01, 0.02 and 0.05. Report and plot changes in the execution time and accuracy.

Answer:

→ Describe for us what is the meaning and purpose of functions used in step 5 and step 6.

```
# Step 5: define loss function
# use cross entropy of softmax of logits as the loss function
entropy = tf.nn.softmax_cross_entropy_with_logits(logits=logits,labels=Y,
name='loss')
loss = tf.reduce_mean(entropy) # computes the mean over all the examples in the
batch
```

Terms Used:

1. **Logits** simply means that the function operates on the unscaled output of earlier layers and that the relative scale to understand the units is linear. It means, in particular, the sum of the inputs may not equal 1, that the values are *not* probabilities (you might have an input of 5).

2. In mathematics, the **softmax function**, or normalized exponential function, is a generalization of the logistic function that "squashes" a K-dimensional vector of arbitrary real values to a K-dimensional vector of real values in the range [0, 1] that add up to 1.

3. **Cross entropy** can be used to define the loss function in machine learning and optimization.

Meaning:
tf.nn.softmax_cross_entropy_with_logits computes the cross entropy of the result after applying the softmax function (but it does it all together in a more mathematically careful way) between logics and labels.

Purpose:
It measures the probability error in discrete classification tasks in which the classes are mutually exclusive (each entry is in exactly one class). For example, each CIFAR-10 image is labeled with one and only one label: an image can be a dog or a truck, but not both.

Assignment 11
CSCI E 63 - Big Data Analytics

```
# Step 6: define training op
# using gradient descent with learning rate of 0.01 to minimize loss
optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(loss)
```

Terms Used:

1. **Gradient descent** is a first-order iterative optimization algorithm for finding the minimum of a function. Here, we look to minimize the loss function.

2. **GradientDescentOptimizer** is an optimizer that implements the gradient descent algorithm. Here, we initialize it for the provided learning rate.

Meaning:
The minimize() function usage is two part:

i. compute_gradients(loss)
Compute gradients of loss for the variables.
This is the first part of minimize(). It returns a list of (gradient, variable) pairs where "gradient" is the gradient for "variable". Note that "gradient" can be a Tensor, an IndexedSlices, or None if there is no gradient for the given variable.

ii. apply_gradients
This is the second part of minimize(). It returns an operation that applies gradients.

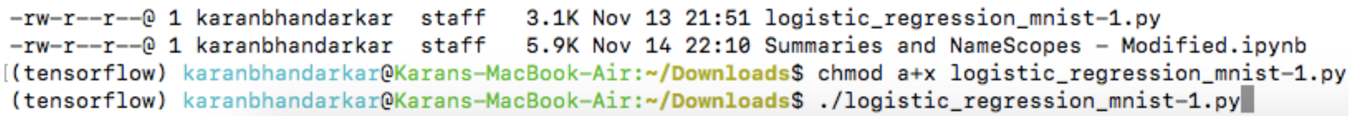
Purpose:
The purpose is to use gradient descent with learning rate of 0.01 to minimize loss.

Assignment 11
CSCI E 63 - Big Data Analytics

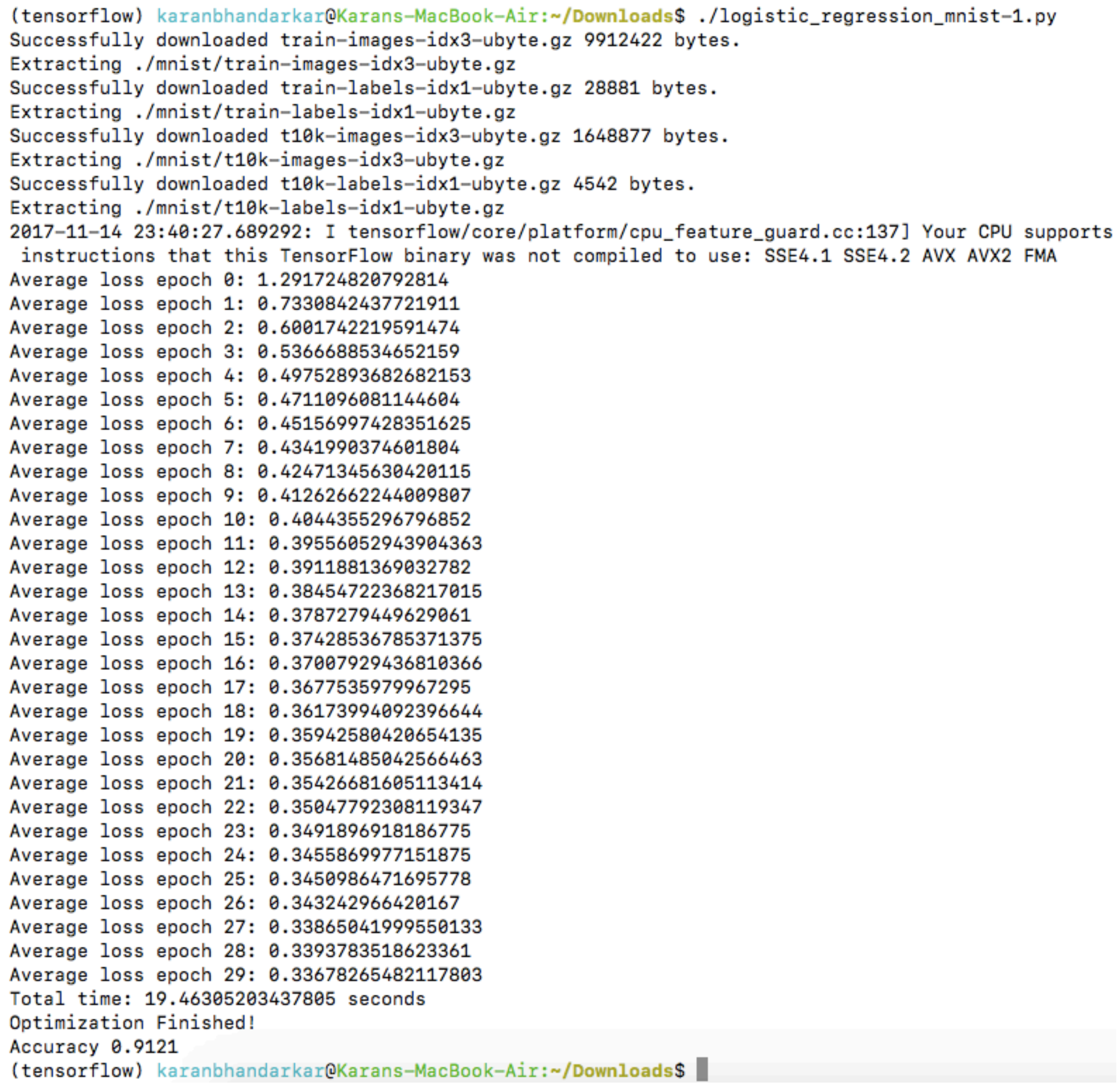
→ Demonstrate that you can run the code successfully.

Line added to provided file:

```
#!/usr/bin/env python3
```



```
-rw-r--r--@ 1 karanbhandarkar  staff   3.1K Nov 13 21:51 logistic_regression_mnist-1.py
-rw-r--r--@ 1 karanbhandarkar  staff   5.9K Nov 14 22:10 Summaries and NameScopes - Modified.ipynb
[(tensorflow) karanbhandarkar@Karans-MacBook-Air:~/Downloads$ chmod a+x logistic_regression_mnist-1.py
(tensorflow) karanbhandarkar@Karans-MacBook-Air:~/Downloads$ ./logistic_regression_mnist-1.py
```



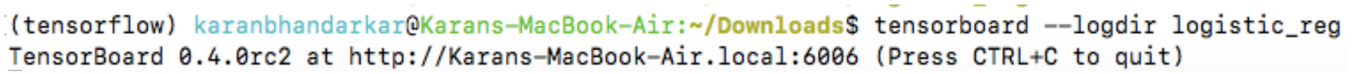
```
(tensorflow) karanbhandarkar@Karans-MacBook-Air:~/Downloads$ ./logistic_regression_mnist-1.py
Successfully downloaded train-images-idx3-ubyte.gz 9912422 bytes.
Extracting ./mnist/train-images-idx3-ubyte.gz
Successfully downloaded train-labels-idx1-ubyte.gz 28881 bytes.
Extracting ./mnist/train-labels-idx1-ubyte.gz
Successfully downloaded t10k-images-idx3-ubyte.gz 1648877 bytes.
Extracting ./mnist/t10k-images-idx3-ubyte.gz
Successfully downloaded t10k-labels-idx1-ubyte.gz 4542 bytes.
Extracting ./mnist/t10k-labels-idx1-ubyte.gz
2017-11-14 23:40:27.689292: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU supports
 instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
Average loss epoch 0: 1.291724820792814
Average loss epoch 1: 0.7330842437721911
Average loss epoch 2: 0.6001742219591474
Average loss epoch 3: 0.5366688534652159
Average loss epoch 4: 0.49752893682682153
Average loss epoch 5: 0.4711096081144604
Average loss epoch 6: 0.45156997428351625
Average loss epoch 7: 0.4341990374601804
Average loss epoch 8: 0.42471345630420115
Average loss epoch 9: 0.41262662244009807
Average loss epoch 10: 0.4044355296796852
Average loss epoch 11: 0.39556052943904363
Average loss epoch 12: 0.3911881369032782
Average loss epoch 13: 0.38454722368217015
Average loss epoch 14: 0.3787279449629061
Average loss epoch 15: 0.37428536785371375
Average loss epoch 16: 0.37007929436810366
Average loss epoch 17: 0.3677535979967295
Average loss epoch 18: 0.36173994092396644
Average loss epoch 19: 0.35942580420654135
Average loss epoch 20: 0.35681485042566463
Average loss epoch 21: 0.35426681605113414
Average loss epoch 22: 0.35047792308119347
Average loss epoch 23: 0.3491896918186775
Average loss epoch 24: 0.3455869977151875
Average loss epoch 25: 0.3450986471695778
Average loss epoch 26: 0.343242966420167
Average loss epoch 27: 0.33865041999550133
Average loss epoch 28: 0.3393783518623361
Average loss epoch 29: 0.33678265482117803
Total time: 19.46305203437805 seconds
Optimization Finished!
Accuracy 0.9121
(tensorflow) karanbhandarkar@Karans-MacBook-Air:~/Downloads$
```

Karan A. Bhandarkar

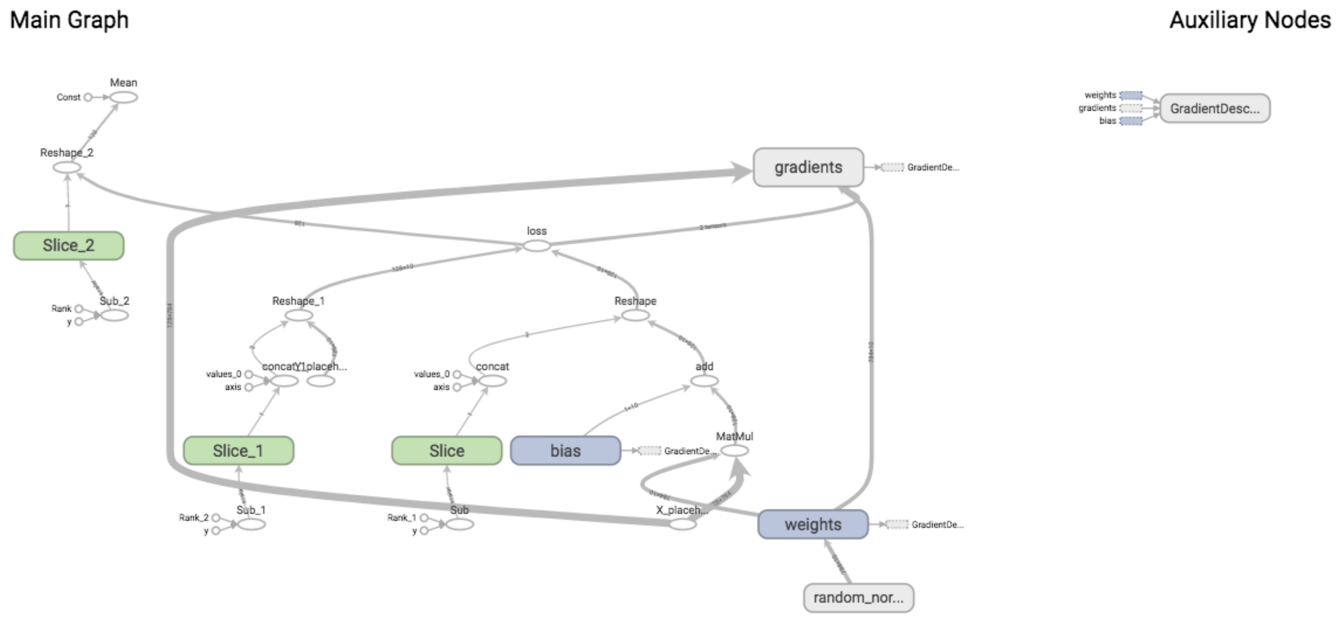
Assignment 11
CSCI E 63 - Big Data Analytics

→ Fetch for us the TensorBoard Graph(Separately attached).

```
tensorboard --logdir logistic_reg
```



```
(tensorflow) karanbhandarkar@Karans-MacBook-Air:~/Downloads$ tensorboard --logdir logistic_reg
TensorBoard 0.4.0rc2 at http://Karans-MacBook-Air.local:6006 (Press CTRL+C to quit)
```



Karan A. Bhandarkar

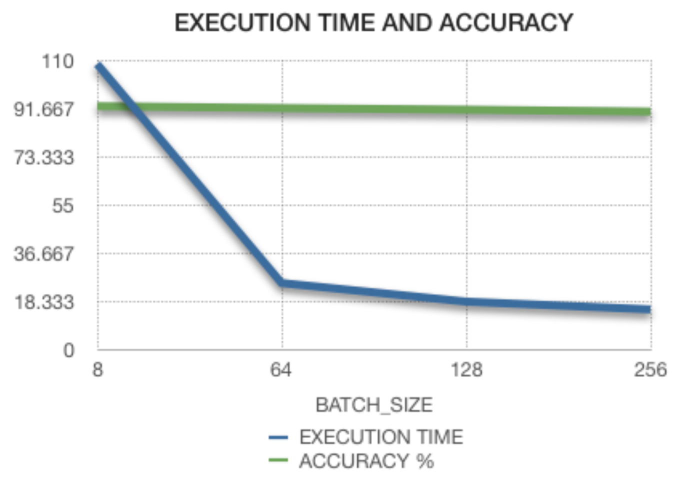
Assignment 11
CSCI E 63 - Big Data Analytics

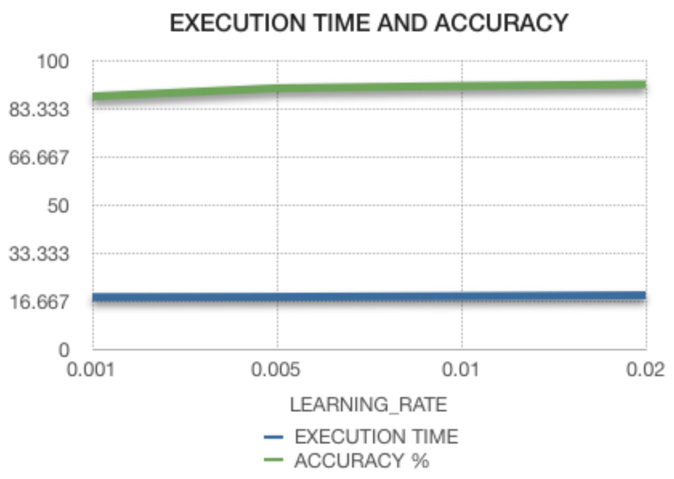
→ Vary parameter `batch_size` through values: 8, 64, 128, 256 and report and plot changes in the execution time and accuracy.

batch_size:	8	64	128	256
Execution Time	108.5636	25.2578	18.1806	15.2257
Accuracy	0.9253	0.9182	0.9121	0.9047

→ Vary parameter `learning_rate` through values 0.001, 0.005, 0.01, 0.02 and 0.05. Report and plot changes in the execution time and accuracy.

learning_rate:	0.001	0.005	0.01	0.02
Execution Time	18.0387	18.1301	18.3789	18.6934
Accuracy	0.8756	0.904	0.9117	0.9174





Karan A. Bhandarkar

Assignment 11
CSCI E 63 - Big Data Analytics

Problem 3:
Fetch Iris Dataset from https://archive.ics.uci.edu/ml/datasets/Iris and make attached Python script, `softmax_irises.py` work. You might have to upgrade the script to TF 1.x API. Generate TensorBoard graph of the process and use scalar summary to presenting variation of the loss function during the training process. Report the results of the evaluation process.

Answer:

→ Ran out of time. Submitted most corrected version os softmax_irises.py