Predictive Analytics World / Deep Learning World
Exercises - TensorFlow

1. Locate Fischer's Iris Dataset and prepare it for use by a TensorFlow (TF) program. Encode setosa as (1,0,0), versicolor as (0,1,0), virginica as (0,0,1). Split the dataset into a training file (120 items, first 40 of each species) and a test file (the remaining 10 of each species).

2. Write a program to create a simple 4-5-3 neural network classifier prediction model. Use tanh for hidden layer activation. Set up stochastic gradient descent training. Run your program.

3. Modify your program to a 4-(5-5)-3 architecture.

4. Which statement is most accurate?

a.) A tf.placeholder is similar to a variable in most programming languages.
b.) A tf.placeholder is similar to a const in most programming languages.
c.) A tf.placeholder is similar to lambda expression in many other programming languages.

5. Which statement is most accurate?

a.) A tf.Variable is used to hold non-numeric data such as char and string.
b.) A tf.Variable is used to hold integer data such as tf.int32 or tf.int64.
c.) A tf.Variable is used to hold numeric variables that will get values via training.

6. What is the difference between a tf.Session and a tf.InteractiveSession?

a.) There is no real difference -- tf.InteractiveSession is now deprecated.
b.) A tf.Session is used for a normal program; a tf.InterractiveSession is used with IPython.
c.) A tf.Session uses only program data; a tf.InteractiveSession can accept user input during run time.

7. Why does TF typically not use softmax activation on output nodes in a neural network classifier?

a.) The tf.nn.softmax_cross_entropy_with_logits_v2() function applies softmax during training.
b.) The tf.nn.cross_entropy() function applies softmax only if preliminary outputs do not sum to 1.0.
c.) There is no built-in softmax() function in TF.

8. Which statement is most accurate?

a.) When training using the tf.Session.run() function, you should randomize the order of training items.
b.) When training using the tf.Session.run() function, you not should randomize the order of training items.
c.) When training using tf.Session.run() it doesn't make any difference if you randomize training order.

```
# iris_tf.py

import numpy as np
import tensorflow as tf
import os
os.environ['TF_CPP_MIN_LOG_LEVEL']='2'

def main():
  np.random.seed(1)
  tfv = tf.__version__
  print("\nUsing TensorFlow version " + str(tfv))

  print("\nLoading iris train and test data \n")
  train_file = ".\\Data\\iris_train_data.txt"
  test_file = ".\\Data\\iris_test_data.txt"

  train_x = np.loadtxt(train_file, usecols=[0,1,2,3], delimiter=",",  skiprows=0, dtype=np.float32)
  train_y = np.loadtxt(train_file, usecols=[4,5,6], delimiter=",", skiprows=0, dtype=np.float32)

  test_x = np.loadtxt(test_file, usecols=[0,1,2,3], delimiter=",",  skiprows=0, dtype=np.float32)
  test_y = np.loadtxt(test_file, usecols=[4,5,6], delimiter=",", skiprows=0, dtype=np.float32)

  input_dim = 4; hidden_dim = 5; output_dim = 3

  X = tf.placeholder(tf.float32, shape=[None, input_dim])
  y = tf.placeholder(tf.float32, shape=[None, output_dim])

  ih_wts = tf.Variable(tf.random_uniform([input_dim, hidden_dim], dtype=tf.float32, seed=1))
  h_biases = tf.Variable(0.0, dtype=tf.float32)
  ho_wts = tf.Variable(tf.random_uniform([hidden_dim, output_dim], dtype=tf.float32, seed=1))
  o_biases = tf.Variable(0.0, dtype=tf.float32)

  h_nodes = tf.add(tf.matmul(X, ih_wts), h_biases)
  h_nodes = tf.nn.tanh(h_nodes)
  o_nodes = tf.add(tf.matmul(h_nodes, ho_wts), o_biases)
  y_predict = tf.argmax(o_nodes, axis=1)  # 0, 1, 2

  learn_rate = 0.01
  max_epochs = 200
  cee = tf.nn.softmax_cross_entropy_with_logits_v2(labels=y, logits=o_nodes)
  cost = tf.reduce_mean(cee)
  optimizer = tf.train.GradientDescentOptimizer(learn_rate)
  trainer = optimizer.minimize(cost)

  init = tf.global_variables_initializer()
  sess = tf.Session()
  sess.run(init)
  print("Starting training")
  for epoch in range(max_epochs):
    indices = np.arange(len(train_x))
    # np.random.shuffle(indices)  # hmm --does this matter?
    for ii in range(len(indices)):
      i = indices[ii]
      sess.run(trainer, feed_dict={X: train_x[i:i+1], y: train_y[i:i+1]})
    train_acc = np.mean(np.argmax(train_y, axis=1) == sess.run(y_predict, feed_dict={X:train_x,\
 y:train_y}))
    if epoch > 0 and epoch % 10 == 0:
      print("epoch = %4d, train accuracy = %.4f " % (epoch, train_acc))
  print("Training complete \n")

  test_acc = np.mean(np.argmax(test_y, axis=1) == sess.run(y_predict, feed_dict={X:test_x, y:test_y}))
  print("Accuracy on test data = %.4f " % test_acc)
  sess.close()

  print("\nEnd demo")

if __name__ == "__main__":
  main()
```