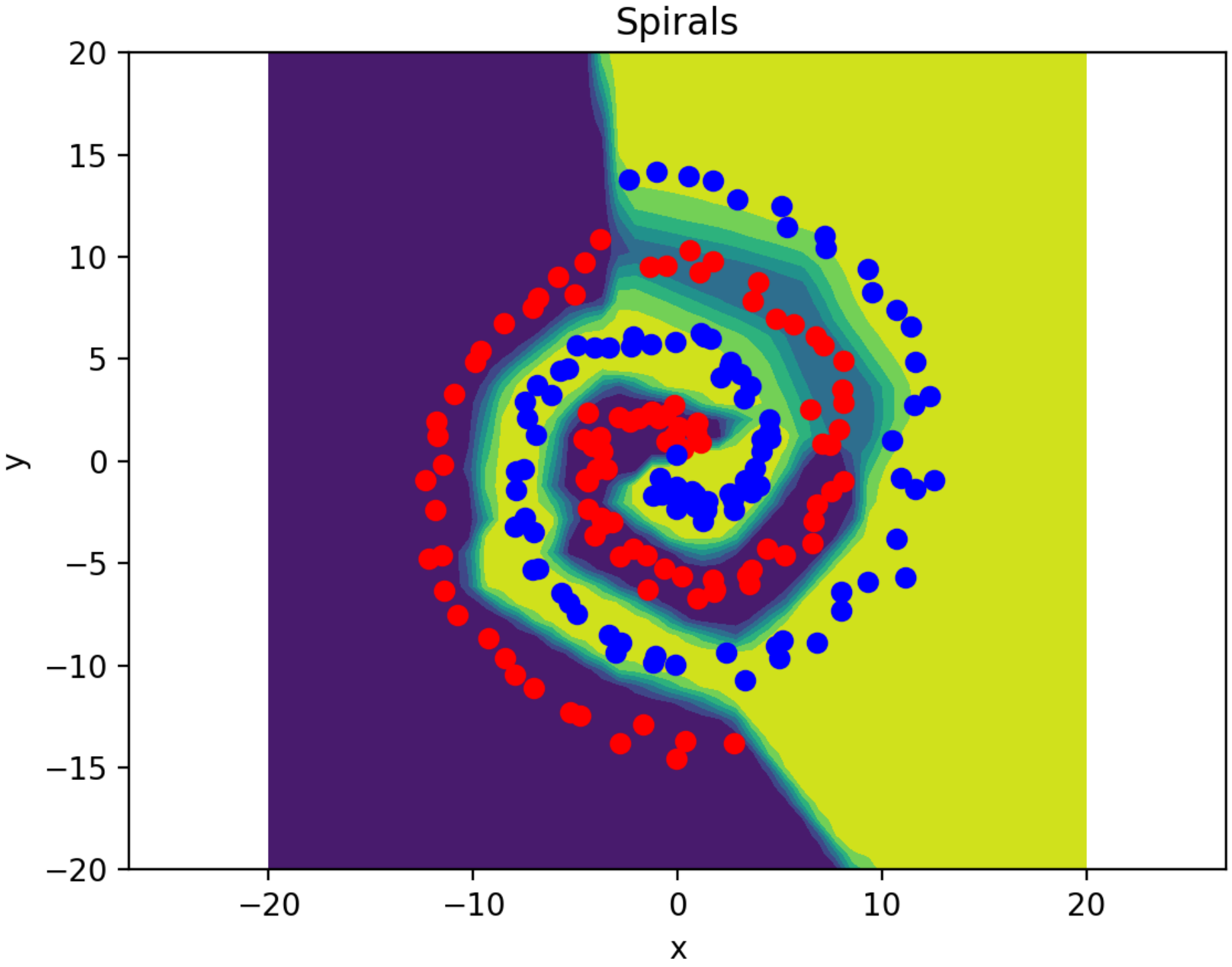
```
'''
ECE471 Selected Topics in Machine Learning - Assignment 2
Submit by Sept. 19, 10PM
tldr: Perform binary classification on the spirals dataset using a multi-layer
perceptron. You must generate the data yourself.
'''

import numpy as np
import matplotlib.pyplot as plt
from sklearn.utils import shuffle
import tensorflow as tf

# Make data
NUM_DATAPOINTS = 200
BATCH_SIZE = 20  # Must evenly divide NUM_DATAPOINTS... sorry

datapoints_per_class = NUM_DATAPOINTS // 2
t = np.linspace(1, 14, datapoints_per_class)
r = t
theta = 0.8*t

x1 = r*np.cos(theta) + np.random.normal(0, 0.5, datapoints_per_class)
y1 = r*np.sin(theta) + np.random.normal(0, 0.5, datapoints_per_class)
x2 = r*np.cos(theta + np.pi) + np.random.normal(0, 0.5, datapoints_per_class)
y2 = r*np.sin(theta + np.pi) + np.random.normal(0, 0.5, datapoints_per_class)

data = np.concatenate([np.vstack([x1, y1]).T,
                       np.vstack([x2, y2]).T])
labels = np.append(np.zeros(datapoints_per_class),
                   np.ones(datapoints_per_class))
data, labels = shuffle(data, labels)

data_batches = np.split(data, NUM_DATAPOINTS // BATCH_SIZE)
label_batches = np.split(labels, NUM_DATAPOINTS // BATCH_SIZE)
data_batches = [x.reshape(x.shape[0], -1) for x in data_batches]
label_batches = [y.reshape(y.shape[0], -1) for y in label_batches]

# Train
LAYER_1_DIM = 10
LAYER_2_DIM = 10
NUM_EPOCHS = 2000

x = tf.placeholder(tf.float32, [None, 2])
y = tf.placeholder(tf.float32, [None, 1])

W1 = tf.get_variable('W1', [2, LAYER_1_DIM], tf.float32,
                     tf.random_normal_initializer())
b1 = tf.get_variable('b1', [1, LAYER_1_DIM], tf.float32,
                     tf.zeros_initializer())

W2 = tf.get_variable('W2', [LAYER_1_DIM, LAYER_2_DIM], tf.float32,
                     tf.random_normal_initializer())
b2 = tf.get_variable('b2', [1, LAYER_2_DIM], tf.float32,
                     tf.zeros_initializer())

W3 = tf.get_variable('W3', [LAYER_2_DIM, 1], tf.float32,
                     tf.random_normal_initializer())
b3 = tf.get_variable('b3', [1, 1], tf.float32,
                     tf.zeros_initializer())

activation1 = tf.nn.relu(tf.matmul(x, W1) + b1)
activation2 = tf.nn.relu(tf.matmul(activation1, W2) + b2)
```



```
logits = tf.matmul(activation2, W3) + b3
activation3 = tf.sigmoid(logits)
clf = tf.round(activation3)

lam = 0.001
l2 = lam * tf.reduce_sum([tf.nn.l2_loss(v)
                          for v in tf.trainable_variables()])
loss = tf.losses.sigmoid_cross_entropy(y, logits) + l2
optim = (tf.train.GradientDescentOptimizer(learning_rate=0.05)
                 .minimize(loss))
num_errs = tf.reduce_sum(tf.abs(y - clf))

sess = tf.Session()
sess.run(tf.global_variables_initializer())
for _ in range(NUM_EPOCHS):
    for data_batch, label_batch in zip(data_batches, label_batches):
        sess.run(optim, feed_dict={x: data_batch, y: label_batch})
    loss_, num_errs_ = sess.run([loss, num_errs],
                                feed_dict={x: data_batch, y: label_batch})
    print(loss_, num_errs_)

# Plot
xx, yy = np.meshgrid(np.linspace(-20, 20), np.linspace(-20, 20))
points = np.array(list(zip(xx.flatten(), yy.flatten())))
zz = sess.run(activation3, feed_dict={x: points}).reshape(xx.shape)
plt.contourf(xx, yy, zz)

plt.scatter(x1, y1, c='r')
plt.scatter(x2, y2, c='b')

plt.xlabel('x')
plt.ylabel('y')
plt.title('Spirals')
plt.axis('equal')

plt.show()
```


Spirals
20
15
10
5
0
−5
−10
−15
−20
y
−20
−10
0
10
20
x