Test | Deep Learning, Spring 2019 | February 21, 2019 | Total Points = 60

This is a closed book exam. No electronics or cheat sheets are allowed. All questions carry equal points.

Name:

1. Following code loads the Pima Indian Diabetes Dataset into the `pima` variable. The shape of `pima` is shown below (as output). Write a line of numpy (Python) code to create a variable `pima_train_x` that will have the first 600 rows and first 8 columns of the data in `pima`. Write another line of Numpy (Python) to load the first 8 columns of the last 100 rows to `pima_valid_x`.

```
url = "https://raw.githubusercontent.com/badriadhikari/2019-Spring-DL/"
url = url + "master/course-content/Module1-Intro2ML/pima-indians-diabetes.csv"
pima = np.loadtxt(url, delimiter=",")
print(pima.shape)
```

```
(768, 9)
```

2. Complete the code below to display key/value pairs, one-by-one, in the Python dictionary `d`. Note that there is no comma between the key and the value pairs. Hint: Use a for loop.

```
d = {'person': 2, 'cat': 4, 'spider': 8}
```

```
person 2
cat 4
spider 8
```

3. What will the output plot of the following code look like? Draw the plot/s with appropriate axis range values. Of the three curves, one is ReLU activation, one is sigmoid, and one is linear function. Label the three curves in your plot correctly identifying which is which function. Show your input axis and output axis as well.

```
import numpy
import matplotlib.pyplot as plt

# gives numbers between -3 and 3 at steps of 0.1
x = numpy.arange(-3, 3, 0.1)

activation1 = 1 / (1 + numpy.exp(-x))
plt.plot(x, activation1)

activation2 = numpy.maximum(x, 0)
plt.plot(x, activation2)

activation3 = x
plt.plot(x, activation3)

plt.show()
```

4. In the context of a binary classification problem, below are two vectors - one with correct output labels (Y) and another one with confidences (probabilities) predicted by a neural network. Calculate the (a) accuracy, (b) precision, and (c) recall for the model.

```
True Labels:
[1.00 1.00 1.00 0.00 1.00 1.00 1.00 1.00 1.00 1.00]

Predictions:
[0.70 0.17 0.70 0.13 0.62 0.23 0.18 0.35 0.62 0.38]
```

5. What will be the output of the following code?

```
import numpy as np
x = np.array([[1,2],[3,4]])
print np.sum(x)
print np.sum(x, axis=0)
print np.sum(x, axis=1)
```

6. What will be the total number of parameters of the following neural network? Explain.

```
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
print(model.summary())
```

7. What will be the output of the following code?

```
nums = [0, 1, 2, 3, 4]
even_squares = [x ** 3 for x in nums if x % 2 == 0]
print even_squares
```

8. What will be the output of the following code?

```
v = np.array([1,2,3])
w = np.array([4,5])
print np.reshape(v, (3, 1)) * w
```

9. The code below uses a simple CNN architecture to train a model on the standard MNIST digits dataset (0 to 9 digits). Complete the code below to have the following architecture:
(a) 16 convolutional filters in the first layer, each of size 3 x 3, with sigmoid activations
(b) 4 convolutional filters in the second layer, each of size 3x3, with sigmoid activations
(c) A dense layer as output layer

```
from keras.datasets import mnist
from keras.utils import to_categorical
from keras import layers, models
(train_images, train_labels), (validation_images, validation_labels)
                = mnist.load_data()
train_images = train_images.reshape( ( 60000, 28, 28, 1 ) )
train_images = train_images.astype( 'float32' ) / 255
validation_images = validation_images.reshape( ( 10000, 28, 28, 1 ) )
validation_images = validation_images.astype( 'float32' ) / 255
train_labels = to_categorical( train_labels )
validation_labels = to_categorical( validation_labels )
model = models.Sequential()
model.add(layers.Conv2D(                                                   ))
model.add(layers.Conv2D(                                                   ))
model.add(                                                                  )
model.add(layers.Dense(                                                    ))
model.compile( optimizer='rmsprop', loss='categorical_crossentropy'
               , metrics = [ 'accuracy' ] )
model.fit( train_images, train_labels, epochs = 1, batch_size = 256 )
```

10. What does the `model.evaluate()` function return in the following code? Hint: It will return two objects.

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
print(model.summary())
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(XTRAIN, YTRAIN, epochs=15, batch_size=10)

scores = model.evaluate(XTRAIN, YTRAIN)
```

11. The output below shows the summary of a convolutional neural network constructed. How is the total number of parameters in the second convolutional layer 580? Explain in detail.

```
from keras.datasets import mnist
from keras.utils import to_categorical
from keras import layers, models

model = models.Sequential()
...
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_3 (Conv2D)            (None, 26, 26, 16)        160       
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 24, 24, 4)         580       
_________________________________________________________________
flatten_2 (Flatten)          (None, 2304)              0         
_________________________________________________________________
dense_59 (Dense)             (None, 10)                23050     
=================================================================
Total params: 23,790
Trainable params: 23,790
Non-trainable params: 0
_________________________________________________________________
```

12. What will be the output of the following code?

```
nums = range(5)
print nums[2:]
print nums[:2]
print nums[:]
print nums[:-1]
nums[2:4] = [8, 9]
print nums
```