by Thomas Hill 12/12/21

Ex. 2 -- As a mini-project, install the *keras* package and learn how to use it. Then, carry out various tasks that may be useful to your project and studies.

```
import tensorflow
```

```
import keras
```

```
import numpy as np
np.random.seed(1212)
from sklearn import datasets
iris = datasets.load_iris()
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense
```

```
x = iris.data #measurements
y = iris.target.reshape(-1,1) #data labels in a single column
```

```
model = Sequential() #define model
```

```
train_x, test_x, train_y, test_y = train_test_split(x, y, test_size=0.20) #80% as training se
```

```
model.add(Dense(4, input_shape=(4,), activation='relu'))
model.add(Dense(1, activation='softmax', name='output'))

print(model.output_shape)
```

```
     (None, 1)
```

```
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

```
print(model.summary())
```

```
     Model: "sequential"
     _________________________________________________________________
      Layer (type)                Output Shape              Param #
     =================================================================
      dense (Dense)               (None, 4)                 20
```

```
 output (Dense)              (None, 1)                 5

=================================================================
Total params: 25
Trainable params: 25
Non-trainable params: 0
_________________________________________________________________
None
```

```
model.fit(train_x, train_y, verbose=2, batch_size=5, epochs=200)
```

```
Epoch 1/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 457ms/epoch - 19ms/step
Epoch 2/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 25ms/epoch - 1ms/step
Epoch 3/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 40ms/epoch - 2ms/step
Epoch 4/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 32ms/epoch - 1ms/step
Epoch 5/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 28ms/epoch - 1ms/step
Epoch 6/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 34ms/epoch - 1ms/step
Epoch 7/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 35ms/epoch - 1ms/step
Epoch 8/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 35ms/epoch - 1ms/step
Epoch 9/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 37ms/epoch - 2ms/step
Epoch 10/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 33ms/epoch - 1ms/step
Epoch 11/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 38ms/epoch - 2ms/step
Epoch 12/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 29ms/epoch - 1ms/step
Epoch 13/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 44ms/epoch - 2ms/step
Epoch 14/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 31ms/epoch - 1ms/step
Epoch 15/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 31ms/epoch - 1ms/step
Epoch 16/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 33ms/epoch - 1ms/step
Epoch 17/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 38ms/epoch - 2ms/step
Epoch 18/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 34ms/epoch - 1ms/step
Epoch 19/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 33ms/epoch - 1ms/step
Epoch 20/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 37ms/epoch - 2ms/step
Epoch 21/200
24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 35ms/epoch - 1ms/step
Epoch 22/200
```

```
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 31ms/epoch - 1ms/step
    Epoch 23/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 33ms/epoch - 1ms/step
    Epoch 24/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 31ms/epoch - 1ms/step
    Epoch 25/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 32ms/epoch - 1ms/step
    Epoch 26/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 33ms/epoch - 1ms/step
    Epoch 27/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 37ms/epoch - 2ms/step
    Epoch 28/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 32ms/epoch - 1ms/step
    Epoch 29/200
    24/24 - 0s - loss: 0.0000e+00 - accuracy: 0.3167 - 39ms/epoch - 2ms/step
```

```
model_results = model.evaluate(test_x, test_y, verbose= 0)
```

```
print(model_results) #(Loss, Accuracy)
```

```
    [0.0, 0.4000000059604645]
```

The results above are the loss and accuracy of this model evaluated on the test set. The loss is zero, which is expected. However, the model was only 40% accurate in picking the best category. This model could be improved by allowing dropout between the Dense layers to avoid overfitting.