Explore Other Approaches

To this point, you've been largely guided through different loss functions and optimizers that can be used when training a network. It's good to explore them for yourself, as well as understand how to declare them in TensorFlow, particularly those that can accept parameters!

Note that there are generally 2 ways that we can declare these functions -- by name, in a string literal, or by object, by defining the class name of the function we want to use.

Here's an example of doing it by name:

```
optimizer = 'adam'
```

And one of doing it using the functional syntax:

```
from tensorflow.keras.optimizers import Adam

opt = Adam(learning_rate=0.001)

optimizer = opt
```

Using the former method is obviously quicker and easier, and we don't need any imports, which can be easy to forget, in particular if you're copying and pasting code from elsewhere! Using the latter has the distinct advantage of letting us set internal hyperparameters, such as the learning rate, giving us more fine-grained control over how our network learns.

You can learn more about the suite of **optimizers** in TensorFlow at https://www.tensorflow.org/api_docs/python/tf/keras/optimizers -- to this point we've seen SGD, RMSProp and Adam. Also consider reading into some of the others, in particular the enhancements to the Adam algorithm that are available.

Similarly you can learn about the **loss functions** in TensorFlow at https://www.tensorflow.org/api_docs/python/tf/keras/losses, -- to this point we've seen Mean Squared Error, Binary CrossEntropy and Categorical CrossEntropy. Read into them to see how they work, and also look into some of the others that are enhancements to these.

Experiment with different loss functions or optimizers in the tutorials. In particular tweak hyper parameters such as the learning rate to see if you can improve your models!