Iterator

Set up an Iterator to extract data from an image dataset.

Chapter Goals:

- Set up an `Iterator` to extract data from a pixel array dataset

A. Using Iterator

The way we can extract the decoded image data from our `Dataset` is through a `tf.data.Iterator`. For a more in-depth look at `tf.data.Iterator`, check out the Industry Case Study course on Educative.

We use the `get_next` function to obtain a *next-element tensor*, which is used for data extraction. Note that the next-element tensor doesn't have an actual value until we execute the iteration process using `tf.Session` (see next chapter).

Time to Code!

In this chapter you'll be working on the `get_image_data` function. This function uses an `Iterator` object to get decoded image data from a dataset.

Using the `get_dataset` function from the previous chapter (not shown), we can create our image pixel `Dataset`.

Set `dataset` equal to `get_dataset` with arguments `image_paths`, `image_type`, `resize_shape`, and `channels`.

We'll now make an `Iterator` for `dataset`.

Set `iterator` equal to `dataset.make_one_shot_iterator` with no arguments.

The one-shot iterator is a very simple iterator. It is associated with a particular dataset and only iterates through it once.

Finally, we set up the next-element tensor for extracting data from `dataset`.

Set `next_image` equal to `iterator.get_next` with no arguments.

```
import tensorflow as tf

# Get the decoded image data from the input image file paths
def get_image_data(image_paths, image_type=None, resize_shape=None, channels=0):
    # CODE HERE
    pass
```