a dataset of 1000 samples. Since 128 does not evenly divide into 1000, you'd wind up with 7 batches of 128 samples, and 1 batch of 104 samples. (7*128 + 1*104 = 1000)

In that case, the size of the batches would vary, so you need to take advantage of TensorFlow's `tf.placeholder()` function to receive the varying batch sizes.

Continuing the example, if each sample had `n_input = 784` features and `n_classes = 10` possible labels, the dimensions for `features` would be `[None, n_input]` and `labels` would be `[None, n_classes]`.

```
# Features and Labels
features = tf.placeholder(tf.float32, [None, n_input])
labels = tf.placeholder(tf.float32, [None, n_classes])
```

What does `None` do here?

The `None` dimension is a placeholder for the batch size. At runtime, TensorFlow will accept any batch size greater than 0.

Going back to our earlier example, this setup allows you to feed `features` and `labels` into the model as either the batches of 128 samples or the single batch of 104 samples.

## Question 2

Use the parameters below, how many batches are there, and what is the last batch size?

*features is (50000, 400)*

*labels is (50000, 10)*

*batch_size is 128*

### ⚠ This quiz is no longer available

This quiz is unavailable because the Nanodegree program term has come to an end