The keras.models.Sequential class is a wrapper for the neural network model that treats the network as a sequence of layers. It implements the Keras model interface with common methods like `compile()`, `fit()`, and `evaluate()` that are used to train and run the model. We'll cover these functions soon, but first let's start looking at the layers of the model.

Layers

The Keras Layer class provides a common interface for a variety of standard neural network layers. There are fully connected layers, max pool layers, activation layers, and more. You can add a layer to a model using the model's `add()` method. For example, a simple model with a single hidden layer might look like this:

```
import numpy as np
from keras.models import Sequential
from keras.layers.core import Dense, Activation

# X has shape (num_rows, num_cols), where the training data are stored
# as row vectors
X = np.array([[0, 0], [0, 1], [1, 0], [1, 1]], dtype=np.float32)

# y must have an output vector for each input vector
y = np.array([[0], [0], [0], [1]], dtype=np.float32)

# Create the Sequential model
model = Sequential()

# 1st Layer - Add an input layer of 32 nodes with the same input shape as
# the training samples in X
model.add(Dense(32, input_dim=X.shape[1]))

# Add a softmax activation layer
model.add(Activation('softmax'))

# 2nd Layer - Add a fully connected output layer
model.add(Dense(1))

# Add a sigmoid activation layer
model.add(Activation('sigmoid'))
```

Keras requires the input shape to be specified in the first layer, but it will automatically