Abstract Environment

Let's start by defining an abstract `Environment` object that encodes the state of a system in a list of NumPy objects (Example 8-1). This `Environment` object is quite general (adapted from DeepChem's reinforcement learning engine) so it can easily serve as a template for other reinforcement learning projects you might seek to implement.

Example 8-1. This class defines a template for constructing new environments

```
class Environment(object):
  """An environment in which an actor performs actions to accomplish a task.

  An environment has a current state, which is represented as either a single NumPy
  array, or optionally a list of NumPy arrays.  When an action is taken, that causes
  the state to be updated.  Exactly what is meant by an "action" is defined by each
  subclass.  As far as this interface is concerned, it is simply an arbitrary object.
  The environment also computes a reward for each action, and reports when the task
  has been terminated (meaning that no more actions may be taken).
  """

  def __init__(self, state_shape, n_actions, state_dtype=None):
    """Subclasses should call the superclass constructor in addition to doing their
       own initialization."""
    self.state_shape = state_shape
    self.n_actions = n_actions
    if state_dtype is None:
      # Assume all arrays are float32.
      if isinstance(state_shape[0], collections.Sequence):
        self.state_dtype = [np.float32] * len(state_shape)
      else:
        self.state_dtype = np.float32
    else:
      self.state_dtype = state_dtype
```

Tic-Tac-Toe Environment

We need to specialize the `Environment` class to create a `TicTacToeEnvironment` suitable for our needs. To do this, we construct a *subclass* of `Environment` that adds on more features, while retaining the core functionality of the original *superclass*. In Example 8-2, we define `TicTacToeEnvironment` as a subclass of `Environment` that adds details specific to tic-tac-toe.

Example 8-2. The TicTacToeEnvironment class defines a template for constructing new tic-tac-toe environments

```
class TicTacToeEnvironment(dc.rl.Environment):
  """
  Play tictactoe against a randomly acting opponent
```