Coding Assignment 3

EC 400

October 28, 2021

Setup. Using code from the last coding assignment, run the command

```
python pacman.py -p PacmanQAgent -x 2000 -n 2010 -l smallGrid
```

You may need to replace "python" with "python3," or if you are using Anaconda, you will need to put these arguments in under "Run," and then "Configuration per file."

After some training, this should produce a video of Pacman playing on a small grid. Training here means that the agent will run about 2,000 episodes of Q-learning; each "episode" consists of generating state-action pairs until termination[1]. What is then shown are the last ten episodes of Q-learning from the batch of 2,000.

This code should solve Pacman the vast majority of the time provided your last coding assignment was correctly done. If not, you may wish to download the solutions to the last coding assignment from blackboard and begin from there.

Assignment. Q-learning cannot solve larger Pacman boards because the number of states gets to be too large. In this coding assignment, you will write code for Pacman on a large board using linear approximation. Specifically, your assignment is to write an implementation of the ApproximateQAgent class in the file qlearningAgents.py.

Your first task is to write the function getQvalue in this class. Once you fill in this code, this will overwrite the getQvalue function you wrote earlier.

Note: you should be able to get a dictionary of features of a state-action pair by writing

```
feature = self.featExtractor.getFeatures(state, action)
```

Given a state and action pair, this returns a collection of features $f_i(s, a)$. Your approximation of the Q-value function should be

$$Q(s, a) = \sum_i f_i(s, a) w_i$$

where w_i are your weights.

Your second task is to write the function "update" in the ApproximateQLearning class.

Once your code works, you can quickly test it by running it on a small instance by running

```
python pacman.py -p ApproximateQAgent -x 2000 -n 2010 -l smallGrid
```

However, you code should also work on the larger Pacman board, though it will take longer to train; you should be able to test this by running

```
python pacman.py -p ApproximateQAgent -a extractor=SimpleExtractor -x 50 -n 60 -l mediumClassic
```

[1] Recall, from the solution of one of your homework assignments, then when we restart and create a new sample path, we do not forget the Q-learning estimates we have already produced