Anthony Galczak
CS238 Project 2 README

Algorithm
I implemented two major algorithms during the course of this project.

Value Iteration

- Iterate over the data set and pre-compute transition probabilities T and reward function R
- Loop over every state in possible states (10*10 cells)
    - Look at every action and find the one that maximizes bellman equation.
    - Save off the best action and its value into a map.
- End after k iterations over all states.

Normally in policy iteration, we would set an explicit stopping criteria using the Bellman residual, but I found this to be unnecessary. We only need ~20 iterations to get a pretty ideal answer for small. I found value iteration to be fairly intractable for medium and large.
By far, the T and R calculations are the most expensive step in my algorithm.

Q-Learning

- Loop over every item in our data set
    - Increment N count upon encountering a (state, action) pair.
    - Find the best action from the next state, save it for Q(s,a) update.
    - Use Q-Learning update step to calculate new Q(s,a) value.
        - Use 1/N(s,a) for learning rate
    - Update Q map with new Q(s,a)
- End after k iterations over all rows in data.

Some notes about my implementation:
I found out, empirically, that using a learning rate of 1 (no learning rate) was better for medium. Thus, I have a parameter for q-learning on whether I use the 1/N learning rate. This learning rate was inspired by a Piazza post by Prof. Kochenderfer.

We have 312020 total states and approximately only 500 given to us. I toyed around with approximation and didn't get anywhere. So, I instead toyed around with a default policy for states we haven't seen in our data. I found that a default policy of selecting 1 was best after trying random values and other defaults.

I use gamma = 1 for medium and gamma = 0.95 for large as per the prompt.

Analysis
A small comment about runtime is the difference between using iterrows with a pandas dataframe vs. using itertuples. Big runtime savings by using itertuples.

Dataset	**Algorithm**	**Runtime**	**Score**
Small	Value Iteration	6s (20 iters)	27.4
Medium	Q-Learning	38s (100 iters)	95.0
Medium	Q-Learning	3m10s (500 iters)	102.1
Medium	Q-Learning	303m17s (50000 its)	102.1
Large	Q-Learning	41s (100 iters)	7460.7
Large	Q-Learning	3m15s (500 iters)	7747.7
Large	Q-Learning	35m45s (5000 iters)	7807.3
Large	Q-Learning	313m17s (50000 its)	8003.2

I was able to get good scores with 100 iterations of Q-Learning. Runtime under 1 minute also seems fairly reasonable, but by running tons of iterations I was getting close to an upper bound for straight Q-Learning scores.