General constraints for code submissions Please adhere to these rules to make our and your life easier! We will deduct points if your solution does not fulfill the following:

- If not stated otherwise, we will use exclusively Python 3.5.
- If not stated otherwise, we expect a Python script, which we will invoke exactly as stated on the exercise sheet.
- Your solution exactly returns the required output (neither less nor more) – you can implement a `--verbose` option to increase the verbosity level for developing.
- Add comments and docstrings, so we can understand your solution.
- (If applicable) The `README` describes how to install requirements or provides addition information.
- (If applicable) Add required additional packages to `requirements.txt`. Explain in your `README` what this package does, why you use that package and provide a link to it's documentation or GitHub page.
- (If applicable) All prepared unittests have to pass.
- (If applicable) You can (and sometimes have to) reuse code from previous exercises.

Now that you have learned about the application of reinforcement learning for learning to learn and algorithm control, you will implement a basic RL agent yourself.

1. **Q-Learning** [9 points]

   We present you a simple environment (based on the OpenAI gym standard), in which your agent has to learn to approximate a Sigmoid function.

   (a) Implement a tabular function approximator that is able to handle float data, by mapping float state feature to the closest integer. [2pt.]

   (b) Implement the Q-learning algorithm. Train your agent on the provided environment in its default configuration for 10 000 training episodes. Your agent will have to learn a policy of length 11 with action space $\{0, 1, 2\}$. The optimal policy will result in a reward of roughly 10.16 and looks like $0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 2$. [5pt.]

   If your agent at first does not produce the optimal policy play around with the learning rate α and the exploration factor ϵ. In your PDF report all learning rates and exploration factors you tried and the resulting policies.

   In your own words explain how the learning rate and the exploration factor influence the Q-learning on this deterministic environment.

   (c) In the same folder you called `main.py` from, the results of training will be stored. Plot the true reward against the expected reward for all learning rates and exploration factors you tried and add them to your PDF. [2pt.]

2. **Feedback** [Bonus: 0.5 points]

   For each question in this assignment, state:

   - How long you worked on it.
   - What you learned.
   - Anything you would improve in this question if you were teaching the course.

This assignment is due on 12.07.19 (10:00). Submit your solution for the tasks by uploading a PDF to your groups BitBucket repository. The PDF has to include the name of the submitter(s).