Assignment No. 7
EECS 658
Introduction to Machine Learning
Due: 11:59 PM, Tuesday, November 26, 2024
Submit deliverables in a single zip file to Canvas
Files in other formats (e.g., .tar) will not be graded
Name of the zip file: FirstnameLastname_Assignment7 (with your first and last name)
Name of the Assignment folder within the zip file: FirstnameLastname_Assignment7

Deliverables:

1. Copy of Rubric7.docx with your name and ID filled out (do not submit a PDF)
2. Python source code.
3. Screen print showing the successful execution of your Python code. (Copy and paste the output from the Python console screen to a Word document and PDF it).
4. Plot of the error value vs. t with ϵ labeled for Part 1. (This may be included in your screen print).
5. Answer to Part 1, Question 1.
6. Plot of the error value vs. t with ϵ labeled for Part 2. (This may be included in your screen print).
7. Answer to Part 2, Question 2.
8. Answer to Part 2, Question 3.

Assignment:

- For both parts, we are going to use a modified version of the Gridtask World described in the lectures. The only difference is that the grid is a 5-by-5 grid instead of a 4-by-4 grid:

	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	

- Everything else is the same:
    - The goal of the Gridworld Task is for a robot, starting at any square in the grid, to move through the grid and end up in a termination state (grey squares) which ends the game.
    - Each grid square is a state (s).
    - The actions (a) that can be taken are up, down, left, or right.
    - The rewards (r) are:
        - -1 on all transitions
        - 0 terminal state
        - -1 when it hits a wall (no transition, but still a reward)

- Assume:
  - $p(r_{t+1}|s_t,a_t) = 0.25$
  - $p(s_{t+1}|s_t,a_t) = 0.25$
  - $\gamma = 1$

Part 1: RL Policy Iteration Algorithm

- Write a Python program that uses the RL Policy Iteration algorithm to develop an optimal policy (π^*).
- The program should display the optimal policy (π^*) as a Python array similar to this:

```
[[0.0 -0.9 -0.8 -0.7 -0.6]
 [-0.9 -0.9 -0.8 -0.7 -0.6]
 [-0.9 -0.9 -0.8 -0.7 -0.6]
 [-0.9 -0.9 -0.8 -0.7 -0.6]
 [-0.9 -0.9 -0.8 -0.7 0.0]]
```

- The policy the robot follows, no matter what square it is in, is to go to the square next to it with the highest value.
- If it follows this policy, it will end up in one of the termination squares in the least amount of moves.
- Note: The values in this array are NOT the ones you will get.
- The program should print out the policy array with the iteration number for iteration 0 (the initial values), iteration 1, iteration 10, and the final iteration.
- Determine a method for deciding when the Policy Iteration algorithm has converged (see the lecture slides for some ideas).
- Plot the error value (e.g., $|q_t(s,a) - q_{t-1}(s,a)|$) vs. t and label ϵ.
- Question 1: Explain the convergence method and why you picked it. There is no wrong answer. You will get credit for any method you pick as long as it converges and you provide a reasonable explanation of why you picked it.

Part 2: RL Value Iteration Algorithm

- Write a Python program that uses the RL Value Iteration algorithm to develop an optimal policy (π^*).
- The program should display the optimal policy (π^*) as a Python array similar to this:

```
[[ 0 -1 -2 -3 -4]
 [-5 -6 -7 -8 -9]
 [-5 -6 -7 -8 -9]
 [-5 -6 -7 -8 -9]
 [-5 -6 -7 -8 0]]
```

- The policy the robot follows, no matter what square it is in, is to go to the square next to it with the highest value.
- If it follows this policy, it will end up in one of the termination squares in the least amount of moves.
- Note: The values in this array are NOT the ones you will get.
- The program should print out the policy array with the iteration number for iteration 0 (the initial values), iteration 1, iteration 2, and the final iteration.

- Determine a method for deciding when the Value Iteration algorithm has converged.
- Plot the error value (e.g., $|q_t(s,a) - q_{t-1}(s,a)|$) vs. t and label ϵ.
- Question 2: Explain the convergence method and why you picked it. There is no wrong answer. You will get credit for any method you pick as long as it converges and you provide a reasonable explanation of why you picked it.
- Question 3: Compare the convergence times between using Policy Iteration and Value Iteration and give reasons for why you would ever use Policy Iteration.

Rubric for Program Comments		
Exceeds Expectations (90-100%)	**Meets Expectations (80-89%)**	**Unsatisfactory (0-79%)**
Software is adequately commented with prologue comments, comments summarizing major blocks of code, and comments on every line.	Prologue comments are present but missing some items or some major blocks of code are not commented or there are inadequate comments on each line.	Prologue comments are missing all together or there are no comments on major blocks of code or there are very few comments on each line.

Adequate Prologue Comments:

- Name of program contained in the file (e.g., EECS 658 Assignment 1)
- Brief description of the program, e.g.,
  - Check versions of Python & create ML "Hello World!" program
- Inputs (e.g., none, for a function, it would be the parameters passed to it)
- Output, e.g.,
  - Prints out the versions of Python, scipy, numpy, pandas, and sklearn
  - Prints out "Hello World!"
  - Prints out the overall accuracy of the classifier.
  - Prints out the confusion matrix.
  - Prints out the P, R, and F1 score for each of the 3 varieties of iris.
- All collaborators
- Other sources for the code ChatGPT, stackOverflow, etc.
- Author's full name
- Creation date: The date you first create the file, i.e., the date you write this comment

Adequate comments summarizing major blocks of code and comments on every line:

- Provide comments that explain what each line of code is doing.
- You may comment each line of code (e.g., using //) and/or provide a multi-line comment (e.g., using /* and */) that explains what a group of lines does.
- Multi-line comments should be detailed enough that it is clear what each line of code is doing.
- Each block of code must indicate whether you authored the code, you obtained it from one of the sources listed in the prolog, or one of your collaborators authored the code, or if it was a combination of all of these.

Collaboration and other sources for code:

- When you collaborate with other students or use other sources for the code (e.g., ChatGPT, stackOverflow):
    - Your comments must be significantly different from your collaborators.
    - More scrutiny will be applied to grading your comments in particular explaining the code “in your own words”, not the source’s comments (e.g., ChatGPT’s comments).
- Failure to identify collaborators or other sources of code will not only result in a 0 on the assignment but will be considered an act of Academic Misconduct.
- Students who violate conduct policies will be subject to severe penalties, up through and including dismissal from the School of Engineering.