Assignment 1 Problem Statement

SoC: RL for NLP

We will focus on ε-greedy algorithms, which the textbook must have introduced you to. If you have gone through the probability texts, you must by now be familiar with distributions like Poisson, Exponential and Gaussian(Normal).

We are going to apply these concepts in a coding task. But before I show you the problem, a few things. This task is expected to be done in Python. There are a few libraries you must be familiar with before you start coding. While the implementation aspect will be rather simple, this week's primary target is to get you familiarized with NumPy and Matplotlib.

In the following week, the problem statement will be joint for Week3 and Week4 and that will be a bulky coding exercise. It is imperative that you are familiar with these libraries before you enter those weeks.

NumPy

This library is central to almost any proper Python code. You can find it documented here. The most important feature this library brings to the table is matrices(in the form of NumPy arrays). In the context of internships, many companies even target knowledge of NumPy arrays during tests.

This week, you will use the three aforementioned probability distributions for your task, Gaussian,Exponential and Poisson. There is a lot more to NumPy than just random number generation, but for now we restrict our attention to this domain. When matrices become a necessary tool for us, we will come back to NumPy again.

Matplotlib

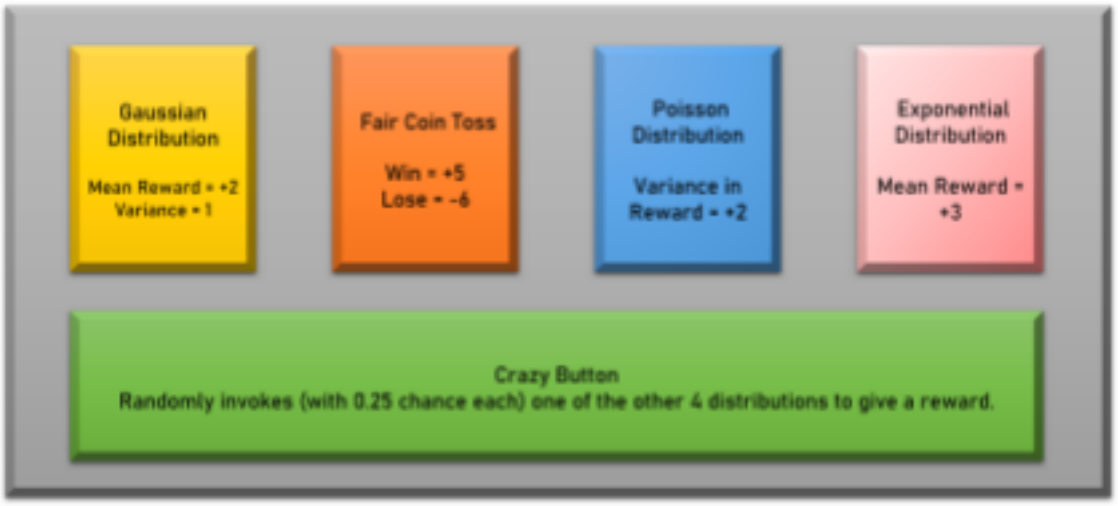
This library is known for producing elegantly formatted graphs. It can be frustrating in the beginning to figure out matplotlib, but it is worth the patient work. Links: Installation, General Tutorials.

We will use this to produce certain graphs for progress visualization, just like the textbook does.

If more resources are needed on any of the above 2 libraries, let me know.

The problem is to look at a multi armed bandit. If you haven't yet realized, multi-armed bandits are a MDP that always offers you the same choice of actions at every time-step.

This bandit has 5 arms(i.e., at each time-step you are offered to choose between 5 options as shown in the figure. The agent doesn't know about the actual probability distributions, the agent can only see 5 buttons and has to figure out an optimal strategy.



The image, I hope, is self-explanatory. At each time-step the agent is faced with choosing one of these actions.

One episode is supposed to be 100 time-steps long. You must implement an epsilon-greedy agent. Play around with the values of ε as 0.1, 0.01 and 0. Plot graphs of “Reward at the end of episode” vs “Episode” for each of these three epsilon greedy algorithms. Simulate at least 1K episodes. If possible, plot the 3 curves in the same graph.

A sample algorithm will be released soon after this statement is released.

Submission Details:
Submit only two files `bandit.py` and `desc.txt` on Google Classroom.

Submit `desc.txt` describing your code implementation and any sources you referred to that were not explicitly given by me. This includes GPT and stackoverflow sources.

Feel free to use the internet or talk to me; this is not a semester course; it's a learning project!