Faculty of Science

Course:	CSCI 4030U Big Data Analytics
Lab:	#9-10
Topic:	Online Learning
Due Date:	Apr. 2 at 11:59pm

Objective:
The objective of this project is to explore Online Learning: designing and analyzing algorithms to minimize regret, to guarantee that the algorithm is "learning" something.

Resources:
A_Modern_Introduction_to_Online_Learning_v5.pdf
Note this textbook is an **extremely** hard read, just reading the first two pages will suffice for this lab.

Programming Language
We will be using Python exclusively for labs 9-10.

Tasks and Schedule

1. Introduction to Online Learning and Regret (Lab 9/10).
    - Implement an algorithm that achieves sublinear regret, implying it is learning effectively.

Starter Code
Please download **lab_9-10_starter_code.py** here:
https://drive.google.com/file/d/1oZKd_s9dCsNv4mtezCR1h7ckyNxPAnB0/view?usp=sharing

The starter code details exactly what is required for these two labs, read it carefully!

Online Learning (in a nutshell but **please read**)

Imagine the following repeated game:
In each round $t = 1, \ldots, T$

- An adversary choose a real number in $y_t \in [0, 1]$ and he keeps it secret;
- You try to guess the real number, choosing $x_t \in [0, 1]$;
- The adversary's number is revealed and you pay the squared difference $(x_t - y_t)^2$.

Basically, we want to guess a sequence of numbers as precisely as possible. To be a game, we now have to decide what is the "winning condition". Let's see what makes sense to consider as a winning condition.

First, let's simplify the game a bit. Let's assume that the adversary is drawing the numbers i.i.d. from some fixed distribution over [0, 1]. However, he is still free to decide which distribution at the beginning of the game. In fact, the numbers can be generated in any way, even in an adaptive way based on our strategy. Indeed, they can be chosen adversarially, that is explicitly trying to make us lose the game. This is why we call the mechanism generating the number the adversary.

Now we will win the game if:

$$\text{Regret}_T := \sum_{t=1}^{T}(x_t - y_t)^2 - \min_{x\in[0,1]} \sum_{t=1}^{T}(x - y_t)^2$$

grows sublinearly with T. The quantity above is called the Regret, because it measures how much the algorithm regrets for not sticking on all the rounds to the optimal choice in hindsight. We will denote it by Regret_T. In many cases, the goal of online learning algorithms is to minimize the regret over time, meaning the learner aims to minimize the difference between its performance and the best possible performance in hindsight.

Sublinear regret suggests that the learning algorithm is effectively adapting to the environment and learning optimal or near-optimal strategies over time. It indicates that the algorithm is able to make better decisions as it gathers more information.

In labs 9/10 you will implement an algorithm where its Regret grows sublinearly, when you run the main method in the **lab_9-10_starter_code.py**, you will see a graph generated that shows the Regret over the number of rounds.

If your algorithm is learning effectively, the graph should look logarithmic in nature. An example is provided below: