ESE545-Project3

April 14, 2020

Due: April 29, 2020 at 11:59pm

- You are required to solve the problems using Python. You are allowed to use only python standard libraries, numpy, pandas, and scipy.

- All the algorithms mentioned in the assignment should be written from scratch without the help of any optimization libraries.

- You are allowed to work in groups up to 3 members. You should submit your code with a brief report containing responses to each part.

- You should submit one code per group but each individual should submit their own report.

- Submit the Report and Code separately on Gradescope. For code, upload the zip file.

- Note that this Project is divided into 2 modules. The 1st module is based on recommendation systems and the 2nd module is based on distance-based clustering.

Module 1: Recommendation System

(5 points) Problem 1 For this module, we will be designing a recommendation system to recommend movies to the users. Please download and load the dataset from Canvas (recommendationMovie.csv). Each row of the dataset corresponds to a movie and each column corresponds to a day. On a given day, 1 signifies that some user has given high rating to that movie and similarly, 0 signifies low or no rating.

The goal of this project is to recommend movies in an online fashion. So, every day the system recommends one movie to the user and on the basis of the data, the system receives a reward (either 0 or 1). Using this information, formulate it as a multi-armed bandit problem. Your goal should be to achieve the lowest regret possible over the given time period (i.e. the number of days).

(30 points) Problem 2 Design a strategy to solve your multi-armed bandit problem with only partial feed-back. In other words, you may only observe the reward for the movie you choose to serve at any given time. You may use any methods we have seen so far in the course to solve this problem. We expect you to experiment with stochastic and non-stochastic algorithms and compare their performances. The analysis should include the loss (algorithm's loss along with the optimal) and regret curves.

(40 points) Problem 3 Design a strategy to solve your multi-armed bandit problem with full feedback, i.e., with observing all rewards after you serve the recommendation for that round. We expect you to experiment with stochastic and non-stochastic algorithms. Perform the same analysis as you did in **Problem 2**. Additionally, compare the results from partial and full feedback strategies.

(15 points) Problem 4 After designing the multi-armed bandits from **Problem 3** (full feedback and non-stochastic setting), we will now analyze how the recommendation probabilities for movies changes over time. For our analysis, we will focus on top 10 highly rated movies, you can devise any metric to pick these movies. After making the movie selection, plot the recommendation probabilities for these movies over the given time period (ten plots in one graph). Reason these plots and report your observations.

(20 points) Problem 5 (Bonus) The 3 groups with the smallest regret for partial feedback will receive bonus points. You must explain your implementation in detail and share your reasoning behind any assumptions/parameters.

Module 2: Clustering

(5 points) Problem 1 For this module, we will be working with the distance-based clustering algorithms to cluster movies based on their attributes. We will be using a subset of the IMDb title.basic dataset, please download and load the dataset from Canvas (Movies.csv). Each row of the dataset corresponds to a movie and each column corresponds to a unique attribute of that movie.

The goal of this module is to cluster the movies on their attributes such that similar movies are clustered together. Using this information, formulate a clustering problem. You may change the data representation or create new features from this data, if you wish.

(25 points) Problem 2 Implement an online version of the k-means clustering algorithm. For this problem, your initial cluster centroids should be chosen randomly. Once the cluster centroids are chosen, the next step is to iteratively improve the solution until a local optimum is achieved.
As discussed in the lectures, Lloyd's heuristic is one approach, but this requires taking a pass over the entire dataset each time, which is not feasible here. Instead implement mini-batch k-means, which will adjust the centroids by taking a subset of the data each time.

(30 points) Problem 3 In this setting, the optimization problem being solved is highly non-convex, and many local minima exist; hence, the choice of the initial centroids can dramatically influence results. For this problem, implement the k-means++ initialization as discussed in class for the online mini-batch algorithm.

For each initialization method mentioned in **Problem 2** and **Problem 3**, generate plots showing the mean, minimum, and maximum distance to the cluster centroids for the dataset, and run this for several values of the number of centroids k, ranging from $k = 5, \dots, 500$. Which k has the lowest error, and how does that change based on the initialization? Which initialization is best?