Project
**Data Mining: Learning from Large Data Sets**
FS 2015

**LAS Group, Machine Learning Institute**
Dept. of Computer Science, ETH Zürich
**Prof. Dr. Andreas Krause**
Web: las.ethz.ch/courses/datamining-f15/
**Email questions to:**
Baharan Mirzasoleiman <baharanm@inf.ethz.ch>
Hamed Hassani <hamed@inf.ethz.ch>

# Project 4, Dec 3, 2015

# Explore-Exploit tradeoffs in Recommender Systems

## 1 Introduction

The goal of this task is to learn a policy that explores and exploits among available choices in order to learn user preferences and recommend news articles to users. For this task we will use real-world log data shared by Yahoo!. The data was collected over a 10 day period and consists of 5 million log lines that capture the interaction between user visits and 271 news articles, one of which was randomly displayed for every user visit. In each round, you are given a user context and a list of available articles for recommendation.

Your task is to then select one article from this pool. If the article you selected matches the one displayed to the user (in the log file), then your policy is evaluated for that log line. Otherwise, the line is discarded. Since the articles were displayed uniformly at random during the data collection phase, there is approximately 1 in 20 chance that any given line will be evaluated.

## 2 Input and Output Specification

The 1000 given lines of log in *log.txt* are purely for testing the syntax and expected behavior of your code and is not reflective of the actual values in the real log. Every call to your policy contains the following as parameters:

- timestamp: integer
- user features: 6 dimensional vector of doubles
- available articles: list of article IDs that are available for selection

In addition, you are given access to a separate file (*articles.txt*) that consists of the six dimensional feature vector of every article. Each line is formatted as follows:

- ArticleID feature1 feature2 feature3 feature4 feature5 feature6

The file in the handout is *different* from the one on the server.

## 3 Code

In the handout for this project, you will find the 1000 lines of log data (`log.txt`), and article features for articles present in the set of 1000 log lines (`articles.txt`).In addition, there are two Python files (`policy` and `evaluator`). Your task is to complete the functions `recommend` , `update` and `set_articles` in the policy file. The provided evaluator will call your `recommend` function for every line in `log.txt` and will call your `update` only if your policy was evaluated. Please ensure that your policy always returns an article from the given list. Failing to do so will result in an exception and the execution will then be aborted. If your chosen article matches the

displayed article in the log, the result of choosing this article (click or no click) is fed back to your policy and you have a chance to update the current model accordingly. This is achieved via the `update` method.

# 4 Evaluation and Grading

An iteration is evaluated only if you have chosen the same article as the one chosen by the random policy used to generate the data set. Upon submission, we will run your algorithm on the entire log file. In addition, all the articles will be provided to the both the `evaluation` and `policy` scripts containing the features of all the articles present in the evaluation. As the articles were picked uniformly at random 5M/20 = 250k times. Your final score is the actual CTR achieved by your policy. Only those selections of your policy that match the log line are counted as an impression.

We will compare the CTR of your submission to two baseline solutions: a weak one (called "baseline easy") and a strong one (called "baseline hard"). We denote the scores of these baselines QBE and QBH respectively.

Performing better than the weak baseline will give you 50% of the grade, and matching or exceeding the strong baseline on the **test set** will give you 100% of the grade. This allows you to check if you are getting at least 50% of the grade by looking at the ranking. If your score is in between the baselines, the grade is computed as:

$$\text{Grade} = \left(1 - \frac{\text{CTR}_{BH} - \text{CTR}}{\text{CTR}_{BH} - \text{CTR}_{BE}}\right) \times 50\% + 50\%$$

**The number of submissions per team is limited to 25. Time limit per submission is 15 minutes. There is also a memory limit of 2G.**

## 4.1 Report

You should update the template given with the first task and add a section titled "Explore-Exploit tradeoffs in Recommender Systems". The maximum length of a task report is 2 pages (which means that the final report will contain a maximum of 8 pages). Submit the final report as the report for this task.

## 4.2 Deadline

The submission system will be open from **Sunday, 6.12.2015, 10:00** until **Sunday, 20.12.2015, 23:59:59**.