Recommender Systems Handout

lucic@inf.ethz.ch

April 17, 2013

1 Code template

The code template contains two Java packages:

1. `package myPolicy` - contains only one class, `MyPolicy` which you have to fill in as described in the problem definition.
2. `package org.ethz.las.bandit` - the changes to files in this package will not be submitted and the original package will be used for evaluation. All changes that you want to introduce have to be in the `MyPolicy` package.

Here is a brief description of the most important classes found in `org.ethz.las.bandit`:

- `Article` - Contains the article ID.
- `ArticleFeatures` - Contains the article ID and an array of features. It is separate from the article class since it should be read once and stored, while the article gets instantiated for every log line.
- `User` - contains the timestamp of the visit and an array of user features.
- `YahooLogLine` - Contains `User`, `Article` and `Click` information.
- `YahooLogLineReader` - reads the log file line by line and creates instances of `YahooLogLine`.
- `Evaluator` - Generates log lines one by one and calls your policy to get the action for a given line. It then calls `MyEvaluationPolicy.evaluate` to update the number of evaluations and clicks (and log those to standard output). It then possibly calls `MyPolicy.updatePolicy` to allow you to update your policy based on the feedback.
- `MyEvaluationPolicy` - Evaluates the Action (chosen article) for a log line. If the chosen article matches the one in the log line it updates the number of evaluations and the number of clicks.
- `Main` - Initializing up log line reader/generator, the evaluation policy and your contextual bandit policy. It then instantiates the evaluator and calls `evaluator.runEvaluation` .

Other useful classes:

- `org.ethz.las.bandit.RandomPolicy` - Example of a policy that chooses a random article in every iteration.
- `org.ethz.las.bandit.logs.RandomGenerator` - Generates random log lines.
- `org.ethz.las.bandit.ArrayHelper` - Convert arrays of strings to arrays of doubles/ints. Can be useful when loading the articles.

2 Importing to Eclipse

If you don't use Eclipse, you can directly use Ant from the root of the project. Otherwise, follow these instructions:

1. Unpack the archive locally.
2. In eclipse go to File → New project → Java project from existing Ant buildfile → browse to the `build.xml` file in the folder where you unpacked the archive.
3. The (dummy) log file and features file are present in the `data` folder. You can add these files to the RunConfiguration arguments or hard-code the path to these files in org.ethz.las.bandit.Main. We will make sure that for evaluation we pass the correct files around. You have to implement only the `MyPolicy` class and submit **ONLY** the `myPolicy` package in the JAR. Instructions on how to do that are in the **Ant** section
4. Feel free to add new classes but make sure to add them only in the `mypolicy` folder.
5. You can include external libraries as well by copying the JAR file to the `lib` folder of your project (**IMPORTANT!**) and importing the JAR file in Eclipse using:
   Project → Properties → Java Build Path → Libraries → Add JARs.
6. **IMPORTANT!**: Be sure to remove all statements that print to standard output. Submissions that do not adhere to this rule may be penalized and marked as failed.

3 Ant

Apache Ant is a software tool for automating software build processes. It is similar to Make but is implemented using the Java language, requires the Java platform, and is best suited to building Java projects. For more info check `http://en.wikipedia.org/wiki/Apache_Ant`. For installation instructions check `http://ant.apache.org/manual/index.html`. The `build.xml` defines all tasks that are needed for this project (you don't need to change this file). If you are using Eclipse, compiling and running is handled by Eclipse. **However, you will need to run `ant jar` in order to create the submission file.** Here are some commands that you may find useful:

1. `ant compile` will compile your code. The location of the source files, the classpath and external libraries folder are all defined in the build file.
2. `ant run` will run the `org.ethz.las.bandit.Main` class.
3. `ant jar` will package the `myPolicy` package as a JAR **which you need to submit!**

4 Linear Algebra for Java

A good Java Linear Algebra library is provided in `lib` folder. For more information about the library check `http://mikiobraun.github.io/jblas/`. You are free to use any other library, just remove this one from `lib`. **If the library is not in the said folder, it will not be packaged together with the rest of your files and the submission will fail.**