# Assignment 3: LSH and Recommendation Systems

Spring 2016

**100 points**

Due: 11:59pm, 3/22/2016

In this assignment, we will consider making movie recommendation and use LSH to efficiently find similar users for recommendation.

## Problem 1: Finding similar users (50 points)

Suppose there are 100 different movies, numbered from 0 to 99. A user is represented as a set of movies. Jaccard coefficient is used to measure the similarity of sets.

Apply minhash to obtain a signature of 20 values for each user. Recall that this is done by permuting the rows of characteristic matrix of movie-user matrix (i.e., row are movies and columns represent users).

Assume that the i-th hash function for the signature: h(x,i) = (3x + i) % 100 (i = 1…20), where x is the original row number in the matrix.

Apply LSH to speed up the process of finding similar users, where the signature is divided into 5 bands, with 4 values in each band. Name your script for this problem `firstname_lastname_Lsh.py`.

### Input format

You are provided with a json file where each line represents a user (with ID "U1" for example) and a list of movies the user has watched (e.g., movie #0, 12, 45). All movies are represented by integers in sorted order.

```
["U1", [0,12,45]]
["U2", [2,3,5,99]]
…
```

You can test your solution to this problem using the sample data file input1.json/input2.json:

`Python firstname_lastname_Lsh.py input1.json`
(or: `Python firstname_lastname_Lsh.py input2.json`)

### Output format

The output of this problem should be a set of candidate pairs that are found similar by LSH, in following format:

```
["U1", "U3"]
["U14", "U8"]
…
```

The order among pairs is not important; however, you should sort every pair in alphabetic order.

Refer to Lsh_output1.json/Lsh_output2.json for sample output from Problem 1. Remember the output from this problem will be given as the input to Problem 2. Also as explained in Assignment 2, don't save the output to any file; but for the purpose of testing your code for Problem 2, you may redirect it to your own p1_output file.

## Problem 2: Making recommendations (50 points)

Based on the LSH result, for each user U, find top-5 users who are most similar to U (by their Jaccard similarity), and recommend movies that have been watched by at least 3 of these users (but not by U yet) to U. Name your script for this problem `firstname_lastname_Recommendation.py`.

**Note:**

1) If one has fewer than 5 similar users found by LSH, then all of them are considered to be "Top-5".
2) If there's no movie that has been watched by at least 3 of U's similar users, then no recommendation is needed.

### Input format

There are 2 input files to this problem:

1) The same input file of Problem 1, with each user and his movie list in a line;
2) The output from Problem 1, with each line represents a candidate similar pair.

You can test your solution to this problem using this command:

```
Python firstname_lastname_Recommendation.py input1.json
Lsh_output1.json
```
(or: `Python firstname_lastname_Recommendation.py input2.json Lsh_output2.json`)

## Output format

For each user, output the recommended movies as follows.

```
["U1", [5,20,38]]
["U2", [1,5,12]]
…
```

The order among users (i.e. each line) is not important; however, for each user, the recommended movie list should be sorted in ascending order.

Refer to final_output1.json/final_output2.json for sample output from Problem 2.

## Submission

Submit the following 2 Python scripts:

```
firstname_lastname_Lsh.py
firstname_lastname_Recommendation.py
```

DO NOT make them into .zip.

## General Instructions:

1. **Please submit your Python code with clear structure and comments for each module. Try to make the code modular and easy for the graders to go through, as it will help to give partial credits.**
2. **Make sure your code compiles before submitting**
3. **Make sure to follow the output format and the naming format.**
4. **Make sure not to write the output to any files. Use standard output to print them.**
5. **We will be using Moss for plagiarism detection.**