University of Ioannina,
Department of Computer Science & Engineering

Data Mining – CSE 012/CS 059
Fall 2015

Assignment 1

This is the second part of Assignment 1. The deadline for this part of the assignment is November 12, 1:00 pm, at the beginning of the class. You should turn in the Iron Python notebook, all the code you have written and a pdf with your report. For late submissions the late policy on the page of the course will be applied. Details for the turn-in, and how to write reports are on the Assignments web page of the course.

Question 1

On the Assignments page of the course there are two files: "data1.csv" and "data2.csv". Each file contains three comma-separated columns of 100 values, with column names A, B, and C. Your goal is to find the relationship between column A and columns B and C, for each of the files. Create an Iron Python Notebook, which should contain the code for processing the data, the plots and computations that you did, and a report with your conclusions.

Question 2

For this question we will use the Yelp Academic Dataset. Download the data from the link on the Material web page. The data are in JSON format (one object per line), so you will need to use a JSON parser to obtain the data in a form that you can process. You can use an existing JSON parser.

Yelp keeps information for users and venues. Users write reviews and tips for venues. There is also a social network between the users. In this assignment we will use mostly the tips data. Usually, a user leaves a tip when they check-in to some venue. We will thus assume that each tip corresponds to a check-in on the same date.

You will use the data to solve the following two problems:

1. We want to find groups of venues (of size at least 2) that users often visit on the same day. (This information is useful so that, for example, if a user visits one of the venues, we can recommend one of the others in the group.) Design and implement an algorithm that uses frequent itemset mining to find these groups. Use a large enough support threshold (e.g., 20) to obtain a relatively small number of pairs, and then lower it to obtain a few triplets. Return the pairs and triplets, with the names of the venues. Comment on the results and point out interesting associations.
2. Transposing the data, we are now interested in finding groups of users (of size at least 2) that often go out together. (This information is useful, since we could, for example, make a group offer to these users.) We assume that a group of users go out together if they all leave a tip at the same venue on the same day, and they are all friends with each other in the social graph. Design and implement an algorithm that uses frequent itemset mining to find such groups. In this case we are interested in the number of such groups we can find, so report the sizes for different support thresholds.

You can use any programming language you want for your implementation. The use of Python is recommended, because of the flexibility it offers in the use of data structures, and the Pandas library, but it is not mandatory. For mining frequent itemsets you can also use some existing implementation. There are several implementations in the FIMI web page (there is a link in the Material page of the course), and also in WEKA (the link to WEKA is also on the Material page). For some implementations you will need to do some data transformations.

Except for the code, you should also turn in a report where you describe the design of your algorithm, the different steps for running your code, and a commentary on the results. Comment on whether you see some interesting associations between the venues that appear frequently together, and if it is possible to find groups of friends that go out together.

Question 3

In Question 2, except for the data in which we are looking for frequent itemsets, we are also given a graph with the pairwise relationships between the items. We are interested in finding k-itemsets that are frequent, and at the same item all items are connected to each other in the graph. Describe how we can use this information to speed-up the APriori algorithm, by restricting the generation of the candidate itemsets. Your answer should be concise and not too long: describe the necessary changes in the algorithm and the implementation of APriori. Hand in a pdf with your report.