Data Mining

Homework 4

Due: 24/5/2015, 23:59.

Instructions

You must hand in the homeworks electronically and before the due date and time.

Handing in: You must hand in the homeworks by the due date and time by an email to the instructor that will contain as attachment (not links!) a .zip or .tar.gz file with all your answers and subject
`[Data Mining class] Homework #`
where `#` is the homework number. After you submit, you will receive an acknowledgement email that your homework has been received and at what date and time. If you have not received an acknowledgement email within 1 day after you submit then contact the instructor.

The solutions for the theoretical exercises must contain your answers either typed up or hand written clearly and scanned.

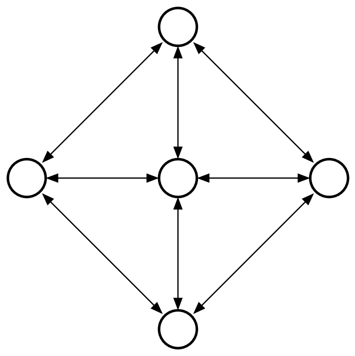
The solutions for the programming assignments must contain the source code, instructions to run it, and the output generated (to the screen or to files).

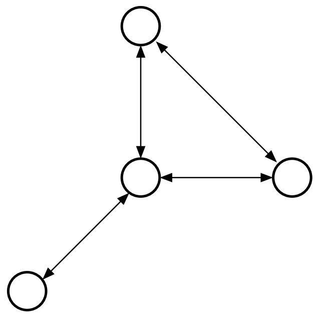
For information about collaboration, and about being late check the web page.

Most of the questions are not very hard but require time and thought. **You are advised to start as early as possible, to work in groups, and to ask the instructor in case of questions.**

Problem 1. Here we will compute the PageRank in a special case and we will prove a rule.

1. Compute the PageRank scores of the nodes of the following two graphs, for teleporting probability equal to zero (i.e., $\beta = 1$), using the equations of the stationary distribution. Take advantage of the symmetries to reduce the number of unknown variables: are there nodes that we know a priori that they have the same PageRank score?


(A)


(B)

2. Notice that here we have a special case: All the edges are bidirectional and we have $\beta = 1$. After observing the scores of the nodes that you computed in these examples, make a conjecture about the PageRank score of a node in this special case, and prove it.

Problem 2.

In class we saw an algorithm for estimating the second moment in a stream. Here we will see another method and we will implement it.

We have a universe $U = \{e_1, \dots, e_\ell\}$ of ℓ elements. Assume that the stream that arrives is the

$$\mathbf{s} = (s_1, s_2, \dots, s_n),$$

where each $s_j \in U$, and for each $e_i \in U$ let $m(e_i)$ be the number of times that e_i appears in $\mathbf{s}$:

$$m(e_i) \triangleq |\{j;\ s_j = e_i\}|\,.$$

Let $\mathbf{Y} = (Y_1, Y_2, \dots, Y_\ell)$ be a random vector, where each Y_i is a random value in $\{-1,1\}$, each with probability 1/2, all Y_is being mutually independent.

Define

$$X \triangleq \sum_{i=1}^{\ell} Y_i \cdot m(e_i).$$

Part 1.

1. Compute $\mathbf{E}[X]$ and $\mathbf{E}\left[X^2\right]$.
2. Use the above idea to propose an algorithm for estimating the second moment of stream $\mathbf{s}$.

Part 2.

Now we will use this to estimate the second moment of the number of twitter users in Rome in a given time period.

1. First we will obtain the stream of tweets as they are generated. There are various ways to access the twitter API. The most direct is to use the package `python-twitter`, which gives direct access to the API. However, there are several high-level packages, which hide several of the details. One of the easiest ones is the `twython` package. You are encouraged to use that one, but feel free to use whichever you prefer.

   To access the twitter streaming API, you should register as a user, create a twitter application, and generate four keys, which allow you to authenticate from your application. After you do that, you can use the `twython` library to set a query and listen for tweets that are being created in Rome (for that you need to define a bounding box to filter tweets and keep only tweets that are created in Rome). Thus the goal of this part is to grab the tweets in Rome as they are generated. In particular, we are interested in the location of the tweet, the screen name of the user, and the text of the tweet.

   To learn more, you need to consult the documentation for `twython` and that of the twitter API.

2. After you familiarize yourself with the API, select a time period, say a day, and collect the geolocated tweets provided by the twitter streaming API in Rome. As the tweets arrive, apply the approach of Part 1, to estimate the second moment of the number of users who tweet in Rome. You apply the algorithm online, but at the same time store the tweets on disk.

3. Compute also the actualy second moment from the file, using nonstreaming techniques, and compare it with the value that you obtain.

You will have to make several decisions while you implement your algorithm and you solve the exercise. Take decisions that you believe are reasonable.