Data Mining

Homework 1

Due: 27/10/2019, 23:59

Instructions

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

The first homework has to be done by each **person individually**.

Handing in: You must hand in the homeworks by the due date and time by an email to Andrea (`mastropietro.1652886@studenti.uniroma1.it`) that will contain as attachment **(not links to some file-uploading server!)** a .zip file with your answers. The filename of the attachment should be
`DM_Homework_1__StudentID_StudentName_StudentLastname.zip`;
for example:
`DM_Homework_1__1235711_Robert_Anthony_De_Niro.zip`.
The email subject should be
`[Data Mining] Homework_1 StudentID StudentName StudentLastname`;
For example:
`[Data Mining] Homework_1 1235711 Robert Anthony De Niro`.
After you submit, you will receive an acknowledgement email that your project has been received and at what date and time. If you have not received an acknowledgement email within 2 days after you submit then contact Andrea.

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

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

Problem 1. A family has two kids, each being a boy or a girl with probability 1/2 and born in a random day of the week.

1. Define a sample space sufficient to answer the third question and define the probabilities to the points in the sample space.
2. If we know that one kid is a girl, what is the probablity that the other kid is a girl?
3. If we know that one kid is a girl born on Sunday, what is the probablity that the other kid is a girl?

Problem 2. Aris and Andrea each pick a different page from the textbook IIR. Your goal is to find who chose the page that appears earlier in the book. Note that we do not choose the pages randomly; in fact, we may choose the pages adversarially so that we make your task really hard (for example, by choosing pages 1 and 2). However, to help you we give you the chance to ask one of us randomly (each with probability 1/2) what is the page number he has chosen. Note that it is easy to find a strategy with probability of winning exactly 1/2; for example, pick Aris or Andrea with probability 1/2. Surprisingly, there are ways to achieve a probability of winning strictly greater then 1/2. Devise a strategy that does that.

Hint: Assume that you know that a given number x is between the two page numbers. Then you know that if the random person you picked reveals a number $y > x$ then you need to respond that

the other person has picked the smallest page number. Of course you do not really know x but this hint should help you.

Problem 3. The Erdős-Rényi $G_{n,p}$ random-graph model, is a mathematical model for creating random graphs. Fix a positive integer n and a value $p \in [0,1]$. Then a graph created according to the $G_{n,p}$ model, has n nodes, and each pair of distinct nodes is connected with an edge with probability p, all pairs being independent from each other.

1. Define an appropriate probability space Ω to describe the $G_{n,p}$ model. What is its size?
2. What is the probability of each element of Ω?
3. What is the probability that a graph created according to $G_{n,p}$ contains exactly only one triangle (three nodes connected with each other and no other edges exist)?
4. What is the probability that all the n nodes are connected in a line (with no other edges present)?
5. Assume that we create a graph according to $G_{n,p}$, what is its expected number of edges?
6. Consider a random graph $G = (V,E)$ with $|V| = n$ nodes created according to the $G_{n,p}$ model. Let us define a *3-star* to be a subgraph $V' = \{v_0,v_1,v_2,v_3\} \subseteq V$ of V such that all the edges (v_0,v_i) for $i = 1,2,3$ exist in G, and none other of the edges (v_1,v_2), (v_1,v_3), and (v_2,v_3) exist in the graph (but other edges may exist, including edges between nodes in V' and other nodes in V). What is the expected number of 3-stars in G?
7. Define similarly a k-star. What is the expected number of k-stars?

Problem 4. In this question we will start getting our hands a bit dirty. We will practice a little bit with UNIX and then we will start playing with Python. At
`http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html`
you can find a dataset from Last.fm, containing songs that users have listened to. The initial file that we will work on is the
`userid-timestamp-artid-artname-traid-traname.tsv`,
which contains among other information, what song each user has listened to. Our goal is to find out what combinations of songs are the most common. But we want to restrict ourselves to songs that users listen to frequently.

- In the first part we will use UNIX commands to simplify and make the file shorter. Quite often you can analyze your data just by using simple UNIX tools and after one knows them he can use them for several simple tasks. Some usefull commands are the `grep`, `sort`, `uniq`, `cut`, `sed`, `awk`, `join`, `head`, `tail`, `wget`, `curl`. You can find more information using the `man` command or by checking the web. Shell scripting can help you even more.

  The file `userid-timestamp-artid-artname-traid-traname.tsv` contains several fields, some are empty, whereas some songs appear multiple times for each user. Our goal is to process it and create a file with the following format:

  $$\texttt{userid}_i\texttt{<TAB>track-id}_{i1}\ \ \texttt{track-id}_{i2}\ \ \ldots$$

where `userid`$_i$ is the ith user and `track-id`$_{ij}$ is the jth track id that will contain only the songs that each user has listened at least 20 times. So every line must contain all the tracks that a user has listened to at least 20 times, and each track-id of those must appear just once. Note that in the initial file, some track ids are missing; ignore these songs.

After downloading the file, use some of the above commands to convert the initial file to the format requested. You will need to use a bunch of the commands above. (**Hint:** You can do it with a single—but long, you'll need `awk`—command line, by chaining commands through pipes!)

- For the second part you need to write a Python program that finds what are pairs of songs that a lot of users like. From Part 1 we know for each user what tracks he has listened for at least 20 times. Write a Python program that outputs the top-10 pairs of track ids, that is, the track ids that correspond to the pairs of songs that have been listened at least 20 times by the most number of users.

Problem 5. In this exercise we will get some practice in using python and some libraries, for downloading web pages, parsing them, and performing some first analysis. We will obtain data about apartments available for rent in Rome.

For downloading the web pages you may use the package `Requests` or the package `urllib2`. To parse the page you can either use regular expressions through the package `re` (it is anyway a good idea become familiar with regular expressions), or, probably better, use an HTML/XML parser. The `Beautiful Soup` package is a good one but it loads the whole file in memory. This is fine for this problem, since the pages to parse are small, but be careful if you want to use it on large XML files; for those ones check the `lxml` library and the tutorial at

`http://www.ibm.com/developerworks/xml/library/x-hiperfparse/`

Write a program that will download from `http://www.kijiji.it` and parse all the apartments for rent in Rome. Download regular and top announcements, but not sponsored ads. Save in a tab-separated value (TSV) file, for every apartment (one line per apartment), the *title*, *short description* (from the result page), *the location*, the *price*, the *timestamp* of the apartment announcement, and the *URL link* to its web page. **Because you will make a lot of calls to the kijiji site, make sure that you have a delay (use: `sys.sleep()`) between different downloads of kijiji pages, to avoid being blocked.**

After you download the web pages, compare the different locations by calculating for each of them:

1. The number of announcements.
2. The average apartment price.