

# Exam Advances in Data Mining

Wojtek Kowalczyk

[w.j.kowalczyk@liacs.leidenuniv.nl](mailto:w.j.kowalczyk@liacs.leidenuniv.nl)

29.10.2018

It is a closed book exam: you are not allowed to use any notes, books, calculators, smartphones, etc. The number of points attached to each question reflects the (subjective) level of question's difficulty. In total you may get 100 points. The final grade for the exam is the total number of points you receive divided by 10.

The exam consists of a number of questions with a "single choice answer". It means that for each question you should select exactly one answer. For every correct choice you get some points; for an incorrect choice or no choice you get 0 points.

Mark your choices by crossing the selected option. In case you want to "undo" your choice put a circle around the cross. For example, on the left side the option **b** is selected; on the right side nothing is select – the selection of **b** is "undone":

a) bla bla  
☒ b) ble ble  
c) bli ble

a) bla bla  
☐ b) ble ble  
c) bli ble

If you think that your marking is no longer readable, put your final choice on the left margin (e.g., by writing "a" if you want to select "a"). Finally, you are free to add to your answers your comments (in a free space) – it may help us to understand your way of thinking.

**Before starting answering the questions, fill in the following entries:**

**Name:**

**Student number:**

**Study type (ICT, Astronomy, ... ):**

<b>6 points</b>	<b>1. Finding the most frequent word</b>
Question	<p>Suppose that you have a huge text file that contains a list of words, one word per line, many of them repeating many times, and your task is to find the most frequent word and its frequency. Additionally, we assume that although the file is too big to fit into RAM, the list of unique words that occur in the file is relatively small and can easily fit into RAM.</p> <p>Which method is the fastest one:</p>
Answer options	<p>a) Initialize a dictionary of counters, indexed by words, and then scan the file updating the dictionary and the counters. Simultaneously, maintain a pair &lt;most frequent word, count&gt;, and update it when needed.</p> <p>b) Sort the file and then scan it, calculating frequencies of all words (when the file is sorted counting frequencies is easy). Simultaneously, when scanning the sorted file, maintain a pair &lt;most frequent word, count&gt; updating it when needed.</p> <p>c) Apply the reservoir sampling algorithm to generate an unbiased sample of words, find the most frequent word in the sample and then scan the original file to count the number of occurrences of the found word.</p>

A dictionary is a very efficient data structure (constant time for access, insert, and delete operations) so in practice the option (a) will take as long as a single read of the file.

The option (b) requires sorting the file on a harddisk which is very slow (the file will be read and copied several times).

The option (c) is slower than (a) as we have to scan the file twice: once for sampling and another time for counting. Moreover, there is no guarantee that the most frequent item will be found by the reservoir sampling algorithm.

<b>6 points</b>	<b>2. Matrix Factorization: regularization</b>
Question	<p>Let us suppose that after some extensive tuning of various parameters a Matrix Factorization model for the Netflix Challenge was developed with help of the gradient descent algorithm. In particular, an optimal value of the regularization parameter <math>\lambda</math> was found. A few years later Netflix published an updated, much bigger data set that involved the same users and movies as in the original competition: the new utility matrix had the same shape but it became more dense as users rated more movies. When retraining the old model on the new data, data scientists ask themselves how the value of <math>\lambda</math> should be changed. Assuming that they have time to try only one option, which one would you advice:</p>
Answer options	<p>a) Make <math>\lambda</math> bigger</p> <p>b) Make <math>\lambda</math> smaller</p> <p>c) Don't change <math>\lambda</math> at all</p> <p>d) It is impossible to predict which option would lead to better results, so make your choice (between a, b, c) at random</p>

Regularization is used to prevent overfitting – preventing the network to memorize the training data. The more data we have the lower the risk of overfitting, therefore the “L2-punishment” should be smaller.

<b>10 points</b>	<b>3. Matrix Factorization: initialization (this is a difficult/tricky question!)</b>
Question	What is the <u>worst</u> method for initializing the movie and user factors, when applying the gradient descent algorithm to train a model:
Answer options	<p>a) Set all factors to <math>\sqrt{\text{global\_average\_rating}/k}</math>, where <math>k</math> is the number of factors, so the product <math>\text{user\_factors} * \text{movie\_factors}</math> will be exactly the global average rating.</p> <p>b) Set factors to some random values around 0, e.g., drawn from the standard normal distribution with 0 mean and standard deviation 1.</p> <p>c) Set factors to some random values close to the global mean.</p> <p>d) Set factors to some random values uniformly drawn from <math>[0,1]</math>.</p>

The option (a) is the worst one: if all factors are initialized to the same value then the gradients of the error function will be the same for all these factors, so in practice it means that effectively we train a model with a single factor. This observation was made in the "gravity-Tikk.pdf" paper: "Note that if the rows of  $U$  and the columns of  $M$  are constant vectors, each row of  $U$  and each column of  $M$  remains a constant vector. That is, why we initialize randomly."

<b>4 points</b>	<b>4. Hadoop</b>
Question	<p>Suppose that a huge collection of documents <math>D</math> is stored on a Hadoop system. One term that is involved in the definition of TF.IDF is Document Frequency: for any word <math>w</math>, document frequency of <math>w</math> is the number of documents that contain <math>w</math>. It can be easily computed within the MapReduce framework. Let the map function be defined as follows:</p> <p style="text-align: center;"><code>Map(d): for every unique word w in d emit(w, d)</code></p> <p>Which of the following Reduce functions computes Document Frequency:</p>
Answer options	<p>a) <code>Reduce(w, [d1,...,dk]) = emit(w, length([d1,...,dk]))</code></p> <p>b) <code>Reduce(w, [d1,...,dk]) = emit(length([d1,...,dk]))</code></p> <p>c) <code>Reduce(w, [d1,...,dk]) = emit(unique([d1,...,dk]))</code></p> <p>d) <code>Reduce(w, [d1,...,dk]) = emit(w, length(unique([d1,...,dk])))</code></p> <p>e) None</p>

By definition, we want to know, for each word, the number of documents that contain this word. So option (d) is the right one. Moreover, as we require that the Map function emits  $(w,d)$  for every unique word in  $d$ , the answer a) is also correct and even faster as the unique function is not called. Therefore both answers are correct; all others are incorrect.

<b>8 points</b>	<b>5. Bloom filter</b>
Question	Consider a Bloom filter which uses $h$ hash functions to "store" $k$ objects in a vector of $n$ bits. Which formula correctly defines the false positive rate (the chance that a randomly chosen object passes the filter) as a function of $h, k, n$ :
Answer options	<p>a) <math>\left(1 - e^{-\frac{hk}{n}}\right)^h</math></p> <p>b) <math>\left(1 - e^{-\frac{hn}{k}}\right)^h</math></p> <p>c) <math>\left(1 - e^{-\frac{k}{n}}\right)^h</math></p> <p>d) <math>\left(1 - e^{-\frac{hk}{n}}\right)^k</math></p> <p>e) None of the above</p>

It's the know formula (e.g., from slide 18, Streams\_1); only different letters are used to denote the number of hash functions (h instead of k), the number of objects (k instead of n) and the number of bits (n instead of N).

<b>10 points</b>	<b>6. Randomness</b>
Question	<p>Consider the following Python program which generates 1 billion (<math>n=10^9</math>) random integers between 0 and <math>N=2^{32}</math>:</p> <pre>import numpy as np n=10**9 N=2**32 r=np.random.randint(0,2**32, dtype='uint32', size=10**9)</pre> <p>Assuming that Python's random number generator is perfect and all integers have the same chance of being selected, what can you say about the number of unique elements of <math>r</math>, i.e., the value of <math>\text{len}(\text{np.unique}(r))</math>?</p> <p><u>Hint:</u> it might be helpful to think about a Bloom filter with <math>N</math> bits and <math>n</math> objects that are hashed into this filter.</p>
Answer options	<p>a) It's about 1 billion (almost all generated numbers are different)</p> <p>b) It's about <math>n(1 - e^{-n/N})</math></p> <p><b>c) It's about <math>N(1 - e^{-n/N})</math></b></p> <p>d) It's about <math>ne^{-n/N}</math></p> <p>e) It's about <math>Ne^{-n/N}</math></p>

See the example 4.3, page 153, from the textbook.  $n$  darts are thrown, one after another, at random into  $N$  bins. Let us select a single bin. What is the chance that after throwing the first dart the bin remains empty? Of course,  $(N-1)/N = 1-1/N$ . After the second dart is thrown the bin stays empty if both darts missed the target and that happens with probability  $(1-1/N)^2$ . And the chance that after throwing  $n$  darts the bin is still empty is  $(1-1/N)^n$ . We looked at one bin only, but there are  $N$  bins and each has the same chance of staying empty after all darts are thrown at it. So the expected number of empty bins is  $N(1-1/N)^n$ , therefore the expected number of non-empty bins is  $N - N(1-1/N)^n = N(1-(1-1/N)^n) = N(1-(1-1/N)^{nn/N}) = N(1-e^{-n/N})$ . Clearly, every dart can be associated with a random integer from  $1:N$  – the position of the bin where the dart landed. And the list of positions of non-empty bins correspond to the list of all unique integers that were generated in this experiment.

<b>6 points</b>	<b>7. Surprise Number (Second Moment)</b>
Question	What is the surprise number (second moment) of the following sequence: C, A, D, C, A, A, C, D, B, A, B
Answer options	<p>a) 25</p> <p>b) 30</p> <p><b>c) 33</b></p> <p>d) Something else</p> <p>e) The question doesn't make sense as the stream elements are not numbers</p>

Frequencies of A, B, C, D are 4, 2, 3, 2, resp., and  $4^2+2^2+3^2+2^2=33$ .

<b>4 points</b>	<b>8. Counting Ones: The Datar-Gionis-Indyk-Motwani Algorithm</b>
Question	How much memory is needed to maintain an approximation of the number of ones in the last $N$ bits of the stream of bits?
Answer options	<p>a) <math>O(\log(N))</math></p> <p>b) <math>O(N\log(N))</math></p> <p>c) <math>O(\log(N^2))</math></p> <p>d) <math>O(\log^2(N))</math> (we have <math>\log(N)</math> bins, each one needs <math>\log(N)</math> bits to keep the timestamp)</p> <p>e) None of the above answers is correct</p>

<b>6 points</b>	<b>9. Counting distinct elements in a stream</b>
Question	How many memory bytes are sufficient to estimate the count of distinct elements in a stream with help of the <i>loglog counting</i> algorithm of Durand&Flajolet, assuming that we know up front that this count is smaller than $10^9$ and that the required accuracy of the estimate can be achieved by averaging 1024 basic estimators?
Answer options	<p>a) Less than 1 kB (<math>10^9 &lt; 2^{32}</math> and <math>\log\log(32)=5</math>, 1024 counters of 5 bits is 640 bytes)</p> <p>b) 1 kB</p> <p>c) 5kB</p> <p>d) 8kB</p> <p>e) More than 8kB (here 1kB=1024 bytes)</p>

<b>8 points</b>	<b>10. LSH: the key formula</b>
Question	Which of the formulas specified below expresses the probability of “being detected”, $p(s, b, r)$ , as a function of: the actual similarity $s$ , the number of bands, $b$ , and the number of rows per band, $r$ ?
Answer options	<p>a) <math>p(s, b, r) = (1 - s^b)^r</math></p> <p>b) <math>p(s, b, r) = 1 - (1 - s^b)^r</math></p> <p>c) <math>p(s, b, r) = (1 - s^r)^b</math></p> <p>d) <math>p(s, b, r) = 1 - (1 - s^r)^b</math> (The last formula on slide 35, Sim_1)</p> <p>e) None of the above</p>

<b>5 points</b>	<b>11. LSH: the impact of changing the number of bands</b>
Question	<p>For given values of <math>b</math> and <math>r</math>, the plot of <math>f(s) = p(s, b, r)</math> has an S-shape, where the central point <math>s_0</math>, such that <math>f(s_0)=0.5</math>, separates pairs with low probability of being detected from pairs with high probability of being detected.</p> <p>What happens with this plot when the number of bands <math>b</math> is increased while the number of rows per band, <math>r</math>, is decreased in such a way that the signature length, <math>b*r</math> stays the same?</p>

Answer options	<p>a) the point <math>s_0</math> shifts to the left (LSH is more sensitive on pairs with lower similarity), [check slide 42, Sim_1]</p> <p>b) the point <math>s_0</math> shifts to the right (LSH is less sensitive on pairs with lower similarity),</p> <p>c) the point <math>s_0</math> stays at the same place, but the shape of the S-curve is more sharp, i.e., it resembles more a jump function,</p> <p>d) none of the above.</p>
----------------	--

<b>5 points</b>	<b>12. LSH: the impact of changing the length of the signature</b>
Question	<p>For given values of <math>b</math> and <math>r</math>, the plot of <math>f(s) = p(s, b, r)</math> has an S-shape, where the central point <math>s_0</math>, such that <math>f(s_0) = 0.5</math>, separates pairs with low probability of being detected from pairs with high probability of being detected.</p> <p>What happens with this plot when the number of rows per band <math>r</math> is increased while the number of bands, <math>b</math>, stays the same (i.e., the signature length is also increased)?</p>
Answer options	<p>a) the point <math>s_0</math> shifts to the left (LSH is more sensitive on pairs with lower similarity)</p> <p>b) the point <math>s_0</math> stays at the same place, but the shape of the S-curve is more sharp, i.e., it resembles more a jump function</p> <p>c) the point <math>s_0</math> shifts to the right and the shape of the S-curve is more sharp</p> <p>d) none of the above</p>

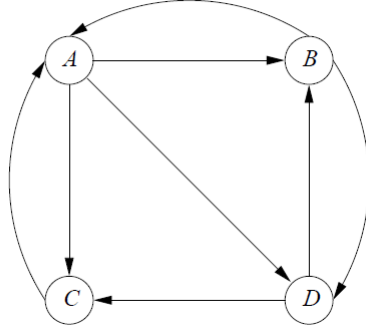
(c) is the correct answer. When the number of rows in a band is increasing then it is more difficult (less likely) to find candidate pairs in this band – therefore  $s_0$  moves to the right. More, the last term in the key formula,  $(1 - s^r)^b$  is converging more rapidly to 0 as  $(1 - s^r)$  becomes smaller ( $r$  is growing), therefore the S-curve rises to 1 faster. Just make several plots of this curve to find it out!

<b>6 points</b>	<b>13. LSH: the size of the signature matrix</b>
Question	<p>Let us consider a large collection of <math>N</math> sets of integers of type <code>uint16</code> (i.e., 16 bits long) that is stored on a hard disk. In order to find in this collection pairs of similar sets (using the Jaccard similarity measure), we need to calculate, for each set, a minhash signature of length <math>L</math>, so the signatures can be stored in a big matrix that can fit in RAM.</p> <p>What is the minimal number of bytes of RAM that is needed to store the signature matrix, assuming <math>L=100</math>, <math>N=10.000.000</math>?</p> <p>(1 byte = 8bits, 1GB = <math>10^9</math> bytes)</p>
Answer options	<p>a) 1GB</p> <p>b) 2GB</p> <p>c) 4GB</p> <p>d) 8GB</p> <p>e) Less than 1GB</p>

The matrix has size  $L \times N = 10^2 \times 10^7 = 10^9$ . Each entry of this matrix is not bigger than  $2^{16}$  – there are at most  $2^{16}$  different integers in the sets from the collections – therefore 2 bytes are sufficient to represent all entries of the signature matrix. Thus we need memory for 1 billion entries, each entry 2 bytes long, which leads to the 2GB memory requirement.

<b>4 points</b>	<b>14. PageRank Algorithm</b>
Question	Let $M$ denote the transition matrix of a strongly connected directed graph $G$ and let $v_0$ denote the initial probability distribution over nodes of $G$ . Which additional condition must be satisfied in order to guarantee that the iterative algorithm $v_{n+1} = Mv_n$ will converge to a fixed point $v$ such that $v = Mv$ ?
Answer options	a) $G$ does not contain cycles b) $G$ does not contain dead-ends c) $G$ does not contain spider traps d) All of the above e) None of the above

The answer (b) is suggested in the textbook on page 179. Unfortunately, when the graph contains dead ends, the algorithm will also converge to a fixed point (with all probabilities being 0), so formally, even in this case we have convergence. However, the initial matrix is not stochastic, i.e., not all columns sum up to 1. This assumption was missing in the question and that led to a confusion. **Therefore, no matter what you selected, you will get 4 points for this question.**

<b>4*3 points</b>	<b>15. Page Rank: Example</b>
Question	<p>Consider the following graph:</p>  <p>Let us assume that we start surfing the graph at a randomly selected node (i.e., all nodes have the same probability of being selected as a starting point), following the links at random. What are the probabilities <math>p_A, p_B, p_C, p_D</math> that in the next step we will visit node A, B, C, or D, respectively? Write down the calculated values:</p>
Exceptionally, this is not a choice question! You should calculate these probabilities and write them down! If possible, use fractions, e.g., 7/23.	<p><math>p_A = 9/24</math></p> <p><math>p_B = 5/24</math></p> <p><math>p_C = 5/24</math></p> <p><math>p_D = 5/24</math></p> <p>Check slide 6 from PageRank or page 180 of the textbook.</p>