Name:	Matrikelnummer:

Chair for Algorithms
and Data Structures
Prof. Dr. Hannah Bast
Patrick Brosi

Information Retrieval
WS 2016/2017

http://ad-wiki.informatik.uni-freiburg.de/teaching

UNI FREIBURG

Exam

Tuesday, February 21, 2017, 14:00 - 16:30, Hörsaal 026+036 in Building 101

General instructions:

There are six tasks, of which you can select *five tasks of your choice*. Each task is worth 20 points. If you do all six tasks, we will only count the best five, that is, you can reach a maximum number of 100 points.

You need 50 points to pass the exam. You have 150 minutes of time overall. If you do five tasks, this is 30 minutes per task on average.

You are allowed to use any amount of paper, books, etc. You are not allowed to use any computing devices or mobile phones, in particular nothing with which you can communicate with others or connect to the Internet or parallel universes.

You may write down your solutions in either English or German.

Please write your solutions on this hand-out, below the description of the tasks! You can also use the back side of the pages. Please write your name and Matrikelnummer on the top of this cover sheet in the framed box. If you need additional pages, please write your name and Matrikelnummer on each of them, too.

Important:

For the programming tasks, you can use Python, Java, or C++. None of your functions should be longer than TEN lines, otherwise you risk point reduction.

For all other tasks: do not simply write down the final result; it should also be clear how you derived it.

Good luck!

Task 1 (Ranking and evaluation, 20 points)

Consider the following deeply meaningful collection of four documents $D_1, \ldots, D_4$:

D_1: first nice document
D_2: second document
D_3: more nice text
D_4: more text

1.1 (10 points) Write down the term-document matrix for this collection with *tf.idf* scores.

1.2 (5 points) Write a function that, given an arbitrary term-document matrix with *tf* scores and the number of terms m and the number of documents n, changes the scores in the matrix to *tf.idf* scores. Don't use matrix operations but simple for-loops. You can assume a function *log2*.

1.3 (5 points) Find a query, such that the dot-product similarities of the four documents above (using your *tf.idf* scores from 1.1) to that query are all different. Rank the four documents by these scores (highest score first). Define a set of *two* relevant documents such that the precision at 2 (P@2) and the average precision (AP) are both 50% (the relevance does not need to make sense, just make sure that P@2 and AP are as requested).

Task 2 (Encodings, 20 points)

Consider the following code: for number x, encode $\lfloor \log_2 x \rfloor$ using Golomb-encoding with modulus $M = 4$ and then write x in binary without the leading 1. For example, the correct code for $x = 10$ is 111010.

2.1 (10 points) Write down the codes for $x = 1, 2, \ldots, 10$. Use a different color for the Golomb part and the binary part. If you don't have colors (shame on you), draw a dot between the two parts. It helps when you write down the 10 codes one below the other.

2.2 (5 points) Write a function that for a given integer $x < 16$ returns the code for x (as an array of bits) according to the encoding above. You can assume functions *floor* and *log2* and *extend* (to extend an array by another array). In Python, you can write an *if* with just a single statement in one line. Note that the restriction $x < 16$ is to make this task easier for you: the function for arbitrary x is slightly more complex.

2.3 (5 points) What is the exact formula for the length of the code for x in that encoding (any x now, not necessarily $x < 16$). Do not just write down a formula, but explain how you derived it.

Task 3 (Web applications and UTF-8, 20 points)

3.1 (5 points) Write valid HTML to display two input fields in the browser. In front of the first input field, write *Centimetres*. In front of the second input field, write *Inches*. Include a JavaScript file *convert.js* (see 3.2) and provide the necessary ids for that JavaScript to work.

3.2 (5 points) Write the content of *convert.js* (you can use *jQuery*) such that, whenever something is entered into one input field, that something is interpreted as a number and the content of the other input field is updated accordingly. Hint: 1 inch = 2.54 cm. You can read and write the content of an input field using *val* and when reading you can just assume that the content evaluates to a real number. To detect a keypress in one of the fields, use *keyup*.

3.3 (5 points) The € character has a codepoint of 8364 in Unicode. Compute its *binary* representation in UTF-8. Use a different color for the code point part. You can use without proof that 8364 = 32 * 256 + 172.

3.4 (5 points) Write a function that, given a byte array that represents a *valid* UTF-8 sequence, returns the number of characters. For example, for an array consisting of the three bytes that represent the € character, the function should return 1.

Task 4 (Naive Bayes and k-means, 20 points)

Consider the following three points in R^2: $P_1 = (3, 1)$, $P_2 = (1, 2)$, $P_3 = (1, 4)$. The distance between two points (x_1, y_1) and (x_2, y_2) is defined as $|x_1 - x_2| + |y_1 - y_2|$, that is, the sum of the absolute differences of the individual components. For example, the distance between P_1 and P_2 is 3.

4.1 (5 points) Do the steps of k-means, using the distance measure defined above, until the clustering does not change anymore. As initial cluster centroids, use $C_1 = (3, 0)$ and $C_2 = (0, 3)$. For each of the steps, write down the 3×2 assignment matrix A, where $A[i, j]$ is 1 if and only if P_i is assigned to C_j, and 0 otherwise.

4.2 (5 points) Write a function that, given an arbitrary $n \times k$ assignment matrix A (like in 4.1) and an arbitrary $n \times 2$ matrix P with n points from R^2, returns the $k \times 2$ matrix containing the centroids. You can use standard linear algebra operations like multiplication, transposition, and computing the $L1$-norm or $L2$-norm of all columns or rows of a matrix.

4.3 (5 points) Assume point P_1 above is labeled X and points P_2 and P_3 are labeled Y. Perform the learning step of Naive Bayes, taking the points as feature vectors. Determine the w and b of the linear classifier computed by Naive Bayes.

4.4 (5 points) Provide an example input (of points in R^2) such that Naive Bayes will classify a point (x, y) as class X if and only if $x > y$ (you can ignore the case $x = y$).

Task 5 (Latent Semantic Indexing, 20 points)

Consider the following matrices:

$$A = \begin{pmatrix} 1 & 1 & 1 & 6 & 1 \\ 3 & 3 & 3 & 2 & 3 \end{pmatrix}, U = 1/2 \cdot \sqrt{2} \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, V = 1/4 \cdot \sqrt{2} \cdot \begin{pmatrix} 1 & -1 & 1 & 2 & 1 \\ -1 & 1 & -1 & 2 & -1 \end{pmatrix}$$

5.1 (5 points) Show that V is row-orthonormal (that is, $V \cdot V^T$ is the identity matrix). You don't have to show that U is column-orthornormal but you can assume it without proof for 5.2.

5.2 (5 points) Compute the missing diagonal matrix S such that $A = U \cdot S \cdot V$ is the singular value decomposition of A. Hint: you can either derive this by doing the matrix multiplication, or (more elegantly) by making use of the fact that the columns of U are the eigenvectors of $A \cdot A^T$.

5.3 (5 points) What is the rank of matrix A above? Do not just state the rank but provide an explanation. Can you add a *column* to the matrix such that the rank becomes 3? Provide an example or argue that this is not possible.

5.4 (5 points) Write a function that, given an n-dimensional vector (represented in an ordinary array in the obvious way), normalizes the vector such that its $L2$-norm (sum of the squares of the entries) is 1. Your function should run in $O(n)$ time.

Task 6 (Miscellaneous, 20 points)

6.1 (5 points)

Given a vocabulary $V = \{v_1, \ldots, v_n\}$ of words, provide a formula (simplified as much as possible) for the *exact* total number of items in the inverted lists of a 3-gram index when each word is padded on both sides with \$\$. For clarification: if a word contains a 3-gram k times, that word is contained k times in the inverted list of that 3-gram.

6.2 (5 points)

Assume a knowledge base stored in three tables of a database, each with two columns: a table *founded_by* (columns: company and person), a table *based_in* (columns: company and city), and a table *born_in* (columns: person and city). Formulate a SQL (not SPARQL) query that returns exactly those persons which have founded a company in the same city they were born in. It is ok, if the same person is returned multiple times.

6.3 (10 points)

What is the maximum entropy of a probability distribution given by $p_1, \ldots, p_n$. Hint: use Lagrangian optimization and don't forget the second derivative. Hint: $(f \cdot g)' = f' \cdot g + f \cdot g'$.