CSCI-B565: Data Mining

Homework Assignment # 1

Assigned: 01/19/2016 *Due: 02/04/2016, 11:59pm, through Oncourse*

Five questions, 130 points in total. Good luck!

Prof. Predrag Radivojac, Indiana University, Bloomington

Problem 1. (10 points) Term frequency-inverse document frequency transformation. Suppose you are given a document-term matrix corresponding to a set of n documents and a dictionary of m terms (words). Suppose further m_{ij} is the number of times that the j-th term appears in the i-th document, m_i is the number of terms in the i-th document and n_j is the number of documents containing the j-th term in the data set. Consider the following feature representation for the ij-th element of the document-term matrix

$$x_{ij} = \frac{m_{ij}}{m_i} \cdot \log \frac{n}{n_j},$$

and answer the following questions:

a) (4 points) What might be the benefits of this encoding in mining text documents?

b) (4 points) What might be disadvantages of this encoding compared to $x_{ij} = \frac{m_{ij}}{m_i}$ or $x_{ij} = m_{ij}$?

c) (2 points) What is the effect of this transformation if a term occurs in one document or in every document?

Problem 2. (10 points) Let $\mathbf{x}$ and $\mathbf{y}$ be k-dimensional column vectors from $\mathbb{R}^k$. Prove that

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{\|\mathbf{x}\| \, \|\mathbf{y}\|}$$

where $\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots x_k^2}$ is the length of vector $\mathbf{x}$. A way to start might be to consider a right triangle and the fact that the cosine of an angle is defined as the ratio of the lengths of the adjacent side and the hypotenuse.

Problem 3. (45 points) Metrics on sets. Let each $d : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ below be a distance function on pairs of sets from space $\mathcal{S}$. For each of the functions below, either prove it is a metric or provide a counterexample. Consider six distance functions:

a) (5 points) $d_1(A, B) = |A - B| + |B - A|$

b) (5 points) $d_2(A, B) = \frac{|A-B|+|B-A|}{|A \cup B|}$

c) (5 points) $d_3(A, B) = 1 - \left(\frac{1}{2} \cdot \frac{|A \cap B|}{|A|} + \frac{1}{2} \cdot \frac{|A \cap B|}{|B|} \right)$

d) (10 points) $d_4(A, B) = 1 - \left(\frac{1}{2} \cdot \frac{|A|}{|A \cap B|} + \frac{1}{2} \cdot \frac{|B|}{|A \cap B|} \right)^{-1}$

e) (5 points) $d_5(A,B) = (|A-B|^p + |B-A|^p)^{\frac{1}{p}}$, $p \geq 1$

f) (15 points) $d_6(A,B) = \frac{(|A-B|^p+|B-A|^p)^{1/p}}{|A \cup B|}$, $p \geq 1$

where $A-B$ is the set difference, $A \cup B$ is the set union, $A \cap B$ is the set intersection, and $|A|$ is the number of elements in the set A, also called the cardinality of A.

Problem 4. (20 points) Understanding the curse of dimensionality. Consider the following experiment: generate n data points with dimensionality k. Let each data point be generated using a uniform random number generator with values between 0 and 1. Now, for a given k, calculate

$$r(k) = \log_{10} \frac{d_{\max}(k) - d_{\min}(k)}{d_{\min}(k)}$$

where $d_{\max}(k)$ is the maximum distance between any pair of points and $d_{\min}(k)$ is minimum distance between any pair of points (you cannot use identical points to obtain the minimum distance of 0). Let k take each value from $\{1, 2, \ldots, 99, 100\}$. Repeat each experiment multiple times to get stable values by averaging the quantities over multiple runs for each k.

a) (15 points) Plot $r(k)$ as a function of k for three different values of n; $n \in \{100, 1000, 10000\}$. Label and scale each axis properly to be able to make comparisons over different n's. Embed your final picture(s) in the file you are submitting for this assignment.

b) (5 points) Discuss your observations and also compare the results to your expectations before you carried out the experiment.

Problem 5. (45 points) Building a recommendation system. In this exercise you will test different distance functions for the movie recommendation systems. First, familiarize yourself with the MovieLens data sets that are available at

`http://grouplens.org/datasets/movielens/`

Next, design and evaluate your own recommendation system based on the following principles:

- For each user i and each movie j they did not see, find top k most similar users who have seen j and then use them to infer the user i's rating on movie j. Handle all exceptions in a reasonable way and report your strategy if you did so; e.g., if you cannot find k users for some movie j, then take all users who have seen it.
- Test the performance of your system using cross-validation. For each data set, the MovieLens database already provides a split of the initial data set into $N = 5$ folds. This means you will run your algorithm N times; in each step, use the training partition to make predictions for each user on all terms rated in the test partition (by that user). When you complete all N iterations, you will have a large number of user-movie pairs from the 5 test partitions on which you can evaluate the performance of your system.
- Measure the performance of your recommendation system using the mean absolute difference (MAD); that is,

$$\text{MAD} = \frac{1}{\sum_i \sum_j r_{ij}} \cdot \sum_i \sum_j r_{ij} \cdot |p_{ij} - t_{ij}|$$

where p_{ij} is the predicted rating of user i on movie j, t_{ij} is the true rating available in the test partition, and r_{ij} is the indicator variable of the availability of rating for the pair (i, j). That is, $r_{ij} = 1$ if the rating for (i, j) is available; otherwise, $r_{ij} = 0$.

- Compare all your algorithms with a simple algorithm that gives each user-movie pair a rating that equals the average score over all users who rated the movie. Note that here each user receives the same rating (prediction) for a particular movie. Ideally, your algorithm will outperform this terribly naive scheme.

Your task in this exercise is to train and evaluate several recommendation systems.

a) (15 points) Use the "100K Dataset" to evaluate three different distance metrics: the Euclidean distance, the Manhattan distance and the $L_{\max}$ distance using the entire vectors of ratings over all movies. Calculate the performance and find the best k from a hand-selected set of 3-5 values (your choice) for each specific case.

b) (15 points) Continue with the "100K Dataset", but modify your distance metrics appropriately to incorporate other information such as user's gender, movie genre, etc. You should also at this stage change distance metrics into the ones you believe might perform better. For each new metric you tested, explain the rationale for choosing it. It is not required, but you can also attempt one modification of the first principle of the recommendation system stated above (the algorithm itself).

c) (10 points) Switch to the "10M Dataset" and evaluate the performance of the top three algorithms you devised in the previous steps (this includes variations over distance functions and parameters). Note that here you will need to use the script to generate training and test partitions. Discuss your observations and findings.

d) (5 points) Briefly comment on what might be a good next set of steps to improve this recommendation system.

We have created directories `/l/b565/ml-100k` and `/l/b565/ml-10M100K` on the Hulk server and ran the .sh script in the latter case. These files are readable by anyone. Therefore, please avoid creating copies of these files on the Indiana University servers. To log on to Hulk, go to

`hulk.soic.indiana.edu`

and use your university ID and password.

Homework Directions and Policies

Submit a single package containing all answers, results and code. You submission package should be compressed and have extension .zip. In your package there should be a single pdf file named main.pdf that will contain answers to all questions, all figures, and all relevant results. Your solutions and answers must be typed[1] and make sure that you type your name and IU username (email) at the beginning of the file. The rest of the package should contain all code that you used. The code should be properly organized in folders and subfolders, one for each question or problem. All code, if applicable, should be turned in when you submit your assignment as it may be necessary to demo your programs to the associate instructors. Use Matlab, Python, R, Java, or C/C++.

Unless there are legitimate circumstances, late assignments will be accepted up to 5 days after the due date and graded using the following rules:

on time: your score $\times$ 1

1 day late: your score $\times$ 0.9

2 days late: your score $\times$ 0.7

3 days late: your score $\times$ 0.5

4 days late: your score $\times$ 0.3

5 days late: your score $\times$ 0.1

For example, this means that if you submit 3 days late and get 80 points for your answers, your total number of points will be $80 \times 0.5 = 40$ points.

All assignments are individual, except when collaboration is explicitly allowed. All the sources used for problem solution must be acknowledged; e.g., web sites, books, research papers, personal communication with people, etc. Academic honesty is taken seriously! For detailed information see Indiana University Code of Student Rights, Responsibilities, and Conduct.

[1] We recommend Latex; in particular, TexShop-MacTeX combination for a Mac and TeXnicCenter-MiKTex combination on Windows. An easy way to start with Latex is to use the freely available Lyx. You can also use Microsoft Word or other programs that can display formulas professionally.