CSCI-B565: Data Mining

Homework Assignment # 4

Assigned: 03/26/2016 *Due: 04/09/2016, 11:59pm, through Oncourse*

Five questions, 130 points in total. Good luck!

Prof. Predrag Radivojac, Indiana University, Bloomington

Problem 1. (15 points) Consider a data set $\{(x_i, y_i)\}_{i=1}^n$ where $y_i \in \{-1, +1\}$ and a set of predictions $\{p_i\}_{i=1}^n$, where $p_i \in \mathbb{R}$, obtained through cross-validation. What are the minimum and maximum number of thresholds for which the true positive and false positive rates have to be computed in order to calculate the area under the ROC curve? Prove it. You may assume that the data set contains n_+ positive and $n_- = n - n_+$ negative examples.

Problem 2. (15 points) Let $\{d(x_i, x_j)\}_{ij}$ be a set of $\binom{n}{2}$ Euclidean distances between pairs of n distinct objects from some space $\mathcal{X}$. Suppose the data set of objects is partitioned into m groups, where $\{c_j\}_{j=1}^m$ is a set of m centroids for each (non-overlapping) subset of objects j; that is, for each subgroup of objects $j \in \{1, 2 \dots m\}$, c_j is their centroid. Now, if two of the m groups of objects are to be merged to minimize the sum of squared errors between each data point and its respective centroid, show how you will compute which two clusters to merge out of $\binom{m}{2}$ pairs in total. Note that you are only allowed to work with distances between these points because you are not given the original objects x_i. You are not allowed to use any embedding techniques such as multidimensional scaling, to map these points into a Euclidean space. Prove any complex statements in your algorithm.

Problem 3. (70 points) Apriori algorithm. Implement the *Apriori algorithm* by first determining frequent itemsets and then proceeding to identify association rules. Consider that the input to your program is a sparse matrix where the rows are transactions and columns are items. Each value in your matrix is a binary variable from $\{0, 1\}$ that indicates presence of an item in the transaction.

a) (15 points) Implement both $\mathbf{F}_{k-1} \times \mathbf{F}_1$ and $\mathbf{F}_{k-1} \times \mathbf{F}_{k-1}$ methods. Allow in your code to track the number of generated candidate itemsets as well as the total number of frequent itemsets.

b) (10 points) Use three data sets from the UCI Machine learning repository to test your algorithms. The data sets should contain at least 1000 examples, and at least one data set should contain 10,000 examples or more. You can convert any classification or regression data set into a set of transactions and you are allowed to discretize all numerical features into two or more categorical features. Compare these two candidate generation methods on each of the three data sets for three different meaningful levels of the minimum support threshold (the thresholds should allow you to properly compare different methods and make useful conclusions). Provide the numbers of candidate itemsets considered in a table and discuss the observed savings that one of these methods achieves.

c) (10 points) Enumerate the number of frequent closed itemsets as well as maximal frequent itemsets on each of your data sets for each of the minimum support thresholds from the previous question. Compare those numbers with the numbers of frequent itemsets.

d) (15 points) Implement confidence-based pruning to enumerate all association rules for a given set of frequent itemsets. Use the previous data sets, with three levels of support and three levels of confidence to quantify the savings in the number of generated confident rules compared to the brute-force method.

e) (10 points) For each data set and each minimum support threshold, select three confidence levels for which you will generate association rules. Identify top 10 association rules for each combination of support and confidence thresholds and discuss them (i.e. comment on their quality or peculiarity). Select data sets where you can more easily provide meaningful comments regarding the validity of rules.

f) (10 points) Instead of confidence, use *lift* as your measure of rule interestingness. Identify top 10 rules for each of the previous situations and discuss the relationship between confidence and lift.

No specialized libraries are allowed for this task. Make sure that you code runs and submit all the code you used in this task, including the code that converts data sets from UCI Machine Learning repository into a transaction data set. Do not include raw data sets in your supplement; however, do provide links to the data sets you used such that your code can be run independently and its performance can be verified.

Problem 4. (30 points) Formalizing clustering is difficult. In this question you will read two scientific publications and in your own words summarize each of them in 1000 words or less. Both papers are available on-line. Your summary should demonstrate the ability to succinctly present the main points of these papers and provide a short critical assessment of each paper. A critical assessment should demonstrate strengths and weaknesses and well as your opinion on the quality of these papers.

a) (15 points) Kleinberg J. The impossibility theorem for clustering. *Advances in Neural Information Processing Systems*, NIPS 2002.

b) (15 points) Ackerman M, Ben-David S. Measures of clustering quality: a working set of axioms for clustering. *Advances in Neural Information Processing Systems*, NIPS 2008.

It is absolutely not allowed to copy any single sentence from the papers and use it in your summary. All sentences must be your own.

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 commented and 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 strictly individual. No discussion about this assignment is allowed with anyone other than the instructors. All the sources used for problem solution must be acknowledged; e.g., web sites, books, or research papers. 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.