Project Assignments

CPE403-CSC403 Advanced Data Management Techniques (Data Mining)

Due Date: **29 March, 2013 (Friday)**

Reminders

- This assignment consists of THREE parts.
- You are NOT allowed to COPY code/report from Internet or others (unless specified for some special cases). Any plagiarism case will be seriously punished!
- You need to implement the source code by yourself. It is NOT allowed to simply call some existing functions/procedures (unless specified for special cases).
- For late submission, a penalty of **1 mark** per day will be applied after the deadline. The assignment will not be accepted if more than **7-day delay**. Please do submit your assignment before the deadline.
- Programming language: Java/C/C++ (MATLAB is not allowed).
- Operating System Platform: Windows / Linux

Marking Scheme (Total: 30 marks)

PART 1:	10 marks
PART 2:	10 marks
PART 3:	10 marks

PART 1: Implementation of Basic K-Nearest Neighbor Classification algorithm

Problem 1. This task is to implement the basic k-Nearest Neighbor Classification (k-NN) algorithm and examine its performance on the given classification tasks/datasets. The detailed requirements are described below:

(1) The number of nearest neighbors *k* can be set by some specific value (default k=1).

(2) To run your k-NN algorithm, your program should be able to run with the command line arguments similar to the following:

knn *training_set_file test_set_file* -k *k_value* -d *metric_type*

where *training_set_file* is the filename of input training data set, *test_set_file* is the filename of the test data set, *k_value* is the number of nearest neighbors used for k-NN classification, and *metric_type* is a number that indicates the type of distance metric (**0** - Cosine similarity, **1** - L1 distance, **2** - L2/Euclidean (default))

(3) The output of your program should include the following:
(1) A prediction file: *test_set_prediction_file*, in which each row is a predicted class label corresponding to one test example in the test data set.
(2) Display the prediction **accuracy** of the classification task (assume that class labels are given) and the total **time cost** (seconds) for doing the classification task.

PART 2: Implementation of Basic K-means Clustering algorithm

This task is to implement the basic k-means clustering algorithm and examine its performance on several given datasets. The detailed requirements are described below:

(1) The number of clusters *k* should be able to set for a specific value (default k=2).

(2) The set of *k* initial cluster centroids should be randomly generated, by either a set of *k* random *d*-dimensional points or a set of *k* examples randomly chosen from the dataset (where d is the dimensionality of the data).

(3) For a k-means clustering task, the number of iterations *t* can be either a small or large value, which usually depends on the problem and the dataset. In practice, to avoid too many rounds of iterations, you can provide an option to set a *maximum number of iterations* (default *max_it*=50), such that when the number of iterations is greater than this number, the algorithm will terminate.

(4) As k-means is sensitive to the initial centroids, your k-means algorithm should be able to repeat the clustering process multiple times (default *n_run*=1) (Note that each run should use a different set of initial centroids.) To evaluate the quality of different runs, you need to compute the within-cluster SSE (sum-of-square error) for each clustering results. Based on the SSE score, you can output the final clustering results with the minimal SSE score obtained among these runs.

(5) The requirements of the input and output formats can be found in the attached *dataset_format.doc* file.

(6) To run your k-means algorithm, your program should be able to run with the command line arguments similar to the following:

kmeans *datasetfile outputfile* -k *k_value* -m *max_it* -r *n_run*

where *datasetfile* is the input data filename, *outputfile* is the output filename, *k_value* is the number of clusters, *max_it* is the maximal number of iterations, *n_run* is the number of times for repeating the k-means clustering multiple times (each run is with different initial centroids).

Example 1:
The following command is to cluster the data examples in the *toydata.txt* file with 4 clusters, 50 maximal number of iterations, and 20 different runs

kmeans *toydata.txt toydata_out.txt* -k 4 -m 50 -r 20

For PART 1 and PART 2, please also read the details of *Experimental Evaluation* section.

PART 3: Making K-means or k-NN Practical for Mining Big Data

The goal of this task is to improve the efficiency and scalability of the traditional k-means or k-NN algorithms, for addressing the challenges of mining big data. This part aims to improve the time and/or space efficiency for the basic classification (k-NN) or basic clustering (k-means) algorithm. The detailed requirements are described below:

1. You can choose either to improve the basic k-means or the basic k-NN algorithm.

2. You have the following two options to improve the efficiency of the basic algorithm:

A. Implement one or multiple existing techniques in recent research studies (you may refer to some example papers in the end of this section (References).

B. Propose and implement your own novel technique for improving basic algorithms;

Note that option B is encouraged (better grade may be awarded if you can justify the novelty and show reasonably good results). In general, your grade for this part would depend on how significant result is achieved by your new algorithm, and the novelty of your proposed technique.

3. The requirements of the input and output file formats are the same to PART1/2.

4. You should report the performance evaluation of both algorithms (the original one and your proposed new algorithm). If your goal is to achieve a speedup of efficiency over the basic algorithm (in terms of time or memory cost or both), you need to report the speedup evaluation in your report. If your goal is to improve classification efficacy, you should report classification predictive(e.g., accuracy) performance.

References:

Below are some of example papers. However, you can go beyond these papers by searching for any related papers from internet or other sources. Take note that you should cite the papers properly in your report.

Example references that mainly aim to enhance **classification efficiency**

[1] Abdelhamid Djouadi, Essaid Bouktache: A Fast Algorithm for the Nearest-Neighbor Classifier. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 19(3):277-282 (1997)

[2] Bin Zhang, Sargur N. Srihari: Fast k-Nearest Neighbor Classification Using Cluster-Based Trees. IEEE Trans. Pattern Anal. Mach. Intell. 26(4): 525-528 (2004)

[3] Luisa Micó, José Oncina, Rafael C. Carrasco: A fast branch & bound nearest neighbour classifier in metric spaces. Pattern Recognition Letters (PRL) 17(7):731-739 (1996)

[4] Ting Liu, Andrew W. Moore, Alexander G. Gray, Ke Yang: An Investigation of Practical Approximate Nearest Neighbor Algorithms. NIPS 2004

[5] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, Angela Y. Wu: An Optimal Algorithm for Approximate Nearest Neighbor Searching Fixed Dimensions. J. ACM (JACM) 45(6):891-923 (1998)

[6] Marius Muja, David G. Lowe: Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. VISAPP 2009:331-340
[7] Robert F. Sproull: Refinements to Nearest-Neighbor Searching in k-Dimensional Trees. Algorithmica 6(4):579-589 (1991)
[8] Aristides Gionis, Piotr Indyk, Rajeev Motwani: Similarity Search in High Dimensions via Hashing. VLDB 1999:518-529

Example references that mainly aim to enhance **classification accuracy**

[9] Yoshihiko Hamamoto, Shunji Uchimura, Shingo Tomita: A Bootstrap Technique for Nearest Neighbor Classifier Design. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 19(1):73-79 (1997)
[10] Noam Shental, Tomer Hertz, Daphna Weinshall, Misha Pavel: Adjustment Learning and Relevant Component Analysis. ECCV 2002:776-792
[11] Carlotta Domeniconi, Jing Peng, Dimitrios Gunopulos: Locally Adaptive Metric Nearest-Neighbor Classification. IEEE Trans. Pattern Anal. Mach. Intell. 24(9): 1281-1285 (2002)
[12] Kilian Q. Weinberger, Lawrence K. Saul: Distance Metric Learning for Large Margin Nearest Neighbor Classification. Journal of Machine Learning Research 10: 207-244 (2009)
[13] Jing Peng, Douglas R. Heisterkamp, H. K. Dai: LDA/SVM driven nearest neighbor classification. IEEE Transactions on Neural Networks 14(4): 940-942 (2003)
[14] Eui-Hong Han, George Karypis, Vipin Kumar: Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification. PAKDD 2001: 53-65
[15] Trevor Hastie, Robert Tibshirani: Discriminant Adaptive Nearest Neighbor Classification. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 18(6):607-616 (1996)
[16] Carlotta Domeniconi, Dimitrios Gunopulos, Jing Peng: Large margin nearest neighbor classifiers. IEEE Transactions on Neural Networks (TNN) 16(4):899-909 (2005)
[17] Jacob Goldberger, Sam T. Roweis, Geoffrey E. Hinton, Ruslan Salakhutdinov: Neighbourhood Components Analysis. NIPS 2004
[18] Hao Zhang, Alexander C. Berg, Michael Maire, Jitendra Malik: SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. CVPR 2006:2126-2136

Example references that aim to improve **k-means clustering**

[19] Kanungo, T.; Mount, D. M.; Netanyahu, N. S.; Piatko, C. D.; Silverman, R.; Wu, A. Y. "An efficient k-means clustering algorithm: Analysis and implementation". IEEE Trans. Pattern Analysis and Machine Intelligence 24: 881–892.
[20] Elkan, C. "Using the triangle inequality to accelerate k-means". Proceedings of the Twentieth International Conference on Machine Learning (ICML). 2003
[21] Frahling, G.; Sohler, C. (2006). "A fast k-means implementation using coresets". Proceedings of the twenty-second annual symposium on Computational geometry (SoCG).
[22] Jing Wang, Jingdong Wang, Qifa Ke, Gang Zeng, and Shipeng Li, "Fast Approximate k-Means via Cluster Closures", CVPR, 2012

Experimental Evaluation:

PART 1

Table 1. Evaluation of Classification Performance

Dataset	Toydata	iris	glass	vowel	vehicle	letter	DNA
K	1	1	1	3	3	3	5
metric_type	2	1	0	2	1	0	2
Accuracy	..	..					
Time (seconds)	..	..					

PART 2

For each dataset, <u>**run**</u> the clustering experiments with the predefined parameters below:

Dataset	Toydata	iris	glass	vowel	vehicle	letter	DNA
K	4	3	6	11	4	26	3
max_it	50	50	50	50	50	50	50
n_run	20	20	20	20	20	20	20

In your report, you should describe the details of your implemented methods clearly. You also need to complete the following table for performance evaluation.

Table 2. Comparison of minimal within-cluster SSE among 20 random runs

Dataset	toydata	Glass	iris	vehicle	vowel	wine
K-means						

Table 3. Comparison of Time Efficiency Performance (seconds)

Dataset	Toydata	Glass	Iris	vehicle	vowel	Wine	**Mean**
K-means	...	...	...				

PART 3

No a formal template. Free Style for this part. In general, for your experimental evaluation, you can download some large-scale data sets from the internet to test the efficiency and scalability of your algorithms. The following links are some sources of data sets for your reference:

http://archive.ics.uci.edu/ml/

http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Submission Guidelines

What To Submit:

1. Source code.
   - Well-commented code
   - Include Makefiles if necessary
   - Remove the binary executable program if any
2. A README file. Please name it **README.txt** This file should include three sections:
   - Your group ID and group member names
   - Program language used and instructions on how to run your programs.
3. A file called **Asg_Report_GroupXX.pdf**, including Pseudocode description of the algorithms, experimental evaluations and discussions, and the conclusions drawn from your experiments. Please show your group members' names and IDs in the cover page of your report.
4. The page limit of the report is 20 pages. The over-length case may be penalized. Please do not simply attach your source code in the report. However, if necessary, you can show some code segment or pseudo code to describe your algorithm.
5. Please use the given template report and remember to fill in the Table in the cover page of your report

Submission Instructions

Please package all of your files (including the source code, the README.txt file, and your report "Asg_Report_GroupXX.pdf") into a ZIP file, named as “Asg_GroupXX.zip”, where XX is your group ID.

Submit the package file with the Subject “**ASG SUBMISSION**” to my E-mail: chhoi@ntu.edu.sg where XX is your group ID. (Please do use upper case in the Subject).

You also need to submit a hardcopy of your repot to my office: **N4-2a-08** by hand (or submit to my pigeonhole in general office if I am not in my office).