COS 435, Spring 2012 - Problem Set 5

Due at 1:30pm, Wednesday, April 11, 2012.

Collaboration and Reference Policy

You may discuss the general methods of solving the problems with other students in the class. However, each student must work out the details and write up his or her own solution to each problem independently.

Some problems have been used in previous offerings of COS 435. You are NOT allowed to use any solutions posted for previous offerings of COS 435 or any solutions produced by anyone else for the assigned problems. You may use other reference materials; you must give citations to all reference materials that you use.

Lateness Policy

A late penalty will be applied, unless there are extraordinary circumstances and/or prior arrangements:

- No penalty if in Prof. LaPaugh's office or inbox by 5pm Wednesday (4/11/12).
- Penalized 10% of the earned score if submitted by 11:59 pm Wed (4/11/12).
- Penalized 25% of the earned score if submitted by 5pm Friday (4/13/12).
- Penalized 50% if submitted later than 5pm Friday (4/13/12).

Problem 1 (2011 Exam 2 problem)

Part A

Develop a recommendation method that combines collaborative filtering and content-based recommending. Specifically, the method should recommend an item for purchase to a consumer based on whether the item is similar to other items the consumer has purchased or whether similar consumers have purchased the item. Similarity between consumers is determined by the similarity of the items they have purchased. Note that similar consumers need not purchase the same items but purchase similar items. Items are characterized by 100 attributes. Each item has a 100-dimensional 0/1 vector that indicates which attributes the item possesses (the 1-valued components of the vector). The only information available about a consumer is the set of items that he or she has purchased.

Make the above method concrete by giving an equation or algorithm to compute item similarity, an equation or algorithm to compute user similarity, and an algorithm to determine whether an item should be recommended to a consumer. (Imagine that items are recommended each time a consumer visits the shopping site.) Your method must consider ***efficiency as well as effectiveness***.

Part B

For your concrete method of Part A, what is the time necessary to

i. Compute the similarity of a new item to existing items.

ii. Update the similarity between consumers when a consumer purchases an item.

iii. Determine whether a specific item should be recommended to a consumer.

Assume the information is stored in main memory. That is, do not worry about disk reads/writes.

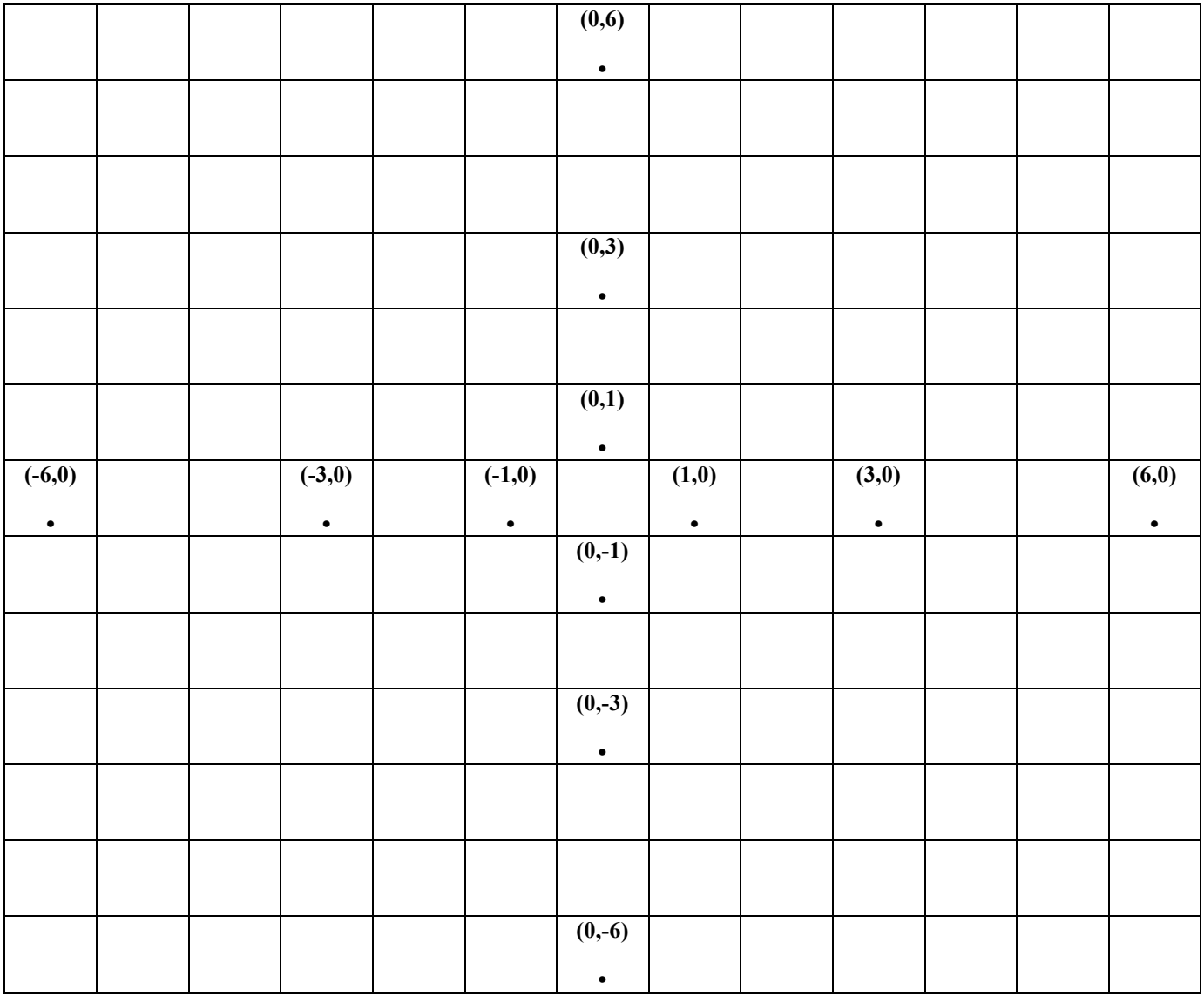
Problem 2:

Consider the following set of 2-dimensional points in the plane:

{(1,0), (0,-1), (-1,0), (0,1), (3,0), (6,0), (0,-3), (0,-6), (-3,0), (-6,0), (0,3), (0,6)}

(See Figure 2 below.) **Measure the distance between two points using Euclidean distance.** Similarity between two points is determined by their distance: more distant is less similar.

Figure 2



Part A: Give the dendrogram for the execution of the hierarchical agglomerative algorithm using ***complete-link cluster similarity*** on the given set of points. The height of each dendrogram horizontal bar should be the similarity of the two clusters merged at that point. Stop when seven clusters are left. What are the seven clusters?

Part B: What is the ***first*** merge in your execution of part A that ***must*** be different if you use ***single-link cluster similarity*** instead of complete-link cluster similarity? That is, if the hierarchical agglomerative algorithm is executed using single-link cluster similarity and ties are always broken to agree with what the complete-link version does, what is the first merge for which the two versions of the algorithm differ? What clusters are merged under single-link cluster similarity?

Problem 3:

The algorithm for hierarchical agglomerative clustering giving in Figure 17.8 of *Introduction to Information Retrieval* uses one priority queue for each cluster to efficiently find the most similar pair of clusters to merge. The priority queues are updated for each merge step by deleting the two clusters that have been merged and inserting the new combined cluster. Consider ***breaking ties*** when selecting the pair of clusters to merge by choosing the pair that results in the smallest combined cluster. What modifications would be needed in the algorithm and data structures of Figure 17.8? Would the running time be affected? Explain.