# Introduction to Data Mining -- Project 2

Each student develops Java programs to handle the three inter-related tasks described below. There will be two separate submissions to two different dropboxes on pilot (called Proj2a for Task 1 and Proj2b for Tasks 2 & 3 respectively) on two different due days (to be announced on pilot news).

Task 1 (60 pts): The input to the program for this task are: a dataset D in the csv format, a class ID c, and an integer k. For this project we will work with the planning.csv dataset as an example; for this dataset a class ID can be 1 or 2. In general, a class ID is a string or a number. Below for each c = 1,2, we will use Dc to refer to the subset of instances in D whose class value is c. Assume all attributes of D except the class are numerical, and the last attribute is the class. We refer to the nth column of D as attribute An; you should follow this in your reports and output files. The information about this csv file also applies to the other tasks.

For this task your program will find some top k patterns P1,…,Pk maximizing
GSO(P1,…,Pk)=avgGR*avgSupp*(1/(avgOvlp+0.01)), where

- avgGR is defined to be the average of GR(P1), …, GR(Pk). For this project, for each pattern P, we define GR(P) = (count(P,c)+1)/(count(P,~c)+1), where count(P,c)=|mds(P,Dc)| and count(P,~c)=|mds(P,D-Dc)|.
- avgSupp is defined to be the average of supp(P1,D), …, supp(Pk,D); supps are assumed to be in fractions.
- avgOvlp is defined to be average$_{1<=i < j <=k}$ (|mds(Pi) INTERSECT mds(Pj) | / |D|). In words, avgOvlp is the average of the sizes (divided by |D|) of the intersections of the mds of pairs of different patterns in {P1,…,Pk}.

How this search is done is up to you. Possible approaches include direct mining, searching by seeding-and-improving over frequent patterns of Dc, searching by (smart) sampling (evaluating a large number of sets having k (frequent) patterns), and so on. Novel efficient methods are welcome.

This task produces two output files:

(a) A file (called binningItemMap.csv), where each row contains four comma-separated values “Ai, lb,rb,j”, Ai is an attribute ID, lb is a number denoting the left bound, and rb is a number denoting the right bound, of an interval/bin, and j is the integer to represent the interval in the itemized data used in the pattern mining process.

(b) A file (called topPatterns.csv) containing two types of rows describing the k selected patterns. The first type of rows describe the patterns and the second type of rows describe the overlap among the patterns. More specifically, for each selected pattern Pi={x1,x2,…,xm}, there is a row containing “Pattern,i, GR(Pi), supp(Pi,c), supp(Pi,~c), supp(P,D), x1,x2,…,xm”; for each pair of distinct patterns Pi and Pj, there is a row “Overlap,i,j, |mds(Pi) INTERSECT mds(Pj) | / |D|”.

In the above, x1,…,xm are items which are described in binningItemMap.csv.

It may be very time consuming to find the k patterns that have the true maximal GSO value. Moreover, which binning method you use can also affect the GSO; you can use any of the binning methods in this project. The mark you receive will depend on how close your result is to the true optimal solution. Work using Excel that lead to results not too far from the optimal will also get some marks (a very small fraction of the available marks).

For this task, your submission should contain a jar file P2a.jar and a report P2aReport (in word, pdf or txt) which describes your output for c=2 and k=4 on the planning.csv file. The top line of P2aReport gives the GSO value, the second line gives the binning method used, the third line states how you found the result (e.g. by submitted java program, by working with Excel). The next four lines describe the patterns, and the final group of lines describe the overlap among the patterns. Those lines about the patterns and overlap should be the same as those produced by your program. The last line of the report file gives info on the author and the source URL of the frequent pattern mining program you are using.

The submitted P2a.jar file should be executable using three command-line arguments: D c k.

Task 2 (25 pts): First, use the code developed in Task 1 to select k patterns PC1={P1,…,Pk} for class 1 (using command-line arguments D, 1, k) and another k patterns PC2={Q1,…,Qk} for class 2 (using command-line arguments D, 2, k). Then, define a CAEP-like classification model using those selected patterns, as follows: For each data instance t and each class c, let score(t,c) = sum_{P in PCc, t matches P} [supp(P,Dc) * (supp(P,Dc) +1)/ supp(P,D)]; the classifier predicts t to belong to the class j such that score(t,c=j) is the larger among score(t,c=1) and score(t,c=2).

Task 3 (15 pts): Improve the program you wrote for Task 1, to select k patterns to maximize the accuracy of the CAEP-like classification model constructed in a way similarly to Task 2. You can change how the scores are defined (see above) in your classification model and also change the objective function of the search.

The submitted P2b.jar file should be executable using two command-line arguments: D k. D is a csv files similar to that of planning.csv; D is used for both training and testing (for the sake of simplicity). k is the number described above in the description of Tasks 2 and 3.

For Tasks 2 and 3, your submission should contain a jar file P2b.jar and a report P2bReport (in word, pdf or txt) describing your results. For Task 2 the report gives the accuracy of the classifier on D. For Task 3 the report gives the accuracy of the classifier on D, together with descriptions of how you define the scores and what is your objective function for the search.

The output of Task 2 is one csv file called “P2aOut.csv” containing the following: The first line gives the accuracy of the classifier on D and the following lines give the classification result of the classifier on the rows in D -- for each row of D, the csv file contains a line of the form “i, corginal, cpredicted” (which are row ID, the original class in D, and the predicted class, resp.)

The output of Task 3 is one csv file called “P2bOut.csv” containing the following: The first line gives the accuracy of the classifier on D and the following lines gives the classification result of the classifier on the rows in D; for each row of D, the csv file contains a line of the form “i, corginal, cpredicted” (which are row ID, the original class in D, and the predicted class, resp.). The two files P2aOut.csv and P2bOut.csv have the same format but they are for different classifiers.

For all tasks, assume **the input file will be placed in the same folder where your root program is**. All output files are to be written in the same folder where your root program is.

The submitted jar file and the report files will be used to evaluate your project. Executability, correctness, efficiency, and quality of the findings will be important factors for marking.

You can use any frequent pattern mining programs written in Java in your program. You cannot use emerging/contrast pattern mining programs written by others; of course you can implement your own.

Your submitted files should not include the dataset files and not include the produced results by your programs.

Collaboration between students is not allowed. No students can access results produced by others or the programs written by others.