Data Mining - Report 2

Damage

1 Problem and Approaches

The task was to determine patterns in the choice of courses computer science students of University of Helsinki take. We were given a data set containing the courses and their metadata taken by students over a number of years. The methods of representation we used in a previous attempt, frequent itemsets, was not optimal. It did solve our problem, but it also represented mundane and redundant information. Therefore, we were introduced to the methods of maximum and closed itemsets during class.
We employed a new approach to solve the problem this week. We extended our own implementation for itemset generation to create maximal and closed frequent itemset. Furthermore we concentrated on the second part of the association rule mining stategy, the actual rule generation. In our implementation, we also considered different measurement methods to determine the value of a rule and an itemset.
The eclat software was still used, but mainly for the purpose of verifying the outcome of our own implementation.

2 Data

In addition to the course information from last week, we now had more specific data about the courses. This metadata consisted of the time period the course was held, the term, its level and compulsority. Furthermore, it contained information about subprograms, which it might belong to.

This missing information caused insufficiencies in the interpretation of last weeks results, therefore we were now able to analyze the data more intensly. However, integrating this information caused problems as well, which are further discussed in the next section.

3 Transformation and Command line arguments

Both the meta data in course_details.txt and actual FID data in course_num.txt were transformed. Firstly, only courses with meta data information were taken into account and FID's not present in course_details were omitted when reading course_num. Secondly all the courses were grouped to single entity by FID.

Each so acquired course instance then had following attributes:

1. FID - fid
2. NAME - course name, lower case and slugified
3. YEAR - sequence of years the course has been taught, i.e. [1999, 2000, 2004]
4. SUBPROGRAM - subprogram of the course
5. COMPULSORY - P:yes V:no ?:not known

The code and semester information were omitted because they were thought to be non-relevant. After the data was transformed it was easy to only take into account the courses that are compulsory, taught on certain year interval and so on. We also added some command line tools to restrict the courses.

For example:
> python prob2.py t=0.4 c=0.2 year=2006-2011 compulsory=V strip=2

would only look at the non-compulsory courses that were taught in years from 2006 to 2011 and after it would strip of all the transactions with 2 or less items. In this case minimum support would be 0.4 and minimum confidence 0.2. The reason to apply these kind of restrictions is to look only subset of courses which might give more interesting results.

Restrictions can still be quite tricky if not handled correctly. We had a discussion of what was the most appropriate way to restrist course years. If we did restrict courses to only those which have been taught in time interval t we can't be sure with this data if course c - which has been taught both in and outside t - is particular transaction was inside t or not.

We had both forward and against opinions in group about the matter. Against opinions main point was that it would not make any good assumptions and the noise (c iterations outside t) in transactions would only increase. Forward opinion thought that we could probably make some sound assumptions if we also cutted transactions with low item count out.

As mentioned above, we removed courses without metadata from our dataset. However, we had considered different ways of handling those courses. Each of the three possibility had some disadvantages, which would affect our results and interpretation.

A first possibility was to leave those course with unknown values for the metadata. However, this would have led to futher complications, as soon as the metadata was used to limit the considered data to certain courses.

A second possibility was to just skip those course and exclude them from the data. Yet, this would affect statistical data such as the average amount of course taken by a student, but also the frequency of the other courses. Furthermore, if a transaction only consisted of the courses with unknown metadata, the whole transaction would be missing.

A third possiblity was to remove the whole transaction, if one of its courses was without metadata. This would not have affected basic statistics, such as an average number of courses assuming that courses without metadata were distributed evenly, which we could not guarantee. Nonetheless, as we would minimize our dataset of transactions anyway, we might as well remove those.

After some discussions, we decided on the secound option, removing the courses form our dataset. The disadvantages seemed to be the easiest to handle, so it was the least of the three evils.

4 Implementation

We began by devising and implementing rigid architecture to address future changes that might be required by subsequent problems . The code is now separated to following steps: input and data transforming, data processing and algorithms, output handling. We were able to reuse most of the code implemented during the first week, but the old code needed some optimizations. The frequency calculation from transactions ($\frac{0}{1}$ matrix) was rewritten with matrix operations reducing the time consumed from about 1 minute to 1 second. Candidate generation was also fixed resulting 1/4 decrease in time consumption.

In addition to data handling and transformation code, we implemented apriori algorithm for rule generation. The implementation follows almost precisely the pseudo code in the course book. It was noticed that the algorithm of the book is erroneous. The algorithm 6.2 calls the rule generation function described in algorithm 6.3. The initial parameters of the 6.3 are $f_k \in F_k = \{f \mid f \in \text{frequent itemsets and } |f| = k\}$ and $H_k = \{\{a\} \mid a \in f_k\}$. In the algorithm 6.3 the first assigments are: $k = |f_k|$ and $m = |H_m| = 1$. Then the rest of the function is inside an if clause testing $k > m + 1$. Assuming that $k = 2$ we get $2 > 1 + 1$ which is clearly false. Therefore the pseudo code of the book does not even start when searching for rules in 2-itemsets. If we assume that $k > 2$, then the content of the if clause is executed. The first line in the loop assigns $H_{m+1} = \mathbf{apriori} - \mathbf{gen}(H_m)$ and H_{m+1} is then processed. This means that the algorithm does not take into account rules of form $f_k \setminus h_1 \Rightarrow g_1$ where $h_1 \in H_1$.

Two separate group members that were not implementing the algorithms found a bug in it. According to that fact it seems that not only the coders read the code.

5 Results and Conclusion

The last week we concentrated on our implementation. Still, we have some interesting results and conclusions. We were also able to find our first pattern.
We did use our own implementation with a support threshold of 0.3 and a confidence threshold of 0.2 and had the following results.

introduction_to_programming -> introduction_to_the_use_of_computers 0.596906
introduction_to_programming -> programming_in_java 0.598547
introduction_to_the_use_of_computers -> introduction_to_programming 0.730427
programming_in_java -> introduction_to_programming 0.816235

A closer look reveals, that according to our association rules, introduction_to_programming implies programming_in_java and the other way around. Therefore we concluded, that the direction and the order, in which the courses are taken, matters. We know, that usually the course introduction_to_programming is taken before programming_in_java. Therefore we had the feeling, that the confidence measurement is unintuitive. We started to implement and integrate some other measurement. Those were:

- Interest factor aka Lift
- IS
- Mutual Information
- Certainty factor

Those either symmetric or asymmetric measurement will give us a different view on the data in the future.

We also managed to find our first pattern, which seems to resemble the choices a group of student make. When we limited our considered courses to the non obligatory ones, we found this interesting group of rules.

digital_media_technology software_architecture -> the_metalanguage_xml 0.814696
digital_media_technology software_design_java -> the_metalanguage_xml 0.846154
software_processes_and_quality -> software_architecture 0.778862

One can easily see, that those courses can be easily grouped together. They belong to the same subprogram.

6 Teamwork Evaluation

Comparing to the last week we used IRC more often to communicate and delegate task samong the group members. All group members joined the github and took part in coding. Roughly half the group had problems with understanding the logic behind git, but this minor problem did not cause any show-stoppers in the end. As far as the measure methods are concerned, they need to be discussed more and an appropriate implementation will be chosen based on the provided data. This will be improved in the later problems. Overall our group worked more fluently in this iteration.