Clustering Students to help Evaluate Learning

Agathe Merceron, Kalina Yacef

CLUSTERING STUDENTS TO HELP EVALUATE LEARNING

Agathe Merceron
Computer Science Department
Engineering School, Technical University Leonard de Vinci
Paris-La Defense, France
agathe.merceron@devinci.fr

Kalina Yacef
School of Information Technologies
University of Sydney
Sydney, Australia
kalina@it.usyd.edu.au

Abstract In this paper we show how clustering techniques can be applied to student answers generated from a web-based tutoring tool. In particular we are interested in extracting clusters of students based on the mistakes they made using the tool, with the aim of obtaining pedagogically relevant information and providing this feedback to the teacher. The data we used comes from the Logic-ITA, a web-based tutoring tool to practice formal proofs currently in use in the School of Information Technologies at the University of Sydney.

Keywords: Tutoring systems, Data Mining, Clustering.

Introduction

Distance education, flexible education as well as increasing number of students in some fields make all use of new technologies to enhance learning. On-line tutoring tools belong to these new technologies. Such tutoring systems allow for storing complete student answers, including mistakes, in a database. It becomes possible to mine this database to extract pedagogically relevant information and provide feedback to the teacher.

Towards this end, Data Mining forms the basis for finding new patterns in data. When data is stored in a database or data warehouse, some relational associations of data elements are defined by the database designer. However, it may well be that the data that has been stored contains more information,

more relationships than what has been intentionally put while designing the database. The aim of Data Mining is to find those hidden patterns. Data Mining uses various techniques and algorithms that are quite different in nature. These comprise simple counting and visualization techniques including histograms, as well as more complex algorithms such as clustering, association rules, decision trees, neural networks etc [4].

Data Mining has been used mainly in business with success but also failure stories. There are two pitfalls to avoid. Not discovering existing hidden patterns, or discovering false hidden patterns. Therefore, when applying Data Mining algorithms, it is essential to select carefully the data, to tune up the algorithms carefully and to interpret the results carefully. Naturally this also applies to mining data collected from learning systems.

In this paper, we report on our experiment using clustering on data collected from the Logic-ITA, [1, 5], a web-based Intelligent Teaching Assistant system that is currently used within the School of Information Technologies, University of Sydney. It allows students to practice formal proofs in propositional logic whilst receiving feedback and also keeps the lecturer informed about the progress the class is making and problems encountered. The system embeds the Logic Tutor, a web-based intelligent tutoring system destined to the students, along with tools dedicated to the teacher for managing teaching configuration settings and material as well as for collecting and analysing data.

The contribution of this paper is to show a simple yet relevant way of using clustering, based on students homework as opposed to logs, yielding two homogeneous groups of students that suggest to teachers two different kinds of learners.

The aim of applying clustering to data from the Logic-ITA was to characterize students with difficulties. Clustering algorithms rest on a distance between individuals (or records). In our case, the choice of a distance was not obvious at all because of the heterogeneous way the tool is used. Students practice with the tool at their own pace and with different frequency of usage: Some students use it a lot, on a big variety of exercises, and others make little use of it. There is no fixed set of exercises that can be used to compare students with each others.

Mining data from learning systems begins to receive attention from the research community. In the context of web-based systems, web logs are mined to search for interesting learning-related patterns of usage [10]. Other researchers make use of genetic algorithms and association rules for finding information about students in an adaptive hypermedia system [7]. Association rules have been applied to our system, the Logic-ITA, to extract mistakes often made together by students [6]. Clustering techniques are argued to be useful to find student sequences through learning resources in [8]. The closest work to the work presented in this paper is [3]. They report on clustering students using, as

data, not only students' homeworks, as we do, but also students' interactions with the learning system. The clusters they have obtained were sensible and easy to interpret for about half of the population only.

The paper is organized as follows. Section 2 introduces briefly the Logic-ITA, in order for the reader to undertsand the nature of an exercise and possible mistakes using the system. Section 3 presents clustering, how we haved used it on our data and the results. Section 4 concludes the paper and discusses future work.

1.1 The Logic-ITA

The Logic-ITA is an intelligent teaching assistant system for the domain of formal proofs in propositional logic containing components for the students and for the teacher. The Logic Tutor, the component dedicated to students, provides an environment where students can practice formal proofs of logic at their own discretion, receiving step-by-step, contextualised feedback. They can choose to create new exercises, select exercises in the exercise database, or ask the system for one adapted to their needs. The system stores, for each student, every step entered by the student, along with any mistake they may have made. We will describe shortly the nature of an exercise.

Then the Logic-ITA collates all this information into a database that the teacher can query, in order to retrieve information about the way students accomplished the exercises and re-adjust the content and material of the teaching.

We will not describe in great details the whole system (the reader can refer to [5]) but we need to explain how some of this data is generated to make the following sections clearer.

Description of an exercise

Exercises start with a given set of premises, i.e. a set of well-formed formulae (wff) of propositional logic, and exactly one wff, the conclusion. The task then consists of deriving the conclusion from the premises, step-by-step, using laws of equivalence and rules of inference (we will refer to both of these as rules for the rest of this paper). Figure 1 shows a screen shot of the interface. Here the student was given the first two lines (lines 0 and 1) and the conclusion C. For each step, the student must fill out a new line, entered at the bottom of the screen. The student needs to do the following:
- enter a formula in the *Formula* section,
- choose, from a pop-up menu, the rule used to derive this formula from one or more previous line(s) (*Rules*),
- the references of those previous lines (*Line References*) and
- the premises the formula relies on (*Premises*).

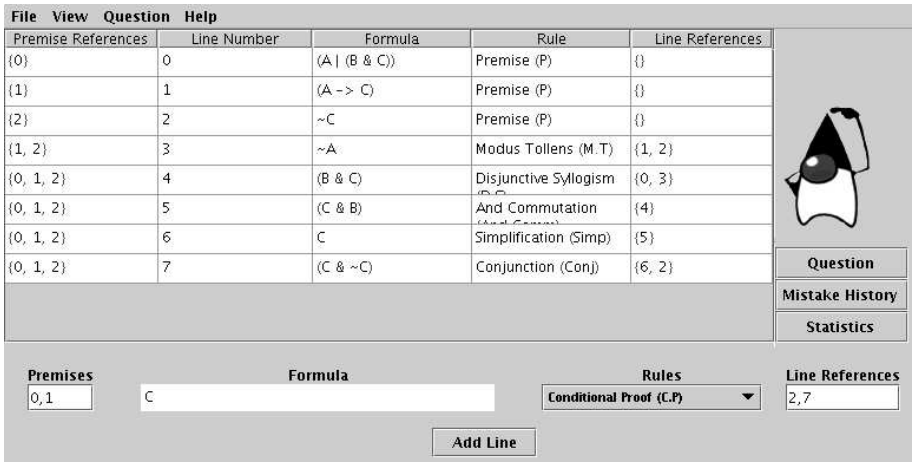

Figure 1. Screenshot during an exercise.

For example in Figure 1, the student is currently deriving the formula C, using the rule *Indirect Proof* and the formulae of lines 2 and 7. Because lines 2 and 7 rely respectively on premises 2 and 0,1,2 (as can be seen in the first column of the screen) and *Indirect proof* removes the premise 2, the line entered therefore relies on premises 0,1. It is actually the last step of this exercise, deriving the conclusion.

There are often many ways to prove an argument valid. The important aspect is that the reasoning must be sound. The actual path followed is not important, as long as each step is valid. Hence students have total freedom in the reasoning they choose to follow.

Exercises come from various sources. Some belong to the core database of the system: they are assigned a difficulty level and are accessible by all the students. Others are created by the system and are downloadable locally. Students can also create exercises. Locally created exercises as well as exercises created by students all have a unique exercise identifier. This means that identical exercises (either created by the system for different students or created by different students) may appear as different when they are stored in the LT-Analyser since they have different identifiers. Furthermore, local exercises as well as exercises created by students have an unknown difficulty level. As a consequence, analysing how students managed a particular exercise only makes sense with exercises belonging to the central database of exercises, with a central and unique identifier.

Mistakes

At each step, the system checks the validity of the data entered by the student. There are different types of mistakes, and, each of them is labelled with a meaningful title for the teacher. Some are generic: *Wrong reference lines*, and others are more specific *Simplification before Commutation*. In addition, the rule specified at the time of the mistake is also recorded and linked to the mistake.

This means that there are two important aspects in a mistake: its type (for example *Wrong reference lines*) and the rule involved (for example *Modus Ponens*).

The student models, all centralized in one place on the server, contains the history of all exercises attempted and mistakes made. The LT-Analyser regularly scans all the student models and builds a database collating all that information. The database contains various tables. The table that is here most interesting for us is the table *Mistake* which provides an index to mistakes for each question attempted by each students. Each line (or record) of this table contains student's login (*login*), exercise id (*qId*), type of mistake made (*mistake*), rule involved (*rule*) and date. When part of the date is missing, the exercise is unfinished. The database is in Microsoft Access and is connected to Microsoft Excel. The teacher has then the choice of querying the database with either software and visualise graphics in MS Excel. The aim of the LT-Analyser is to provide information to the teacher about the class, so that s/he can adapt his/her teacher accordingly, or be aware of individuals needing assistance. Currently, we are extending the LT-Analyser with Data Mining facilities.

1.2 Clustering students

One need, for teachers, is to cluster students into homogeneous groups with respect to their abilities concerning the course material. In particular, it is important to be aware of groups with difficulties to take proper action. Two well known methods for clustering a population are k-means clustering and hierarchical clustering. In this section we present these methods, how we have used them on student answers from Logic-ITA and discuss the results.

K-means and hierarchical clustering

Both k-means and hierarchical clustering rest on a distance concept between individuals. Distance is taken here almost in its mathematical meaning with $d(x,x) = 0$, $d(x,y) = d(y,x)$ and $d(x,y) \leq d(x,z) + d(z,y)$ for any individuals x, y and z. It is the user's responsibility to fix a distance for the population before performing a clustering. Most of the time, changing the distance changes the resulting clustering.

K-means clustering partitions the populations into k-classes where k is fixed in advance by the user. The algorithm works as follows.

k-means clustering algorithm.
Select a population and a distance between individuals.
Choose k random individuals as initial centers.
Repeat
For each individual,
Calculate its distance with all the centers and
put it in the cluster with the nearest center.
Calculate new centers by taking the mean of each cluster.
Till there is no more change in the clusters.

Hierarchical clustering does not ask for an initial number of clusters. Rather, classification is stopped when the distance between two groups is too large and no longer guarantees the homogeneity of the individuals grouped together. There are several ways to measure the distance between two groups. One common way is to calculate the distance between any two pairs and take the average. The algorithm works as follows.

Hierarchical clustering algorithm.
Select a population and a distance between individuals.
Each individual forms an initial group.
Calculate distances between all groups to form a distance matrix.
While there is more than one group and distance between two
nearest groups is below a given threshold
Repeat
Cluster these two nearest groups into one.
Recalculate distances between all other groups
and this newly formed group.

The advantage of hierarchical clustering over k-means clustering is that its result is unbiaised by initial parameters. Indeed, the random choice of initial centers as well as a given number of clusters influence the result for k-means clustering. However, the complexity of k-means clustering is linear, while hierarchical clustering is polynomial. Therefore many Data Mining softwares offer a combination of both, known as *two-steps* clustering. First individuals are grouped according to k-means clustering, where k is chosen to be far larger that the number of expected clusters. Its effect is to reduce the original size of the population in linear time in something which is manageable for hierarchical clustering. Then, hierarchical clustering is performed on these k clusters to yield final clusters.

Use of the clustering with Logic-ITA

We have chosen to perform a *two-steps* clustering on the data collected from the Logic-ITA using the Data Mining software Clementine® [9].

The first thing to do is to select the population. Our aim was to cluster students who seemed to have difficulties. Therefore, we have taken the table *mistake* and selected only the lines corresponding to non-finished exercises. Indeed, finishing an exercise means being able to complete a full proof and, thus, correct all the mistakes made on the way. From the initial population of 149 students having made mistakes, we went down to 60 students.

The second thing to do is to fix a distance between two individuals. This is no trivial task. Indeed, students have not attempted the same exercises, nor the same number of exercises. Furthermore, a large numer of exercises attempted by students does not come from the database of exercises provided by the teacher. Indeed, students are free to type in their own exercises, or to try exercises from the exercise generator. These latter exercises have a unique identifier and an unknown level of difficulty. Thus, exercises attempted by students can hardly be compared by level. After various attempts, what worked best was the simplest one could do: count all mistakes made per student, which gives exactly one number and use the difference in absolute value as a distance between two students (which is equivalent in that case to the Euclidean distance): let $CountMistake_x$ and $CountMistake_y$ be the total number of mistakes made by student x and y respectively. We get $d(x, y) = | CountMistake_x - CountMistake_y |$. Running the algorithm led to two clusters:

- *cluster_1* containing 21 records (in our case students), average = 39.1, standard deviation = 15.6.
- *cluster_2* containing 39 records, average = 7.5, standard deviation = 4.7.

Thus students from *cluster_1* made a lot more mistakes than students from *cluster_2*.

Apart from their number of mistakes, is there any other way to qualify these two groups and interpret the result? To answer this question, we have used the graphical visualization of relations offered by Clementine® between clusters, rules and mistakes. To build the relationship, the number of occurrences of any two items in a line of the table is counted. For example, counter(*cluster_1*, *Modus Ponens*) is the total number of lines in the table where some student of *cluster_1* made a mistake with the rule *Modus Ponens*. Similarly, counter(*cluster_1*, *Wrong rule used*) is the total number of lines in the table where some student of *cluster_1* made a mistake called *Wrong rule used*. This relation is displayed graphically as a graph where the nodes are the different items: *cluster_1*, *cluster_2*, all rules (like *Modus Ponens*, etc...) and

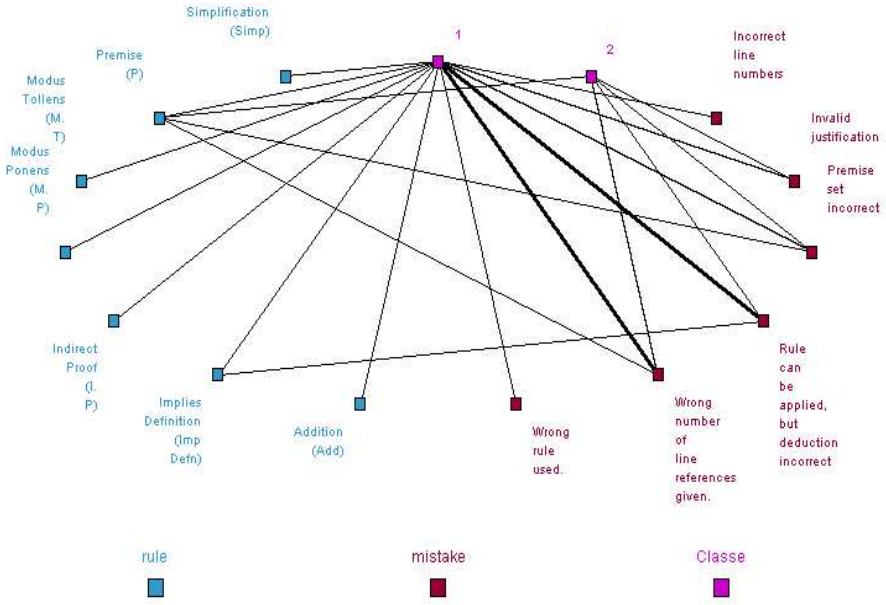

Figure 2. The relations between clusters, rules and mistakes with a threshold of 20 from Clementine®.

all mistakes (like *Wrong rule used* etc...). There is an edge between any two nodes if their counter is bigger than some number. The thickness of an edge is proportional to the counter value. In Figure 2 we show the graph where the threshold value for counters is 20 (a smaller value like 5 gives a spinweb!).

First, confirming our earlier results [6], this graph puts in evidence very common mistakes that belong to the core learning of formal proofs and that are made by almost all students. These mistakes are *Premise set incorrect*, *Rule can be applied, but deduction incorrect* and *Wrong number of line references given*. The first mistake indicates that the premises given by the student are wrong, the second one indicates that the formula given by the student is wrong and the third one indicates that the line numbers given by the student are wrong. It is interesting to note that two mistakes appear only with students from *cluster_1*. These are *Incorrect line numbers* and *Wrong rule used*. The mistake *Incorrect line numbers* indicates that the student provided only 1 line number when 2 where expected with the rule used, or vice-versa. The mistake *Wrong rule used* indicates that the rule cannot be applied to the lines given by the student. This graph shows also clearly that students of *cluster_1* have tried a lot more rules.

To go further in the comparison, we have selected the exercise that has been most often attempted by students, which happened to be the exercise with *qid* 1003. We have used the same clustering method as before selecting only students who have attempted that exercise without completing it. This gave 17 students. *Two-steps* clustering gives now four clusters.

- *c_1* containing 3 records, average = 29.7, standard deviation = 1.9
- *c_2* containing 6 records, average = 3.3, standard deviation = 1.5
- *c_3* containing 5 records, average = 8.6, standard deviation = 1.6
- *c_4* containing 3 records, average = 18.0, standard deviation = 2.2

Is this clustering consistent with the one obtained above? If one thinks of *c_2* and *c_3* as corresponding to *cluster_2* and *c_1* and *c_4* corresponding to *cluster_1*, then the two clusterings are quite coherent. All students, except for 1, belonging to *c_1* or *c_4* also belong to *cluster_1*, while almost all students belonging to *c_2* or *c_3* also belong to *cluster_2*. There are altogether 4 mismatches. One belongs to *c_4* and *cluster_2*. He made 16 mistakes on that exercise. Others belong to *c_2* or *c_3* and *cluster_1*. They have made 2 to 6 mistakes on that particular exercise, but many mistakes on the other exercises that they have attempted. The graph relationship for these four clusters is shown in Figure 3 and bears similarities with the previous one.

A further question is: where are the students who have not finished any exercise? Altogether 9 students have not been able to complete any exercise and made mistakes in the exercises they have attempted. A third of them is in in *cluster_1* and the rest is in *cluster_2*.

Discussion

From a teacher point of view, how can we interpret the clusters we have obtained? What do they suggest about the student learning? As a result of the bigger variety of rules used and mistakes made, students of *cluster_1* appear as if they would try out the tool, more than they use it thoughtfully to solve exercises. Students of *cluster_2* appear as being lost or as giving up quickly. We have performed a k-means clustering with the same students using TADA-Ed, a prototype research Data Mining tool [2]. Our tool has a point view module that allows to visualize *login* against *rule*. The resulting graph has vertical lines that shows clearly students trying one rule after the other from the menu (apparently relying heavily on a "guess and test" strategy). This agrees with our interpretation for *cluster_1*.

Analysing the mistakes made in unfinished exercises and finding these two clusters of students is very useful for the teacher as they give an insight on the reasons for the mistakes. The remedial actions a teacher can take are very

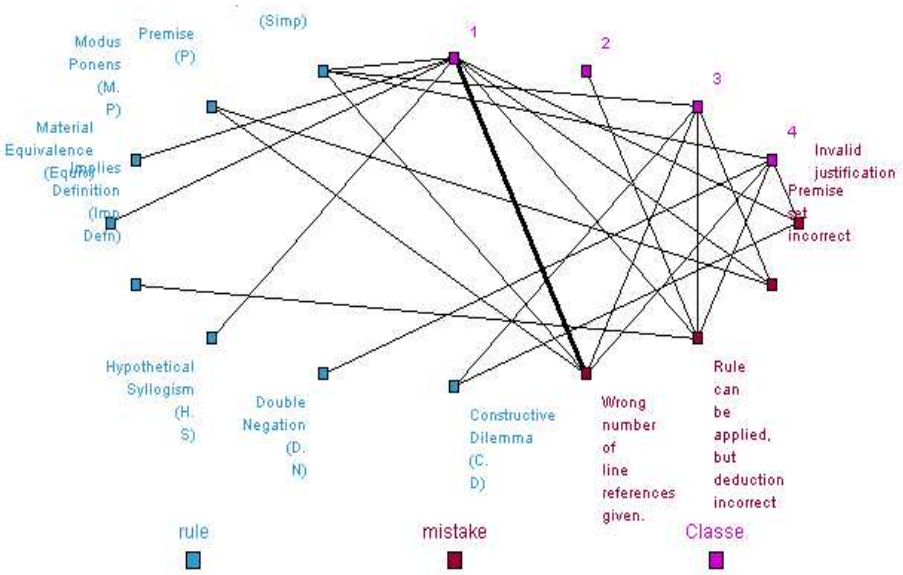

Figure 3. The relations between clusters, rules and mistakes for students not completing question 1003 with a threshold of 4 from Clementine®.

different depending on the efforts made by the students and the reasons for their failures. For those who rely on a "guess and test" strategy for example, the teacher would probably want to motivate these students to think before trying. Whereas for those who are at lost, the teacher would probably choose to readjust the level of difficulty of the exercises, reexplain the concepts and so on.

Note that the partitioning of students into these two groups would not be obtainable by only querying the database.

We have performed this clustering after the whole course took place, so we cannot validate anymore the interpretation of our results against another approach like human observations or surveys. We plan to validate this clustering approach during the coming academic year.

1.3 Conclusion

In this paper, we have reported our experiment on clustering students using their mistakes made with a web-based tutoring tool, the Logic-ITA. The key points in this experiment were (1) to choose an appropriate distance between students in a context of heterogeneous data, since the students have not necessarily attempted the same exercises, neither the same number of exercises,

nor the same level of exercises, (2) to extract useful information for the teacher about the type of difficulties and learning students had.

To perform the clustering, we have used the *two-steps* algorithm available in Clementine®, a commercial Data Mining software. To characterize further the clusters produced by the algorithm, the graphical visualization of relations has been useful. However, Clementine® does not have a plotting facility for nominal variables, thus it is not possible to plot and visualize for example *login* against *rule* on a graph. This confirms to us the need of a mining platform dedicated to a teaching context since the needs of a teacher trying to understand and evaluate his/her students' learning are different to the needs in a business context. Therefore, future work includes improving our Data Mining research tool [2], in particular its visualization facilities to help teachers in their interpretation.

Applying Data Mining to data collected from learning systems could bring new feedback to teachers, giving them unexpected insight on the learning of their students, and on their own teaching. This could lead to the concept of 'pedagogical intelligence' in a similar way as Data Mining to data from business has led to 'business intelligence'. To reach this goal, several sub-goals need to be met such as the selection of relevant algorithms and relevant ways to use them. This paper is a contribution to this sub-goal.

We plan to use similar clustering techniques to other data, in particular to some students homework in mathematics. These students have already been "clustered" using pedagogical criteria. It will be interesting to see what clusters we obtain, and how they compare with the ones already obtained by teachers.

References

[1] Abraham D., Crawford L., Lesta L., Merceron A. and Yacef K., *The Logic Tutor: A Multimedia Presentation*, Interactive Multimedia Electronic Journal of Computer-Enhanced learning, Vol. 3, Nb. 2, Nov. 2001.

[2] Benchaffai M., Debord G., Merceron A., and Yacef K., *TADA-Ed, a tool to visualize and mine students' work*. Submitted paper. 2004

[3] Bisson G., Bronner A., Gordon M.T., Nicaud J.-F., Renaudie D., *Analyse statistique de comportements d'élèves en algèbre*, Proceedings of Environnement Informatiques pour l'Apprentissage Humain, Strasbourg, France, pp.67-78, 2003

[4] Han J., and Kamber M., *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2001

[5] Lesta L. and Yacef K., *An Intelligent Teaching-Assistant System for Logic*, Proceedings of Intelligent Tutoring Systems, Biarritz, France, Springer-Verlag, June 2002.

[6] Merceron A., and Yacef K., *A web-based tutoring tool with mining facilities to Improve Teaching and Learning*. Proceedings of the 11th International Conference on Artificial Intelligence in Education, IOS Press, Sydney, Australia, pp.201-208 2003

[7] Romero, C., Ventura S., de Castro C., Hall W. and Ng M.H., *Using Genetic Algorithms for Data Mining in Web-based Educational Hypermedia Systems*. In Workshop on Adaptive Systems for Web-based Education, Malaga, Spain, 2002.

[8] Tang T.Y., McCalla G., *Student Modeling for a Web-based Learning Environment: a Data Mining Approach*. Eighteenth national conference on Artificial intelligence, Edmonton, Alberta, Canada, pp.967-968, 2002

[9] http://www.spss.com/clementine/

[10] Zaiane, O.R. *Web Usage Mining for a Better Web-Based Learning Environment*. Proceedings of Conference on Advanced Technology for Education (CATE'01). Banff, Alberta 2001.