Failure Analysis in University and Computer Science Contexts With Data Mining

Daniela de Souza Gomes[1], Marcos Henrique Fonseca Ribeiro[1], Giovanni Ventorim Comarela[2], Gabriel Philippe Pereira[1]

[1]Departamento de Informática – Universidade Federal de Viçosa

[2]Departamento de Computação e Eletrônica – Universidade Federal do Espírito Santo

{daniela.s.gomes, marcosh.ribeiro, gabriel.philippe}@ufv.br

gcomarela@inf.ufes.br

Abstract. *High failure rates are a worrying and relevant problem in Brazilian universities. From a data set of student transcripts, we performed a study case for both general and Computer Science contexts, in which Data Mining Techniques were used to find patterns concerning failures. The knowledge acquired can be used for better educational administration and also build intelligent systems to support students' decision making.*

Resumo. *O alto índice de reprovação é uma questão recorrente e preocupante nas universidades brasileiras. Pretende-se então, com este trabalho identificar padrões relacionados à essas reprovações sobre uma base de históricos acadêmicos utilizando métodos consagrados de mineração de dados.*

1. Introduction

Educational Data Mining (EDM) is the field related to the discovery of knowledge from educational environments data [Baradwaj and Pal 2011]. Machine learning models could be used to predict, for instance, dropouts, grades, and failures. One alarming issue in Brazilian Universities is high failure rates. An investigation of this problem would be very useful for better educational planning and management. A similar study can be done for narrower scenarios, like the Computer Science (CS) context. The Computer Science Department from the Federal University of Viçosa[1] (known as DPI and UFV, respectively) is responsible for offering Computer Science Degrees at both undergraduate and graduate levels. Besides that, it offers programming related courses for other majors (which will be called INF courses, from now on). This context is relevant because few students graduated at CS within the proposed time in the last decade and some of those INF courses belong to the set with high failure rates outcomes at UFV.

Thus, our objective is to apply Data Mining (DM) techniques to real data, extracted from UFV historical academic database, to acquire and/or validate knowledge that can be further used for better administration, as well as for building intelligent and knowledge-aware computational systems to support academic counseling. In this work, we analyze failure causes in three different perspectives: among UFV students in general, among CS students alone and INF courses students. This approach intends to compare

[1]This research was partially funded by Capes and UFV's Education Office.

the CS scope to the general one and to investigate which aspects come from the whole context and which are specific and may require special solutions. This paper is organised as follows. Section 2 covers the related works in EDM, while database construction and knowledge extraction are described in section 3. Following, we show the results in section 4 and some conclusions in section 5.

2. Related Works

[Baradwaj and Pal 2011] investigated students' failure, a relevant task in EDM, using a decision tree to evaluate students' performance, grouping them according to their results. [Raji et al. 2017] addressed the problem by proposing a visual analysis system, using over 16 years of collected data, to describe the students' progression. The systems allowed them to question some school policies. Our work is part of ongoing research aimed at building a context-aware model capable of analyzing and predicting the probability of students' failure. Therefore, the study presented here focuses on acquiring and preparing knowledge to support the future development of such model.

3. Dataset and Knowledge Extraction

UFV provided data[2] with records since 2003, wherefrom we built a structured database, composed of 50 majors, 2492 courses, 23636 students, 1832 professors, and 819326 detailed student transcripts (courses taken, semester, professor, grades, etc). However, a smaller data set was extracted following some criteria determined by the University itself, considering only courses with an attendance average of at least 25 students and with failure rate over 40% in, at least, one occasion and also, transcripts from students enrolled from 2013 on. It's worth mentioning that courses statistically considered outliers were removed. The final dataset ended up with a total of 70566 records.

Once data set was created, the next step was trying to extract knowledge from it, represented by patterns. But first, we needed to pre-process the data, to avoid redundant information and scaling problems. It basically consisted of reducing dimensionality and normalization. We started by analyzing the distribution of the admittance grade at UFV versus their performance within the University, which presented a normal distribution, showing that such grade barely affects the students' progress during their majors. Next, failures were confronted against 4 other different attributes: major, UFV adherence method, geographic origin, and quota mode used to join the university. Only in the first case, some relevant relation could be found, indicating that some majors have more failure related issues than others. Then, we applied a Random Forest [Han et al. 2012] classifier to predict students' failure. This approach proved to be too naive to deeply understand the problem. The confusion matrix showed a 63% accuracy when predicting students' approval and 59% when predicting failure. Although the results above are not sounding, they suggest that some patterns related to this problem may be found and also that a more sophisticated classification process can be promising. Both kinds of results could be applied to support better studying planning and help to avoid failure before it happens.

Frequent Patterns Mining looks for recurring relationships in a data set [Han et al. 2012]. A way to represent these patterns are Association Rules [Agrawal et al. 1993], an implication in the form $X \implies Y$, indicating that, if

[2]Personal data, such as student and professor identification, was provided anonymized.

a set X of items occur in a database, there is a probability, with a certain confidence, of a set Y also occurs. X and Y must not be empty sets as well as their intersection. X is called antecedent, while Y is the consequent of a rule. Frequent patterns are easily mined in transactional databases so our data set had to be converted into one. A transaction is an occurrence (a set) of items that appear together. In this version of our database, each transaction reflects a semester in a student's transcript. They are composed by year and semester, courses taken at that period, how many class hours those courses demand, professors responsible for each class, college entrance mode, admittance grade, major, and how many semesters the student has been in UFV, so far. Following the conversion into a transactional database, two algorithms were executed: Apriori [Han et al. 2012] which extracts frequent itemsets and one for detecting Association Rules [Han et al. 2012]. We used the implementation present in the Mlxtend Python library [Raschka 2018] which requires some input parameters, such as minimum support and confidence. According to [Agrawal et al. 1993], the support gives the proportion of transactions containing an item set X and denotes statistical relevance. Confidence is defined as the probability of Y occurs, given that X occurs in a transaction.

We extracted 47 rules from that procedure. Even though they may be considered strong rules, in general, they showed a tendency to only confirm what was expected, for example, semesters in which a student takes many courses associated with a great probability of failing in them. A considered hypothesis then was that maybe there are different course profiles. Each profile would have different values for attributes like failure rate, number of students, dropouts, or other relevant aspects. Considering all these profiles together might increase the difficulty in finding useful patterns, due to more chaotic behavior. Segmenting the data set by grouping similar courses and then searching for patterns inside such groups could help to find stronger and/or more unexpected rules. Since there was no previous categorization of courses in UFV, we performed a study to find "natural" groups with high similarity between elements inside a same group and low similarity between elements from different groups. This task is known as clustering and is well studied in literature. The algorithm chosen for this task was K-means, that demands a parameter, referred to as K, the desired number of clusters. One way of finding the best K is the "elbow method analysis". It says that increasing the number of clusters can help to reduce the sum of the within-cluster variance of each group [Han et al. 2012]. The best value found was $K = 4$, after testing values from 2 to 10. Once defined K, it was important to check if the clusters were different enough to represent course profiles. One possible way of doing so is to compare their centroids, virtual representatives elements from each group usually computed as having the average values of all attributes from all individuals in each cluster.

Each plot inside Figure 1 represents the values of each attribute, for each cluster centroid. Their different shapes indicate that they have distinct characteristics, so we were able to find and isolate different profiles among the courses. The labels are omitted for better legibility, but we considered attributes such as averages and variances of some historical aspects of the courses and number of students, for example. To find patterns considering those clusters, the algorithm of Association Rules extraction was executed again over subsets of the original database, filtered by such groups. Yet, two other filters were also applied: one containing only records from computer science students (referred to as *CS data set*) and another with transcripts containing some introductory programming

courses (referred to as *INF data set*). The results are presented and discussed next section.

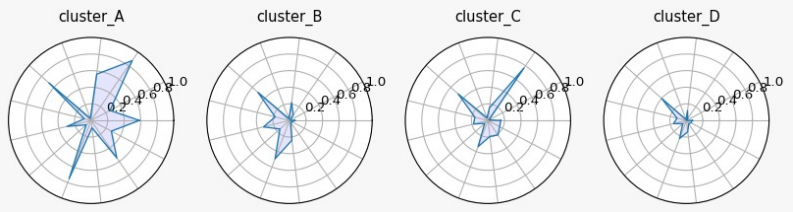

Figure 1. Centroid shapes.

4. Results and Discussion

When analyzing the rules extracted before clustering, it was noticed that there were two major kinds of rules, both with high values for confidence and also recurrent. Rules involving failure at one course leading to a high chance of failing in another one, like the example in the second row of Table 1, and a specific group of courses implying in a high probability of failing in another course, like the example in the first row of the same table. Just to illustrate, it is worth saying that the general probability of failing in MAT 146 is around 69%, independently of any other factor. When considering attending BIO 131 and FIT 190 simultaneously, such probability increases up to around 85%, indicating that this specific set of courses should be avoided or receive greater attention when students and their advisors build the study plans.

Table 1. Top 5 strongest rules, according to confidence, found before clustering

Antecedents	Consequents	Support	Confidence
(BIO 131, FIT 190)	(MAT 146-R)	0.010687	0.857685
(QUI 107-R)	(QUI 100-R)	0.011112	0.828924
(FIT 190)	(MAT 146-R)	0.012200	0.787786
(BIO 112-R)	(BIO 111-R)	0.012744	0.735334
(MAT 141-R)	(FIS 201-R)	0.017780	0.594466

After clustering, it was possible to find more rules and with higher confidence values, as expected. Another effect was to find more diverse kinds of rules, like ones involving specific professors or the number of class hours taken by the student. One undesirable side effect was that there were many rules in which their antecedent are subsets of antecedents of other rules. When this phenomenon does not affect confidence, it is just redundancy, leading to an overload of information. Table 2 shows the top strongest rules found after clustering. One can notice that the increase of confidence was not sounding, but it is compensated by the increased number of rules extracted (over 100 versus the original 47), support and variety of rules, which may lead to a richer analysis and provide more knowledge from the data. Narrowing the scope to Computer Science context, on INF data set the process returned 153 rules while in CS data set were found 36 rules, both with confidence over than 98%, which are considerably stronger results. Most of the rules in INF data set are related to failure in two specific courses: Fundamentals of Elementary Mathematics (MAT105) and Physics II (FIS202), denoting problems related to basic subjects, once rules involving other introductory courses could also be found. In the CS data set, one remarkable result is the behavior of Introduction to Algebra (MAT131). Its general average percentage of failure is about 69%. When a student is taking 5 courses or over, the chances reach up to 89%, while when a specific professor teaches it reaches

98% of chance of failure. It's an alarming result since MAT131 is an extremely important content to the CS core curriculum. These rules should be used as warning notifications in the development of an intelligent study planning system[3].

Table 2. Top 5 strongest rules, according to confidence, found after clustering

Antecedent	Consequent	Support	Confidence
(BIO 131_A, course: 103.0, num_courses: 7)	(MAT 146_R)	0.048560	0.898585
(BIO 131_A, course: 103.0, FIT 190_A, num_courses: 7)	(MAT 146_R)	0.046393	0.896552
(BIO 131_A, FIT 190_A, num_courses: 7)	(MAT 146_R)	0.046393	0.896552
(course: 103.0, FIT 190_A, num_courses: 7)	(MAT 146_R)	0.050089	0.895216
(FIT 190_A, num_disc: 7)	(MAT 146_R)	0.050089	0.895216

5. Conclusions

Given the results shown, it could be observed that students of some specific majors have difficulty in specifics courses and the STEM courses largely occur in the rules, suggesting complications in these study fields too. We also noticed that many primary courses were found among the rules, which can indicate a deficiency in basic education or the Federal Universities selection system. Further research could follow that direction. Among CS undergraduates, it's clear that first-year students are the ones with more problems and maybe that's because, in Brazil, programming is not largely taught in high school. Once more, future research could investigate that. More knowledge concerning structural problems related to education administration, study planning, and majors could be extracted and can be further used both by UFV and by researchers in educational fields, especially those from the CS area. Finally, it is very important to notice that all the work done here can be easily applied to similar contexts, because despite many universities and schools have their own systems to manage educational data, many Brazilian universities have a lot of characteristics in common, especially the federal ones. For future work, these rules are intended to be used at building alert systems in curriculum planning and be delivered to the UFV administration to elaborate better teaching policies. Our work has numerically confirmed some evidence of common sense. These indications can now be better investigated to create proposals for solving problems.

References

Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington D.C.

Baradwaj, B. K. and Pal, S. (2011). Mining educational data to analyze students performance. *International Journal of Advanced Computer Science and Applications*, 2(6).

Han, J., Kamber, M., and Pei, J. (2012). *Data mining concepts and techniques, third edition*. Morgan Kaufmann Publishers, Waltham, Mass.

Raji, M., Duggan, J., DeCotes, B., Huang, J., and Zanden, B. T. V. (2017). Visual progression analysis of student records data. *2017 IEEE Visualization in Data Science (VDS)*, pages 31–38.

Raschka, S. (2018). Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *The Journal of Open Source Software*, 3(24).

[3]The complete set of results is available at https://github.com/DaniGomes/failure-analysis