Mining Coherent Evolution Patterns in Education through Biclustering

André Vale

Instituto Superior Técnico, Av. Rovisco Pais,
1049-001 Lisboa, Portugal
andre.vale@tecnico.ulisboa.pt

Abstract. Educational Data Mining (EDM) focus in the development of methods for exploring the types of data that come from an educational context. In this dissertation, we studied the inclusion of an unsupervised technique, *Biclustering* that has been successfully applied in areas such as gene expression and information retrieval, but not used in the educational context. We presented a methodology that allows us to use *Biclustering* algorithms in educational data to get new patterns and use these results as a complement to the classification. By applying this new technique we can improve the accuracy of the classifiers, similarly to other techniques previously used, finding new types of patterns which until now had never been discovered.

Keywords: Educational Data Mining, Biclustering, Coherent Evolution Patterns, Student's Performance

1 Introduction

On the last decade, there has been an increasing interest in applying data mining methods on educational data, making Educational Data Mining (EDM) a new growing research community. The prediction of students' performance has deserved a significant attention in Educational Data Mining (EDM) research, with several distinct approaches being proposed, mostly using classification and regression techniques.

While classification tries to find a model to predict an outcome, non-supervised techniques, as pattern mining and clustering, can explore the data for identifying frequent behaviours. Previous studies (Antunes, 2008; Barracosa & Antunes, 2011) have shown that sequential pattern mining is suited to discover patterns able to model students behaviours, which in turn can be used to enrich training data, improving global classification accuracy on more than 10% (Barracosa & Antunes, 2011).

Biclustering algorithms (S. Madeira & Oliveira, 2004) are a recent alternative to traditional clustering methods that allows the discovery of local patterns rather than global ones. Besides discovering sequential patterns identified by pattern mining algorithms, biclustering is able to discover other sequential patterns that reveal coherent evolutions (Ben-Dor & Chor, 2003; Murali & Kasif, 2003).

This dissertation intends to study the use of biclustering to discover patterns in educational data. Algorithms that are currently state of the art in the area will be studied and analyzed the suitability and adaptability to the problem of analysis of educational data. We will explore the ability of biclustering to discover new patterns in educational data and make use of these patterns to enrich training data in

order to improve the prediction of students' performance. Finally at the end of the dissertation approaches based on biclustering for analyzing educational data will be proposed which will be tested using data from the *Educare* project[1].

This, to the best of our knowledge, is the only work that effectively applies biclustering algorithms, to educational data, what allowed us to understand the advantages and disadvantages of these methods in the field of EDM. The results show that biclustering can slightly improve the classifiers and that we can extract important information through the obtained biclusters.

The rest of this document is organized as follows: in **Section 2** we present a brief Literature Review. **Section 3** presents our dissertation statement. In **Section 4** we describe our main case study, followed by the experiments performed and a comparison with a method of pattern mining. **Section 5** contains our second case study, a matrix with a time variable. Finally, in **Section 6**, we present the conclusions of this work and point some future work.

2 Literature Review

2.1 Biclustering

Biclustering can be applied whenever the data to analyze has the form of a real-valued or symbolic matrix A, where the value a_{ij} represents the relation between row i and column j, and the goal is to identify subsets of rows with certain coherence properties in a subset of the columns. The goal of biclustering algorithms is to identify a set of biclusters. Let A be a matrix defined by its set of rows, R, and its set of columns, C. Then we can define bicluster as follows (S. C. Madeira, Teixeira, Sá-Correia, & Oliveira, 2010):

Definition 1 (Bicluster). A bicluster $B = (I, J)$ is a submatrix A_{IJ} defined by $I \subseteq R$, a subset of rows, and $J \subseteq C$, a subset of columns. A bicluster with only one row or one column is called *trivial*.

2.2 Related Work on EDM and Biclustering

The literature about biclustering on educational data is almost non-existent. Only recently Trivedi et al. proposed this technique on education area (Trivedi, Pardos, Sarkozy, & Heffernan, 2012). In their work they used the idea of co-clustering (namely biclustering) students and their tutor interaction features and interleave it with a bagging strategy which was used previously with clustering (Trivedi, Pardos, & Heffernan, 2011; Trivedi, Pardos, Sárközy, & Heffernan, 2011).

The results that they obtained were better than the baseline and also indicated that the dynamic assessment condition returns a much better prediction of student test scores as compared to the static condition. However, this technique has some limitations. The datasets that were used were not vast and did not have a large number of columns (only six). Moreover, they used the *k-means* algorithm twice, on columns and rows, not getting such good results as would obtain using biclustering algorithms.

[1] Project *Educare* - https://sites.google.com/site/istprojecteducare/

3 Dissertation Statement

This dissertation is intended to demonstrate the added value that the use of biclustering brings to the discovery of information in educational data.

Biclustering has already been used with very success in other areas, with this study we purpose to bring the application of biclustering for the first time in education. We intend to realize if the biclustering in EDM can bring new results that have not been previously accomplished. Biclustering also allows us to discover information that other techniques cannot easily find in EDM.

In order to reach some conclusions from the application of biclustering in EDM, we decided to study two cases in particular:

- Case study 1: We applied four biclustering algorithms that allow to find different types of patterns. We used these biclusters to improve classifiers and compared the results with another technique that finds some similar patterns that had already been used in EDM.

- Case Study 2: We applied biclustering to a matrix that has the time variable, which allowed us to realize how biclustering behaves when we have time educational data.

With these case studies we intend to introduce and study the suitability and adaptability of biclustering algorithms to the problem of analysis of educational data.

4 Case Study - Students' Grades

The dataset used as input was collected in the Educare project[2], a project that studies the Computer Science students' of Instituto Superior Técnico (IST) from Universidade de Lisboa between 1997 and 2012.

4.1 Data Analysis

To reach the final dataset, we had to pre-process the data. In order to simplify the pre-processing, we decided to use only the students who entered after Bologna Process because we do not need to do an equivalence between the subjects before and after the Bologna Process. Hence, data has two main groups of students: from graduation (LEIC) and master (MEIC) programs. LEIC has a duration of three years (6 semesters) with 30 subjects and MEIC is a specialization of LEIC that has a duration of two years (4 semesters) with 15 subjects.

We obtained a Bologna dataset with 3764 instances. First, we removed those who has less than 20 approved subjects (out of 30). With this we removed students who dropped the course and those who have not made many subjects. We managed to reduce the percentage of missing values (which are the subjects that have not been made by students) from 68.32% to 8.64% obtaining a dataset with 648 instances. Then we removed all students who have not yet started MEIC, i.e., those who still have no subjects approved in MEIC, so we could calculate an average for each student in MEIC. Later we will this MEIC average as a class in the classifiers. In the end we obtained a matrix with 442 students and 30 subjects with only 4,69% of missing values.

[2] Project *Educare* - https://sites.google.com/site/istprojecteducare/

Thus, we built our matrix with data from LEIC (students x subjects matrix) where we have students in the rows and subjects in the columns, and the values in the matrix are the students' grades. These values are numbers between 10 and 20.

In the case of student's grades, a training dataset can be created using the *n* subjects from LEIC as attributes for the *m* instances of students, with the class being the average grades at MEIC, discretized into 3 bins (Fair, Good and Very Good). As normal, the *Very Good* class is in a smaller number than the other classes. In this manner is possible to train a model to anticipate the average grades of new students enrolling in MEIC.

To obtain more reliable results with biclustering analysis, it was decided to discretize the matrix in order to get a small number of symbols. Instead of having grades between 10 and 20 (corresponding to 11 symbols), we produce a dataset with five symbols, A to E.

Thus, in the end we got a matrix with a configuration equal to the **Table 1** where we have values of $x \epsilon$ {A, B, C, D, E} and $y \epsilon$ {F, G, VG}.

Table 1. ***Grade matrix having in the last column the corresponding Class for each student.***

	Subject 1	…	Subject *j*	…	Subject *m*	Class
Student 1	x_{11}	…	x_{1j}	…	x_{1m}	$y_{1(m+1)}$
Student …	…	…	…	…	…	…
Student *i*	x_{i1}	…	x_{ij}	…	x_{im}	$y_{i(m+1)}$
Student …	…	…	…	…	…	…
Student *n*	x_{n1}	…	x_{nj}	…	x_{nm}	$y_{n(m+1)}$

4.2 Biclustering Analysis

After pre-processing, we applied the four biclustering algorithms mentioned before to the matrix and obtained 16 biclusters with *OPSM*, 975 with *xMotifs*, 308 with *ISA* and 39 with *Bimax*. With the biclusters obtained, our goal is to enrich a training dataset in order to improve the accuracy of the classification that predicts the average grades of students of MEIC.

The biclusters were found using the *BicAT* tool (Barkow, Bleuler, Prelic, Zimmermann, & Zitzler, 2006). We can observe that each algorithm has a most predominant number of subjects, for example, *OPSM* find patterns where frequently appears ACED, EO, PO and IPM. On the other hand, in *Bimax* mainly appears CDI-I, CDI-II and PEst subjects with the highest incidence.

4.3 Evaluation

As in (Barracosa & Antunes, 2011), a new dataset can be obtained from the previous one, enlarged *k* Boolean attributes, one for each bicluster. Each bicluster attribute is then filled with the true value whenever the bicluster has the student instance and false otherwise. As such, the new instances take the following format:

Subject1, Subject2, … , SubjectN, Bicluster1, Bicluster2 , … , BiclusterK, Class

Methodology.

To help us on feature selection, cross-validation and classification, we used the data mining open source software *Weka* 3.6.10[3] (Hall et al., 2009). The data flow used in *Weka* consists of first loading the data, using a feature selection method to get only the best columns, making cross-validation with 10 folds since we did not have defined training and test sets, and finally applying a classifier for obtaining the results (**Figure 1**).

Figure 1. ***Data Flow used.***

Feature Selection.

Feature Selection (FS), sometimes referred as data selection, discover relevant features in a dataset. This step is of huge importance to simplify the data, and minimize the confusion presented to the classifier. We use a FS method that allowed us to choose the best attributes of our dataset, including the best biclusters. In order to understand what is the best FS method for our problem, all these existing methods of feature selection in *Weka* were tested (Hall et al., 2009): *ChiSquaredAttributeEval*, *ClassifierSubsetEval*, *ConsistencySubsetEval*, *CfsSubsetEval*, *FilteredAttributeEval*, *FilteredSubsetEval*, *WrapperSubsetEval*. These methods were run with every compatible search method from *Weka*.

Cross-validation.

K-fold cross-validation partitions the dataset into *k* mutually exclusive subsets, called folds, with approximatively the same size. In cross-validation, each fold contains the same proportion of instances from each class. Training and testing are executed *k* times, using in each iteration, one fold in the last and the others in the older ones. During the learning, each fold is used just once for testing and *k-1* times for training. The accuracy estimate is given by the overall number of correct classifications from the *k* iterations, divided by the amount of tuples in the initial dataset. In all the tests was used cross-validation with 10 folds and a random seed with value of 1 for the performance evaluation of classification algorithms.

Classification.

As we explained, we have three classes in our dataset {Fair, Good, Very Good} that will serve to train a model to anticipate the average grades of new students enrolling in MEIC. For the Classification, we used AdaBoost (AB) (Freund & Schapire, 1995), Decision Trees (DT) (Quinlan, 1993), Naïve Bayes (NB) (John & Langley, 1995) and Random Forest (RF) (Breiman, 2001).

[3] *Weka 3: Data Mining Software in Java* - http://www.cs.waikato.ac.nz/ml/weka/

Results.

As the *WrapperSubsetEval* method was the one that had the best results, we did experiments putting the biclusters as attributes and compare the two classification algorithms who obtained better results with the FS, Naive Bayes and Decision Tree. Naive Bayes always obtained better results compared to the Decision Tree, with or without biclusters.

Figure 2 shows the result of applying the classification without biclusters and with biclusters. We can confirm that there are improvements in classifiers with biclusters compared with the results without biclusters. The accuracy of *Naive Bayes* got an increase of more than 5% (from 60.9% to 66.5%).

Figure 2. ***Overall results.***

4.4 Biclustering and Frequent Item-Set Mining

Algorithms for frequent pattern mining can be classified into three categories, according to the types of patterns that they aim to obtain: transactional, sequential or structured pattern mining. We will only focus on the transactional algorithms since it approaches the type of biclustering patterns obtained in previous section.

Transactional Pattern Mining can be defined as an item-set of a non-empty set of items that appear together in a transactional dataset. The patterns can be expressed in the form of association rules, as proposed originally in (Agrawal & Srikant, 1994).

Let $L = \{i_1, i_2, \ldots, i_n\}$ be a set of literals representing *items*. The problem of discovering frequent item-sets can be formally stated as follows: let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq L$ and has a *TID* (transaction id), which is a unique identifier. X is a set of some items in L, or a *k-item-set* if it contains k items. A transaction is defined as a tuple $T = (TID, X)$. A transaction is said to contain X if $X \subseteq T$.

So, in order to have a base for comparison with the results obtained, we decided to compare with the Frequent Item-Set Mining (FIM) using the same methodology we used for the biclusters. We applied the discovered patterns as features to use in classification.

The algorithm used to obtain the patterns was FP-Growth (Han, Pei, & Yin, 2000) with a minimum support of 2%. The value of 2% was chosen since we only have biclusters with more than ten students and 10/442 (size dataset) = ~ 2%. We initially got 26067 patterns, we removed all patterns that have only 1 or 2 subjects because they are not relevant, so we finished with 22323 patterns.

As we expected, the patterns found are very similar to the patterns found by *ISA* and *Bimax* algorithm because they can find the same type of patterns.

We then placed the patterns obtained with FIM on a matrix, like we did with the biclusters, and run a classifier (*Naive Bayes*, in this case) and finally compared the results obtained with the results obtained with the biclusters (**Figure 3**). As expected, we obtained better results with the biclusters since we can find more types of patterns than with Frequent Item-Set Mining.

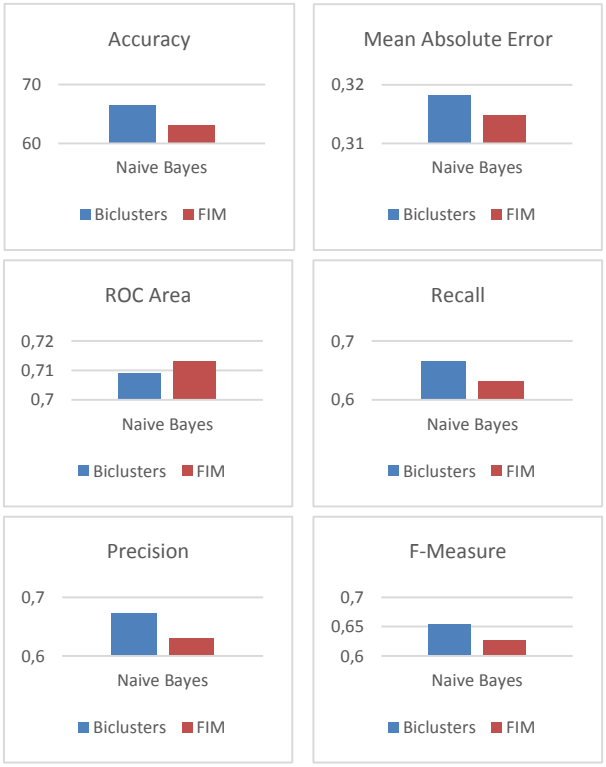

Figure 3. ***Comparison between classification with biclusters and classification with FIM patterns.***

5 Case Study - Subjects' Approval Rate

For this case study we decided to use a different type of matrix where in this case we use in the column a time variable to realize if we can get some interesting results with the biclustering patterns applied to the EDM. First we will make a brief introduction to biclustering with time series and its related work.

5.1 Biclustering Over Time

Biclustering with time series are typically used for expression time series in gene expression. The great difference of biclusters with time series for biclusters that we have mentioned previously is that, in this case, we are always interested to have biclusters that have continuous columns, which correspond to coherent expression patterns shared by a group of rows in consecutive time points. *CCC-Biclustering* algorithm (S. C. Madeira et al., 2010) was developed to work with genes and uses a generalized suffix tree to identify, in time linear on the size of the expression matrix, all maximal biclusters with contiguous columns that exhibit coherent expression evolutions over time.

Then, we will see how this algorithm behaves in educational data on a specific matrix and understand whether these patterns bring somewhat important information for education.

5.2 Data Analysis

In this case study we also used data from the *Educare* project[4]. But we made a matrix that has the subjects in rows and a time variable in columns (semesters of the subjects). The values of the matrix corresponds to the number of approved students, in percentage, to the subject of the respective semester.

The matrix that we used for this case study has 30 subjects of graduation program (LEIC) in rows and six semesters of these subjects in columns, which is related to six academic years (annual subjects). Unfortunately it is a matrix that has a small number of instances, but this is one of the major problems of EDM, the obtaining of data with many instances.

The data of rate approvals was in percentage with a range between 0% and 100% (0% corresponds to saying that zero students were approved and 100% that all enrolled students were approved), however, the matrix was discretized to CCC-biclustering algorithm present the patterns properly (to five symbols).

5.3 Biclustering Analysis

To obtain the bicluster patterns we used a tool that has the CCC-Biclustering algorithm implemented, the *BiGGEsTS* (Gonçalves, Madeira, & Oliveira, 2009). We obtained 39 patterns, which for the size of the matrix are not few. Generally the patterns are small, with an average of 3 rows by 3 columns.

---

[4] Project *Educare* - https://sites.google.com/site/istprojecteducare/

5.4 Evaluation

We can recognize that the patterns of biclusters have some interesting behaviours. For example, we can realize that the subjects PO and SS have almost always the same behaviour. We can realize that in semester S5 occurred changes in TC and Redes subjects who did improved the approval rate. And, for example, in semester S3 the rates decreased, which may indicate some evidence of some negative event that happened in that period.

6 Conclusion

In this dissertation, we have proposed to explore biclustering to discover new patterns in educational data and make use of these patterns to enrich training data in order to improve the prediction of students' performance. Something that proved difficult was the pre-processing, because the data had very few instances and had many missing values. We also had to test various parameters of biclusters algorithms to reach to more consistent results.

Results were satisfactory and the objectives achieved were successful. The different studies have enabled a new perspective on the application of biclustering in education context. In conclusion, we sincerely hope that this work be a starting point to new collaborations about biclustering in EDM.

There are multiple possible problems with different type of educational data that can bring new interesting results and that should be explored. Many other approaches can be tested but one that we think would be interesting to experiment are three-dimensional (3D) datasets. There are already algorithms that address three-dimensional matrices that can be adapt to the context of education. For example, we have the *triCluster* (Zhao & Zaki, 2005). Another interesting future work would be to develop more efficient techniques to choose the biclusters that have more relevance for the problem at hand.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proc. 20th Int. Conf. Very Large Data Bases, VLDB, 1215*, 487–499. doi:10.1.1.40.6757

Antunes, C. (2008). Acquiring Background Knowledge for Intelligent Tutoring Systems. *EDM*.

Barkow, S., Bleuler, S., Prelic, A., Zimmermann, P., & Zitzler, E. (2006). BicAT: a biclustering analysis toolbox. *Bioinformatics (Oxford, England)*. doi:10.1093/bioinformatics/btl099

Barracosa, J., & Antunes, C. (2011). Anticipating teachers' performance. *KDD 2011 Workshop*.

Ben-Dor, A., & Chor, B. (2003). Discovering local structure in gene expression data: the order-preserving submatrix problem. *Journal of Computational*. doi:10.1089/10665270360688075

Breiman, L. (2001). Random forests. *Machine Learning*, 5–32. doi:10.1023/A:1010933404324

Freund, Y., & Schapire, R. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory*, *55*(1).

Gonçalves, J. P., Madeira, S. C., & Oliveira, A. L. (2009). BiGGEsTS: integrated environment for biclustering analysis of time series gene expression data. *BMC Research Notes*, *2*, 124.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software. *ACM SIGKDD Explorations*, *11*(1), 10–18. doi:10.1145/1656274.1656278

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *ACM SIGMOD Record*. doi:10.1145/335191.335372

John, G. H. G., & Langley, P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. In *CONFERENCE ON UNCERTAINTY IN ARTIFICIAL INTELLIGENCE. Montreal, Quebec, Canada* (Vol. 1, pp. 338–345). Morgan Kaufmann.

Madeira, S. C., Teixeira, M. C., Sá-Correia, I., & Oliveira, A. L. (2010). Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Madeira, S., & Oliveira, A. (2004). Biclustering algorithms for biological data analysis: a survey. *Biology and Bioinformatics, IEEE*.

Murali, T., & Kasif, S. (2003). Extracting conserved gene expression motifs from gene expression data. *Pacific Symposium on Biocomputing*.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. (M. Kaufmann, Ed.)*Morgan Kaufmann San Mateo California* (Vol. 1, p. 302). Morgan Kaufmann.

Trivedi, S., Pardos, Z. A., & Heffernan, N. T. (2011). Clustering Students to Generate an Ensemble to Improve Standard Test Score Predictions. In A. Biswas, G and Bull, S and Kay, J and Mitrovic (Ed.), *AIED* (Vol. 6738).

Trivedi, S., Pardos, Z. A., Sarkozy, G. N., & Heffernan, N. T. (2012). Co-Clustering by Bipartite Spectral Graph Partitioning for Out-of-Tutor Prediction. *International Educational Data Mining Society*. Retrieved from http://eric.ed.gov/?id=ED537191

Trivedi, S., Pardos, Z. A., Sárközy, G. N., & Heffernan, N. T. (2011). Spectral Clustering in Educational Data Mining. In *Educational Data Mining*.

Zhao, L., & Zaki, M. J. (2005). triCluster : An Effective Algorithm for Mining Coherent Clusters in 3D Microarray Data. *Sigmod*, 694–705.