Using Large Margin Nearest Neighbor Regression Algorithm to Predict Student Grades Based on Social Media Traces

Florin Leon[1] and Elvira Popescu[2(✉)]

[1] Department of Computer Science and Engineering,
Technical University "Gheorghe Asachi" of Iasi, Iasi, Romania
florin.leon@tuiasi.ro
[2] Computers and Information Technology Department,
University of Craiova, Craiova, Romania
popescu_elvira@software.ucv.ro

Abstract. Predicting students' performance is a popular objective of learning analytics, aimed at identifying indicators for learning success. Various data mining approaches have been applied for this purpose on student data collected from learning management systems or intelligent tutoring systems. However, the emerging social media-based learning environments have been less explored so far. Hence, in this paper we present an approach for predicting students' performance based on their contributions on wiki, blog and microblogging tool. An innovative algorithm (Large Margin Nearest Neighbor Regression) is applied, and comparisons with other algorithms are conducted. Very good correlation coefficients are obtained, outperforming commonly used regression algorithms. Overall, results indicate that students' active participation on social media tools is a good predictor of learning performance.

Keywords: Educational data mining · Performance prediction · Large Margin Nearest Neighbor Regression · Social learning environment · Social media

1 Introduction

Learning analytics is a growing field of research which deals with collecting and analyzing student data in order to understand and improve the learning process and the learning environments [4]. Prediction of student performance is one of the most popular goals, which aims to estimate future learning outcomes and identify indicators for learning success [10]. The predictive information can be used by the instructor to monitor learning progress and provide personalized feedback and interventions, especially for students at-risk, who are in need of more assistance [14]. Prediction results could also be employed in a formative assessment tool or simply to increase students' awareness [10, 14].

The goal of performance prediction is to develop a model which can infer students' outcome (i.e., the *predicted variable*, generally in the form of grades or scores) from a

P. Vittorini et al. (eds.), *Methodologies and Intelligent Systems for Technology Enhanced Learning*, Advances in Intelligent Systems and Computing 617,
DOI 10.1007/978-3-319-60819-8_2

combination of various indicators (i.e., *predictor variables*) from the educational dataset [1]. A large variety of indicators can be used, such as: number of content pages viewed per student, number of threads started per student, number of messages read on forum per student, number of assignments submitted per student, students' tags of learning resources, etc. [2]. These depend also on the particular learning environments in which the study takes place (e.g., Learning Management Systems, Intelligent Tutoring Systems, Massive Open Online Courses) which influence the type and amount of data collected.

As far as computational techniques are involved, a wide variety of methods have been applied so far for predicting students' performance, such as linear regression [9], decision trees [13], neural networks [11], Bayesian networks [3], or genetic algorithms [14].

In the current paper, we investigate the less explored context of social learning environments; more specifically, an innovative algorithm, called Large Margin Nearest Neighbor Regression (LMNNR) is applied in order to predict academic performance based on students' activity on social media tools (blog, wiki, microblogging tool). The novelty of our approach consists in the successful application of the algorithm in the educational domain, on students' social media traces. The context of study and data collection are described in the next section. The Large Margin Nearest Neighbor Regression algorithm is briefly presented in Sect. 3. The analysis results are reported and discussed in Sect. 4. Some conclusions and future research directions are outlined in Sect. 5.

2 Educational Dataset

Data was collected during a Web Applications Design course, taking place at the University of Craiova, Romania, in 2015/2016 winter semester. A social learning environment called eMUSE [8] was used to implement a project-based learning scenario. 75 undergraduate students worked in groups of 3–4 peers in order to develop a relatively complex web application; they used three social media tools (Blogger, Twitter, Mediawiki) to communicate and collaborate for the project activities. The learner tracking mechanism provided by eMUSE collected students' actions on the social media tools: blog posts and comments, tweets, wiki page revisions and file uploads. Based on these actions, a set of 14 numeric features were computed for each student:

- *NO_BLOG_POSTS* (the number of blog posts)
- *NO_BLOG_COM* (the number of blog comments)
- *AVG_BLOG_POST_LENGTH* (the average length of a blog post)
- *AVG_BLOG_COM_LENGTH* (the average length of a blog comment)
- *NO_ACTIVE_DAYS_BLOG* (the number of days in which a student was active on the blog, i.e., wrote a post or a comment)
- *NO_ACTIVE_DAYS_BLOG_POST* (the number of days in which a student wrote a post on the blog)
- *NO_ACTIVE_DAYS_BLOG_COM* (the number of days in which a student wrote a comment on the blog)
- *NO_TWEETS* (the number of tweets)

- *NO_ACTIVE_DAYS_TWITTER* (the number of days in which a student was active on Twitter, i.e., posted at least one tweet)
- *NO_WIKI_REV* (the number of wiki page revisions)
- *NO_WIKI_FILES* (the number of files uploaded on the wiki)
- *NO_ACTIVE_DAYS_WIKI* (the number of days in which a student was active on the wiki, i.e., revised a page or uploaded a file)
- *NO_ACTIVE_DAYS_WIKI_REV* (the number of days in which a student revised a wiki page)
- *NO_ACTIVE_DAYS_WIKI_FILES* (the number of days in which a student uploaded a file on the wiki)

In addition, the final project grade obtained by the student (on a 1 to 10 scale) was included as predicted variable. The algorithm used to analyze this dataset is briefly described in the next section.

3 The Large Margin Nearest Neighbor Regression Algorithm

In a classification context, support vector machines rely on the idea of finding a large margin between classes by solving an optimization problem. This idea was used in conjunction with the k-Nearest Neighbor method [12] to change the distance metric of the kNN space by using a matrix:

$$d_M(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i - \mathbf{x}_j\right)^T \mathbf{M}\left(\mathbf{x}_i - \mathbf{x}_j\right). \tag{1}$$

This method was further developed for regression purposes, resulting in an original algorithm, *Large Margin Nearest Neighbor Regression* (LMNNR) [6, 7], briefly described as follows.

For simplicity, it assumes that $\mathbf{M}$ is a diagonal matrix, and in this case the weights of the neighbors are:

$$w_{d_M}(\mathbf{x}, \mathbf{x}') = \frac{1}{d_M(\mathbf{x}, \mathbf{x}')} = \frac{1}{\sum_{i=1}^{n} m_{ii} \cdot \left(x_i - x_i'\right)^2}. \tag{2}$$

These weights can also be interpreted as the coefficients that stretch or shrink the axes of the input space according to the importance of the attributes.

Equation 2 involves a single, global matrix $\mathbf{M}$ for all the instances. However, it is possible to have different distance metrics for different instances or groups of instances. The LMNNR algorithm allows the use of *prototypes*, which are special locations in the input space of the problem. Each prototype P has its own $\mathbf{M}^P$ matrix. When computing the distance weight to a new point, an instance will use the weights of its nearest prototype, i.e., m_{ii}^P instead of m_{ii} in Eq. (2).

Finding the appropriate matrices is achieved by solving an optimization problem. In a simplified formulation, the objective function F, which is to be minimized, takes into account two criteria F_1 and F_2, defined below, with equal weights.

In order to briefly explain the expressions of these functions, let us make the following notations: $d_{ij} = d_M(\mathbf{x}_i, \mathbf{x}_j)$, $d_{ik} = d_M(\mathbf{x}_i, \mathbf{x}_k)$, $g_{ij} = \left|f(\mathbf{x}_i) - f(\mathbf{x}_j)\right|$ and $g_{ik} = \left|f(\mathbf{x}_i) - f(\mathbf{x}_k)\right|$, where d_M means the weighted square distance function using the weights one searches for.

Then, the first criterion is:

$$F_1 = \sum_{i=1}^{n} \sum_{j \in N(i)} d_{ij} \cdot (1 - g_{ij}), \tag{3}$$

where $N(i)$ is the set of the nearest k neighbors of instance i, e.g., 3. This criterion reflects the fact that the nearest neighbors of i should have similar values to the one of i, and more distant ones should have different values.

The second criterion is expressed as follows:

$$F_2 = \sum_{i=1}^{n} \sum_{j \in N(i)} \sum_{l \in N(i)} \max\left(1 + d_{ij} \cdot (1 - g_{ij}) - d_{ik} \cdot (1 - g_{il}),\, 0\right). \tag{4}$$

The distance to the neighbors with close values (the positive term) is minimized, while simultaneously trying to maximize the distance to the neighbors with distant values (the negative term). An arbitrary margin of at least 1 should be present between an instance with a close value and another with a distant value.

4 Results and Discussion

Even if the grades are integers, there is a definite order between their numeric values. Also, subtle differences in student evaluation may sometimes occur, which cannot be taken into account in a classification problem with discrete, unrelated values. Therefore, we considered that this problem is a suitable one to be addressed by means of regression algorithms.

4.1 Performance of Classic Regression Algorithms

In order to assess the performance of the LMNNR algorithm, a comparison with the algorithms implemented in *Weka* [5] was attempted.

As mentioned in Sect. 2, the original dataset has 14 attributes and a numerical class that represents the project grade. First, different *Weka* algorithms that belong to different classification/regression paradigms, were applied, e.g., k-nearest neighbors, neural networks, decision trees, support vector machines. The correlation coefficient was used as a performance measure. The best results obtained by different algorithms for 10-fold cross-validation were:

- *Random Forest* with 100 trees: $r = 0.6795$;
- *k-Nearest Neighbors*, with k obtained by cross-validation and inverse-distance weighting of the neighbors: $r = 0.569$.

In order to improve on these results, feature selection was also attempted. By taking into account the indications of the *ReliefF* algorithm, only 5 attributes were selected: *AVG_BLOG_POST_LENGTH, AVG_BLOG_COM_LENGTH, NO_ACTIVE_DAYS_BLOG_POST, NO_ACTIVE_DAYS_TWITTER,* and *NO_ACTIVE_DAYS_WIKI.*

Under these circumstances, the same regression algorithms in *Weka* performed the best, however with similar results as before:

- Random Forest: $r = 0.6887$;
- k-Nearest Neighbors: $r = 0.5535$.

One can notice that the random forest algorithm achieved slightly better results, while k-nearest neighbor achieved slightly worse ones.

Because the grades are natural numbers, it is also possible to treat them as distinct values, and transform the problem into a classification problem, instead of a regression one. However, the results were not promising; in this case, the random forest algorithm only achieved a 44% success rate in correctly classifying the instances, also in a 10-fold cross-validation scenario.

4.2 Performance of the LMNNR Algorithm

Next, we turned our attention to the evaluation of the original LMNNR algorithm for the given problem. The first attempt was to assess the influence of the feature selection on its performance. The comparison was made for the simplest parameter settings of the algorithm, with 1 prototype and 3 classification neighbors:

- *LMNNR* with feature selection: $r = 0.818721$;
- *LMNNR* without feature selection: $r = 0.826438$.

There is no significant difference between these values, and there is even a slight increase of correlation when the full set of attributes is used. This can be explained by the nature of the algorithm, which implicitly searches for the importance of the input attributes. Therefore, there is no need to manually reduce the number of attributes; on the contrary, it seems that the algorithm is able to use the additional information better than other regression algorithms.

One can immediately notice that the overall results are far better than those obtained with the well-established methods implemented in *Weka*, with a 20% increase in the quality of the results.

Table 1 presents the performance of the algorithm when the number of classification neighbors (i.e., the number of close instances that are actually used to compute the weighted output) and the number of prototypes vary. It must be mentioned that the algorithm also has an additional parameter, the number of optimization neighbors which are used to solve the optimization problem defined by Eqs. 3 and 4. However, from previous studies, it was found that a value of 3 is sufficient in most cases, while higher values only increase the computation time without providing better results.

Table 1. Performance of the LMNNR algorithm for different parameter values

Number of neighbors	Number of prototypes	Correlation coefficient *r*
3	1	0.826438
3	2	0.821534
3	3	0.822922
5	1	0.800335
5	2	0.818194
5	3	0.814019

By observing the results in Table 1, one can notice that the best combination of parameter values is also the simplest one: 1 prototype and 3 neighbors. The configuration with one prototype can be viewed as a particular case of the configuration with two or more prototypes, in which all the prototypes have the same location and weights. However, from the practical point of view, when there are more prototypes, the search is actually performed in a much larger space, and this can affect the result quality in a negative way.

This also confirms the general empirical finding that simpler models usually tend to generalize better than complex ones. In our case, the generalization capability was the only criterion used in the analysis.

Figure 1 displays a comparison between the test results and the desired outputs. The scale of the chart is the [0, 1] interval because all the data is normalized attribute-wise, before applying the algorithm. One should notice the fact that these data are obtained on the combined test bins in the 10-fold cross-validation. Since it is an instance-based method, the results of LMNNR on data that belongs to the training set are always perfect.

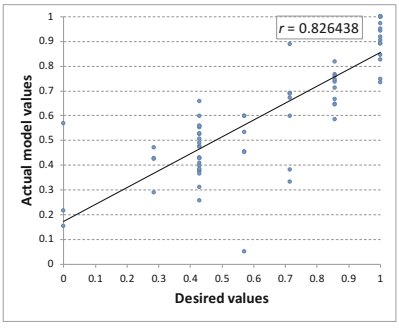

Fig. 1. Comparison between the predictions of the model and the expected data

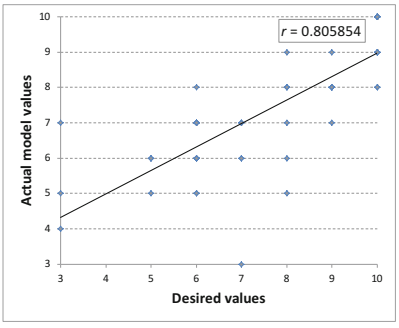

Fig. 2. Comparison between the predictions of the model and the expected data transformed into integer grades

While the chart in Fig. 1 displays a comparison between normalized output data, Fig. 2 displays a comparison between output data rounded to the nearest integer grade. The grades in the dataset are between 3 and 10, where 10 is the best grade.

Because of the rounding, one can see that the correlation coefficient slightly decreases by roughly 2.5% to $r = 0.805854$. However, this chart can offer a better understanding of the results by translating them into their original domain.

Finally, Fig. 3 shows the errors of the model, as actual differences between the predicted and desired grade values. One can notice the shape of the graph, which looks like a skewed Gaussian distribution, which is mainly caused by the number of the training instances, which is not so large: there are only 75 instances in the dataset. More importantly, this analysis shows that 85% of predictions are within only 1 point of the actual grade. This emphasizes the fact that the model is capable of good approximation for our particular problem.

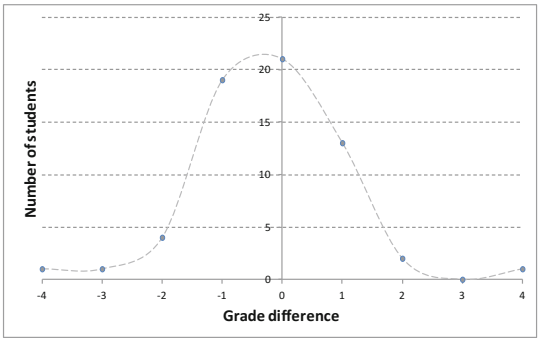

Fig. 3. The differences between the predictions of the model and the expected data transformed into integer grades

Regarding the larger differences, they are caused by the random splitting of the data into the training and testing sets of the cross-validation bins. As an instance-based method, the LMNNR algorithm cannot extrapolate to values outside the range of its training instances. For example, the small grades of 3 and 4 are rare in the dataset. If a training set contains only higher grades, and the small ones are only placed into the testing set, there is no way for the algorithm to compute proper predictions for them as a weighted sum of its training data.

5 Conclusions

The paper presented an approach for predicting students' performance based on their traces on social media tools (i.e., blog posts and comments, tweets, wiki page revisions and file uploads). An innovative algorithm, called Large Margin Nearest Neighbor Regression, was applied and its performance was assessed by means of comparisons with various algorithms implemented in *Weka*. Very good correlation coefficients were obtained (greater than 0.8), outperforming commonly used regression algorithms. Overall, results showed that students' active participation on social media tools was a good indicator of learning performance.

A potential limitation of the study is the relatively small number of students involved (75). Therefore, as future work, we plan to extend the analysis to a larger dataset, by including several cohorts of students. Investigating the algorithm performance on student data collected from different years and slightly different instructional scenarios is an interesting research direction; a more comprehensive perspective on academic performance predictors in a social learning environment could thus be obtained.

Acknowledgements. This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS – UEFISCDI, project number PN-II-RU-TE-2014-4-2604.

References

1. Baker, R.S., Inventado, P.S.: Educational data mining and learning analytics. In: Larusson, J.A., White, B. (eds.) Learning Analytics: From Research to Practice, pp. 61–75. Springer, New York (2014)
2. Dyckhoff, A.L., Lukarov, V., Muslim, A., Chatti, M.A, Schroeder, U.: Supporting action research with learning analytics. In: Proceedings of the LAK 2013, pp. 220–229. ACM Press (2013)
3. Fancsali, S.: Variable construction for predictive and causal modeling of online education data. In: Proceedings of the LAK 2011, pp. 54–63. ACM Press (2011)
4. Ferguson, R., Brasher, A., Clow, D., Cooper, A., Hillaire, G., Mittelmeier, J., Rienties, B., Ullmann, T., Vuorikari, R.: Research evidence on the use of learning analytics - implications for education policy. In: Joint Research Centre Science for Policy Report (2016). doi: 10.2791/955210
5. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. **11**(1), 10–18 (2009)
6. Leon, F., Curteanu, S.: Evolutionary algorithm for large margin nearest neighbour regression. In: Proceedings of the ICCCI 2015. LNAI, vol. 9329, pp. 286–296. Springer (2015)
7. Leon, F., Curteanu, S.: Large margin nearest neighbour regression using different optimization techniques. J. Intell. Fuzzy Syst. **32**(2), 1321–1332 (2017)
8. Popescu, E.: Providing collaborative learning support with social media in an integrated environment. World Wide Web **17**(2), 199–212 (2014). Springer
9. Roberge, D., Rojas, A., Baker, R.S.: Does the length of time off-task matter? In: Proceedings of the LAK 2012, pp. 234–237. ACM Press (2012)
10. Steiner, C., Kickmeier-Rust, M. Türker, M.A.: Review article about LA and EDM approaches (Deliverable D3.1) (2014). http://css-kmi.tugraz.at/mkrwww/leas-box/downloads/D3.1.pdf
11. Wang, Y.H., Liao, H.C.: Data mining for adaptive learning in a TESL-based e-learning system. Expert Syst. Appl. **38**(6), 6480–6485 (2011)
12. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. **10**, 207–244 (2009)
13. Wolff, A., Zdrahal, Z., Nikolov, A., Pantucek, M.: Improving retention: predicting at-risk students by analysing clicking behaviour in a virtual learning environment. In: Proceedings of the LAK 2013, pp. 145–149. ACM Press (2013)
14. Xing, W., Guo, R., Petakovic, E., Goggins, S.: Participation-based student final performance prediction model through interpretable genetic programming: Integrating learning analytics, educational data mining and theory. Comput. Hum. Behav. **47**, 168–181 (2015)