Predicting Student Exam's Scores by Analyzing Social Network Data

Michael Fire, Gilad Katz, Yuval Elovici, Bracha Shapira, and Lior Rokach

Telekom Innovation Laboratories and Information Systems Engineering Department,
Ben-Gurion University of the Negev, Beer-Sheva, Israel
{mickyfi, katzgila, elovici, bshapira, liorrk}@bgu.ac.il

Abstract. In this paper, we propose a novel method for the prediction of a person's success in an academic course. By extracting log data from the course's website and applying network analysis methods, we were able to model and visualize the social interactions among the students in a course. For our analysis, we extracted a variety of features by using both graph theory and social networks analysis. Finally, we successfully used several regression and machine learning techniques to predict the success of student in a course. An interesting fact uncovered by this research is that the proposed model has a shown a high correlation between the grade of a student and that of his "best" friend.

Keywords: Social Network Analysis, Data Mining, Score Prediction, Machine Learning, Web Log Analysis, Multi Graph

1 Introduction

The ability to predict individual or group success in exams and courses has been researched in the past four decades [1, 2]. Accurately predicting students' exam or course grades has the potential to help students in various ways; by using accurate predictions we can detect early on students who have difficulties with the course materials and help them to improve. Moreover, using this kind of prediction technique can help in several other education-related areas [3]:

1. Discriminating among enrolment applicants.
2. Advising students on their majors.
3. Identifying productive programmers.
4. Identifying employees who might profit best from additional training.
5. Improving computer classes for non-CIS majors.
6. Determining the importance of oft-cited predictors of computer competency, such as gender or math ability.
7. Exploring the relationship between programming abilities and other cognitive reasoning processes.

One common approach for solving this type of prediction problem is to extract as many attributes as possible, sometimes as many as hundreds. By evaluating

the value of each attribute, researchers can attempt to predict exam grades or other variables using linear regression or multiple regression methods. Usually, when using regression, one tries to predict the dependent variables' values using independent attributes of different types. The number of independent variables is very large and includes age and gender [4,5], GPA grades [6], educational level of parents [7], emotional and social factors [8], and even the complexity measure of teachers' lecture notes [9]. Dependent variables include course grade [5, 6, 10, 11], exam grade [8, 12] and even aptitude in computer programming [13]. Other methods used in tackling the grade prediction problem are the factor analysis or other classification schemes with statistical analysis [5,10,14,15]. For example, Rountree et al. [15] used Decision Trees analysis in order to identify combinations of factors that may assist in predicting success or failure in a CS1 class. In this paper, we present a method for building a social network of students in a course for the purpose of predicting their final exam grades. The method was tested on the course "Computer and Network Security", which was a mandatory course taught by two of the authors of this paper at Ben-Gurion University. The social network contains data collected from 185 participating students from two different departments. The course's social network was created by analyzing the implicit and explicit cooperation among the students while doing their homework assignments. Using this social network, we extracted various social attributes, such as the grade of the student's "best friend". We demonstrate - using linear regression - that the best friend's grade has a direct influence on a student's exam grade. We also demonstrate - by using multiple regression and machine learning - that other social parameters are also influential in determining a student's grade. Moreover, they can help predict which students are likely to fail the test.

2 Related Work

In the past four decades, a considerable amount of research has gone into detecting factors that affect students' exam and course grades. Most of these studies used regression in order to detect the relevant factors that influence students' grades. Petersen and Howe [11] tried to predict grades in introductory CS courses. They found that previous success in high school mathematics and science had a positive correlation with the CS course grade. Mazlack [16] found that a low correlation exists between the final exam grade in a FORTRAN programming course and a student's personal background factors in addition to a students' results in an IBM's Programmer Aptitude Test (PAT). Konvalina et al. [12] used students' high school performance and background in mathematics in order to predict final exam grades in the course "Introduction to Computer Science". Butcher and Muth [6] found a linear correlation between different ACT grades and the final CS course grade. Evans and Simkin [3] utilized a 100 question survey in order to predict students' grades. They discovered that the Myers-Briggs cognitive style [17] can assist in predicting students' grades. Henry et al. [18] examined the relationships among attributions and performance in a computer science course. They found that people with an optimistic attribu-

tional style performed better in a computer programming course than those with a pessimistic one. Chamillard [19] used students' performance in prior academic courses in order to predict their performance in a particular course. Bennedsen and Caspersen [8] studied the influence of emotional and social factors on students' learning outcomes in an introductory computer science courses. Surprisingly, they did not find a linear correlation between the exam grade and the social well-being and emotional health of the students. Recently, Joseph and Suzanne [2] examined peer tutoring's influence on students' performance in CS1 and CS2 courses. They found indications that peer tutoring had a positive impact on a students' course performances, but did not have a significant impact on the course final exam grades. In this paper we too use several regression and machine learning techniques in order to predict students' final exam grade in the course "Computer and Network Security". In order to carry out these predictions, we mainly used attributes extracted from the course's social network, created by us. Similar techniques that involve social-network analysis and regression were used by Lee [20] in analyzing the behavior of del.icio.us users, by Christakis [21] in researching the spread of obesity, and by Altshuler et al. in predicting the individual parameters and social links of smart-phones users [22].

3 Methods and Experiments

To cope with the challenge of predicting students' grades,we extracted features from the course's social network and used regression and machine learning to analyze them.. We constructed the social network of the course by using different types of homework assignments. Some assignments were individual (and done online), while others required a group effort. After constructing the course's social networks, the next step was to extract the relevant features from these networks. We developed a Python code using the Networkx graph package [23] for this purpose. We then combined the social network graph features with other student data collected throughout the course, such as their grades in the assignments and in the final exam. Finally, we used the collected features in order to test our hypothesis with that a student's social network can influence that particular student's grades. The rest of this section describes in detail the steps taken in order to run our experiments and prove our hypothesis.

3.1 Constructing the Course Social Networks

Different Homework Assignments We constructed the course social network using the course homework assignments which were a part of the students' final course grade. There were three types of assignments:

- **Online Assignments:** The students received five different online assignments. These assignments were individual and every student was expected to solve them without help. Every student got a different solution, according to his private email address. Despite the difference in the solution itself,

all online assignments could be solved by using the same techniques. These assignments were stored in a dedicated website, with every access and every submission to the website recorded in the site's log. By analyzing these logs we were able to identify the students who had cooperated with each other.
- **Coding Assignment:** Students were given one large coding assignment to be completed in pairs. The assignment consisted of developing a web proxy with multiple features.
- **Theoretical Writing Assignments:** The students received two different theoretical writing assignments. The first assignment was to be done in pairs, while the second could be done in a group of up to four students.

3.2 Creating the Class Cooperation Social Network Graphs

By using homework assignments and analyzing the website logs (which contained 10,759 entries), we were able to construct social networks of explicit and implicit cooperation among the students. We defined the course's social network as a weighted multi-graph $G =< V, E >$, where V is the set of vertices in the multi-graph, and each vertex $v \in V$ contains unique information about one of the students that participated in the course. We defined E as the multi-set of links in the graph, with each link $e \in E$ defined to be a tuple $e = (u, v, t, w) \in E$, where $u, v \in V$, w is the weight of the link, and t can be one of the following values: $t \in \{\text{partner's links}, \text{same computer link}, \text{same time link}\}$. We constructed the cooperation multi-graph links in the following manner. First, we defined the partner's links to be the explicit connection among the students who submitted theoretical or coding assignments together as partners. Secondly, we defined the same computer links to be the implicit connection among students who used the same computer while solving the online assignments. In order to discover the same computer links, we used the online assignments logs to extract the following data for each user: user name, IP address, and the browser user-agent string. We used the combination of user IP address and browser user-agent strings to fingerprint computers in almost unique manner [24]. By collecting computer fingerprints, we were able to conclude with a high degree of confidence which users had worked together on the same computer. We then were able to define a link between every pair of users that used the computer. Finally, we defined the same time links to be the second implicit connection among users who probably solved the assignments together but submitted their solution from different computers. To uncover the same time links, we used the online assignments to extract the timestamp of each submission. We defined two users as connected to each other by a time link if the two had accessed our website in almost the same time on several occasions. For each link, we assigned a weight according to the number of times such a correlation between the users was found. For example, if two students submitted, as partners, three different assignments, then the weight of the partner's link between them would be three. In addition, we also added several information layers to the multi-graph based on the student's information. For each vertex, we added the following information: the student's department, assignments grades, and the final exam score. At the end of the construction

phase, a multi-graph with 184 vertices and 360 links was generated. We used the Cytoscape [25] software tool in order to visualize our multi-graph (see Figure 1). In the visualized graph, different link types were displayed by using different colors; red for "partner's link", yellow for "same computer link", and blue for "same time link". The strength of each link in the multi-graph was calculated according to the weight of the link between the vertices where stronger connections were visualized by bolder lines. Additionally, the following visualization elements were also added to the vertices of the multi-graph: a) Each vertex is represented by a different color according to the student's department; blue vertices for students from the Information Systems Engineering (ISE) department and red for students from the Computer Science (CS) department. b) Different vertex sizes were assigned according to the student's final test score; the higher the student's test grade, the bigger the size of his vertex.

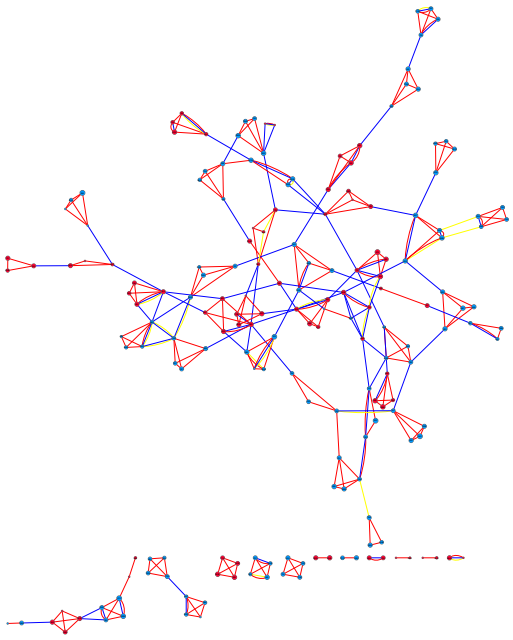

Fig. 1. The course social network multi-graph. Each link color represents connection of a different type; *red* is "partner's link", *yellow* is "same computer link" and *blue* is "same time link". The color of each vertex defines the student's department and each vertex size is in accordance with the student's final test score.

Using the course weighted multi-graph, we also created a graph describing the social network of the students who participated in the course. This network

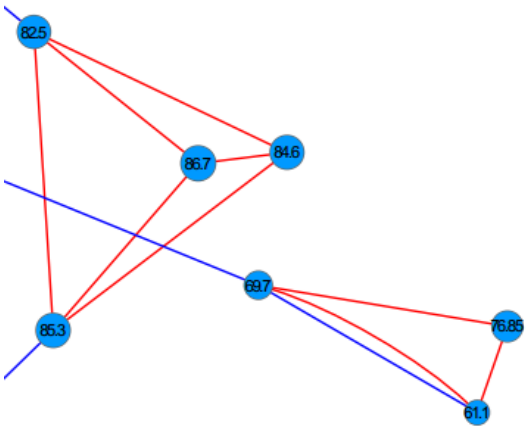

Fig. 2. Closer Look Into The Social Network. Connected students got similar final test grades.

Table 1. Multi-Graph Visualization Characteristics

Characteristics	Color	Amount
Graph Components Number	-	12
Total Links	-	360
Total Vertices	-	184
Total Vertices With Final Grade	-	169
ISE Students Vertices	Blue	109
CS Students Vertices	Red	75
Partner's Links	Red	240
Same Computer Links	Yellow	23
Same Time Link	Blue	96

is a weighted graph where each link's strength is the accumulated weight of the links between a pair of students. Namely, we define $G' =< V, E' >$, where $E' = \{(u, v, w') | u, v \in V \text{ and } w' = \sum_{((u,v,t,w)\in E)} w\}$.

Using the graphs mentioned above, we attempted to estimate the accuracy of our construction process and whether or not the topology and attributes of the graphs could be used in order to predict a student's final test grade.

Calculating the Graph Characteristics We calculated several graph characteristics (see Table 1): a) general graph statistics (to be explained later on in this paper), b) the number of links of each type, and c) the number of graph components. To evaluate the integrity of our representation of the implicit and explicit links, we calculated several statistics. For each link type, we calculated the ratio of links among students from the same department and links among students from different departments (inter and intra-departmental links). We

also calculated the percentage of connections among vertices that were both explicit and implicit.

3.3 Predicting Students‘ Final Test Scores

Based on the social network graphs, we developed a Python code using the Networkx graph package [23]. This code extracted a feature vector for each student based on the social network graphs described in the previous section. Using the extracted features together with the students' personal information, such as a student's department, and assignments score, we attempted to predict the student's final test grade (*final-grade*). This was done by applying several regression and machine learning techniques.

Features Extraction Using the students‘ personal information combined with the social network graphs, we extracted the following features for each student $u \in V$:
Personal Information Features.

- *Student assignments scores* the score the student received on each one of his assignments.
- *Students department* - the department the student belongs to.
- *Student Final Test Grade* - The student's final test grade between 0 and 100.

Topological Features.

- *Student's degree (number of friends)* - used to define the student's centrality in the network and defined as $d(u) := |\{v|(u,v) \in E‘\}|$
- *The number of friends of u by type* - using the multi-graph G, we can extract the number friends u has with respect to a specific connection type:

$$\begin{aligned}\text{partners-number}(u) :=&|\{(u,v,t,w) \in G|t = \text{'partners link'}\}|.\\ \text{same-time-number}(u) :=&|\{(u,v,t,w) \in G|t = \text{'same time link'}\}|.\\ \text{same-comp-number}(u) :=&|\{(u,v,t,w) \in G|t = \text{'same computer link'}\}|.\end{aligned}$$

Friends Grades Features. We defined several functions and sets designed to make the friends features definitions clearer and easier to understand:

- *The best friend's score* - we attempted to determine the influence one's best friend has on one's scores. One's best friends are defined as the friends with the maximum connection weight to the student in $G‘$[1]

$$\text{best-friend}(u,G‘) := \text{bf}(u,G‘) := \text{first-element}(\{v|(u,v,w) \in G‘ \text{ and } w \geq w‘, \forall (u,v‘,w‘) \in G‘\}).$$

[1] If there is more than one "best" friend, we arbitrarily chose one vertex from the set.

– *The second best friend's score* the same as the previous feature. If a student is u's best friend after we remove the first best friend from G‘, then the second best friend of u is defined to be:

$$\text{second-best-friend}(u, G\text{‘}) := \text{sf}(u, G\text{‘}) := best - friend(u, G\text{‘}_{bf(u)})$$

where $G_u :=< V - u, E >$.

– We define $w_(u, v), u, v \in V$ to be the weight of edge (u, v) in the weighted social graph, namely:

$$w_{(u,v)} := \{w|(u, v, w) \in E\text{‘}\}$$

Using the above definition we can define the following features:

– *The best friend test grade* - defined as:

$$\text{best-friend-grade}(u) := \text{final-grade}(\text{bf}(u))$$

– *The average and weighted average score of the two best friends* - defined as:

$$\text{two-best-friends-avg-grade}(u) := \frac{\text{final-grade}(\text{bf}(u)) + \text{final-grade}(\text{sf}(u))}{2}$$

, and

$$\text{two-best-friends-weighted-avg-grade}(u) := \frac{w_{u,bf(u)}\text{final-grade}(\text{bf}(u)) + w_{u,sf(u)}\text{final-grade}(\text{sf}(u))}{2}$$

– *Student's friends maximum score* - defined as

$$\text{max-fscore}(u) := max(\{\text{final-grade}(v)|(u, v) \in E\text{‘}\})$$

– *Student's friends minimum score* - defined as

$$\text{min-fscore}(u) := min(\{\text{final-grade}(v)|(u, v) \in E\text{‘}\})$$

– *Student's average and weighted average friends scores* - defined as

$$\text{avg-friends-score}(u) := \frac{\sum_{(u,v)\in E} \text{final-grade}(v)}{d(u)}$$

Predicting Grades Using the R-project statistical software [26], WEKA [27] (a popular suite of machine learning) and the features defined in the above subsections, we ran several regression and machine learning algorithms. Our goal was to find a correlation between the different features and the students' final test grade and attempt to predict who among the students will receive a score of below 60, thus failing the final test. Using regression, the following two experiments were run: a) a simple linear-regression in order to find a correlation

between students' grades and their best friends' grades, and b) a multiple linear regression and step-wise regression in order to build a module for predicting the students' grades. Each of the modules was evaluated by calculating the regression *P-value* and *R-square* values. We also evaluated different supervised learning algorithms in an attempt to predict which students are most likely to fail in the final test. We used WEKAs C4.5 (J48) decision tree, IBk, NaiveBayes, SMO, Bagging, AdaBoostM1, RotationForest and RandomForest implementations of the corresponding algorithms. The Bagging, AdaBoostM1, and RotationForest algorithms were evaluated using the J48 as the base classifier.

4 Results

In the following section, we present the results obtained using the methods described in the previous section. The results consist of three parts. First, we present the results of the statistical test used to validate the integrity of the constructed social network graph. Secondly, we present the results of the regression analysis techniques mentioned above. Finally, we present the results of the machine learning algorithms mentioned in Section 3.3.

4.1 Graph Construction Integrity

To assess the integrity of the constructed social networks graph, we determined how many of the links in the graph are among students of the same department. The results of the analysis showed that 99.58% of the explicit "partners links", 100% of the implicit "same computer links", and 68.75% of the implicit "same time links" were among students from the same department (see Figure 3).

In addition, we calculated how many of our implicit links (i.e., "same computer link" and "same time link") were also explicit links (i.e., "partners links"). The results showed that 52% (12 out of 23) of the implicit "same computer links" and 32% (31 out of 97) of the implicit "same time links" were also explicit partners links.

These results are particularly significant considering the a-priori probabilities. With 109 students in the Information Systems Engineering department and 75 in the Computer Science department, the a-priori probability of intra-departmental connection ranges between 0.4 and 0.59 (as one can see, there are hardly any). Moreover, the a-priori probability of an implicit link to be explicit as well is 0.007, while more than half of our implicit links share this trait.

4.2 Regression Analysis Results

Using the R-project software, we ran several regression algorithms based on the full features vectors that we extracted from 163 students, out of which 41 failed the final exam (with a grade lower than 60). Using the regression algorithms, we generated and evaluated several prediction models in order to predict the students' final grades. These models were mainly based on their social network attributes.

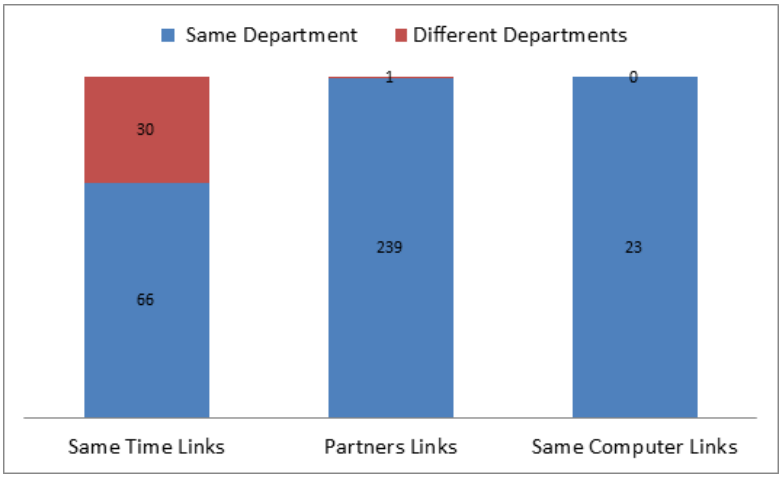

Fig. 3. The distribution of links between students by departments. Each bar illustrates the link type distribution between inter and intra-departmental links.

Simple Linear Regression Using a subset of the social network features described in Section 3.3, we attempted to use the simple linear regression model:

$$y = \beta_0 + \beta_1 x_1$$

Four different features were tested in order to create this simple regression model: a) Using only the Best-Friend-Grade feature in our regression model produced a regression model with a positive slope of 0.2525 ($\beta_1 = 0.2525$) with R^2 (*R-square*) of 0.0553, *mean absolute error* (MAE) of 11.08 and a *p-value* of 0.002 (see Figure 4), b) Using the *Two-Best-Friends-Avg-Grade* feature produced a regression model with a positive slope of 0.3219, with R^2 of 0.0506, *MAE* of 11.045, and a *p-value* of 0.0038, c) Using the Final exercise grade (Final-Ex-Grade) feature produced a model with R^2 of 0.05458, MAE of 10.929, and *p-value* of 0.0027, and d) Using the number of friends with "same-computer" link type (same-comp-number) feature produced a model with a slope of -4.708 with R^2 of 0.019, and *p-value* of 0.079.

Multiple Regression In this section, we present the results obtained using the features presented in Section 3.3 and the general multiple linear regression model:

$$y = \beta_0 + \beta_1 x_1 + \beta x_2 + \ldots + \beta_n x_n$$

We constructed a model to predict the final test grades of the students using different feature combinations. When using multiple regression algorithms with all the extracted features, the result was a model that predicted the students' final test grades with *Multiple R-squared* of 0.174, *MAE* of 10.377 and p-value

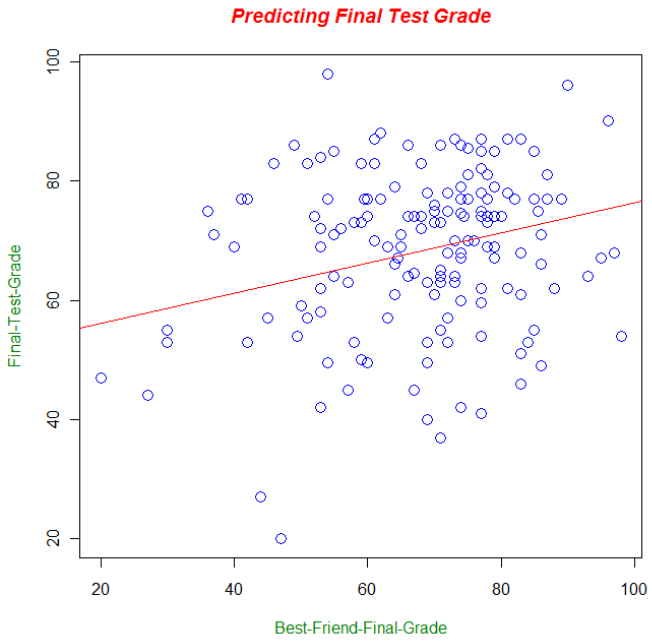

Fig. 4. Linear Regression. Predicting the student's final test grade using the "Best-Friend-Grade" feature.

of 0.009. When a regression model that uses only the four exercises grades was created, the result was a *p-value* of 0.02 with *MAE* 10.98. Next, we generated a regression prediction model using backwards stepwise regression. This resulted in a worse *Multiple R-squared* value (with a value of 0.1445), but a slightly better p-value of 0.0001 and with *MAE* of 10.644. The model only used five features: the best friend final grade, first and third exercises grades, friend's maximum grade, Final-Ex-Grade, and same computer links number.

Machine Learning Results Using the WEKA software, we ran machine learning algorithms in order to predict which of the 163 students will receive a grade of below 60 and fail the final test. Each of our classifiers was evaluated using 10-fold cross validation approach. We used the area-under-curve (AUC) measure in order to evaluate our results. As expected, the ensemble methods fared best, especially the Rotation Forest and Adaboost classifiers. When all the features were used, the Rotation Forest classifier performed best with an AUC of 0.672. Afterwards, we used T-tests with significance of 0.05 to compare between the different classifiers. According to T-test results, the RotationForest classifiers returned better AUC results than the naive ZeroR and the simple OneR classifiers.

Table 2. Regressions Results

Features	**p-value**	R^2	**MAE**
Best-Friend-Grade	0.002	0.055	11.08
Two-Best-Friends-Avg-Grade	0.004	0.051	11.05
Final-Ex-Grade	0.003	0.055	10.93
Four Exercises Grades	0.02	0.07	10.98
All-features	0.009	0.174	10.38
Backwards Stepwise Selected Features	0.0001	0.145	10.64

5 Conclusions

To paraphrase an old saying, we believe that, "If you tell me what your friend's grades are, I will tell you what yours will be". We attempted to prove this hypothesis using the social networks of a course taught during the spring semester of 2010. We constructed the course's social network graphs using information we collected from the log of the course website and by analyzing the course homework assignments. We tested the integrity of our graphs by comparing the link distribution of the implicit links to that of the explicit links and looked for correlation. In addition, we also presented a visualization of the social network graphs of the course. Using this visualization, we were able to detect an interesting phenomenon; when looking closely into the social network graph, one can see that students' final grades are closely related to those of his friends' grades (See Figure 2). We attempted to find an explanation to this phenomenon by using different regression and machine learning techniques. Using multiple linear regressions, we were able to prove that a correlation exists between a students' final grade and that of their friends. This correlation has a small p-value, which strongly supports this hypothesis. Furthermore, the regression models presented in this paper have R-squared values ranging from 0.1445 to 0.174, and mean absolute error ranging from 10.337 to 10.98 (See Table 2). These values are not uncommon for human behavioural studies (this is especially true in the field of grade prediction, according to Evans and Smikin [3]). We also discovered interesting phenomenon where there is negative correlation between the final grade and the *same-comp-number* feature (with a slope of -4.708 and *p-value* of 0.079). This correlation may indicate that students whom cheat on their homework assignments and solve their assignments with their friends on the same computer tend to receive lower final test grades. Using supervised learning algorithms we created different classifiers that predicated which students are most likely fail the test. Our Rotation Forest classifier received an AUC of 0.672.

We believe this work has two main future research directions. The first is to use different classification methods combined with machine learning algorithms to identify students who are likely to fail their final exam in other courses. The second future research direction is to use the social network of the course combined with text analysis techniques (used on the homework assignments) to follow the diffusion of information across the network. We believe it could

be very interesting to examine the correlation between homework plagiarism, knowledge diffusion patterns, and success in the course's final exam.

6 Availability

Anonymous version of the students' multi-graph social network topology is available for other researchers to use on our research group website `http://proj.ise.bgu.ac.il/sns/`.

References

1. C. Alspaugh, "Identification of some components of computer programming aptitude," *Journal for Research in Mathematics Education*, pp. 89–98, 1972.
2. J. Cottam, S. Menzel, and J. Greenblatt, "Tutoring for retention," in *Proceedings of the 42nd ACM technical symposium on Computer science education*. ACM, 2011, pp. 213–218.
3. G. Evans and M. Simkin, "What best predicts computer proficiency?" *Communications of the ACM*, vol. 32, no. 11, pp. 1322–1327, 1989.
4. R. Deckro and H. Woundenberg, "Mba admission criteria and academic success," *Decision Sciences*, vol. 8, no. 4, pp. 765–769, 1977.
5. T. Cronan, P. Embry, and S. White, "Identifying factors that influence performance of non-computing majors in the business computer information systems course," *Journal of Research on Computing in Education*, vol. 21, no. 4, pp. 431–446, 1989.
6. D. Butcher and W. Muth, "Predicting performance in an introductory computer science course," *Communications of the ACM*, vol. 28, no. 3, pp. 263–268, 1985.
7. S. Ting and T. Robinson, "First-year academic success: A prediction combining cognitive and psychosocial variables for caucasian and african american students." *Journal of College Student Development*, 1998.
8. J. Bennedsen and M. Caspersen, "Optimists have more fun, but do they learn better? on the influence of emotional and social factors on learning introductory computer science," *Computer Science Education*, vol. 18, no. 1, pp. 1–16, 2008.
9. K. Keen and L. Etzkorn, "Predicting students' grades in computer science courses based on complexity measures of teacher's lecture notes," *Journal of Computing Sciences in Colleges*, vol. 24, no. 5, pp. 44–48, 2009.
10. G. Fowler and L. Glorfeld, "Predicting aptitude in introductory computing: A classification model." *AEDS Journal*, vol. 14, no. 2, pp. 96–109, 1981.
11. C. Petersen and T. Howe, "Predicting academic success in introduction to computers." *AEDS Journal*, vol. 12, no. 4, pp. 182–91, 1979.
12. J. Konvalina *et al.*, "Identifying factors influencing computer science aptitude and achievement." *AEDS Journal*, vol. 16, no. 2, pp. 106–12, 1983.
13. T. Hostetler, "Predicting student success in an introductory programming course," *ACM SIGCSE Bulletin*, vol. 15, no. 3, pp. 40–43, 1983.
14. P. Campbell and G. McCabe, "Predicting the success of freshmen in a computer science major," *Communications of the ACM*, vol. 27, no. 11, pp. 1108–1113, 1984.
15. N. Rountree, J. Rountree, A. Robins, and R. Hannah, "Interacting factors that predict success and failure in a cs1 course," in *ACM SIGCSE Bulletin*, vol. 36, no. 4. ACM, 2004, pp. 101–104.

16. L. Mazlack, "Identifying potential to acquire programming skill," *Communications of the ACM*, vol. 23, no. 1, pp. 14–17, 1980.
17. C. Allinson and J. Hayes, "The cognitive style index: A measure of intuition-analysis for organizational research," *Journal of Management studies*, vol. 33, no. 1, pp. 119–135, 1996.
18. J. Henry, M. Martinko, and M. Pierce, "Attributional style as a predictor of success in a first computer science course," *Computers in human behavior*, vol. 9, no. 4, pp. 341–352, 1993.
19. A. Chamillard, "Using student performance predictions in a computer science curriculum," in *ACM SIGCSE Bulletin*, vol. 38, no. 3. ACM, 2006, pp. 260–264.
20. K. Lee, "What goes around comes around: an analysis of del. icio. us as social space," in *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*. ACM, 2006, pp. 191–194.
21. N. Christakis and J. Fowler, "The spread of obesity in a large social network over 32 years," *New England Journal of Medicine*, vol. 357, no. 4, pp. 370–379, 2007.
22. Y. Altshuler, N. Aharony, M. Fire, Y. Elovici, and A. Pentland, "Incremental learning with accuracy prediction of social and individual properties from mobile-phone data," *Arxiv preprint arXiv:1111.4645*, 2011.
23. A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," *In Proceedings of the 7th Python in Science Conference (SciPy2008)*, 2008.
24. P. Eckersley, "How unique is your web browser?" in *Privacy Enhancing Technologies*. Springer, 2010, pp. 1–18.
25. P. Shannon, A. Markiel, O. Ozier, N. Baliga, J. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome research*, vol. 13, no. 11, pp. 2498–2504, 2003.
26. R. Team *et al.*, "R: A language and environment for statistical computing," *R Foundation for Statistical Computing Vienna Austria*, no. 01/19, 2010.
27. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, November 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278