Better Safe than Sorry: Visualizing, Predicting, and Successfully Guiding Courses of Study

Alexander Kerth[1] Felix Schuhknecht[2] Lukas Pensel[3] Justus Henneberg[4]

Abstract:

Successfully going through a course of study is a lengthy and challenging task. To obtain a degree, many obstacles must be overcome and the right decisions must be made at the right point in time, often overwhelming students. To reduce the amount of dropouts, the goal of study advisors is to reach out to endangered students in time and to provide them help and guidance. To support the work of study advisors, who typically have to monitor a large amount of students simultaneously, we present in this demonstration an easy-to-use graphical tool that (a) allows the advisor to visualize all relevant information of study data in a responsive graph in order to overview the current study situation. In addition to visualization, our tool provides (b) a forecasting functionality based on pre-trained models and (c) a warning feature to identify endangered students early on. In the on-site demonstration, the audience will be able to step into the role of a study advisor and use our tool and all of its features to identify and guide struggling students within anonymized real-world study data.

Keywords: Study monitoring; Study Prediction; Visualization; Machine Learning; Graph Databases

1 Introduction

Successfully studying and obtaining a degree is challenging. While some students are able to successfully make it through the forest of lectures, seminars, and labs on their own, many struggle in finding the right individual path by themselves. To counter this problem, many universities employ so-called *study advisors*. Their task is essentially threefold: (1) To identify struggling students, for instance by analyzing their performance over the past semesters. (2) To provide guidance for the identified students on how to improve their individual situation. (3) To monitor whether the guidance actually helps and improves the situation of the students. With their work, study advisors help in reducing (avoidable) dropouts and eliminate stress and uncertainty on the side of students.

Unfortunately, all three tasks are highly difficult for the study advisor in the present situation. In terms of (1), typically, a large number of students must be monitored, namely all currently enrolled students in all stages of their studies. Therefore, overseeing the sheer amount

[1] Johannes Gutenberg University Mainz, Institute of Computer Science, Staudingerweg 9, 55128 Mainz, Germany alkerth@students.uni-mainz.de
[2] schuhknecht@uni-mainz.de
[3] pensel@uni-mainz.de
[4] henneberg@uni-mainz.de

doi:10.18420/BTW2023-36

of data is already challenging when done (pseudo-)manually. Connected to this is the problem that the study data is typically materialized in some sort of course management system (CMS). The primary purpose of such a database is to reliably log all entered study data, but not to perform any form of sophisticated analysis on it. This leads to the unpleasant situation that accessing and analyzing the existing study data is already a cumbersome step for the study advisor. Assuming that struggling students have been identified somehow in step (1), step (2) is even more challenging. Now, the study advisor has to come up with guidances and recommendations that fit to the individual problems of each student. This requires a deep individual analysis of the study performance so far and the design of a fitting counter-steering measure. Here, different courses could be advised to be taken in a specific order. The optimization goal is basically to find a suitable trade-off between progress, interests, and pressure for the student. When guidance has been given to a struggling student, in step (3), the performance of the student must be monitored and evaluated throughout the next semesters in some sort of feedback loop. Here, the study advisor has to carefully analyze the development of the student over a longer period of time, requiring cumbersome (repetitive) data analysis and comparison.

To support study advisors in their challenging tasks, in the following we propose a tool to assist their workflow in all three previously mentioned steps. Note that we specifically use the term *"assist"* and not *"replace"*, as we believe that human advising is (and will always be) essential in this sensitive field. Precisely, our tool assists the advisor in the following aspects:

(a) It makes study data accessible for the study advisor by connecting students, lecturers, courses, and exam results with each other and by presenting their relation in a *graph-based visualization*. We provide different views for different analysis purposes: For example, it is possible to visualize for a particular student all taken courses and exam results. Or it is possible to focus on a specific course and to see all exam performances of a specific semester or over multiple semesters for all students. Any available meta-data, such as passing/failure ratio, is also presented to ease the interpretation of the data.

(b) It contains an *early warning system* that automatically identifies potentially struggling students. To do so, it predicts the likelihood of finishing the studies using an ML model, which has been trained on available study data, and combines it with the number of exams failed so far. The generated list of students can be ordered and post-filtered by the study advisor to contact the students in danger.

(c) It allows the advisors to create a *course of study forecast* for a particular student to give recommendations on which courses to take. This can come in handy if the curriculum contains many courses to choose from. Again, we use pre-trained ML models to generate this forecast and visualize it in an easy-to-digest manner. Besides the proposed courses, the forecast also contains the predicted grade range and passing probability for each course.

Note that we designed our tool to be usable by *non-computer-scientists*, i.e., there is no

programming or query writing required. Therefore, our tool can be applied university-wide by study advisors of all faculties.

2 Architecture and Setup

Our tool runs in a web browser and uses a multitude of technologies: For the frontend, we use a combination of PHP, JavaScript, and Bootstrap, running on an Apache web server. The backend is twofold: To manage the data, we use Neo4j [ht22] as a graph-based database management system that is queried by the frontend via Cypher [Fr18] queries. Since all study data in the CMS is typically in relational format, we use a converter that translates the relational data into the corresponding graph representation by turning foreign-key relationships into edges. In our case, we extracted the (anonymized) study data in relational format from JoGu-StINe [Jo22], the CMS of JGU Mainz, and then converted the data into a corresponding graph representation.

The machine learning part, which is required for forecasting and warning, is realized in Tensorflow and largely builds upon the N-RELAGGS [PK19] work. The core idea is to feed a neural network with input features composed of joined study data. The joined data contains information about enrollment, semester statistics, and exam results for each student. Underneath, Tensorflow and Keras are used to train the model and to forecast the performance of students of interest. Note that other works [Zh20, Má13] also tried to predict the performance of students using data mining and ML-assisted approaches. In general, these approaches could be integrated into our prediction backend as well.

3 Visualization

Figure 1 shows a screenshot of the main view of our tool running in a browser. The depicted center view shows the visualization of the performance of a specific student (turquoise node). Around the student, all taken courses of that student are arranged (blue nodes) with the semester in which the course was taken (yellow nodes). Linked to each course/semester combination is the grade of the corresponding exam (green nodes for passing, red node for failing). The border of each course node additionally shows the passing/failing ratio of this particular course over all semesters. The same applies to exam nodes.

Note that at any time, the user can click a node in the center view to focus the view on that node. Focusing on a node does more than rearranging the currently visible nodes as it might trigger the loading of new nodes. For example, when clicking on a specific exam, the center view will load and show all students of that particular exam. To go back to the previous view, the user can click on the “return” button on the top of the view at any time.

Alongside the center view, on the right side we list various statistics that support the current visualization. For example, we show the average grade of the student, the fraction of passed exams, the number of courses taken, and the passing ratio of all exams. Note that the

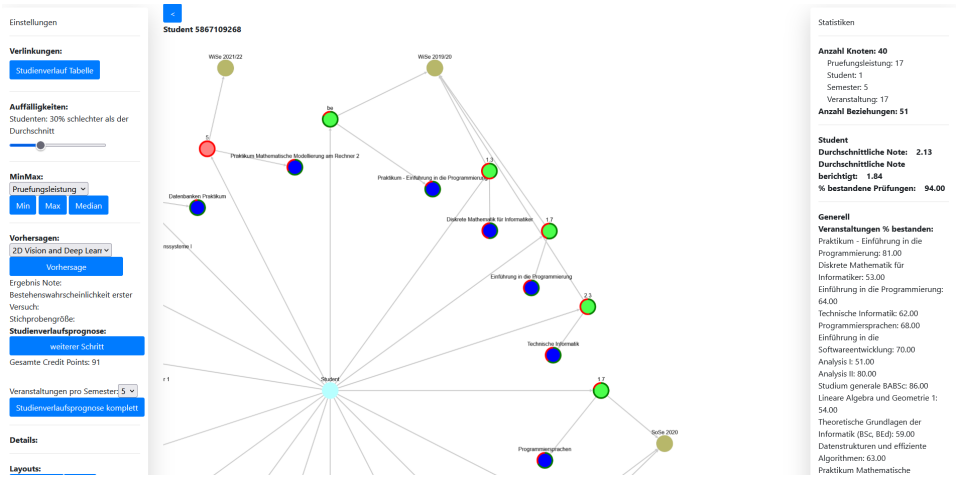

Fig. 1: Frontend overview of our tool running in a web browser.

right-side view automatically adapts to the current visualization of the center view. The left view of the tool serves as a settings and action panel that can be used to adjust the current visualization and to trigger the loading of new visualizations. This includes triggering the the course of study forecast (Section 4) and the early warning system (Section 5).

4 Course of Study Forecast

Besides visualization, our tool supports forecasting the performance of a particular student in a particular course or even the whole (remaining) course of study. To enable this feature, the course of study regulations must be encoded in our tool in advance, which contain information on which courses are available and which combination of course (types) must be passed to obtain the degree. For this demo, we encoded the study regulations of the B.Sc. in Computer Science (2016 version) of JGU Mainz.

We now trained four different models for each course on the available study data, where each model is tailored towards a different stage of study. We train the first model on the first semester study data of all students, whereas the second model is trained on the first *and* second semester study data of all students, and so on. We consider only courses that contained at least 40 students and feed the model with exam results as well as meta-data such as achieved credit points per semester and achieved credit points overall. When forecasting for a particular student, the system automatically picks the model that fits best to the semester in which the student is currently enrolled. For each recommended course, the model predicts one of the outcomes “good” (grade range 1-2), “satisfactory” (grade range 3-4), or “failed” (grade 5).

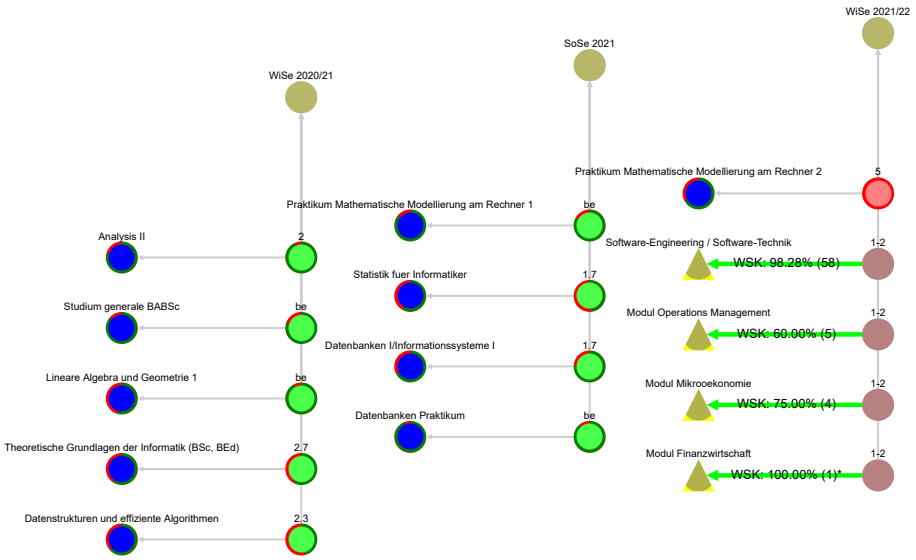

Fig. 2: Course of study forecast for a particular student for winter semester 2021/2022 based on the performance of the previous two semesters.

Additionally, a passing probability is predicted, which might be more relevant for endangered students than the actual grade. It is based on a Naïve Bayes classifier, which has been used in this context successfully in [DG17, PP21], using one of two metrics: In the first variant, which is used by default, it computes the probability of passing a particular exam in the first try depending on how many previous exams were passed in the first try. In the second variant, which is used if the sample size is too small, it computes the passing probability depending on the results achieved in already taken exams. Figure 2 shows how the complete forecast visualization looks like. For the student of interest, we show for each semester the already taken exams (winter semester 20/21 and summer semester 21 in this example) as well as the predicted semester (winter semester 21/22 in this example). For the predicted semester, we show all recommended courses along with the predicted grade ranges, passing probabilities, and sample sizes.

5 Early Warning System

In addition to forecasting the performance of a particular student, our tool also supports the automatic identification of potentially struggling students, so that the study advisor can help these students as early as possible.

To do so, we again utilize our four pre-trained models and predict for each currently enrolled student whether he or she will acquire the degree or not. We then combine this prediction with the amount of already failed exams to compute a "risk score", by which we can order all students. The user can then view all students whose risk score lies above a certain threshold in the frontend, as shown in Figure 3. From this visualization, the study advisor can then directly perform a closer inspection of the performance of the endangered students.

Gefährdung	Student	Ø Note	Fachsemester	Studiengang	Link
9.83	4579990613	4.17	8	Bachelor Informatik	Visualisierung
9.71	4874479461	5.00	4	Bachelor Informatik	Visualisierung
9.70	5237004224	4.23	6	Bachelor Informatik	Visualisierung

Fig. 3: List of "endangered students", who have a high risk score.

Note that we also support the visualization of "risky courses" that seem to be related to dropouts. Here, we define risk as the number of students who failed an exam and then dropped out in relation to the total number of students who took the corresponding course. Figure 4 shows an example list representation of such identified courses.

Veranstaltung	Anzahl gescheiterter Studenten	Anzahl Studenten an Veranstaltung teilgenommen	Anteil
Einführung in die Höhere Mathematik (Analysis)	18	90	20.00%
Elementare Algebra und Zahlentheorie	28	204	13.73%
Diskrete Mathematik für Informatiker	93	701	13.27%

Fig. 4: List of "risky courses" that are potentially related to dropouts of students.

6 Demonstration

In our on-site demonstration, the users will be able to freely interact with our tool, which we pre-load with anonymized real-world student data from the computer science B.Sc. course of study of JGU Mainz. The audience will step into the role of a study advisor and experience the entire tool-assisted workflow: Identifying potentially endangered students via the warning system, exploring their individual problems by navigating through the different visualizations, and finally predicting the best courses to take alongside with their chances of success. By inspecting the courses year-by-year, the audience can also identify temporal trends in the passing/failing ratio and the number of students visiting the course.

Acknowledgements: We would like to thank Hans-Jürgen Schröder for his contributions to early stages of this work and his efforts in acquiring the anonymized study data. Also, we would like to thank JGU Mainz for providing the anonymized study data, as well as Markus Blumenstock for his input as a study advisor.

Bibliography

[DG17] Dake, Delali Kwasi; Gyimah, Esther: Students Grades Predictor using Naïve Bayes Classifier – A Case Study of University of Education, Winneba. International Journal of Innovative Research in Science, Engineering and Technology, 6(10), 2017.

[Fr18] Francis, Nadime; Green, Alastair; Guagliardo, Paolo; Libkin, Leonid; Lindaaker, Tobias; Marsault, Victor; Plantikow, Stefan; Rydberg, Mats; Selmer, Petra; Taylor, Andrés: Cypher: An Evolving Query Language for Property Graphs. In (Das, Gautam; Jermaine, Christopher M.; Bernstein, Philip A., eds): Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018. ACM, pp. 1433–1445, 2018.

[ht22] https://neo4j.com, 2022.

[Jo22] Johannes Gutenberg University Mainz: , https://jogustine.uni-mainz.de/, 2022.

[Má13] Márquez-Vera, Carlos; Cano, Alberto; Romero, Cristóbal; Ventura, Sebastián: Predicting student failure at school using genetic programming and different data mining approaches with high dimensional and imbalanced data. Appl. Intell., 38(3):315–330, 2013.

[PK19] Pensel, Lukas; Kramer, Stefan: Forecast of Study Success in the STEM Disciplines Based Solely on Academic Records. In (Cellier, Peggy; Driessens, Kurt, eds): Machine Learning and Knowledge Discovery in Databases - International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16-20, 2019, Proceedings, Part I. volume 1167 of Communications in Computer and Information Science. Springer, pp. 647–657, 2019.

[PP21] Perez, Joann Galopo; Perez, Eugene S.: Predicting Student Program Completion Using Naïve Bayes Classification Algorithm. International Journal of Modern Education and Computer Science (IJMECS), 13(3):57–67, 2021.

[Zh20] Zhao, Yijun; Xu, Qiangwen; Chen, Ming; Weiss, Gary: Predicting Student Performance in a Master's Program in Data Science using Admissions Data. In (Rafferty, Anna N.; Whitehill, Jacob; Romero, Cristóbal; Cavalli-Sforza, Violetta, eds): Proceedings of the 13th International Conference on Educational Data Mining, EDM 2020, Fully virtual conference, July 10-13, 2020. International Educational Data Mining Society, 2020.