Poster: Teaching Predictive Modeling to Junior Software Engineers—Seminar Format and Its Evaluation

Katsiaryna Labunets
University of Trento
Trento, Italy
katsiaryna.labunets@unitn.it

Andrea Janes
Free University of Bozen-Bolzano
Bolzano, Italy
andrea.janes@unibz.it

Michael Felderer
University of Innsbruck
Innsbruck, Austria
michael.felderer@uibk.ac.at

Fabio Massacci
University of Trento
Trento, Italy
fabio.massacci@unitn.it

***Abstract*—Due to the increased importance of machine learning in software and security engineering, effective trainings are needed that allow software engineers to learn the required basic knowledge to understand and successfully apply prediction models fast. In this paper, we present a two-days seminar to teach machine learning-based prediction in software engineering and the evaluation of its learning effects based on Bloom's taxonomy. As a teaching scenario for the practical part, we used a paper reporting a research study on the application of machine learning techniques to predict vulnerabilities in the code. The results of the evaluation showed that the seminar is an appropriate format for teaching predictive modeling to software engineers. The participants were very enthusiastic and self-motivated to learn about the topic and the empirical investigation based on Bloom's taxonomy showed positive learning effects on the knowledge, comprehension, application, analysis, and evaluation level.**

***Keywords*-Machine Learning, Predictive models, Empirical Software Engineering, Bloom's taxonomy**

I. Introduction

The demand for data analytics know-how in software projects to support informed decision-making grows rapidly [1].

In this paper, we present a two-day seminar to teach machine learning-based prediction in software engineering and the evaluation of its learning effects. The seminar was given to students from the Free University of Bozen-Bolzano, the University of Trento, and the University of Innsbruck in a jointly funded seminar. As the students had working experience and a good background in programming and software engineering, the situation is comparable to software engineers in practice [2].

The paper is structured as follows: Section II provides an overview of the research questions, Section III presents the base study and teaching workflow of the provided training, Section IV presents the underlying data collection and data analysis, and Section V then discusses the results.

II. Research Design

The overall goal of our study was to develop a suitable seminar format to teach the basics of predictive modeling to software and security engineering students and to empirically evaluate the learning effects. The seminar format has been selected for pragmatic reasons. However, software developers who work in a company do not have time to attend a several days course with lectures on prediction modeling, while a two-day seminar is easy to adopt by companies as a part of their education program. At the same time, seminar provides theoretical background and practical skills in prediction modeling. Seminars of 2-3 days is a widely used learning format in IT industry.

We investigated the following two research questions:

- *RQ1*: How could a seminar to teach prediction models be designed to promote the use of the method among novices?
- *RQ2*: How effective is the chosen format of the seminar to teach prediction models?

III. Teaching Instrument

A. Base Study

To motivate the methods and tools explained and used during the seminar, we decided to base the content of the seminar on a previously published study, which acts as a scenario that explains in which context the presented methods are used and which problems they solve. We chose the paper "Predicting Vulnerable Software Components via Text Mining" by R. Scandariato et al. [3]. We have selected this work for three reasons. First, one of the organizers of the seminar was already familiar with the content of the work, so it took less time to understand and to prepare the replication of the study to be done together with participants. Second, based on our previous experience with students of the participating universities, we knew that security is a topic students find fascinating and, therefore, we expected a higher interest to participate in a seminar dealing with security vulnerabilities than other topics. Third, the paper was published in a high-quality journal, IEEE Transactions on Software Engineering, which indicates the high quality of the study.

B. Teaching Workflow

The seminar had a total duration of 11.5 hours, distributed over two days (8 hours excluding breaks). Table I lists the agenda followed during the seminar.

TABLE I: Time table of the seminar

Day 1: Preparation and background

Time	Activity
17:45	Welcome, consent forms, and Questionnaire 1: Background and demographics.
18:15	Tutorial 1: Empirical methods and software and security engineering.
19:00	Installation of the tools used during the seminar.
20:00	Dinner

Day 2: Execution

Time	Activity
09:00	Tutorial 2: Data collection and data preparation.
09:30	Tutorial 3: Machine learning and WEKA.
11:00	Group exercise 1: Determine the vulnerability warnings of *the last* version of an open-source application and build a classifier.
12:00	Lunch
13:45	Group exercise 2: Determine the vulnerability warnings of *the first* version of an open-source application, build a classifier, and *apply* the generated classifier to *several* versions of an open-source application.
14:15	Group exercise 3: Apply the classifier trained with one of the five open-source applications to *another* one of the five open-source applications.
15:00	Coffee break
15:15	Questionnaire 2: Individual feedback about the seminar
15:30	Wrap-up about the results, presentation of the paper by R. Scandariato et al. and its discussion.
17:30	Event closure.

IV. Research Execution

In this section, we describe the pre- and post-task questionnaires that were distributed to the participants correspondingly at the beginning and at the end of the seminar.

1) Pre-task questionnaire: To control the possible effect of confounding factors, we asked the participants to fill in a *pre-task questionnaire*. This questionnaire assessed (in addition to the host university and the working experience) the participants' perception about their knowledge in Software and Security Engineering, Java programming, Empirical Research, Statistics, and Machine Learning. For each knowledge area, we asked to provide *a)* the overall perceived level of expertise in the area as well as *b)* the perceived level of expertise with three core concepts of the area chosen by us. For instance, in the area of Java programming we asked if the participant is familiar with the concept of "Garbage Collection", in the area of statistics with the concept "Standard Deviation".

As an overall measure of participants' level of knowledge in the area we took the average of the responses to four questions (one self-assessment and three on core concepts).

2) Post-task questionnaire: Table II presents the post-task questions that we used to evaluate the learning effect of the seminar. The questions were designed following Bloom's taxonomy [4], which considers the cognitive levels knowledge, comprehension, application, analysis, and evaluation.

V. Results and Discussion

In response to $RQ1$, we proposed a two-day seminar aiming to introduce participants to the area of machine learning and teach how to use predictive modeling to detect vulnerabilities in source code. The tutorial part of the seminar provides a *a)* general introduction into Empirical Methods and Software and Security Engineering, *b)* detailed explanation on data collection and preparation, and *c)* basics of machine learning and use of WEKA tool. The practical part included three guided exercises aiming to replicate the study by Scandariato et al. [3]. The idea of using a research study as a scenario of the practical part turned out to be a success. Participants appreciated the practical illustration how machine learning techniques can be used to solve a problem that they would never think is possible to apply to. Based on the experience of the first seminar and feedback from the participants, such a seminar should be longer, at least two full days of work (i.e. 16 hours). The additional time should be used for individual exercises helping participants to apply the obtained knowledge.

TABLE II: Post-task questionnaire

No.	Type	Cognitive level	Statement
$Q1.1$	Open	Knowledge	What is machine learning?
$Q1.2$	Open	Knowledge	What is empirical research?
$Q1.3$	Open	Knowledge	To what type of machine learning algorithms does classification belong?
$Q1.4$	Open	Knowledge	How would you explain the difference between supervised and unsupervised machine learning?
$Q1.5$	Open	Knowledge	What is a vulnerability?
$Q1.6$	Open	Knowledge	What is precision?
$Q2.1$	Closed	Comprehension	What are the advantages of predictive modeling? Why?
$Q2.2$	Closed	Comprehension	What are the disadvantages of predictive modeling? Why?
$Q2.3$	Closed	Comprehension	Where do you think predictive models cannot be applied? Why?
$Q2.4$	Closed	Comprehension	Can you explain what is happening when you apply a trained classifier to a test data set?
$Q3.1$	Closed	Application	How would you use prediction models in another software engineering topic (e.g., requirements engineering, cost estimation, risk analysis, code analysis, etc.)? Please, make an example.
$Q3.2$	Closed	Application	How would you use prediction models in your course work (in the future work)? Please, make an example.
$Q3.3$	Closed	Application	How would you apply what you learned to develop your own spam filter?
$Q4.1$	Closed	Analysis	Why should a model trained on an application X work on application Y? Why not? Please explain.
$Q4.2$	Closed	Analysis	In the precision and recall analysis we have accepted a threshold of 80%. Why we did not use 100%?
$Q5.1$	Closed	Evaluation	What do you think about using prediction modeling for vulnerability detection? Why?

Regarding $RQ2$, the results of the post-task questionnaire showed that the participants demonstrated rather high overall quality at the levels of Knowledge (78%), Comprehension (72%) and Application (68%), and medium quality results at the Analysis level (59%). However, we need to provide a better explanation of "how things work" in predictive modeling as the participants experience problems in responding to $Q2.4$. The Analysis level requires better support by the seminar structure as participants showed fair quality of responses at this level. Possibly, having a more interactive format of the practical exercises, as proposed by our participants, will help participants to practice skills related to Application and Analysis levels.

References

[1] A. Begel and T. Zimmermann, "Analyze this! 145 questions for data scientists in software engineering," in *Proc. of ICSE 2014*. ACM, 2014.

[2] M. Höst, B. Regnell, and C. Wohlin, "Using students as subjects—a comparative study of students and professionals in lead-time impact assessment," *Empir. Softw. Eng.*, vol. 5, no. 3, Nov. 2000.

[3] R. Scandariato, J. Walden, A. Hovsepyan, and W. Joosen, "Predicting vulnerable software components via text mining," *IEEE T. Software Eng.*, vol. 40, no. 10, 2014.

[4] B. S. Bloom, "Taxonomy of educational objectives: The classification of educational goals," 1956.