Predicting Student Success in Cybersecurity Exercises with a Support Vector Classifier

Quinn Vinlove[1], Jens Mache[1], Richard Weiss[2]

[1]Lewis & Clark College, Portland, Oregon; [2]The Evergreen State College, Olympia, WA

Introduction

Capture the flag (CTF) exercises have been used in cybersecurity for fun and for training and education for many years. In our own experience, students often become frustrated when they do not make progress. This has also been reported by others [8]. Even with instructor office hours and TA support, many students have trouble and struggle with completing hands-on exercises. Our goal is to provide tools to help students be more engaged by automatically detecting when they are struggling, and to potentially offer a hint when that happens.

We used the 'Introduction to DETER and Unix', or 'Intro' exercise [6] on DeterLab [7], completed by 25 students in a small cybersecurity class, and developed features based on the resulting annotated bash history files produced by ACSLE [1], an automated reporting and log file collection program for testbeds. While the first class we tested our methods on was small, the data collection and annotation system would scale to larger classes and potentially give even more precise results with more data.

We evaluated the classifier using primarily the number of milestones completed, but also the log files themselves with both student input and command line output. We did not have ground truth information on each student as to whether they struggled or not. By constructing three features from this data: the number of related commands between each new milestone achieved, number of repeated commands, and distance from a few 'known good' commands, and using these features as inputs into a multi-dimensional support vector classifier, we were able to predict exercise completion with 80% accuracy. Our work will provide a good basis for a new model to suggest hints automatically.

Methods

To construct inputs for our classifier, we first created three features, computed them for every student, and checked our work by evaluating each log file by hand. These features are described below.

The first metric was the average number of commands between new milestones reached. This could possibly describe how much effort was spent for each learning objective met. Initial analysis showed that students who completed more milestones either entered relatively few commands for each milestone that they reached, or took their time and entered more, possibly indicating that they didn't know the answer at first, but worked hard to complete the exercise.

The second metric was the longest string of repeats of any single command. To count commands that were close, but not identical, we would compute the edit distance (or Levenshtein distance) from each new command to the last.

The last metric we used was the smallest edit distance between a list of 'known good' commands and bash history. This operated on the initial assumption that while the REGEX search method of seeing if a command matches a milestone may be good, it is binary, and sometimes a continuous metric can produce better results.

The exercise log data from two classes at USC and one at LC were loaded into a Jupyter notebook with Pandas, processed with the help of numpy, then normalized with a min-max scalar, shuffled, and split 80-20 between the testing set and validation set. For desired tags, we assigned a true to each student if they got 6 or 7 milestones (all of them) and a false if they didn't. Then, we placed this data into a scikit-learn support vector classifier.

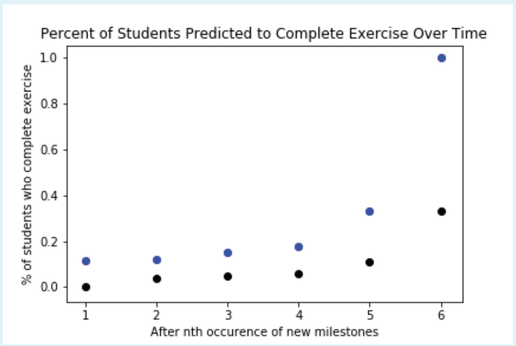

Figure 1: This figure shows the percentage of students who achieved one or more new milestones, then went on to complete the exercise. As students progress and continue to reach new milestones, more stop working. As fewer continue, the ones who do end up with a higher likelihood of completing. The blue dots represent the actual value, and the black dots represent the predicted value.

Results

After training a support vector classifier with the training set, we evaluated our work with the validation set and found that the classifier was able to accurately predict true or false values for 80% of the data. The confusion matrix is shown in Figure 3. Note that the classifier is pessimistic, which is good, since we want to minimize false negatives. There were no false negatives with the sample data. There were some false positives, i.e. it predicted that 4 out of 20 students would not finish but they did.

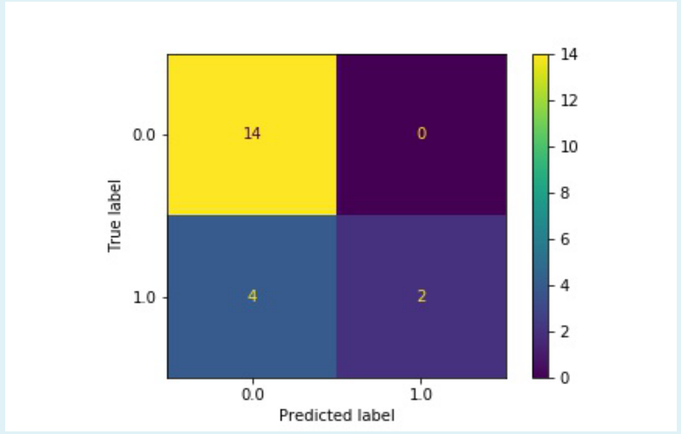

Figure 3: Confusion matrix for the evaluation dataset after training our SVC.Note here that 0.0 represents a 'will not complete' prediction and 1.0 represents a 'will complete' prediction. Each point in the matrix represents a measured state and its outcome.

Works Cited

[1] Jelena Mirkovic, Aashray Aggarwal, David Weinman, Paul Lepe, Jens Mache, and Richard Weiss. 2020. Using Terminal Histories to Monitor Student Progress on Hands-on Exercises. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20). Association for Computing Machinery, New York, NY, USA, 866–872. DOI:https://doi.org/10.1145/3328778.3366935

[2] Pedregosa et al. 2011. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825-2830.

[3] Chris Piech, Mehran Sahami, Daphne Koller, Steve Cooper, and Paulo Blikstein. 2012. Modeling how students learn to program. In Proceedings of the 43rd ACM technical symposium on Computer Science Education (SIGCSE '12). Association for Computing Machinery, New York, NY, USA, 153–160. DOI:https://doi.org/10.1145/2157136.2157182

[4] Anna N. Rafferty, Michelle M. LaMar, and Thomas L. Griffiths. 2014. Inferring Learners' Knowledge From Their Actions. Cognitive Science 39, 3, 584-618. DOI:https://doi.org/10.1111/cogs.12157

[5] Valdemar Švábenský, Jan Vykopal, and Pavel Čeleda. 2020. What Are Cybersecurity Education Papers About? A Systematic Literature Review of SIGCSE and ITiCSE Conferences. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20). Association for Computing Machinery, New York, NY, USA, 2–8. DOI:https://doi.org/10.1145/3328778.3366816

[6] Peter A. H. Peterson, and Peter Reiher, Introduction to DETER and Unix, https://www.isi.deterlab.net/file.php?file=/share/shared/LinuxandDeterLabintro

[7] Jelena Mirkovic and T. Benzel. Teaching Cybersecurity with DeterLab. IEEE Security and Privacy Magazine, 10(1):73-76, January/February 2012

[8] Kevin Chung and Julian Cohen, Learning Obstacles in the Capture The Flag Model, 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14), 2014.

Discussion

Drawing conclusions based on this data set was limited by not having a direct measure of which students actually struggled; nevertheless, we were able to infer this in many cases by reading the bash history logs and counting the milestones achieved.

One of the simplest trends we observed was that more persistent students would score higher on the exercises. Encouraging persistence may increase exercise completion rate. In the future, we plan to interview students afterwards to see what they struggled with and correlate that with their command line history.

Some preliminary work explored how well this model applies when students are just beginning the exercise, not just at the end of it. A variation of this model trained only on the snippets of bash history between new milestones achieved still maintains 80% accuracy (Fig. 2), and points toward increasing completion rates as students persist longer. (Fig. 1) To construct this dataset and generate more data, we sliced the data for each student several times progressively, so that a single student who, for example, achieved three new milestones, had three data points, each representing what their bash history data would look like if they stopped working at that point.

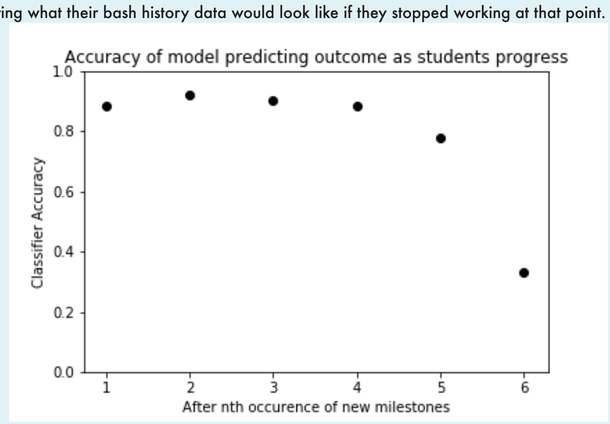

Figure 2: Classifier accuracy for the scale on the x axis described by Fig.1. Our classifier is able to accurately predict exercise completion with 80% accuracy or better for the first five occurrences of new milestones.

Conclusion & Future Work

Since the ACSLE system shows the failed attempts at a milestone, it makes it easier for exercise authors to identify common misconceptions and then produce hints for each of them. One way to use this would be to alert the instructor when a student is struggling, and the instructor could choose one of the pre-recorded hints based on a digest of the student's failed attempts.

In the future, instructors will be able to write a single file that describes commands that students could use in a solution. The string-edit distance between what they typed and those commands would be used to assess progress. Potentially, 'snippets' of them could be used as a hint to assist stuck students after our system detects that they may not finish.

Acknowledgments

We would like to thank Jelena Mirkovic. This work was partially supported by National Science Foundation grants 1723714 and 1723705. Datasets, along with the accompanying Jupyter Notebook, can be found on GitHub at https://github.com/edurange/mlearn.