The Art and Science of Analyzing Software Data; Quantitative Methods

(ICSE Technical Briefing)

Tim Menzies
Computer Science,
North Carolina State University, USA
tim.menzies@gmail.com

Leandro Minku
Comptuer Science
University of Birmingham, UK
l.l.minku@cs.bham.ac.uk

Fayola Peters
Lero, University
of Limmerick, Ireland
fayolapeters@gmail.com

***Abstract*—Using the tools of quantitative data science, software engineers that can predict useful information on new projects based on past projects. This tutorial reflects on the state-of-the-art in quantitative reasoning in this important field. This tutorial discusses the following: (a) when local data is scarce, we show how to adapt data from other organizations to local problems; (b) when working with data of dubious quality, we show how to prune spurious information; (c) when data or models seem too complex, we show how to simplify data mining results; (d) when the world changes, and old models need to be updated, we show how to handle those updates; (e) when the effect is too complex for one model, we show to how reason over ensembles.**

I. Overview

In the age of big data, data science for software engineering is a very active area. In the 21st century, it is now impossible to manually browse all the available software project data. The PROMISE repository of SE data has grown to 100+ projects and this is just one of over a dozen open-source repositories that are readily available to researchers. There are many sources of information; e.g. example, as of October 2012, Mozilla Firefox had 800K reports on projects; and Source-forge.net Github host 324K and 11.2M projects.

There are many successful results in this field (see Figure 1). These results have inspired much interest in this kind of data analysis in the industrial and research communities. A search through Amazon.com reveals dozens of new data mining texts, just in the last 2 years. Yet most of those texts are overly concerned with the specific details of particular data miners. Industrial practitioners should be able to use a range of approaches as decision support tools which allow prediction of useful information on new software projects based on completed projects.

This tutorial is our reflection and summary on the state-of-the-art in quantitative reasoning in this important field. For further information, see our recent book on this topic (see Figure 2).

II. Details

Target audience: Software practitioners and researchers wanting to understand the state of the art in using data mining for software engineering (SE) data.

Pre-requisites: This tutorial makes minimal use of maths of advanced algorithms and would be understandable by developers and technical managers.

Christian Bird found that it was possible to build high-quality software using teams across the planet, just as long as the management is structured around code functionality (who works on what) and not merely on geography (who sits where) [1].

Another result from AT&T [3] (which has been repeated several times [2], [4]) is that static code measures can be used to guide inspection teams to the small parts of the code containing most defects; e.g. inspection teams can find 80% to 88% of the defects after inspecting 20% to 25% of the code.

For more examples, see the long list of applications discussed in the July and September 2013 issues of IEEE Software special issues on Software Analytics, edited by Menzies & Zimemrmann.

Fig. 1. Successful results, analyzing software data.

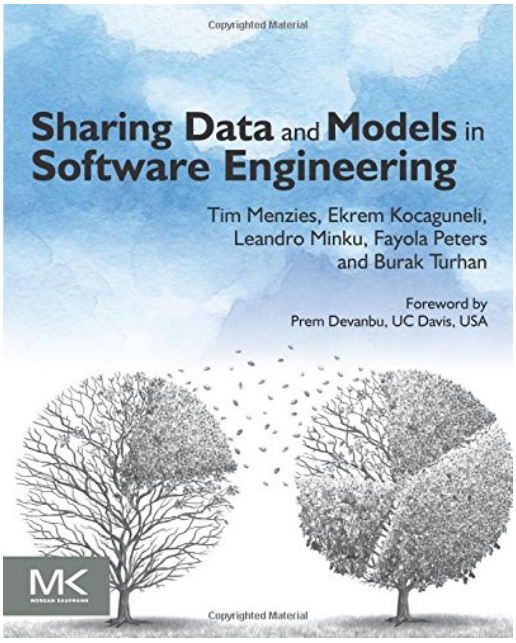

Fig. 2. Sharing Data and Models, available from Amazon.com.

Overall goal of the tutorial: Better data mining by better skilled software engineers.

Why the topic would be of interest to a broad section of the software engineering community:

We show how to acquire knowledge from software engineering data, so that software engineers can make informed decisions.

III. About the Presenters

Tim Menzies (Ph.D., UNSW, 1995) is a full Professor in Computer Science at North Carolina State University where he teaches software engineering and automated software engineering. His research relates to synergies between human and artificial intelligence, with particular application to data mining for software engineering. He is the author of over 230 referred publications; and is one of the 100 most cited authors in software engineering out of over 80,000 researchers (http://goo.gl/SSPX3J). In his career, he has been a lead researcher on projects for NSF, NIJ, DoD, NASA, USDA, as well as joint research work with private companies. Prof. Menzies is the co-founder of the PROMISE conference series devoted to reproducible experiments in software engineering (http://openscience.us/repo). He is an associate editor of IEEE Transactions on Software Engineering, Empirical Software Engineering and the Automated Software Engineering Journal. In 2016, he will serve as co-general chair for ICSME'16.

Leandro L. Minku is a Research Fellow II at the Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science, the University of Birmingham (UK). He received the PhD degree in Computer Science from the University of Birmingham (UK) in 2011, and was an intern at Google Zurich for six months in 2009/2010. He was the recipient of the Overseas Research Students Award (ORSAS) from the British government and several scholarships from the Brazilian Council for Scientific and Technological Development (CNPq). His research focuses on software prediction models, on-line/incremental machine learning for changing environments, and ensembles of learning machines. He has published in internationally renowned journals such as ACM Transactions on Software Engineering and Methodology, IEEE Transactions on Knowledge and Data Engineering.

Fayola Peters is a Postdoctoral Researcher with Lero - The Irish Software Research Center at the University of Limerick in Ireland. She received the PhD in Computer Science from West Virginia University in 2014. Her research focuses on handling privacy issues related to supporting privacy preserving data sharing for data owners (as well as software users). She has published at top software engineering venues like ICSE, IEEE TSE and ESEM. She has also been a curator for the PROMISE repository since 2011.

References

[1] Christian Bird, Nachiappan Nagappan, Premkumar Devanbu, Harald Gall, and Brendan Murphy. Does distributed development affect software quality?: an empirical case study of windows vista. *Commun. ACM*, 52:85–93, August 2009.

[2] Tim Menzies, Z. Milton, B. Turhan, B. Cukic, Y. Jiang, and A. Bener. Defect prediction from static code features: Current results, limitations, new approaches. *Automated Software Engineering*, (4), December 2010. Available from http://menzies.us/pdf/10which.pdf.

[3] Thomas J. Ostrand, Elaine J. Weyuker, and Robert M. Bell. Where the bugs are. In *ISSTA '04: Proceedings of the 2004 ACM SIGSOFT international symposium on Software testing and analysis*, pages 86–96, New York, NY, USA, 2004. ACM.

[4] A. Tosun, A. Bener, and R. Kale. AI-based software defect predictors: Applications and benefits in a case study. In *Twenty-Second IAAI Conference on Artificial Intelligence*, 2010.