Search-Based Predictive Modelling for Software Engineering: How Far Have We Gone?*

Federica Sarro

Department of Computer Science, University College London, f.sarro@ucl.ac.uk
http://www0.cs.ucl.ac.uk/staff/F.Sarro/

Abstract. In this keynote I introduce the use of Predictive Analytics for Software Engineering (SE) and then focus on the use of search-based heuristics to tackle long-standing SE prediction problems including (but not limited to) software development effort estimation and software defect prediction. I review recent research in Search-Based Predictive Modelling for SE in order to assess the maturity of the field and point out promising research directions. I conclude my keynote by discussing best practices for a rigorous and realistic empirical evaluation of search-based predictive models, a condicio sine qua non to facilitate the adoption of prediction models in software industry practices.

Keywords: Predictive Analytics · Predictive Modelling · Search-Based Software Engineering · Machine Learning · Software Analytics

1 Introduction

Nowadays software pervades almost every aspect of our life. This allows the production and collection of a large amount of information about people’s decisions and behaviours. Predictive Analytics exploits such information through intelligent systems which are able to identify patterns and predict future outcomes and trends. Applied to Software Engineering, predictive analytics helps us better understand software processes, products and customers in order to maximise product quality, users satisfaction, and revenues [27].

One of the most important use of Predictive Analytics for Software Engineering is building prediction systems to estimate crucial software aspects and support engineers throughout the software production life-cycle (a.k.a Predictive Modelling for Software Engineering). Examples of software engineering prediction problems are: estimating the amount of effort likely required to develop or maintain software [25, 11], estimating the successes of mobile applications [29] and identifying software that will most likely contain defects [12], cause crashes [33] or fail tests [19].

Predictive Modelling for Software Engineering has been an important and active research field that can be dated back to 1971, when the first attempt to

* This paper provides an outline of the keynote talk given by Dr. Federica Sarro at SSBSE 2019, with pointers to the literature for details of the results covered.
estimate the number of software defects was made [18]. Since then, predictive systems of various nature have been proposed ranging from statistical models and analogy-based techniques to machine learning and search-based methods. In particular, over the past 10 years, search-based prediction systems have been specifically devised to tackle long-standing software engineering prediction problems such as software development effort, defect proneness, maintainability and change proneness [14, 21]. These systems are either stand-alone systems able to build optimal prediction models [6, 10, 25] or ones that are used in combination with other (usually machine learning-based) estimators [3–5, 20, 24]. A variety of meta-heuristics based on both local and global search techniques (e.g., Simulated Annealing, Tabu Search, Genetic Algorithm, Genetic Programming) has been used, with the latter being definitively the most studied [8, 21, 26] and with Multi-Objective Evolutionary Algorithm usually resulting in the most effective approach for different prediction tasks (see e.g. [2, 25]).

In this keynote I explain how to use search-based heuristics to tackle software engineering prediction problems. I also highlight their strengths and weaknesses with respect to more traditional statistical or machine learning-based estimators. Some of these are the possibility to use one or multiple desired measures as a fitness function to evolve optimal prediction models [2, 7, 25, 28] and the need of scalable solutions [9, 23]. I review the most promising results in this field and also envisage novel applications of search-based heuristics to predictive modelling for SE; this includes using them to analyse interesting trade-offs (e.g. models' predictive quality vs. interpretability) and to test machine learning-based predictors, both of which are challenges currently faced by the wider SE community. I conclude my keynote by discussing best practices for a rigorous [1, 16, 22, 30, 31] and realistic [13, 15, 17, 32] empirical assessment and evaluation of search-based predictive models, which is a condicio sine qua non to grow this field and to facilitate the adoption of prediction models in software industry practices.

References


