Effectiveness of Tree based Learners in Incremental Dataset of Software Defect Predictions - A Case Study

Md Rayhanur Rahman[*]
Ph.D. Student
Dept. of CSC
NC State University
Raleigh, NC, USA
mrahman@ncsu.edu

Abstract

In various research fields, data miners are being applied in an intense manner such as in domains of computing, space, business intelligence etc. In software engineering domain, it is also being used extensively. One of the key area where it is being applied is defect prediction. Defect prediction models helps software developers to control the quality issues of the software projects. There are a diversified range of data miners utilized to predict defects from the software metrics such as simple regressors and classifiers as well as complex multi objective models. These days, source codes of the software can be obtained from the github and other repositories easily and code metrics can be computed on the fly. This indicates that datasets for defect prediction can grow to a large volume incrementally over time. In such scenario, scalability challange will appear as the data become larger and larger. In this research, we will try to compare the offline tree based classifiers and online VFDT classifier to observe the case of classifiying defects from large datasets. From our observation, we found out that, online classifier behaves more stable and produces similar results without paying the penalty of time.

Keywords Defect Prediction, Decision Tree, Random Forest, VFDT, FFT, Online Analysis

1 Introduction

These days, software automation has engulfed all the spheres of our life. Millions of software companies automates new business logics along with replacing legacy systems with its modern descendant. Consequently, software development is an ongoing process that never stops and hence, there will be always defects in the software sources that needs to be fixed in constant manner. Fixing these defects is one of the key concern in software quality assurance activities and dedicated human resources spend a significant time in resolving these issues - if unfixed, culminates in loss of money, time, consumer satisfaction and in mission critical cases, casualties.

Finding and fixing software defects was a manual task decades ago. But these days, data miners are there to help the developers find these defects. These data miners work on the defect prediction models that mainly contains software code quality metrics such as line of code, cyclometric complexity etc. Based on these data, those data miners predicts whether a particular software module would contain defects or not.

...To perform defect prediction studies, researchers have explored the use of various classification techniques to train defect prediction models. For example, in early studies, researchers used simple techniques like logistic regression and linear regression to train defect prediction models. In more recent work, researchers have used more advanced techniques like adaptive regressions, ensemble learning etc. Several context sensitive analysis based defect predictors are explored as well.

...Despite the fact that, there have been numerous advanced miners deployed in defect prediction; classification techniques to build defect prediction models have focused on the performance. However, as the volume of software source codes increases in daily basis, so does the size of defect prediction examples from which the classifiers will predict the defects. Hence, in case of traditional learners, memory and sample size will be a dominant obstacle if those are fed with incrementally large amount of data from time to time. Being trained with small number of data also suffers from overfitting. Meanwhile, currently there are many online data-miners are available which are sample size agnostic. Thus exploring the comparison of traditional learners and online data miners have become a priority.

In this paper, we have put three offline classifier named C4.5 decision tree based classifier, Fast Frugal Tree (FFT), Random forest against online classifier named VFDT. We will look into three research questions in particular:

[*] unity id: 200255928

NCSU, FALL 2018, December 7, 2018, Raleigh, NC, USA
2018.

- **RQ1:** How these four classifiers performs in large defect prediction datasets that will increment over time
- **RQ2:** How hyper-parameters changes as the data size changes
- **RQ3:** How much computation resources would be used by these four learners

We have used four defect prediction datasets obtained from (...). These datasets contains around $40,000$ examples on an average. From our observation we found out that, VFDT performs better and more stable manner than the other three as the data size increases. Moreover, the performance differential becomes more prominent among those learners considering datasize increase and finding tuning parameters.

Rest of the paper is organized as follows. In the section II, we will discuss existing literature study on this aspect. In section III, we will discuss baseline criteria upon which we will evaluate our comparison of the learners, In section IV, we will discuss the experiment setup and result analysis. Finally, it will be followed by discussion and future work scope in the section V.

2 Related Work

bla bla bla

3 Background and Motivation

...

3.1 Baseline Criteria for Data Miners

Baseline criteria refers to the important factors, majority of which should be achieved by a learner. It provides key insight to us regarding how effective and efficient the data miners would perform in the real world. Here follows some of the key baseline crietria along with their short description.

3.1.1 Simple and Reasonable

Learners should be simple in a sense that it is easily explicable to the end users. It should also be easy to understand the underlying models, how it works and how to work on to build on further. There are several learners that not very easy to describe such as Naive Bayes classifiers and neural networks. On the other hands, decision tree based models are simple enough to understand and explain to others. However, learners have to be reasonable as well. It should perform well in terms of accuracy and performance. Otherwise being a simple learning producing non-reasonable results does not help the case of data mining activities.

3.1.2 Stable and Robust

Data miners work on examples to build the model and on the basis of that model, make the predictions. Datasets containing loads of examples poses a significant challenge for the miners. Datasets comes with lots of unique cases as well context aware information. Some datasets are also imbalanced while the others might be incomplete. There is also a potential chance of a lot of anomalies hidden in the dataset. All these factors contribute to the fact of being a data miner unstable. This means the learner would produce different type of decisions in case of diversified datasets which is extremely frustrating for business users. It also prevents further improvement of the data miners. Data miners also need to be robust throughout most of the cases of different sample size, splits and validation techniques.

3.1.3 Generic

Data miners should be as generic as possible referring to the fact that they should provide a range of possible solution rather than a simple one point one. The benefit of it is to help the end users with a possible wide outlook of the scenario. But if the miners behave too specific or provides only one solution of a certain scenario, it would confuse the end users more. There is also a chance to miss other corner cases and ignore other possible, equally similar solutions which would turn out truly bad in real world scenario.

3.1.4 Replicable

The learner should build model from the dataset and produce outputs fast enough. This ensures that the learner can be invoked iteratively in case of learning, understanding, rebuilding or tweaking. If it takse several hours to produce outputs for a small data, it would become impossible to work on that learner or tune a little bit. At the same time, the learner must also produce the same output every time if is fed the same input. If the outcome of a certain learner is not replicable, then it is impossible to understand, implement and improve the learner.

3.1.5 Goal Aware

These days, all the real world problems do not focus on a single goal. Earlier, optimizers were used to maximize or minimize a value over a set of constraints but now a days, complex situations are there where there is no optimal solution. Hence, data miners being applied in those fields should be goal aware. This means, rather focusing on a single goal, the miner should produce output in such a manner so that the output should reflect overall realization of multiple goals. There might be conflicting goals and often, the correlation of the goals and datasets are quite complex. None the less, the miner should be able to handle such conflicts, complexity and tradeoff cases and produce results that provides a satisfaction of all the goals.

3.1.6 Anomaly Aware

It is almost certain and usual that the datasets would contain a percentage of noisy and anomalous examples. Data miners should be able to detect and cancel out those. However, sometimes data miners confuses with the true example as anomalous examples and thus discards those or learn something misleading. A good data miner should be able to handle all these cases.

3.1.7 Context Aware

Despite the fact that a dataset should represents a generic scenario of a particular phenomenon, often this is the case that many context specific information are hidden in the datasets. So in the datasets, there might be a region of locality which is almost similar to other generic examples but might vary in one or two attributes and indicates different labels. Those regions are in fact, responsible for difference in decision making process. Thus, a good data miner should be able to find out those context specific local regions and learn the phenomenon protecting the generic learning model intact as well. This baseline is very critical in fields of health, e-commerce etc.

3.1.8 Incremental

Modern days, datasets tend to be large. They also get incremented periodically. So it is in vain if a data miner build models from a datasets and then can't reuse the already built model if it wants consider the newer example. So data miner should be able to extend the existing knowledge of model reading the newer examples. It should also be able to run over infinite stream of datasets as well as relearn in case of anomalies.

3.1.9 Shareable

The datasets and the obtained knowledge by the learner should be shareable so that the obtained learning can be obtained in different contexts. However, while sharing the datasets, privacy should be a big concern. So miners should be able to hide the actual data and interpret the outcomes at the same time so that sensitive information of the datasets are protected.

3.1.10 Tunable

There are a variety of parameters on every learners that can affect the accuracy, performance and stability of the learners. However, for a large datasets, if it is necessary to find the best set of tuned parameters for the miner, then it would take a hefty amount of time to do so. So a data miner should be able to tune itself so that it can be fit to a certain dataset or certain domain specific models to obtain better results.

In addition, there are several other baselines that deserves mention:

- Be applicable to mixed qualitative and quantitative data
- Have no parameters within the modelling process that require tuning.
- Be publicly available via a reference implementation and associatedenvironment for execution
- Be computationally cheap in a reasonable sense

References