Looking For Generality in Cross-Project Defect Prediction

Tomi Prifti, Adam Nelson, William Mensah, Tim Menzies

Summary

Software defect prediction attempts to provide organizations the opportunity to predict the quality of their software before it is used. However, an important aspect of software development could be the similarity between elements used in the construction of data between different companies. This would provide insight into general measures that should (or shouldn't) be taken in order to improve the overall quality of software. The text here describes methods used in data mining in an attempt to find the existence, or lack thereof, of these similarities.

How to Find Generality?

In the deal situation, a company will have large amount of data from a software project and will use that data to build models that can predict defects for future releases of the software. In practice, however, training data from within the same company may not be available. Using data from other projects or other companies is the only option to build prediction models. An ideal model will be able to predict software defects of a project even if it trained on data obtained outside the company. Zimmerman et. al in their paper try to asses the extend that cross-project data can be used to predicts defects also what kind of software systems are good for cross-predictors. Their results show that simply using projects in the same domain does not work to build accurate prediction models. In our work we are trying to find generality between cross data projects following a different approach based on prism rule learner.

PRISM

Prism constructs a set of rules for each class in the data set. The main difference between a decision tree algorithm and prism is that a decision tree takes all classes into consideration while splitting whereas Prism concentrates on one class at a time, disregarding what happens to the other classes. Prism operates by adding tests to a rule that is under construction, always striving to create a rule with maximum accuracy. Prism will stop constructing rules for a class when all the training instances for that class are classified by the set of rules. The algorithm will iterate through all classes in the dataset resulting with a set of rules for each class.

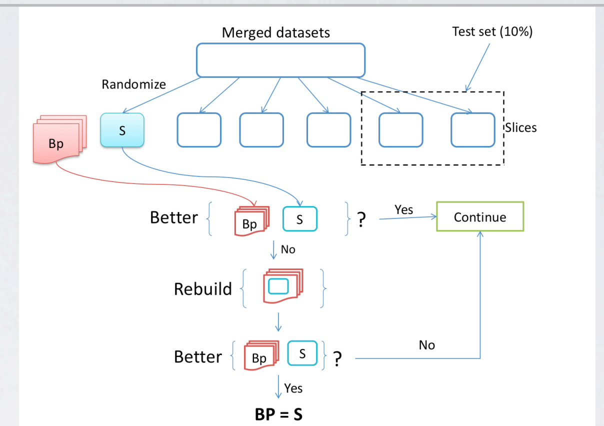
Approach

In general, our experiment involved, combining all the datasets into one gigantic datasets consisting of over 5700 instances and then randomized into 250 slices. Subsequently, 10% of the slices was set aside to be used as the test set and the remaining 90% as the training set. Rules generated for each slice is applied to the test set and then the resulting probability of detection (pd) is tested against that of the Ruleset.

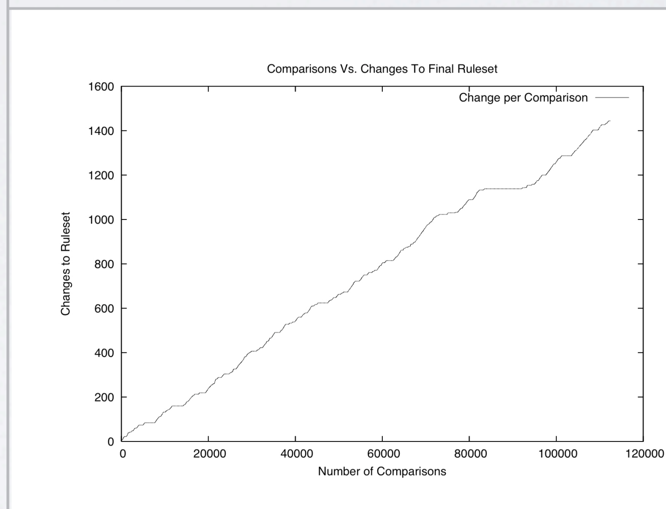
```
for each slice s do
 if Better (ruleset, s)
     continue
 else
     ruleset = Rebuild (s, ruleset)
     if !Better (s, ruleset) then
         ruleset = s
     end if
end for

procedure Better (s, ruleset)
 if pd(ruleset) > pd(s)
     return true
 return false
```

Approach cont.

merged with that of the Ruleset, tested and the comparison is repeated.



Results



The cumulative graph above shows the number of times that Ruleset loses when compared to a new set of rules as we iterate through the training sets. The graph shows clearly that our learner is ***not*** converging to an optimal set of rules that will always win when compared to new set of rules.

The following PD and PF values represent results obtained by training on data set *A*, and attempting to predict on data set *B*. It can be shown that neither learner performs well for predicting cross-company defects.

Results cont.

Rank	Treatment	pd percentiles 25%	50%	75%	2nd quartile median, 3rd quartile
1	NB	12	35	58	
1	PRISM	0	14	28	
					0 50 100

Figure 1: Probability of Detection (PD) results, sorted by median values.

Rank	Treatment	pf percentiles 25%	50%	75%	2nd quartile median, 3rd quartile
1	NB	1	10	18	
1	PRISM	0	1.2	4.3	
					0 50 100

Figure 2: Probability of False Alarm (PF) results, sorted by median values.

Conclusions

According to our approach, and experiments, the prediction methods lack evidence of generality in cross-company data. This evidence is supported by the divergence of rules and the low probabilities of detection. Based on these results, PRISM is not a suitable learner for predicting software defects on data of this nature.

References

1. T. Zimmermann and N. Nagappan and H. Gall and E. Giger and B. Murphy. Cross-project Defect Prediction: A Large Scale Experiment on Data vs. Domain vs. Process. FSE'09

2. Witten, I. H. and Frank, E. Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, 1999.

For More Information

anelson8@mix.wvu.edu
tomi.prifti@gmail.com
wmensah@mix.wvu.edu