A. REPLY TO REVIEWERS

Thank you for your careful reviews of this paper. Our replies to your many useful suggestions are shown below in plain font (and the review comments are shown in *an italic font*).

A.1 EDITOR'S SUMMARY:

I think the constructive substance of reviewer #2's comments can be addressed in a major revision, on top of paying close attention to reviewer #1's suggestions. In particular:

—*Please first pay close attention to and make sure to address reviewer#1's suggestions.*

Please see §2.3.1, §2.3.4, §4.1.1, §6.1.

—*Second, please address Reviewer#2's remarks, especially*

—*Address the regression methods discussed by reviewer#2; actually comparing by means of adding to the study would be most preferable, but if you feel these methods are somehow off the table, please say in detail why.*

Please see §2.3.2.

—*The balance of your related work seems a bit out of whack; please give more space to the other authors in the field listed by reviewer#2, if necessary reducing or compacting your references to your own work.*

We have cut back our own self-references from 24 to 9. We also added more references to the authors mentioned by reviewer 2.

—*Please address the methodological issues raised as well, such as future-vs-present tests and what about using project history?*

Please see the discussion in Reviewer 2's section

Also, reviewer 2 had concerns regarding the practical utility of this work. Clearly, the previous draft was incomplete and did not detail our successful industrial track record. That track record is now described in §2.3.4.

A.2 Reviewer #1

The paper is well written. Its organization is good. I like the background section which gives a lot of information about the related studies and different viewpoints. Generally speaking, I have a favorable opinion of their research.

Thank you for that comment.

However, in this paper, there are important points that deserve consideration and probably additional data analysis and discussion, which requires a major revision. The detailed comments are as follows:

1) A simple ManualUp method gives very good results, which is surprising. A workshop paper is given in the references but is there any other supporting publication or evidence in the literature?

Yes, we have two more journal references on this result from IEEE TSE [Koru et al. 2009] and ESE [Koru et al. 2008].

2) On page 5, the authors try to justify the use of binary measures. However, could this use of binary outcomes be responsible from:

** the curves in Figure 9? It is highly possible that big modules/classes will have more faults. Therefore, counting every defective class as "1" or "true" regardless of its actual count of faults can result in the curves seen in Figure 9.*

** the ceiling effect observed for different learners in many different studies.*

Indeed- one possible explanation for the ceiling effect is that we are *hiding* critical information from our learners and that if we gave them access to all the data, they would do better.

In fact, when we first started working on defect prediction (in 2002), we did try using the actual numbers. However, the data defeated us. For example, in PC1:

—1034 modules have 0 defects
—47 modules have 1 defects
—17 modules have 2 defects
—5 modules have 3 defects
—1 modules has 4 defects
—1 modules has 5 defects
—1 module has 6 defects
—1 module has 7 defects
—1 module has 9 defects

The same pattern repeats in the other data sets: most modules have zero reported defects, and very few have more than one.

But we quite take the reviewer's general point that this issue requires more discussion in the paper. Accordingly, in this draft, we added a new figure 2 (showing the distribution of defects in our data) as well notes on implications of this defect pattern. Specifically, we can't use regression since regression assumes a continuous target variable.

To read that extended discussion, see §2.3.1 and §2.3.2.

3) Re: Figure 9 I am not clear on what corresponds to a point on the x-y axis in this plot. Just for example, let's assume that you have

—100 modules of 10 LOC
—10 modules of 15 LOC
—1 module of 20 LOC

in a product. If the x-axis is %LOC, it becomes necessary to mark the three percentile values for total LOC, at the points 10, 15, and 20 LOC, and their three corresponding PD values. Or, would the authors simply sort all 100 modules of 10 LOC one after each other, and 10 modules of 15 LOC one after another, and so on??

If the authors follow the latter approach, this can greatly affect the shape of the curves. Please elaborate on this issue.

> We follow the latter approach- but the text in the previous version was not clear. We have hence extended the explanation text, see §4.1.1.

It might be useful to look into the mathematical properties of such curves. Solely relying on that an earlier conference paper used this approach (first reference) seems to be naive. Note that, with the latter approach, different distributions of LOC in different data sets could greatly affect the shape of the curves.

> Sometimes, papers can have a wider impact than it would appear, just by looking at the publication venue. Certainly, this is the case with this conference paper. One of the authors of that paper (Briand) is very influential in the field. So much so that, in the period 2007 to 2009 when we conducted this research, we often came across reference to it. Also, in discussions with other researchers on our work, the pd-vs-effort curve of Briand was very often mentioned. So much so, in fact, that this whole paper was motivated by those numerous questions on the nature and value of a pd-vs-pf criteria.

4) If AUC(effort,pd) is the area under the curve, this means that the authors always make comparisons with the worst case scenario (pd jumps to 100% at 100% LOC). Do the authors have any idea about how a random order of modules would perform on average when many random orderings are produced? This kind of randomness must be considered because if the total area under the curve is used as a metric, then the orderings that are indeed worse than random ordering will seem like they are still favorable. This is because there will be always some area under the curve.

> We would defend the current ordering, in order to generate a simple baseline result with manual methods. One of the sobering results of this work is that methods we have advocated for nearly a decade fail (on the AUC(effort,pdf) criteria) when compared to a very simple manual method. We could offer random results- at which point the value of this simple method would be lost in random noise. However, at least to our way of thinking, the utility of this simple manual method is a major result of this work and we want to present it here.

5) Normalizing AUC(effort,pd) by the best line in Figure 9 can result in difficulties while making comparisons rather than simplifying those comparisons. This is

because the best curve can be different from one product to another and normalizing by the best curve can cause unpredictable results and comparisons. Instead, involving the performance of random orderings in the calculations seems to be necessary.

> We agree that exploring random orderings would be useful to (say) show that some learner's conclusions is stable across a range of possible biases. But that is not the intent of this paper- quite the opposite in fact. What we show here is that if switch the evaluation bias from AUC(pf,pd) to AUC(effort,pd) then the set of preferred learners changes dramatically. our conclusion is that *before* embarking on a data mining study, the *first* step must be to map the local business criteria to an evaluation bias (a point we return to after your next comment).

6) The authors state that the recent results have not improved the performances of different learners. This observation is used as a motivation to explore an alternative to the standard goal for learners. Then, the rest of the paper continues on explaining how the use of this new goal results in improved performance for learners. It should be noted that trying different goals for learners randomly is an evolutionary process for research. Consequently, this approach is perhaps too expensive because it spans over many studies.

> Could not agree with you more! Exploring all possible evaluation biases would be a neverending task.
> We therefore need research guidelines on how to NOT explore all possible biases- an issue we return to below for your next point.

The paper presents interesting results. However, it does not discuss how model builders should go about their research design. It is the initial decisions made in the research design which affects all of the results (in this case, similar results from various learners developed in different studies). Would there be more effective research guidelines to be given to software engineering researchers so that they can build learners that matter?

> This is an important point- and one that was not addressed by the previous draft. We have hence added text about research guidelines to the conclusion: see §6.1.

7) Do you assume that effort is proportional to lines of code?

> Yes- based on a previous literature review we conducted on this issue [Menzies et al. 2002].
> But your point is well taken- this issue deserves a little more elaboration in the draft. Hence, based on your question, we added more text to §4.1.3.

A.3 REVIEWER #2

Defect prediction is an important area for software engineering research, and has many potential practical uses in software development. Additionally, I think that it is essential to do empirical studies and believe that replication is respectable and important. But these authors have churned out at least several dozen papers (I think I counted roughly two dozen by subsets of these authors cited in this paper alone) and I have yet to see anything of practical value described in any one of them.

> Thank you for this comment. Clearly, the previous draft was incomplete and did not detail our successful industrial track record. Prompted by this review, we have added notes on our industrial track record to §2.3.4. Based on those notes we must, respectfully, disagree with this reviewer's assessment that our work has never/will never have any practical utility. In fact, based on this reviewer's comments, we would assert that our results are more useful, more industrially relevant that other researchers:
> —This reviewer comments below that *"all of the above-mentioned authors whose work have gone uncited or under-cited are working in an industrial setting and at least have some potential for showing industrial technology transfer, even if their work is not being used yet".*
> —On the other hand, as shown in §2.3.4, our work has been commercialized and applied with demonstrable benefit in the United States and overseas as well.

This is a paper production project, and the closing sentence predicts that more are to come ("We hope that this paper prompts a new cycle of defect prediction research ...")and I predict that their prediction will come true by them.

I see several major flaws in their research:

1) The point of being able to accurately predict which modules will contain defects is to allow the user to predict the FUTURE. I want to be able to look at data that I have NOW and be able to say which modules will contain defects LATER (and of course get the prediction right). However, these authors persist in doing hold out experiments in which they are making predictions about NOW from data collected NOW, and then I suppose that they are claiming that this really tells the user what will happen LATER.

> Our evaluation methods are standard in the field. For example, many of the other researchers that this reviewer is concerned we are not quoting also use hold-out experiments to assess their results.
> Nevertheless, the reviewer's point may hold- is the entire field is in error? To address that issue, we offer the following comments.
> There are many ways to use these defect predictors- including the generation of the $\frac{dDefects}{dt}$ curve (as mentioned above). In our reading of the literature, we have seen that this is mostly done with post-release failures (e.g. a Musa-style analysis). If we had an unambiguous source of post-release failure data (e.g. date-stamped core dumps) then we would certainly generate $\frac{dDefects}{dt}$ curves. However, for other data sources such as the ones processed in this paper, such curves cannot be created since the oracle offer the defect data is a poor temporal oracle.

Without an unambiguous source of temporal data, we fall back to the hold out experiments. Note that, in those experiments, the testing is done on data not used in training. So under the assumption that the test set follows the same distributions as the training set then those test sets could be now or later data, and the predictors will still work. By repeating these hold out experiments many times on random subsets of data, we accumulate evidence on the generalization capabilities of these models.

Another reason to prefer hold out studies is to eliminate bias, which -if exists- affects the decisions of models as well as human-beings. Hold out experiments have nothing to do with time. They are about data points being visible or non-visible to a prediction model during training. The assumption in this field (to the best of our knowledge, there is currently no known way of relaxing this assumption) is that the system producing the data will, in the future, continue to produce data with similar characteristics. Therefore models trained now on visible data are expected to show similar performance on the yet non-visible, or as the reviewer calls, future data.

What is the evidence for that? Would it convince any practitioner to go ahead and apply something from this paper? Software engineering research should ultimately help practitioners improve the way they engineer software. I just don't see this paper moving forward towards this goal.

We offer above several pieces of evidence. For example, in §2.3.4, we list field and controlled studies where our defect predictors have proved demonstrably useful. We apologize for not offering the field study data in the previous draft.

2) Additionally, the authors restrict attention to static code features to make predictions when there is substantial evidence that information about the history of modules such as whether they were defective in the past or have been extensively changed (churn data) is at least as important as static features. But this is not included. Of course since they are not predicting the future, there is no history.

We agree that not only churn data, but also other factors have been shown to be effective in predicting defects. However, this does not invalidate the effectiveness of static code features. The reason for not using churn data is simply a matter of availability. Industrial researchers have access to almost all data resources within their environment, which are, understandably, not shared with other researchers due to confidentiality issues. On the other hand, we can only work on data that are available to us. That is also the reason for employing binary prediction (or classification) rather than regression.

Also, the core of this paper is not model comparison. That is an issue, which has been deeply investigated in our references, specifically Lessmann et.al. We, anyway, compare the proposed method with the currently-best-performing methods (as empirically shown in Lessmann et.al). It is always possible to make comparisons with more and more

methods, like the reviewer suggests the use of binomial regression. However, the possibilities are infinite and we believe that our choices of baseline methods are validated by recent research. The core of this paper is to introduce an innovative, novel idea via the WHICH method: to align, or better embed, business goals with the models used.

3) The paper restricts attention to machine learning predictors to the exclusion of standard statistical regression models. I am not sure why. But given that there is evidence that regression models can be very successful at defect prediction, I do not understand why they do not consider linear or binomial regression, for example.

Thank you for this comment- it prompted us to add notes on regression in §2.3.2. As stated in that section, Regression assumes a continuous target variable and, as discussed in §2.3.1, our target variable is binary and discrete. Also, there is no definitive result showing that regression methods are better/worse than the data miners used in this study. In one of the more elaborate recent studies, Lessmann et al. found no statistically significant advantage of logistic regression over a large range of other algorithms [Lessmann et al. 2008]. Finally, in previous work we have assessed various learning methods (including regression methods and model trees) in terms of their ability be guided by various business considerations. Specifically, we sought learners that could tune their conclusions to user-supplied utility weights about false alarms, probability of detection, etc. Of the fifteen defect prediction methods used in that study, regression and model trees were remarkably *worst* at being able to be guided in this way. The last section of this paper discusses a new learner, called WHICH, that was specially designed to support simple tuning to user-specific criteria.

4) They only consider binary classifications. The module is either buggy or not. They say that that is because they only do pre-release analysis and there might be defects after release. Of course that is true, but that is what a project needs predicted.

We describe above how companies have successfully (even eagerly) used our binary defect predictors. That is, there is more to defect prediction that just learning the $\frac{dDetects}{dt}$ curve.

Another issue of concern is the set of papers cited. As mentioned above, they cite roughly 24 papers by members of this group. You would think that they are the only people doing research of this nature. This is certainly not true. I saw NO papers by Zimmerman, Zeller, or Mockus all of which are certainly presences in the field and have done high-quality defect prediction research. Both Zimmerman and Mockus work in industry, by the way. Additionally, I only noticed one paper by Ostrand and Weyuker who have worked in the field for many years, one paper by Briand, and two by Nagappan (one of his papers appears twice in the references). Each of these researchers have done major studies using industrial software, and all work in industry. The group at AT&T have made predictions for several different systems over many different releases. I am certain that other researchers I have forgotten

have been similarly overlooked. And even when a paper by, say Nagappan or Briand or Weyuker was cited, it was done just in passing and is not necessarily the most relevant paper to cite. Notice that all of the above-mentioned authors whose work have gone uncited or under-cited are working in an industrial setting and at least have some potential for showing industrial technology transfer, even if their work is not being used yet.

We have reduced our self-references from 24 to 9.

We note that this comment actually endorses our use of hold-out experiments to validate our learned models. Many of the researchers mentioned above interesting feature of the above comment is that the researchers that the reviewer notes use hold out experiments in their research.

As to the specifics of the above comment, the lack of reference to Zimmermann and Mockus is a clear mistake. They have been added to this paper.

As to Briand, his presence and impact on this work is very present and clearly documented.

As to Zeller, his work (at least, his more recent work at, say, ASE 2009) is more static code analysis than learning from static defect predictors.

As to other authors, there are many many more we could mention (Khoshgoftaar, Srinivasan and Fisher, Porter and Selby, etc etc). But one paper cannot cover an entire field. Some bias must be imposed, otherwise there would be no room in this paper for new results (the WHICH learner)- only a review of the field.

Instead, we focus on what is reproducible. All the studies you mention, with the partial exception of some of Zimmermann's 2009 work, are all conducted on closed data sets. This paper is all about what can be done with public domain data sets. The other work mentioned by this reviewer is usually analyzes a single product or a product family of a company, whereas our work spans a larger space of products. Hence, we spend more time on the papers discussing reproducible results (e.g. Lessmann; our own work) than others.