Software defect prediction with zero-inflated Poisson models[⋆]

Daniel Rodriguez[1][000−0002−2887−0185], Javier Dolado[2][0000−0002−3301−5650],
Javier Tuya[3][0000−0002−1091−934X], and Dietmar Pfahl[4][0000−0003−2400−501X]

[1] University of Alcala,
[2] Dept of Computer Science, 28805 Alcalá de Henares, Madrid, Spain
daniel.rodriguezg@uah.es
[3] University of the Basque Country, Facultad de Informática, 20018 Donostia, Spain
javier.dolado@ehu.es
[4] University of Oviedo, Campus of Viesques, 33204 Gijón, Asturias, Spain
tuya@uniovi.es
[5] University of Tartu, 50090 Tartu, Estonia
dietmar.pfahl@ut.ee

Abstract. In this work we apply several Poisson and zero-inflated models for software defect prediction. We apply different functions from several R packages such as *pscl*, *MASS*, *R2Jags* and the recent *glmmTMB*. We test the functions using the Equinox dataset. The results show that Zero-inflated models, fitted with either maximum likelihood estimation or with Bayesian approach, are slightly better than other models, using the AIC as selection criterion.

Keywords: Software defect prediction · Zero-inflated models · AIC.

1 Zero-inflated models for Software Defect Prediction

Most software defects datasets follow a distribution with a large number of non-defective modules, i.e., modules with zero bugs. Therefore, these datasets are highly unbalanced. Furthermore, when there are defects, modules tend to have a low number of defects. Most works in the literature have addressed this problem as an unbalanced binary supervised classification problem, i.e., modules are either defective or non-defective no matter the number of defects (e.g.[7]). In this work, we explore several Zero-inflated models (ZIP), including Bayesian estimation, to predict the number of defects in software defect datasets taking into account the previously stated imbalance problem. Although ZIP models have been explored in the past [4] here were compare different ZIP models with different criteria other than the p-value.

Definition of the Zero-inflated Poisson model. The zero-inflated model splits the governing equation for the dependent variable Y in two processes as shown in

[⋆] Supported by TIN2016-76956-C3-R QARE-BadgePeople-TESTEAMOS

Equation (1): the first one generates those extra zeros with probability π, and the second equation follows a Poisson distribution that generates the counts (some may also be zero) [8].

$$\begin{aligned}
\Pr(y_j = 0) &= \pi + (1-\pi)e^{-\lambda} \\
\Pr(y_j = h_i) &= (1-\pi)\frac{\lambda^{h_i}e^{-\lambda}}{h_i!}, \qquad h_i \geq 1
\end{aligned} \tag{1}$$

2 The Equinox dataset

In this work, we use the Equinox framework described by D'Ambros et al [3]. This dataset is part of the *Bug prediction dataset*[6] and corresponds to a Java Framework included the Eclipse project.

Fig. 1 shows the high number of modules with no defects in the Equinox dataset. The second histogram in Fig. 2 shows the distribution of the non-zero values.

Variable selection For the purpose of building the models, we performed several analyses using correlation, Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), giving as result that the best set of variables were: *wmc*: weighted methods per class, that is simple the method count for a class; *rfc*: response for a class is the set of methods that can potentially be executed in response to a message; *cbo*: coupling between objects, i.e., number of classes to which a class is coupled; *lcom*: lack of cohesion. For the excess count of zeros the variable, *number of lines of code* (*nloc*) was selected because it gives the user a clear understanding about what the source of zeros is. However, the variable *nloc* can be safely replaced by *wmc*, giving the latter slightly better results.

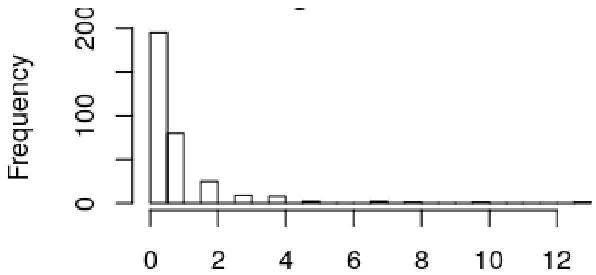

Fig. 1. The x-axis represents the number of bugs found in software modules. The y-axis represents the number of modules that contain x bugs.

[6] http://bug.inf.usi.ch/

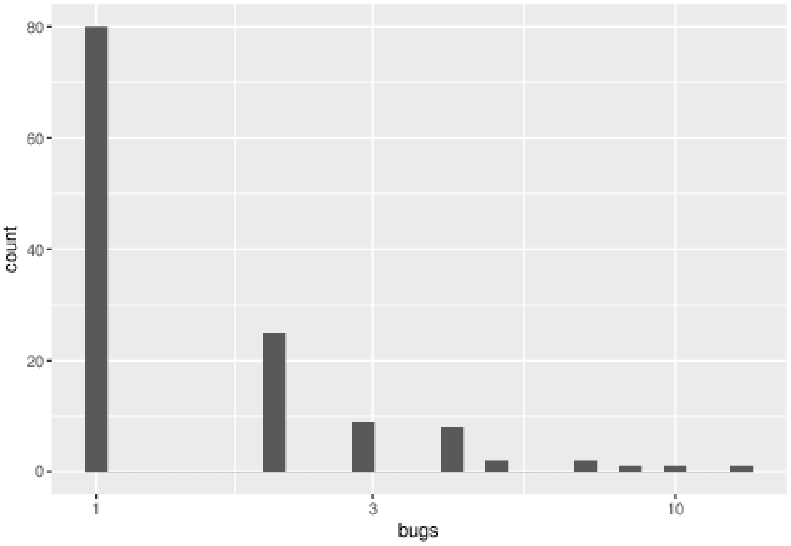

Fig. 2. After removing the zero bugs modules, the histogram still shows a big proportion of modules with few defects.

3 Simulation of ZIP models with R packages

There are several R packages that can be used to analyze zero-inflated models such as *pscl*, *R2Jags*, *MCMCglmm*, *CARBayes*, *R-INLA*, *mgcv* and others. The last addition is the package *glmmTMB* package [1, 2]. Some of them are based on the Maximum Likelihood Estimation and other packages use Bayesian simulation [6][5].

AIC, BIC, DIC. The AIC and the BIC, or Schwarz information criterion, are common measures for model selection. The Deviance Information Critierion (DIC) is used in Bayesian model selection and is a generalization of the AIC.

Table 1 shows that the AIC is low in most of the ZIP models: *pscl* and *glmmTMB* give the same result. The Bayes regression performed with *R2jags* gives similar coefficients (not shown here) as those of the *pscl* version. The column ”bugs predicted” has been computed for the models that had functions readily available. The ZIP model in *pscl* predicts the same number of bugs as the actual value of the Equinox dataset, which was 195.

4 Conclusions

Although ZIP models have presented good results across all R packages, more research is needed to generalize the validity of the ZIP approach. It makes sense to assume that many modules will not contain bugs because they have few lines of code or because they have been heavily tested in the past. Here we used only a single dataset and future work will include more datasets.

Table 1. Summary of the results obtained with different R packages.

Method	***AIC***	***BIC***	***R Package***	***# Bugs predicted***
Regression	**904.8354**	927.5198	MASS	97.76806
Poisson	**632.1547**	651.0584	pscl	188.7356
Poisson	**632.2**	651.1	glmmTMB	n.a
Poisson	**632.1547**	-	mgvc	-
Neg. binom.	**644.5**	-	MASS	195.8165
Neg. binom.	**628.6**	651.2	glmmTMB	n.a.
Neg. binom.	**628.5507**	-	mgvc	-
ZIP	**606.9155**	633.3807	pscl	195.7924
ZIP	**606.9**	633.4	glmmTMB	n.a.
ZIP	**602.9** wmc	629	glmmTMB	n.a.
ZIP	-	**DIC=622.5**	Bayes RJAGS	-
ZIP	**653.4149**	-	mgvc	-
ZIP	**647.9201** wmc	-	mgvc	-
ZINB	**607.5639**	637.8098	pscl	198.2048

References

1. Brooks, M.E., Kristensen, K., van Benthem, K.J., Magnusson, A., Berg, C.W., Nielsen, A., Skaug, H.J., Machler, M., Bolker, B.M.: glmmtmb balances speed and flexibility among packages for zero-inflated generalized linear mixed modeling. The R journal **9**(2), 378–400 (2017)
2. Brooks, M.E., Kristensen, K., van Benthem, K.J., Magnusson, A., Berg, C.W., Nielsen, A., Skaug, H.J., Maechler, M., Bolker, B.M.: Modeling zero-inflated count data with glmmtmb. BioRxiv p. 132753 (2017)
3. D'Ambros, M., Lanza, M., Robbes, R.: An extensive comparison of bug prediction approaches. In: Proceedings of MSR 2010 (7th IEEE Working Conference on Mining Software Repositories). pp. 31 – 41. IEEE CS Press (2010)
4. Khoshgoftaar, T.M., Gao, K., Szabo, R.M.: Comparing software fault predictions of pure and zero-inflated poisson regression models. International Journal of Systems Science **36**(11), 705–715 (2005). https://doi.org/10.1080/00207720500159995
5. Kruschke, J.: Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan. Academic Press (2014)
6. McElreath, R.: Statistical rethinking: A Bayesian course with examples in R and Stan. Chapman and Hall/CRC (2018)
7. Rodriguez, D., Herraiz, I., Harrison, R., Dolado, J., Riquelme, J.C.: Preliminary comparison of techniques for dealing with imbalance in software defect prediction. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering (EASE'14). pp. 43:1–43:10. EASE'14, ACM, New York, NY, USA (2014). https://doi.org/10.1145/2601248.2601294
8. Zuur, A.F., Ieno, E.N.: Beginner's guide to zero-inflated models with R. Highland Statistics Limited Newburgh (2016)