PR101

MATTIA PALMIOTTO – RICCARDO PETRELLA

Performance Analysis of Feature Selection Methods in Software Defect Prediction: A Search Method Approach

Description

SDP can help software engineers in identifying defect-prone modules in a software system. SDP is a classification task, where the class label is either “defective” or “not defective”. It is based on historical data and on software metrics and features. However, the number of features involved is often large, leading to a high-dimensionality problem. Furthermore, some of the features may be less relevant to determine the class than others, and/or redundant.

Feature Selection (FS) is a way to solve the high-dimensionality problem. An efficient FS method should identify and remove as much irrelevant and redundant information as possible, thereby leading to a good predictive performance with low computational cost.

Filter Feature Ranking (FFR) methods assess and rank features in datasets according to certain critical factors, then the set of the *k* top-ranked features is selected, where *k* is established by the modeler. Feature Subset Selection (FFS) methods aims to find a subset of features which have a good predictive capacity collectively.

In the paper, four FFR methods based on Ranker Search and fourteen FSS methods based on either Exhaustive and Heuristic Search are experimented, together with four classifiers: Naïve Bayes, Decision Tree, Logistic Regression and K-Nearest Neighbour. The four classifiers have been applied to five software defect datasets from the NASA both without and with each one of the FS methods. Successively, a prediction performance in terms of accuracy and stability has been assigned to all the prediction models and comparisons are made. In the project, only one FFR method, the Information Gain, and one FFS method, the Genetic Selection, have been considered.

Results and comparisons

By looking at the accuracies scores on the single datasets, we notice that the results obtained for CM1 and PC2 are very similar to the ones displayed in the paper. The situation is different for the other three datasets, especially KC1 and KC3 where the scores of the IG+DT method are particularly low. However, from a global point of view our results are not very different from the ones showed in the paper.

Some average accuracies are better than the ones reported in the paper, while others are worse. However, in most cases we have obtained similar values. The main exception is the IG+DT prediction model, whose average accuracy in the code is remarkably lower than the one mentioned in the paper. By looking at the performance of such a model on the individual datasets we can clearly notice that its accuracy in KC1 and KC3 is much smaller than the ones obtained through the experiments done by the paper's authors (65.63 vs 74.44 and 61.22 vs 80.41 respectively).

The data about the average variations are more discordant, for two reasons. First, all the respective values in the paper are positive (or at least equal to zero), which would imply that Feature Selection improves the predictive capability of the classifiers, whereas our code generates negative values as well. Furthermore, the size of the variations in the accuracies is quite restrained in the paper since it ranges between 0 and 2.87, while it ranges between -7.46 (IG+DT) and +9.57 (GS+NB) in our code.

By looking at this situation, it could seem that our experiment opposes the thesis that Feature Selection always improves the prediction. However, if we confront the two tables of the average training scores and the variations, we promptly see that in the case of the only two relevant negative variation, the corresponding training score is approximately 100%, which leads to suppose that such a variation is due to overfitting.

As regards to the stability, in our project it has a different meaning from the one in the original experiment. The paper's authors have assigned to each class of prediction methods combined with a specific classifier the respective value of the performance metrics (the Coefficient of Variation), which expresses the variations in accuracy between the methods belonging to the same class. However, in our project we have considered only one prediction model for each class of methods, meaning that such an approach would have been infeasible (if we would have done the same way, we would have trivially obtained the index of stability equal to 0). Consequently, we have considered the stability of each of the twelve prediction models on the five datasets instead. In our case, the Coefficient of Variation expresses the variations in accuracy of the same method when applied to the different datasets.

According to our results, the GS method is usually more stable than the IG method and its CVs does not differ too much from those of the four classifiers employed without Feature Selection.

Conclusions

Finally, while the paper's authors decide that FS methods likely have a positive effect on the prediction models, our experiment does not lead immediately to the same conclusion since sometimes our FS methods have had a negative influence on the

predictions. Nonetheless, as anticipated, further investigation is needed for those models which resulted in overfitting to decide if the ensuing variations after having limited the training score would be still negative or not.

Other issues which threaten the validity of the results obtained in the project:

1. External validity: the methods were tested just on five software defect datasets; therefore, any result cannot be generalized to other software defect datasets.
2. Internal validity: besides the overfitting, we have only employed 2 methods based on FS techniques and 2 search methods.
3. Construct validity: we have studied the stability in a different way with respect to the one adopted by the paper's authors. Furthermore, other performance metrics of accuracy and stability could have been employed to evaluate the performance of the prediction models.