# Software Quality Classification

Raymonjo Victorino
Professor: Dr. Kehan Gao
Department of Computer Science

# Introduction

The classification problem tackled is to predict fault prone (FP) software modules and not fault prone (NFP) software modules and find the best classification method to be used to create accurate predictions. This is important as client found faults in software is expensive and so predicting whether a module may be faulty is valuable in project management. A high priority for developers is to prevent any faults discovered by clients.

# Background: A Case Study

Experiments conducted in the case study used software metrics and defect data collected from a very large telecommunications software system. The software measurement datasets consist of 42 software metrics, including 24 product metrics, 14 process metrics, and four execution metrics. More details about these software metrics can be seen in the article [1]. The dependent variable is the class of the program module. A module with one or more faults is considered FP, and NFP otherwise. The software system consists of four successive releases labeled Release1 through Release4, where each release is characterized by the same number and type of software metrics but has a different number of instances (program modules). The experiments were performed on two groups of datasets, one with 42 metrics (a complete set of metrics) and the other with 28 metrics, ignoring any of the processing metrics.

Table I: Distribution of FP and NFP

| Data | # Modules | NFP # | NFP % | FP # | FP % |
|---|---|---|---|---|---|
| **Release 1** | 3649 | 3420 | 94% | 229 | 6% |
| **Release 2** | 3981 | 3792 | 95% | 189 | 5% |
| **Release 3** | 3541 | 3494 | 99% | 47 | 1% |
| **Release 4** | 3978 | 3886 | 98% | 92 | 2% |

.

Table I shows the characteristics of the groups of datasets used in this study. The output of the experiment is the area under the ROC (Receiver Operating Characteristics) curve (AUC) which reveals how a classification model can distinguish between classes such as NFP and FP.

In the experiments, seven learning algorithms were used as classifiers including Naïve Bayes, Multilayer Perceptron, k Nearest Neighbor, Support Vector Machine, Logistic Regression, C4.5, and Random Forest. Unless stated otherwise, the default parameter settings were used for the different learners as specified in WEKA [2]. Parameter settings are changed only when a significant improvement in performance is obtained. In Multilayer Perceptron (MLP) the parameters changed from default are hiddenlayers set to 3 and validationSetSize set to 10. The k Nearest Neighbor (Knn/IBK) had distanceWeighing set to “Weight by 1/distance,” kNN set to 5 and crossValidate set to true. The Support Vector Machine (SMO in WEKA) classifier had c set to 5.0 and buildLogisticModles set to true. Random Forest (RF100) had the number of trees set to 100.

# Prediction Results

Table II : Average Area Under the ROC Curve for Data28metrics

| Datasets | Naïve Bayes | Multilayer Perceptron | IBK | SMO | Logistic | J48 | Random Forest |
|---|---|---|---|---|---|---|---|
| Release 1 | 0.777 | 0.799 | 0.757 | 0.688 | 0.803 | 0.630 | 0.779 |
| Release 2 | 0.815 | 0.824 | 0.775 | 0.646 | 0.824 | 0.657 | 0.818 |
| Release 3 | 0.787 | 0.812 | 0.750 | 0.569 | 0.781 | 0.499 | 0.710 |
| Release 4 | 0.765 | 0.787 | 0.789 | 0.564 | 0.785 | 0.520 | 0.748 |

Table III : Average Area Under the ROC Curve for Data42metrics

| Datasets | Naïve Bayes | Multilayer Perceptron | IBK | SMO | Logistic | J48 | Random Forest |
|---|---|---|---|---|---|---|---|
| Release 1 | 0.792 | 0.813 | 0.778 | 0.740 | 0.815 | 0.587 | 0.802 |
| Release 2 | 0.822 | 0.839 | 0.803 | 0.683 | 0.841 | 0.607 | 0.831 |
| Release 3 | 0.802 | 0.808 | 0.789 | 0.544 | 0.732 | 0.499 | 0.752 |
| Release 4 | 0.785 | 0.805 | 0.814 | 0.627 | 0.784 | 0.539 | 0.810 |

The datasets were analyzed and classified using WEKA experimenter. The output excel file was then processed and the average area under the ROC curve was calculated per release in terms of the corresponding classifiers. Tables II and III show the average AUC (area under ROC curve) value for each classifier (learner) constructed over 10 runs of 5-fold cross validation (CV) on each dataset of the two groups. For all experiments, 10 runs of five-fold CV was employed. That is, for each run the data is randomly divided into five folds, one of which is used as the test data while the other four folds are used as training data. The training data is used to build the classification model and the resulting model is applied to the test fold. This cross-validation is repeated five times (the folds), with each fold used exactly once as the test data. The five results from the five folds then was averaged to produce a single estimation. In order to lower the variance of the CV result, the CV process was repeated with new random splits 10 times. The final estimation is the average results over the 10 runs of 5-fold CV. The AUC value ranges from 0 to 1. The larger the value of AUC, the better the performance of the model.

# Visual Comparisons

Figure 1:

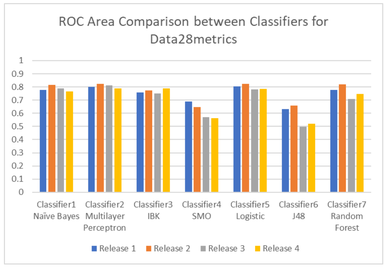



Figure 2:

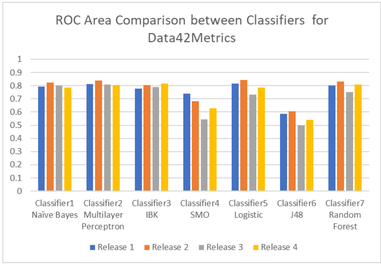



Figure 3:

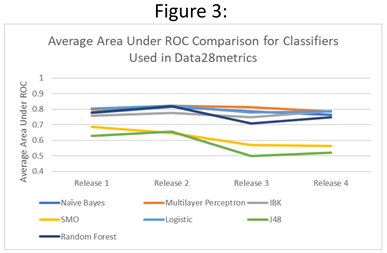
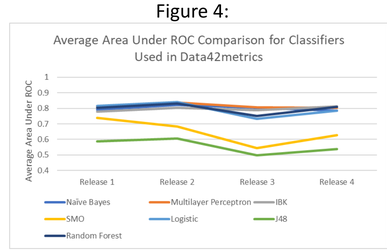



Figure 4:




Support Vector Machine (SMO) and C4.5 (J48) appear to have the least amount of area under ROC curve, while other classifiers (especially Multilayer Perceptron (MLP) and Logistic Regression (LR)) show better performance. There does not appear to be a difference in output between 28 and 42 metrics, although the 42 metric group illustrates a much clearer distinction between the top and bottom classifiers.

Figure 5: Ranked Classifiers for Data28metrics

```
>-<   >   < Resultset
 12  12   0 functions.MultilayerPerceptron
 11  11   0 functions.Logistic
  8   9   1 bayes.NaiveBayes
  5   7   2 trees.RandomForest
 -1   6   7 lazy.IBk
-15   0  15 functions.SMO
-20   0  20 trees.J48
```

Figure 6: Ranked Classifiers for Data42metrics

```
>-<   >   < Resultset
 10  10   0 functions.Logistic
 10  10   0 functions.MultilayerPerceptron
  9   9   0 trees.RandomForest
  8   8   0 bayes.NaiveBayes
  1   7   6 lazy.IBk
-17   1  18 functions.SMO
-21   0  21 trees.J48
```

Classifiers are ranked in terms of the ROC area produced using the Analyze function of WEKA Experimenter configured to be a Paired T-Tester.

- “>” = number of wins
- “<“ = number of loses
- “>-<“ = difference between wins and loses which determines the ranking

# Conclusions

Among all the classifiers used, Multiplayer Perceptron, Logistic Regression, Random Forest, IBK and Naïve Bayes appear to be better suited for the datasets. Support Vector Machine and C4.5 do not appear to be suited for the datasets. Even with 14 metrics removed from one of the grouped datasets, it appears that the predictions of the classifiers were not impacted severely. In other words, the outputs of both groups did not appear to be very different even with the removal of processing metrics.

**References:**

1. K. Gao et al. “Choosing software metrics for defect prediction: an investigation on feature selection techniques”, Software: Practice and Experience. Special Issue: Practical Aspects of Search-Based Software Engineering, vol. 41, no 5, 2011, pp. 579-606. Wiley. DOI:10.1002/spe.1043
2. Waikato Environment for Knowledge Analysis (WEKA), developed at the University of Waikato, New Zealand. It is free software licensed under the GNU General Public License, and the companion software to the book "Data Mining: Practical Machine Learning Tools and Techniques". https://www.cs.waikato.ac.nz/ml/weka/