Abstract

This thesis analyzes software defect prediction from data mining point of view with representations of software characteristics as static code features and defect predictors as defect logs from previous works. Since static code features provide limited information content, it is determined that there is a performance limit at data mining methods for constructing defect predictors. So instead of providing new features, data information content should be increased because gathering new features may cost too much or not applicable to all contexts. In order to increase data information content, substantial data were added to methods of data mining. To do this, some methods are proposed. First, assumptions of data mining methods were relaxed. Then, data from multiple companies are used. Finally, interactions between modules of software are modeled. Results of these methods showed that Naïve Bayes with relaxed assumptions is more preferable than standard Naïve Bayes since it dramatically increases defect prediction performance. For remote data method, it is obtained that remote data can also detect defects as well as local data with cost of high false alarm rates. Even so, a filtering method is proposed for removing this cost. Last method also shows that modeling interactions of modules dramatically decrease false alarm rates. Techniques used in this thesis present pathways for utilizing defect prediction by usage of remote data and increasing performance of defect predictor.

Defect Predictor by Remote Data

Companies may not have or have limited amount of local defect data; so using defect datasets from other companies might be a solution. In order to validate this assumption, an experiment was made with the design that is in table below. Experiments showed that cross company data has high probability of detection like within company data but it comes with the cost of high false alarms. So «k nearest neighbor algorithm» was used to remove irrelevancies in cross company data to make it similar to within company data and this filtering decreased false alarm rate significantly.

```
DATA = [FC1, KC1, KC2, CM1, KC3, MW1, MC2]
LEARNER = [Naïve Bayes]
C_FEATURES <- Find common features IN DATA
FOR EACH data in DATA
   data = Select C_FEATURES in data
END
REPEAT 20 TIMES
   FOR EACH data in DATA
      CC_TRAIN = DATA - data
      WC_TRAIN = random 80% of data
      TEST = data - WC_TRAIN
      CC_PREDICTOR = Train LEARNER with CC_TRAIN
      WC_PREDICTOR = Train LEARNER with WC_TRAIN
      [cc_pd, cc_pf, cc_bal] = CC_PREDICTOR on TEST
      [wc_pd, wc_pf, wc_bal] = WC_PREDICTOR on TEST
   END
END
```

Local Data Amount for Constructing Model

Public datasets of NASA were used in experiments in order to calculate how much data have to be inspected in order to detect defects. Experiments showed that there is no need to inspect thousands of instances in order to build local defect predictor.

```
Public data = [CM1, PC1, PC2, PC3, CM2, PC4, PC5, CM3, PC6, PC7]
```

Future Work

Standard machine learning algorithms lack the business knowledge which characterizes software projects. To add that business knowledge, it is proposed to use human-in-the-loop case based reasoning (CBR) tools. This may sound impractical but since there are reasonable amount of data to be inspected, a human based detection will perform much better than an algorithm does.

Contact: m.sedatgungor@gmail.com

Results

Results showed that relaxing assumptions of Naïve Bayes increases its performance of detecting defects. Weighting features and finding relevancies between them increased probability of detection rate.

Also if a company lack local repository of defect logs, results showed that remote data can be used for defect prediction at least after filtering that data. And it is showed that predictors learned from mere hundred examples perform as well as predictors learned from many more examples. That is, defect predictors tuned to the particulars of one company can be learned using very little data, collected in a very small amount of time.