EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

# Prediction of Cross Project Defects using Ensemble based Multinomial Classifier

Lipika Goel[1,2,*], Mayank Sharma[3], Sunil Kumar Khatri[3], D.Damodaran[4]

[1]Research Scholar, Amity University Noida, India

[2]Assistant Professor, AKG Engineering College GZB, India

[3]Amity Institute of Information Technology, Noida, India

[4]Centre for Reliability, Chennai, India

## Abstract

BACKGROUND: The availability of defect related data of different projects leads to cross project defect prediction an open issue. Many studies have focused on analyzing and improving the performance of Cross project defect prediction.

OBJECTIVE: The multinomial classification has not been much explored. This paper instanced on multiclass/multinomial classification of defect prediction of cross projects.

METHOD: The ensemble based statistical models – Gradient Boosting and Random Forest are used for classification. An empirical study is carried out to determine the performance of multinomial classification for cross project defect prediction. Depending on the number of defects, class level information is classified into one of three defined multiclass class 0, class 1, and class 2.

RESULTS & CONCLUSION: Major outcome of the paper concludes that multinomial/multiclass classification is applicable on cross project data and has comparable results to within project defect data.

**Keywords:** Defect Prediction, CPDP (Cross Project defect prediction), WPDP (Within Project defect prediction), Ensemble learning, Multinomial classification, Homogeneous metrics.

Received on 10 May 2019, accepted on 27 August 2019, published on 09 September 2019



doi: 10.4108/eai.13-7-2018.159974

*Corresponding author. Email:lipika.bose@gmail.com

## 1. Introduction

Identification of the defect prone classes before actual testing reduces the testing cost and efforts. It also leads to a more focused testing thereby enhancing the probability of fault free software [35]. Much research work has been carried out in within projects data. The availability of many different projects data leads to the motivation of cross project defect prediction. In the past decade much research has focused on binary classification of CPDP. The multinomial classification in CPDP is still an open area to be focused.

The advantage of treating this problem as multinomial classification than a regression is that regression does not respect the bounds of zero. Here the target variable that needs to be predicted is a positive quantity. However, regression based approaches in prediction of the defects can give non-integer or negative predictions. This can lead to invalid predictions. Therefore in this study substantiation of multinomial/multiclass classification has been done for cross projects.

The multinomial classification provides information on severity of defect prone classes. *Higher the number of defects*

*in a class higher is that class defect prone*. In this work we have classified the class level defect data into different groups depending on the total number of defects in each class. The three classes defined are as below. The reason for taking 5 as the threshold value is included in Section 4 of the paper.

- Class 0: Class with no bugs.
- Class 1: Class with bugs greater than 0 and less than 5.
- Class 2: Class with bugs greater than and equal to 5.

We have taken 15 open source object oriented projects. The classification is performed for cross projects and within project defect prediction. In our previous work [33] (G. Lipika et al., 2018) of binary classification for CPDP and WPDP, we inferred that Random forest and Gradient Boosting ensemble algorithms outperformed all the other algorithms ( LR, NVB, K-NN). Henceforth, in this paper we are using Gradient Boosting and Random Forest ensemble techniques for modeling. Ensemble learners combine multiple learners for a predictive model. They improve the predictive performance of the model.

The main research question directed is :

RQ: *Whether multinomial classification of CPDP is feasible and comparable to WPDP?*

To answer the above stated question we conducted the experiment for multinomial classification for the CPDP. Multinomial classification was also done for within project defect data. To determine the feasibility of multinomial classification for CPDP we have compared our results of multinomial classification of cross projects to within projects on the basis of AUC-ROC value, f measure, precision and recall. Cross validation is also performed to determine the training accuracy of the model.

This paper is divided into the following sections: Section 2 presents a brief of the literature survey. Section 3 summarizes the datasets, metrics, the performance measures and the models used. Section 4 states the proposed methodology. Section 5 tabulates the results. The threats to validity is stated in Section 6.Section 7 concludes the paper.

## 2. State Of Art

In this section we present a brief summary of the state of art in the field of cross project defect prediction. Since multinomial classification is not much explored therefore the below state of art gives a big picture of the work done in CPDP.

K-NN was used by Turhan et al in 2009 [3] to reduce the data distribution difference, then the Naïve Bayes classifierwas used for defect prediction. The homogeneous metrics were selected and the Naïve Bayes classifier was applied on 10 different cross projects. The results of the cross projects were compared with within project. The author demonstrated the minimum number of data samples required to build an effective predictor.

Logistic Regression model was used by T. Zimmermann et al in 2009 [4] for CPDP thereby giving a new dimension to it. He concluded that selection of dataset characteristic has a significant impact on the performance of the model in defect prediction. He performed a big study on feasibility of CPDP and inferred that decision trees can improve the performance measures.

S.J. Pan et al. in 2010 [5] focused on the categorizing and reviewing the progress of transfer learning in regression, classification and clustering problems. It inferred that transfer learning will be helpful in detection of defects with different training and testing dataset distribution. The authors gave a new dimension to defect prediction for cross projects using transfer learning approach.

Menzies et al. in 2011 [6] used a WHERE algorithm for creating a local model through clustering of the training data. Then the WHICH rule learning algorithm was used for classification. The novel approach to classification showed better results then some existing classifiers.

Ma et al. in 2012 [10] presented a new innovative approach named TNB-Transfer Naïve Bayes for CPDP. The results concluded that TNB showed better results than state of art in terms of area under curve. In inferred that even if few local training data is available , the knowledge from the different data distribution can also be used for defect prediction.

Zhimin He et al. in 2012 [9] investigated the defect prediction in cross project context focusing on selection of training dataset. The author concluded that in some cases training data from cross projects provide better results than from the same project. The authors also proposed a method for automatic selection of training data for projects with no historical data.

TCA- Transfer Component Analysis was implemented by Jaechang Nam et al. in 2013 [11] for CPDP. TCA made the feature distribution in source and target datasets similar. In addition TCA + was also proposed to make better defect predictions.

Herbold in 2013 [13] proposed NN Filtering and EM Clusterin as distance-based strategies for selecting the training data. Different classifiers like Logistic Regression, Naïve Bayes, Random Forest, ANN, Decision trees were used for prediction of the cross projects. Proper selection of the training data can lead to better defect prediction.

Peters et al. in 2013 [17] proposed a new filter called the Peters filter. The results of peter filter were compared with the Burak filter. Peter filter outperformed for CPDP. It analyses the structure of the other available projects and selects the final training dataset.

Dejaeger et al. in 2013 [15] contributed by studying fifteen different bayes networks and compared them to the other machine learning algorithms. The authors also investigated the Markov theory of feature selection.

G. Canfora et al. in 2013 [12] proposed a novel multi-objective model for cross project defect prediction. The multi-objective logistic model used genetic algorithm. This model allowed engineers to choose predictors by achieving a

compromise between effectiveness and lines of code to be tested.

Panichella et al. in 2014 [18] investigated the equivalence of the classifiers. He studied whether different classifiers identified the same defect prone classes. The authors proposed a combined defect predictor which outperformed with higher value of ROC. They concluded that combination of classifiers yields better results than a single classifier.

Peng He et al. in 2014 [19] addressed the problem of imbalanced feature set between the source and the target dataset. Considering this the authors proposed a framework based on distribution characteristic instance mapping. They validated the results on the publicly available datasets an concluded that the proposed model improved the performance of CPDP.

Peters et al. in 2015 [21] proposed an extension of MORPH and CLIFF. LACE2, a novel algorithm did not added all the data from the products to the defined shared cache. It decides which instance should be added. Kawata et al. in 2015 [22] proposed the use of DBSCAN clustering algorithm. In 2015 Y.Zhang et al.[24] used ensemble classifiers Maximum Voting and Average Voting for classification. Amasaki et al. in 2015 [25] proposed a combination of attribute selection and relevance filtering. These were performed on the log transformed data. Nam and Kim in 2015 [20] proposed CLAMI, a novel fully automated unsupervised approach. CLAMI included clustering and labeling of the training data for prediction.

The solution for heterogeneous cross project defect prediction by was provided X. Y. Jing and Nam in 2015 [27] . Ryu, D et al. in 2016 [28] investigated the applicability of class imbalance learning in CPDP scenario. They proposed a novel method as VCB-SVM.The proposed model improved the predictive performance by sampling and class imbalance learning.

Duksan Ryuet et al. in the year 2016 [30] provided a solution to the class imbalance problem by proposing a model using SVM with cognitive boosting.The combined approach of Genetic and Ensemble learning gave good results. Zhang et al. in the year 2016 [29] used unsupervised classification for heterogeneous Cross Project Defect P.rediction.

Herbold Steffen et al. in 2017 [32] benchmarked the CPDP approaches. He concluded that CPDP has still not reached to a level where it can be used for the application in practice.

## 3. Prerequisite Understanding

### 3.1. Metrics used and description of Datasets

We have taken the well-defined datasets from http://openscience.us. The original data is collected by Marian Jureczko, Institute of Computer Engineering, Wroclaw University of Technology. From each dataset the information at the class level is considered. These datasets generally have multiple versions. Table 1 describes the datasets used.

Table 1. Dataset Details

| Project | Language | Total no. of classes | Defect prone classes |
|---|---|---|---|
| Ant -1.5 | Java | 291 | 32 |
| Ant -1.6 | Java | 351 | 94 |
| Ant- 1.7 | Java | 745 | 168 |
| Ivy-1.1 | Java | 111 | 63 |
| Ivy-1.4 | Java | 241 | 17 |
| Ivy-2.0 | Java | 352 | 40 |
| Camel- 1.2 | Java | 608 | 216 |
| Camel- 1.4 | Java | 872 | 145 |
| Camel -1.6 | Java | 965 | 189 |
| Prop1 | Java | 18471 | 2738 |
| Prop2 | Java | 23041 | 2431 |
| Prop3 | Java | 10274 | 1180 |
| Jedit- 4.0 | Java | 306 | 75 |
| Jedit- 4.1 | Java | 312 | 79 |
| Jedit- 4.2 | Java | 367 | 47 |

In this work we have focused on homogeneous CPDP for multinomial classification. The software metrics are important in defect prediction process [36]. The CK metrics (Chidamber and Kemerer, 1994) are selected and are used for software defect prediction. CBO [34,37], RFC and LCOM metrics of the CK suite were associated with the fault proneness of classes. Higher the value of these metrics higher is fault proneness of the class. WMC and RFC metrics of the CK suite to be highly related with number of defects in a class. The higher value of DIT [38] and lower value of NOC metrics of the CK suite to be also related to the fault proneness of the class. These studies infer the high association of the CK metric suite with the number of defects in a class. The table 2 gives the description of the CK metrics.

The CK Metrics used in the analysis are as follows (Verma and Vyas, 2013) [14]:

- DIT
- WMC
- CBO

- NOC
- LCOM
- RFC

Table 2. Software Ck Metrics

| Metrics | Description |
|---|---|
| WMC | It describes the total time and effort required to develop and maintain the class. The level and complexity of WMC should be low. |
| DIT | It gives the maximum distance from the root to the terminal nodes. The level and complexity of DIT should be low. |
| CBO | It tells the number of classes to which a class is coupled. It describes the degree of interdependence between two classes. The level and complexity of CBO should be low to easily test and maintain the .software. |
| NOC | It states the number of subclasses of a class. T The level and complexity of NOC should be low to easily test and maintain the software. |
| LCOM | It describes the degree of relatedness between the methods of a class. The level and complexity of LCOM should be low. |
| RFC | It states the number of methods of the class or any other methods called. The level and complexity of RFC should be low to easily test and maintain the software. |

## 3.2 Ensemble Learning Models

An ensemble contains a combination of base/ weak learners to improve the performance of the model. The prediction accuracy of an ensemble is much higher than the base learners. Base learning algorithms can be neural network, decision tree or any other machine learning algorithm. The majority voting for classification and weighted averaging for the regression are the common strategies of combining base learners. The popular and effective ensemble methods are Bagging, Boosting and Stacking. The figure1. gives a diagrammatic representation of ensemble approach. The figure states that N classifiers are used on a set of input features for prediction. The results from the N classifiers are combined for the final outcome.

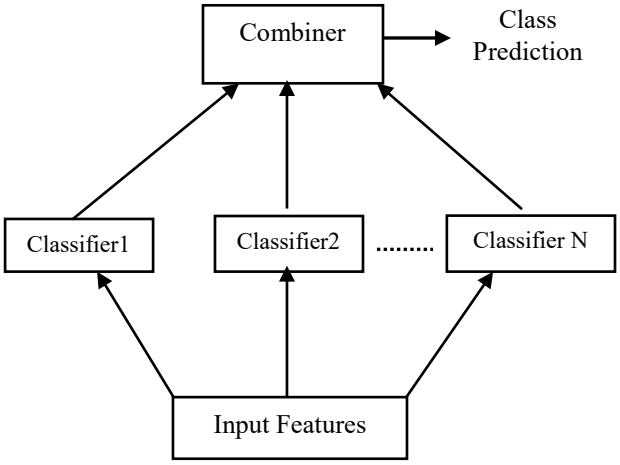


**Figure 1.** Ensemble Learning Approach

The ensemble learning models used in this experiment are Random Forest and Gradient Boosting. Random forest model is a bagging technique whereas Gradient Boosting is a boosting approach.

1) *Random Forest:* Random Forest is a solution to most of the data science problems. It is a flexible machine learning model which is competent to perform both classification and regression. It handles missing values, dimensionality reduction methods, outliers and thereby having better results. It is an ensemble of weak models (decision trees). To classify an object each tree votes, the forest chooses the classification which has the highest number of votes. For regression it computes the average of the output produced by each tree. The major advantage of Random forest is avoiding the problem of overfitting.

Given a training set: - $X = x_1, x_2, ....x_n$ with responses $y_1, y_2, .....y_n$. bagging it B times- selecting random samples. For $b = 1,2,3,....N$ We will train a regression or classification tree on $X_b, Y_b$. After training the prediction of the sample x can be done either by averaging the prediction f all the regression trees or taking the majority votes in the case f classification trees.

$$F = 1/N \sum f_b(x') \quad \text{for } b = 1,2,3,....N$$

The standard deviation of the predictions from different regression trees gives the estimate of uncertainty of the prediction.

$$\sigma = \sum (f_b(x') - F)^2 / (N-1)$$

2) *Gradient Boosting:* It is an ensemble learning technique (Dai, W. et al., 2007) . It has sequential predictors.

The subsequent predictors improve its learning by faults of the existing predictors. It involves three elements:

- A loss function to be optimized.
- A weak learner to make predictions.
- Model to add weak learners.

Gradient boosting uses decision trees as weak learners. A tree is parameterized and its parameters are modified and then it is added to the model to minimize the loss.

For j=1…..n

$F_0(x) = \arg\min \sum L(y_i, \gamma)$

$\gamma_m = \text{argmin} \sum L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$

$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$.

Where $(x_i, y_i)$ is the training set L(y,F(x)) is a differential loss function, M is the no. of iterations, $h_m(x)$ is the base learner, $\gamma_m$ is the multiplier.

## 3.3. Evaluation Measures

The Table 3 summarizes the performance measures which are mostly used for evaluation of classification models.

Table 3. Performance Measures

| Performance Measure | Description |
|---|---|
| Precision | It is the proportion of cases that are correctly identified to belong to class A among all the cases which classifier claims to belong to A. |
| Recall | It is the proportion of cases that are correctly identified to belong to class A among all the cases that truly to belong to A. |
| F-measure | (2*Recall * Precision)/(Recall + Precision) |
| Accuracy | It is the ratio of all correctly identified and classified cases. |
| AUC | It is used in classification analysis to determine the performance of the model ie. Which model predicts the classes best |
| MAP | It is the average value of the maximum precision at different recall values. |

# 4. Proposed Methodology

## 4.1 Data Acquisition and Understanding

On collection of the datasets the identification of target variables and the input features is done. The total number of defects in a class is taken as the target variable. We have taken into assumption that higher the number of defects in a class higher is the class defect prone and higher is the severity. When two or different projects have the same set of metrics then such metrics are called the homogeneous metrics. In our analysis we have taken the different projects with the homogeneous metrics thereby performing homogeneous CPDP.

## 4.2 Data Preprocessing and Preparation

The data preprocessing steps include data encoding, data normalization and feature selection. In the data encoding, label encoder is used to convert the categorical value to the numeric value. Z-score normalization is performed to normalize the data on a common scale.

We have conducted the experiment to classify our data of cross and within project into multi-classes. First of all. we made this problem as a 10 class multinomial classification Problem. Class with 0 defects has lowest severity, class 1, class 2 and so on and the class with 10 defects has highest severity (the maximum defects in a class were not more than 10). On evaluation of the predicted values and actual values, we inferred that most of the values are belonging to class 0 or class 5 among the 10 classes defined above. Hence, in order to evaluate the severity into 3 buckets as low, medium and high we converted the 10 class problem to 3 class problem (multinomial) with 5 being the threshold value. As stated before we defined three classes as follows:

Class 0: Class with no bugs.

Class 1: Class with bugs greater than 0 and less than 5.

Class 2: Class with bugs greater than and equal to 5.

Cross validation method is also performed to evaluate the training accuracy of the model.

## 4.3 Modeling

The modeling includes a description on model fitting and its evaluation.

### 4.3.1 Model Fitting:

Gradient Boosting and Random forest are used for multinomial classification. Hyperparameter tuning of the classifiers are done. Hyperparameter tuning is the settings done to the classifiers to optimize its performance. These are not directly learned from the data and needs to be predefined. In the case of a random forest, Hyperparameter includes the number of decision trees used in the forest, the number of features considered when splitting a node by each tree, the maximum depth of a tree, the minimum number of samples required at leaf node. We used Grid Search to set the range for

each parameter. The range of values for each parameter in Random Forest in the experiment is:
'n_estimators': [10,15,20]
'max_depth': [5,10,12]
min_samples_leaf': [2,3]
min_samples_split': [2,3]
In the case of Gradient Boosting, hyperparameters include n-estimators and the learning rate. The range of values for each parameter in Gradient Boosting in the experiment is:
'n_estimators': [30,40,50]
'learning_rate': [0.2, 0.1]

### 4.3.2 Model Evaluation:

For evaluation of the models we have used three performance measures i.e. F-Measure, AUC-ROC and Mean Average Precision (MAP). F-Measure is the harmonic mean of precision and recall and is widely used in evaluation of classification models. Confusion Matrix is used to evaluate recall, precision and f-measures is calculated.
The AUC-ROC values facilitate the comparison of the models. An AUC value of 1 represents a perfect classifier whereas for random classifiers a value of 0.5 is expected. The MAP is used to evaluate the performance of the classifiers. MAP value can be calculated by computing the average precision of each class and then the average over the class. Higher the value of MAP better it is. Algorithm 1 presents the pseudo code of the experiment conducted. Figure2 gives defect prediction process of multinomial classification. In this, data is collected and preprocessed. As discussed in section 1 of the paper, the dataset is then divided into three classes of 0, 1 and 2. After the selection of CK metrics the training dataset is provided to the model for classification. The ensemble classifiers are used for prediction of defects under the category of three defined classes.

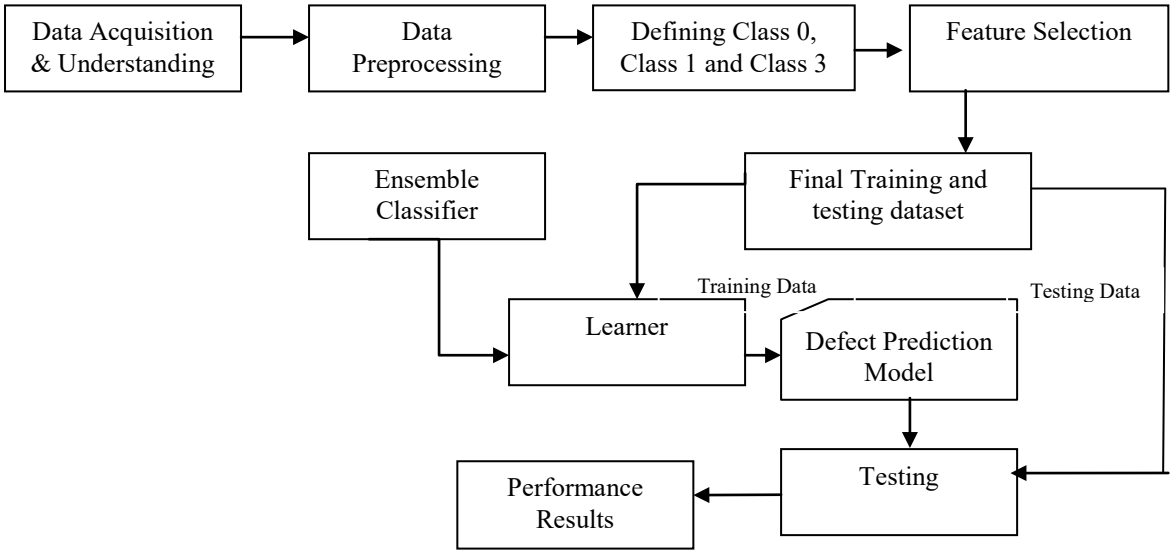


**Figure 2.** Defect Prediction Process Of Multinomial Classification

ALGORITHM 1. PSEUDO CODE OF THE EXPERIMENT

```
Data= Different data from different domains
CP_data = {Ant 1.5 ,Ant 1.6,Ant 1.7, Camel 1.2, Camel 1.4, Camel 1.6, Ivy1.1, Ivy1.4, Jedit 4.2,
Prop1,Jedit 4.0,  Ivy2.0, , Jedit 4.1, Prop2, Prop3 }
#Preprocessing data
CP_data= Unique(CP_data)
defining the target variable as 10-class multi-classification
for categorical_features in input_features do
{perform label encoding}
end for
#Perform k-fold cross validation.
#Selecting homogneous CK Metrics from all the datasets
for data in training_data do
   for each_project in data
      {input_features = select ck metrics from each_project}
end for
# Defining the training and the testing data for Cross Project Defect Prediction
for project belongs to CP_test
CP_train =  CP_data – data
CP_test= data
# Defining the training and the testing data for Within Project Defect Prediction
for project belongs to WP_test
splitting training and testing data with 60:40 ratio
train = {selecting 60% of data us }
test = training_data – train
#Modeling
applying machine learning algorithms
model1 = random_forest_model_training(train)
model2 = gradient_boosting_model_training(train)
Applying Grid Search technique to tune the hyper-parameters for both models
Predicting the results on test data
Conversion of 10-class to 3-class and re-train the models again with above steps
Evaluating the models using metrics
{auc,precision,recall, f1score}
```

## 5. Results & Discussion

In this section we present results of the experiment conducted. Table 4,5,6 ,7 and 8  tabulates the results.

Table 4 gives the value of precision,recall & f-measure for CPDP using random forest & gradient boosting. Table 5 tabulates the values of precision,recall & f-measure for wpdp using random forest & gradient boosting. Table 6 gives the average value of f-score, precision & recall using random forest & gradient boosting for CPDP & WPDP whereas the Table 7 presents the Auc-Roc values using random forest and gradient boodting  for CPDP and WPDP. The  accuracy of the predictive performance of the model is specified in Table 8.

The performance evaluation measures used are f1-score, AUC-ROC value , precision and recall. We have compared the results of multinomial classifiation for CPDP with WPDP. The observations are as below:

**RQ: Whether multinomial classification of CPDP is feasible  and comparable to WPDP?**

From table 7 the average value of AUC-ROC for Cross Project Defect Prediction using Gradient Boosting and Random Forest are 0.610 and 0.605 respectively. These values are comparable to the AUC-ROC values for WPDP as 0.630 and 0.621 using Random Forest and Gradient Boosting respectively. This value is only 3.17% and 2.57% higher than the values observed for CPDP. Since AUC-ROC value depicts the performance of the classifier and facilitates the comparison of the models therefore from the above observations we can infer that multinomial classification for CPDP is also feasible and comparable to WPDP with respect to AUC-ROC values.

Similarly from table 8 the average value of accuracy for Cross Project Defect Prediction using Gradient

Boosting and Random Forest are 0.73 and 0.72 respectively. These values are comparable to the accuracy values for WPDP as 0.79 and 0.81 using Random Forest and Gradient Boosting respectively. Since accuracy is also one of the performance measure of the classifier therefore from the above observations we can infer that multinomial classification for CPDP is also feasible and comparable to WPDP.

From table 6 we observe that when using Random Forest as classification model the average values of F-Score in CPDP for class 0, class 1 and class 2 are 0.9, 0.268 and 0.302 respectively, whereas when we look at the values for WPDP they are 0.838, 0.274 and 0.291 for class 0, class 1 and class 2 respectively. In the case of CPDP, this value was higher by 6.88% for class 0, 2.18% lower class 1 and 3.64% higher for class 2 when compared with WPDP.

On using Gradient Boosting as the ensemble model for classification the average values of F-Score in CPDP for class 0, class 1 and class 2 are 0.92, 0.247 and 0.334 respectively, whereas when we look at the values for WPDP they are 0.846, 0.258 and 0.322 for class 0, class 1 and class 2 respectively. In the case of CPDP, this value was higher by 8.043% for class 0, 4.26% lower class 1 and 3.59% higher for class 2 when compared with WPDP.

From the above observations we can infer that multinomial classification for CPDP is also feasible and comparable to WPDP with respect to F-measure.

*The multinomial classification of CPDP is feasible and can be compared to WPDP.*

Table 4. Values Of Precision,Recall & F-Measure For Cpdp Using Random Forest & Gradient Boosting

| CLASSIFIER | | RANDOM FOREST | | | GRADIENT BOOSTING | | |
|---|---|---|---|---|---|---|---|
| Testing Data | Class | Precision | Recall | .F1 score | Precision | Recall | F1 score |
| Ant 1.5 | 0 | 0.85 | 0.9 | 0.87 | 0.87 | 0.88 | 0.87 |
| | 1 | 0.21 | 0.17 | 0.19 | 0.24 | 0.27 | 0.25 |
| | 2 | 0.5 | 0.06 | 0.11 | 0.42 | 0.09 | 0.14 |
| Ant 1.6 | 0 | 0.96 | 0.85 | 0.9 | 0.89 | 0.78 | 0.83 |
| | 1 | 0.14 | 0.31 | 0.2 | 0.27 | 0.01 | 0.02 |
| | 2 | 0.21 | 0.05 | 0.12 | 0.22 | 0.51 | 0.3 |
| Ant 1.7 | 0 | 0.87 | 0.82 | 0.85 | 0.85 | 0.82 | 0.83 |
| | 1 | 0.44 | 0.47 | 0.46 | 0.43 | 0.46 | 0.44 |
| | 2 | 0.38 | 0.73 | 0.5 | 0.35 | 0.73 | 0.47 |
| Camel 1.2 | 0 | 0.89 | 0.78 | 0.94 | 0.9 | 0.81 | 0.85 |
| | 1 | 0.27 | 0.01 | 0.02 | 0.26 | 0.05 | 0.08 |
| | 2 | 0.22 | 0.51 | 0.3 | 0.22 | 0.52 | 0.3 |
| | 0 | 0.86 | 0.8 | 0.83 | 0.86 | 0.86 | 0.86 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Camel 1.4 | 1 | 0.40 | 0.46 | 0.43 | 0.24 | 0.26 | 0.25 |
| | 2 | 0.27 | 0.5 | 0.35 | 0.25 | 0.06 | 0.1 |
| Camel 1.6 | 0 | 0.93 | 0.9 | 0.91 | 0.93 | 0.92 | 0.9 |
| | 1 | 0.66 | 0.69 | 0.68 | 0.66 | 0.68 | 0.67 |
| | 2 | 0.36 | 0.75 | 0.49 | 0.36 | 0.73 | 0.48 |
| Ivy 1.1 | 0 | 0.77 | 0.99 | 0.87 | 0.82 | 0.93 | 0.87 |
| | 1 | 0.2 | 0.03 | 0.05 | 0.51 | 0.29 | 0.37 |
| | 2 | 1 | 0.09 | 0.17 | 0.01 | 0.04 | 0.02 |
| Ivy 1.4 | 0 | 0.96 | 0.95 | 0.95 | 0.94 | 0.97 | 0.95 |
| | 1 | 0.19 | 0.21 | 0.2 | 0.17 | 0.06 | 0.09 |
| | 2 | 0.04 | 0.06 | 0.05 | 0.23 | 0.07 | 0.12 |
| Ivy 2.0 | 0 | 0.84 | 0.91 | 0.87 | 0.83 | 0.94 | 0.89 |
| | 1 | 0.49 | 0.36 | 0.42 | 0.55 | 0.26 | 0.35 |
| | 2 | 0.01 | 0.04 | 0.05 | 0.38 | 0.5 | 0.43 |
| Jedit 4.0 | 0 | 0.82 | 0.93 | 0.87 | 0.83 | 0.93 | 0.87 |
| | 1 | 0.51 | 0.29 | 0.37 | 0.52 | 0.29 | 0.37 |
| | 2 | 0.01 | 0.04 | 0.02 | 0.01 | 0.05 | 0.03 |
| Jedit4.1 | 0 | 0.95 | 0.98 | 0.97 | 0.79 | 0.99 | 0.88 |
| | 1 | 0.27 | 0.11 | 0.15 | 0.6 | 0.09 | 0.15 |
| | 2 | 0.07 | 0.25 | 0.11 | 0.6 | 0.55 | 0.57 |
| Jedit4.2 | 0 | 0.79 | 0.99 | 0.88 | 0.79 | 0.97 | 0.87 |
| | 1 | 0.6 | 0.09 | 0.15 | 0.62 | 0.09 | 0.15 |
| | 2 | 0.6 | 0.55 | 0.57 | 0.61 | 0.58 | 0.58 |
| Prop1 | 0 | 0.89 | 1 | 0.94 | 0.76 | 1 | 0.87 |
| | 1 | 0.27 | 0.01 | 0.02 | 1 | 0.02 | 0.03 |
| | 2 | 0.22 | 0.51 | 0.6 | 1 | 0.18 | 0.31 |
| | 0 | 0.89 | 1 | 0.94 | 0.86 | 0.96 | 0.9 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Prop2 | 1 | 0.42 | 0.01 | 0.01 | 0.66 | 0.39 | 0.49 |
| | 2 | 0.2 | 0.5 | 0.61 | 0.83 | 0.42 | 0.56 |
| Prop3 | 0 | 0.93 | 0.9 | 0.91 | 0.89 | 1 | 0.94 |
| | 1 | 0.66 | 0.69 | 0.68 | 0.42 | 0.01 | 0.01 |
| | 2 | 0.36 | 0.75 | 0.49 | 0.2 | 0.5 | 0.61 |

Table 5. Values Of Precision,Recall & F-Measure For Wpdp Using Random Forest & Gradient Boosting

| **CLASSIFIER** | | **RANDOM FOREST** | | | **GRADIENT BOOSTING** | | |
|---|---|---|---|---|---|---|---|
| **Testing Data** | **Class** | **Precision** | **Recall** | **F1 score** | **Precision** | **Recall** | **F1 score** |
| Ant 1.6 | 0 | 0.85 | 0.86 | 0.85 | 0.85 | 0.86 | 0.86 |
| | 1 | 0.15 | 0.34 | 0.22 | 0.29 | 0.01 | 0.03 |
| | 2 | 0.20 | 0.06 | 0.14 | 0.23 | 0.54 | 0.32 |
| Camel 1.4 | 0 | 0.84 | 0.82 | 0.76 | 0.86 | 0.86 | 0.77 |
| | 1 | 0.41 | 0.36 | 0.38 | 0.67 | 0.70 | 0.68 |
| | 2 | 0.19 | 0.20 | 0.2 | 0.38 | 0.78 | 0.51 |
| Ivy2.0 | 0 | 0.84 | 0.91 | 0.87 | 0.82 | 0.93 | 0.87 |
| | 1 | 0.49 | 0.36 | 0.42 | 0.51 | 0.29 | 0.37 |
| | 2 | 0.01 | 0.04 | 0.05 | 0.01 | 0.04 | 0.02 |
| Jedit 4.0 | 0 | 0.74 | 0.82 | 0.76 | 0.77 | 0.79 | 0.76 |
| | 1 | 0.29 | 0.12 | 0.17 | 0.8 | 0.09 | 0.16 |
| | 2 | 0.60 | 0.58 | 0.59 | 0.62 | 0.60 | 0.61 |
| Prop2 | 0 | 0.96 | 0.95 | 0.95 | 0.95 | 0.98 | 0.97 |
| | 1 | 0.67 | 0.69 | 0.68 | 0.42 | 0.03 | 0.05 |
| | .2 | 0.19 | 0.21 | 0.2 | 0.27 | 0.11 | 0.15 |

Table 6. Average Value Of F-Score, Precision & Recall Using Random Forest & Gradient Boosting For Cpdp & Wpdp

| | | | Precision | Recall | F-Score | | | | Precision | Recall | F-Score |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CPDP | Random Forest | Class 0 | 0.88 | 0.847 | 0.9 | WPDP | Random Forest | Class 0 | 0.846 | 0.872 | 0.838 |
| | | Class 1 | 0.381 | 0.26 | 0.268 | | | Class 1 | 0.456 | 0.442 | 0.274 |
| | | Class 2 | 0.296 | 0.359 | 0.302 | | | Class 2 | 0.168 | 0.278 | 0.206 |
| CPDP | Gradient Boosting | Class 0 | 0.854 | 0.917 | 0.92 | WPDP | Gradient Boosting | Class 0 | 0.85 | 0.884 | 0.846 |
| | | Class 1 | 0.476 | 0.215 | 0.247 | | | Class 1 | 0.538 | 0.224 | 0.258 |
| | | Class 2 | 0.379 | 0.368 | 0.334 | | | Class 2 | 0.302 | 0.414 | 0.322 |

Table 7. Auc-Roc Values Using Random Forest And Gradient Boodting For Cpdp & Wpdp

| | CPDP | | WPDP | |
|---|---|---|---|---|
| **Project** | **Random Forest** | **Gradient Boosting** | **Random Forest** | **Gradient Boosting** |
| Ant 1.5 | 0.612547 | 0.687451 | 0.698745 | 0.712456 |
| Ant 1.6 | 0.63109 | 0.661218 | 0.651242 | 0.671112 |
| Ant 1.7 | 0.741302 | 0.744314 | 0.73312 | 0.742314 |
| Camel 1.2 | 0.684474 | 0.678049 | 0.694464 | 0.678149 |
| Camel 1.4 | 0.551103 | 0.593971 | 0.591143 | 0.601271 |
| Camel 1.6 | 0.573669 | 0.577769 | 0.601259 | 0.612549 |
| Ivy1.1 | 0.51805 | 0.550849 | 0.52805 | 0.549849 |
| Ivy1.4 | 0.529259 | 0.518523 | 0.535259 | 0.525423 |
| Ivy2.0 | 0.702244 | 0.56859 | 0.713244 | 0.587859 |
| Jedit 4.0 | 0.636752 | 0.626342 | 0.66752 | 0.626342 |
| Jedit 4.1 | 0.676672 | 0.655756 | 0.698632 | 0.689556 |
| Jedit 4.2 | 0.664231 | 0.655756 | 0.694231 | 0.667756 |
| Prop1 | 0.506667 | 0.509004 | 0.523667 | 0.515004 |
| Prop2 | 0.625195 | 0.626286 | 0.671545 | 0.665246 |
| Prop3 | 0.504031 | 0.504056 | 0.524331 | 0.574166 |
| **AVERAGE** | **0.610339** | **0.6050345** | **0.6305505** | **0.621899714** |

Table 8. Acuracy Values Using Random Forest And Gradient Boodting For Cpdp & Wpdp

| | CPDP | | WPDP | |
|---|---|---|---|---|
| **Project** | **Random Forest** | **Gradient Boosting** | **Random Forest** | **Gradient Boosting** |
| Ant 1.5 | 0.63 | 0.65 | 0.78 | 0.82 |
| Ant 1.6 | 0.68 | 0.71 | 0.82 | 0.81 |
| Ant 1.7 | 0.71 | 0.7 | 0.86 | 0.85 |
| Camel 1.2 | 0.62 | 0.63 | .0.81 | 0.83 |
| Camel 1.4 | 0.75 | 0.78 | 0.76 | 0.74 |
| Camel 1.6 | 0.73 | 0.75 | 0.76 | 0.78 |
| Ivy1.1 | 0.78 | 0.82 | 0.82 | 0.82 |
| Ivy1.4 | 0.71 | 0.71 | 0.76 | 0.79 |
| Ivy2.0 | 0.69 | 0.67 | 0.73 | 0.75 |
| Jedit 4.0 | 0.78 | 0.76 | 0.78 | 0.78 |
| Jedit 4.1 | 0.77 | 0.77 | 0.82 | 0.85 |
| Jedit 4.2 | 0.81 | 0.82 | 0.83 | 0.84 |
| Prop1 | 0.81 | 0.83 | 0.86 | 0.89 |
| Ant 1.5 | 0.63 | 0.65 | 0.78 | 0.82 |
| Ant 1.6 | 0.68 | 0.71 | 0.82 | 0.81 |
| **AVERAGE** | **0.72** | **0.73** | **0.79** | **0.81** |

## 6. Threats To Validity

Threats to internal validity states the errors in the experiments. We have checked our datasets and experiments but still there can be errors that we did not notice. An issue that can affect the internal validity is the use of classifier for modeling. There are many classification algorithms available. Any research work includes only a small subset of classification algorithm. In our work, we have focused on only on two algorithms of Ensemble approach (Random Forest and Gradient Boosting).Besides this, all the datasets used in our experiments are of Jureczko, there can be some quality issues among the datasets.

Threats to external validity are related to the generalization of the results. All the datasets used in our experiment are from single source and are open source projects. This threat can be reduced by considering more datasets from different sources for the experiment.

Threats to reliability validity states the possibility of replicating this work. All the datasets used in our experiment are publicly available and the pseudocode is available in this paper. This work can easily be replicated.

The threat to conclusion validity is that we have compared our outcomes of the experiment on the basis of F-score and AUC of within project defect prediction. Since not much of work on multinomial classification for CPDP is present in state of art therefore results obtained are compared with WPDP to prove the feasibility of multinomial classification for CPDP.

## 7. Conclusion & Future Scope

The software reliability is directly proportional to probability of error free software. In this paper we analyzed the performance in multinomial classification of cross and within project defect prediction. The datasets were collected Jureczko. Cross validation was done to estimate the predictive performance of the models. CK based object oriented metrics was effective for CPDP. We first labeled our class information in three different levels depending on the number of defects in each class. The experiment was conducted by training the model on different cross project defect prediction. Ensemble learning approaches were used for classification. The values of F-score, MAP and AUC were used for empirical analysis of the model.

The results indicate that multinomial classification on CPDP is feasible and comparable to WPDP. There are many verticals on which the work can be extended. Multinomial classification of Heterogeneous CPDP can be focused. Proper training dataset selection and class imbalance to optimize the performance of defect prediction are still open issues in CPDP.

## References


[1] Hourani, Hussam, Hiba Wasmi, and Thamer Alrawashdeh. "A Code Complexity Model of Object Oriented Programming (OOP)." 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT). IEEE, 2019.

[2] S. R. Chidamber and C. F. Kemerer. A metrics suite for object oriented design. IEEE Trans. Softw. Eng., 20:476–493, June 1994.

[3] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano. On the relative value of cross-company and within-company data for defect prediction. Empirical Software Engineering, 14(5):540-578, 2009.

[4] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy. Cross-project defect prediction. In Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE), pages 91-100, 2009.

[5] S. J. Pan and Q. Yang. A survey on transfer learning. IEEE Trans. on Knowl. and Data Eng., 22:1345–1359, October 2010.

[6] T. Menzies, A. Butcher, A. Marcus, T. Zimmermann, and Cok, "Local vs. global models for effort estimation and defect prediction," in Proc. 26th IEEE/ACM Int. Conf. on Automated Softw. Eng. (ASE). IEEE Computer Society, 2011.

[7] T. Menzies, A. Butcher, D. Cok, A. Marcus, L. Layman, F. Shull, B. Turhan, and T. Zimmermann, "Local versus global lessons for defect prediction and effort estimation," IEEE

[8] M. D'Ambros, M. Lanza, and R. Robbes. Evaluating defect prediction approaches: A benchmark and an extensive comparison. Empirical Software Engg., 17(4-5):531–577, Aug. 2012.

[9] Z. He, F. Shu, Y. Yang, M. Li, and Q. Wang. An investigation on the feasibility of cross-project defect prediction. Automated Software Engineering, 19(2):167-199, 2012.

[10] Y. Ma, G. Luo, X. Zeng, and A. Chen. Transfer learning for cross-company software defect prediction. Information and Software Technology, 54(3):248-256, 2012.

[11] J. Nam, S. J. Pan, and S. Kim. Transfer defect learning. In Proceedings of the 35th International Conference on Software Engineering (ICSE), pages 382-391, 2013.

[12] G. Canfora, A. De Lucia, M. Di Penta, R. Oliveto, A. Panichella, and S. Panichella. Multi-objective cross-project defect prediction. In IEEE 6th International Conference on Software Testing, Verification and Validation (ICST), pages 252-261, 2013.

[13] S. Herbold. Training data selection for cross-project defect prediction. In Proceedings of the 9th International Conference on Predictive Models in Software Engineering (PROMISE), pages 6-16, 2013.

[14] Singh, P., Verma, S., & Vyas, O. (2013). Cross Company and within Company Fault Prediction using Object Oriented Metrics. International Journal of Computer Applications, 5-11.

[15] Dejaeger, K. (2013). Toward Comprehensible Software Fault Prediction Models Using Bayesian Network Classifiers. IEEE Transactions on Software Engineering, 39(2), 237–257.

[16] Turhan, B., Mısırlı, A., &Bener, A. (2013). Empirical evaluation of the effects of mixed project data on learning defect predictors. Information and Software Technology, 55(6), 1101–1118. doi:10.1016/j.infsof.2012.10.003

[17] Peters, T. Menzies, and A. Marcus. Better cross company defect prediction. In 10th IEEE Working Conference on Mining Software Repositories (MSR), pages 409-418, 2013.

[18] Panichella, R. Oliveto, and A. De Lucia. Cross-project defect prediction models: L'Union fait la force. In Software Evolution Week-IEEE Conference on Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), pages 164-173, 2014.

[19] He, P.; Li, B.;& Ma, Y. (2014). Towards Cross-Project Defect Prediction with Imbalanced Feature Sets. Cornell University Library.

[20] Nam, J., & Kim, S. (2015). Heterogeneous Defect Prediction. Proceeding ESEC/FSE 2015 Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering (pp. 508-519). New York, NY, USA: ACM. doi:10.1145/2786805.2786814

[21] Peters, T. Menzies, and L. Layman, "Lace2: Better privacypreserving data sharing for cross project defect prediction,"in Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEEInternational Conference on, vol. 1, May 2015, pp. 801–811.

[22] K. Kawata, S. Amasaki, and T. Yokogawa, "Improving relevancy filter methods for cross-project defect prediction," in

[23] Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence (ACIT-CSI), 2015 3rd International Conference on, July 2015, pp. 2–7.

[24] Y. Zhang, D. Lo, X. Xia, and J. Sun, "An empirical study of classifier combination for cross-project defect prediction," in Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual, vol. 2, July 2015, pp. 264–269.

[25] S. Amasaki, K. Kawata, and T. Yokogawa, "Improving crossproject defect prediction methods with data simplification," in Software Engineering and Advanced Applications (SEAA), 2015 41st Euromicro Conference on, Aug 2015, pp. 96–103.

[26] J. Nam and S. Kim, "Clami: Defect prediction on unlabeled datasets," in Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on, Nov 2015, pp. 452–463.

[27] X. Y. Jing, F. Wu, X. Dong, F. Qi, and B. Xu, ―Heterogeneous cross company defect prediction by unified metric representation and cca- based transfer learning,‖ In Proceedings of the 10th Joint Meeting on Foundations of Software Engineering, 2015, pp. 496–507

[28] Ryu, D., Jang, J.-I., &Baik, J. (2015). A transfer cost-sensitive boosting approach for cross-project defect prediction. Software Quality Journal, 1-38. doi:10.1007/s11219-015-9287-1

[29] Zhang, Feng, et al. "Cross-project defect prediction using a connectivity-based unsupervised classifier." Proceedings of the 38th International Conference on Software Engineering. ACM, 2016.

[30] Ryu, D., Choi, O., &Baik, J. (2016). Value-cognitive boosting with a support vector machine for cross project defect prediction. Empirical Software Engineering. doi:10.1007/s10664-014-9346-4.

[31] Xia, Xin, et al. "Hydra: Massively compositional model for cross-project defect prediction." IEEE Transactions on software Engineering 42.10 (2016): 977-998.

[32] Steffen Herbold et al. "A Comparative Study to Benchmark Cross-project Defect Prediction Approaches". Ieee Transactions On Software Engineering, 2017.

[33] Lipika Goel, Mayank Sharma, Sunil Kumar Khatri , D.damodaran (In press). An empirical analysis of the statistical learning models for different categories of Cross Project Defect Prediction, Int.Journal of Computer Aided Engineering & Technoloy, Inderscience,2018.

[34] Sharma, Manik, and Gurdev Singh. "Analysis of Static and Dynamic Metrics for Productivity and Time Complexity."International Journal of Computer Applications 30.1 (2011): 7-13.

[35] Misra, Sanjay, and Adewole Adewumi. "Object-oriented cognitive complexity measures: An analysis." Intelligent Systems: Concepts, Methodologies, Tools, and Applications. IGI Global, 2018. 1324-1347.

[36] Sharma, Manik, et al. "A comparative study of static object oriented metrics." International Journal of Advancements in Technology 3.1 (2012): 25-34.

[37] Sunil, Jinu M., Lov Kumar, and Lalita Bhanu Murthy Neti. "Bayesian Logistic Regression for software defect prediction (S)." SEKE. 2018.