Chapter 3

A Supervised SBP Technique Using Reward-Based WMV Ensemble Approach

This chapter focuses on the first contribution of this thesis detail in Section 1.7.1. We provide an introduction, motivation, and key feature for the problem to develop classification-based supervised SBP in Section 3.1. Section 3.2 demonstrates the proposed work to develop the classification-based supervised SBP. The experimental setup is given in Section 3.3. Section 3.4 covers the results obtained using the proposed technique. Section 3.5 reports the threat of validity associated with the chapter. Section 3.6 focuses on the conclusion and future scope of this chapter. Detailed related work is provided in Section 2.1.1.

3.1 Introduction

In recent years, several SBP models have been developed using various ML techniques. These models exhibit an average performance accuracy of approximately 80% to 85% [10, 94]. Hybrid learning techniques have been explored to improve the

performance of SBP models, but there is still room for performance improvement, especially in the majority voting ensemble technique. Hence, we proposed WMV.

The key features of WMV that deal with the aforesaid limitations of existing SBP techniques are as follows: Class imbalance problem is handled using three sampling techniques. To improve the performance, we have trained five BCs with 10-fold cross-validation (FCV) and then the WMV technique is used to predict the final class labels. WMV technique consists of the following three steps: **(a)** A novel weight calculation scheme is proposed to provide an effective weight to each BC based on its performance on test dataset. The weight of each BC is calculated by adding a reward to the BC that predicts the correct label of an instance while no reward/penalty is given for a wrong prediction of an instance by a BC **(b)** Secondly, we obtained the weighted probability of each BC and the corresponding prediction class label using a threshold **(c)** At last, we applied majority voting on this predicted class to obtain the final label (buggy or non-buggy) of an instance (Subsection 3.2.2).

Fig. 3.1 shows the block diagram to implement the proposed weighted majority voting methods. As shown in the Fig. 3.1, firstly, we balanced the considered imbalanced dataset using ROSE (Random oversampling example) method [54], then all the five BCs were trained using 10 FCV. After that, we evaluated the weight of each classifier based on their performance. This weight calculation approach uses reward-based scheme to update the weight of each classifier across all the instances. We then multiplied the prediction probability of a classifier with its weight. Based on the optimal threshold value, we categorized the instances as buggy or non-buggy for each classifier. Now perform majority voting on these label vectors and finally label each instance based on majority class.

3.2 Proposed Approach: WMV

Ensemble Method: Machine learning (ML) ensemble methods are based on the real-world concept of "unity is strength." This phrase expresses the main idea

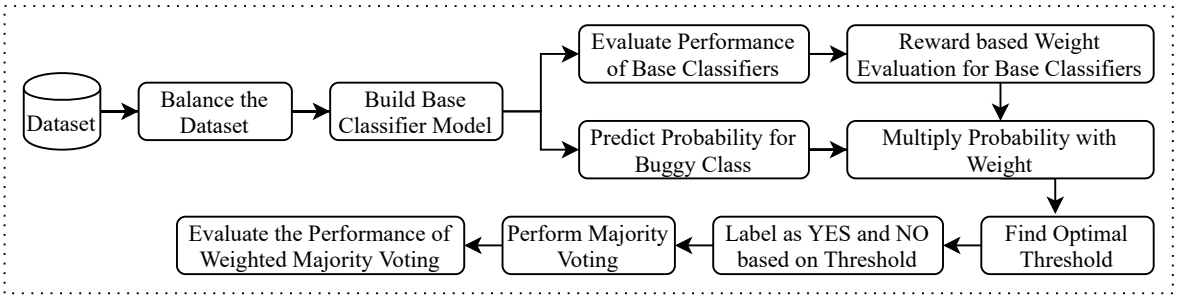

FIGURE 3.1: Block diagram to develop the proposed SBP

behind ensemble methods quite well. It is based on the assumption that combining multiple learning algorithms produces significantly better results than benchmark standalone algorithms. Ensemble methods are ML approaches in which multiple weak learners are trained to solve the same problem and then blended to achieve better results.

In ensemble techniques, weak learners or BCs serve as the foundation for developing a more complex model by combining several of them. These models frequently do not perform well independently due to high bias (low degree of freedom) or high variance (high degree of freedom). As a result of combining these models, strong learners are produced by reducing the bias and/or variance of such weak classifiers, resulting in improved performance [204]. Five BCs have been chosen for this study: K-nearest neighbor (KNN) [198], Naive Bayes (NB) [199], Support vector machine (SVM) [200], Random Forest (RF) [201], and C5.0 (C50) [202].

There are two types of ensemble methods: heterogeneous and homogeneous. Heterogeneous ensemble methods (e.g., stacking, voting) consist different BCs, whereas homogeneous methods (e.g., bagging, boosting) consist the same BC. Bagging is used to minimize variance, boosting is used to minimize bias, and stacking/voting is often used to improve the predictive power of the model. In this study, two ensemble methods, i.e., simple majority voting (SMV) and weighted majority voting (WMV), are implemented.

3.2.1 Simple Majority Voting Ensemble Method (SMV)

SMV is a heterogeneous ensemble technique that combines outcomes based on the BCs' majority voting. The outcome of each classifier on test data is used in this method, and the final outcome of the ensemble is the majority class predicted by BCs. In this type of simple majority voting, all BCs have an equal value of vote, and that is one, i.e., the weight of each classifier is $\omega_k = 1$. Let η be the number of BCs (should be an odd number) chosen in the SMV method and n be the number of instances in the DS. Suppose, $p_{i,k}$ be the prediction probability by BC function ϕ_k on dataset $DS[x_i, y_i]$. Where, $x_i \in X[1:n, 1:m]$ and $y_i \in (NO, YES)$, $i = 1, 2, .., n$ is the number of instances, m is the number of metrics, and y_i is the actual label vector. The prediction probability $p_{i,k}$ by k^{th} BC prediction function ϕ_k $(k = 1, 2, .., \eta)$ is represented using (3.1)

$$p_{i,k} = \phi_k(DS[x_i, y_i]) \quad \forall \quad i = 1, 2, ..., n \tag{3.1}$$

Now, Based on $p_{i,k}$, the prediction label vector $\hat{y}_{i,k}$ of BC is defined using (3.2):

$$\hat{y}_{i,k} = \begin{cases} 'YES' & \text{if } p_{i,k} > 0.5,\ \phi_k \text{ predicts buggy label for } i^{th} \text{ instance} \\ 'NO' & \text{otherwise, } \phi_k \text{ predicts non buggy label for } i^{th} \text{ instance} \end{cases} \tag{3.2}$$

After obtaining the label vector $\hat{y}_{i,k}$ using all BC, the label vector $\delta_{i,SMV}$ is the resultant label vector using SMV method for each instance i. This label can be obtained using (3.3).

$$\delta_{i,SMV} = \underset{\hat{y}_{i,k}=NO}{\overset{YES}{\arg\max}} \sum_{k=1}^{\eta} \hat{y}_{i,k} \quad \forall \quad i = 1, 2, ..., n \tag{3.3}$$

Usually, the performance of BCs is different. Therefore, combining them with equal weights in the ensemble (SMV) may not produce optimal results. So, to mitigate this problem, we assigned different weights to classifiers based on their performance. However, assigning different weights to BCs is a critical issue. So, we designed a

TABLE 3.1: Prediction probability of base classifiers for each instance

Instances	Base classifiers function			
	ϕ_1	ϕ_2	...	ϕ_η
1	$p_{1,1}$	$p_{1,2}$	...	$p_{1,\eta}$
2	$p_{2,1}$	$p_{2,2}$	...	$p_{2,\eta}$
.	.	.	.	.
.	.	.	.	.
n	$p_{n,1}$	$p_{n,2}$	...	$p_{n,\eta}$

weighting scheme that appropriately allocates the weights to classifiers based on their prediction performance. We embedded this scheme in our proposed WMV ensemble method, which is described in the following subsection.

3.2.2 Weighted Majority Voting (WMV) Ensemble Method

WMV is also a heterogeneous ensemble method where different BCs are assigned weights on the basis of their prediction performance. The ensemble predicts the final label of each instance after combining the weighted results of all the BCs. It is emphasized here that if we have to assign weight to multiple classifiers, we assign higher weights to the BCs with a higher number of correctly predicted instances. Fig. 3.1 shows the block diagram to implement the proposed weighted majority voting methods. The step-by-step implementation of the proposed approach is shown in Algorithm 1 and described as follows.

Let $DS[x_i, y_i]$ be the input dataset. Table 3.1 shows the prediction probability $p_{i,k}$ of each instance i of the dataset $DS[x_i, y_i]$ by each BC (k^{th}) using (3.1). Based on prediction probability $p_{i,k}$ in Table 3.1, we have calculated the corresponding label using (3.2) After finding the prediction label by BCs, the reward-based weight evaluation scheme applied (using (3.4)).

3.2.2.1 Reward-Based Weight Evaluation Scheme

The proposed WMV updates the weights of BCs across each instance of testing dataset of 10-FCV. We set the weights of all BCs to 1 at the start (Ini. weights). All instances are navigated and analyzed by η BC once. The weights of the BCs that rightly predict the class label of an instance are increased by the ratio (β_i) of the number of wrong BC (Z) for that instance by BCs to the total number of BCs (η).

Suppose, $\omega_{i,k}$ is the weight of k^{th} BC after predicting label of the i^{th} instance. The weight of each classifier is updated using (3.4):

$$\omega_{i,k} = \begin{cases} \omega_{i-1,k} + \beta_i & \text{if } k^{th} \ \phi \text{ predicts correct label for } i^{th} \text{ instance} \\ \omega_{i-1,k} & \text{if } k^{th} \ \phi \text{ predicts wrong label for } i^{th} \text{ instance} \end{cases} \tag{3.4}$$

Where β_i is the change in weight and is evaluated using (3.5). Z_i is the number of wrong predictions for i^{th} instance by all BCs.

$$\beta_i = \frac{Z_i}{\eta} \quad \forall \quad i = 1, 2, ..., n \tag{3.5}$$

When all the instances are traversed by η BCs once, the final output is saved as the weights and shown in Table 3.2. These weights are used as the effective vote of each BC to predict class label of each instance.

After calculating the final weights (F.weights) $\omega_{n,k}$, we calculated the weighted probability $\mathcal{P}_{i,k}$ using (3.6)

$$\mathcal{P}_{i,k} = p_{i,k} * \omega_{n,k} \quad \forall \quad i = 1, 2, ..., n \tag{3.6}$$

Then, we calculated the threshold θ_k for each BC label vector using (3.7). We considered a threshold value that is around the middle of the weighted probability

TABLE 3.2: Weights of each classifier evaluated after traversing each instance of testing dataset

Instances	Base classifiers (ϕ)			
	ϕ_1	ϕ_2	...	ϕ_η
0	1	1	...	1
1	$\omega_{1,1}$	$\omega_{1,2}$	...	$\omega_{1,\eta}$
2	$\omega_{2,1}$	$\omega_{2,2}$	...	$\omega_{2,\eta}$
.	.	.	.	.
.	.	.	.	.
n	$\omega_{n,1}$	$\omega_{n,2}$	...	$\omega_{n,\eta}$

range i.e. from 0 to $max(\mathcal{P}_{i,k})$. This threshold (half of $max(\mathcal{P}_{i,k})$) is taken as an analogy to the default decision threshold, i.e., 0.5.

$$\theta_k = \frac{\max(\mathcal{P}_{i,k})}{2} \quad \forall \quad k = 1, 2, ..., \eta \tag{3.7}$$

Based on threshold θ_k, we have calculated weighted label $\lambda_{i,k}$ for each classifier using (3.8) for each instance i.

$$\lambda_{i,k} = \begin{cases} 'YES' & \text{if } \mathcal{P}_{i,k} > \theta_k \\ 'NO' & \text{if } \mathcal{P}_{i,k} <= \theta_k \end{cases} \tag{3.8}$$

Finally, in WMV ensemble, all weighted votes for each class (label) are added, and the class with the maximum weighted votes becomes the predicted class label of an instance. The following equation (3.9) is used to predict the class of each instance using proposed weighted majority voting ensemble methods. Label $\delta_{i,WMV}$ is the predicted class label using WMV for each instance i.

$$\delta_{i,WMV} = \underset{\hat{y}_{i,k}=NO}{\overset{YES}{\arg\max}} \sum_{k=1}^{\eta} \lambda_{i,k} \quad \forall \quad i = 1, 2, ..., n \tag{3.9}$$

An example for calculating the weight Let us consider an example to illustrate the proposed approach weighted majority voting (WMV) method. Suppose

TABLE 3.3: An example dataset with predicted probability ($p[i,k]$), corresponding class label using BCs and original class label

Instances	BCs predicted probability, corresponding class label					Original Class
	BC1	BC2	BC3	BC4	BC5	
1	0.8, YES	0.3, NO	0.7, YES	0.7, YES	0.4, NO	YES
2	0.3, NO	0.4, NO	0.6, YES	0.7, YES	0.4, NO	YES
3	0.7, YES	0.3, NO	0.3, NO	0.6, YES	0.4, NO	NO
4	0.2, NO	0.1, NO	0.2, NO	0.1, NO	0.1, NO	NO
5	0.4, NO	0.4, NO	0.6, YES	0.3, NO	0.7, YES	No
6	0.6, YES	0.6, YES	0.7, YES	0.7, YES	0.4, NO	YES

there are five BCs that are applied on six instances for buggy class prediction. Table 3.3 shows the predicted probability and corresponding class label by the five BCs function (ϕ_1, ϕ_2, ϕ_3, ϕ_4, ϕ_5) on 6 instances using (3.2).

Evaluation of weights of each base classifier (BC) after predicting the label of each instance (Table 3.3) are shown in Table 3.4. Firstly, we initialized the weights (Ini. weights) to 1. For each instance, the weight of only correctly predicting BC is increased by the ratio of number of wrongly predicting BCs to the total number of BCs (β_i). Weights of wrongly predicting BCs are not modified. For example, there are three BCs (ϕ_1, ϕ_3, ϕ_4) predicting right class for instance 1 in Table 3.3. So, the weights of these three classifiers are incremented by rewarding them 2/5 (β_i). This process is followed for each instance till we get the final weights for all the classifiers. The weights obtained for last instance (n^{th}) are used as the final weight (Fin.weights) value of each BC. Table 3.5 shows weighted probability calculation using (3.6).

The run-time complexity of Algorithm 1 and Algorithm 2 is the same i.e. O($n*\eta$). The worst-case run time complexity is O(n^2) for worst-case $\eta = n$ (assuming number of BCs is always less than or equal to number of instances).

Algorithm 1: Proposed algorithm to implement WMV approach

Input: Dataset $DS[x_i, y_i]$ for $i = 1, 2, .., n$ instances and $x_i \in X[1:n, 1:m]$ with label vector $y_i \in (NO, YES)$
Set of five BCs function ϕ_k={NB, SVM, KNN, RF, C50}
Output: $\delta_{i,WMV}$: Prediction label using WMV ensemble methods
Performance in terms of Accuracy, FM, and MCC
Preprocessing: Balance $DS[x_i, y_i]$ using ROSE method
Algorithmic Steps:
Step 1: Training BCs using 10-fold cross-validation (FCV)
for $k \leftarrow 1$ *to* η **do**
  Train BC using function $\phi_k(DS[x_i, y_i])$
  Save the prediction probability $p[i, k]$ using (3.1)
  Predict label $\hat{y}[i, k]$ using (3.2)
  Compute the performance of trained classifier ϕ_k on the testing dataset of 10-fold
end
Step 2: Evaluate final weight ω_{nk} of each ϕ_k using reward based mechanism //Algorithm 2
Step 3: Calculate the weighted probability $\mathcal{P}_{i,k}$ using (3.6)
for $i \leftarrow 1$ *to* n **do**
  for $k \leftarrow 1$ *to* η **do**
    $\mathcal{P}[i, k] \leftarrow p[i, k] * \omega[n, k]$
  end
end
Step 4: Compute the threshold θ_k using (3.7)
for $k \leftarrow 1$ *to* η **do**
  $\theta[k] \leftarrow \frac{\max(\mathcal{P}[i,k])}{2}$
end
Step 5: Compute weighted label $\lambda_{i,k}$ for each instances based on threshold θ_k using (3.8)
for $i \leftarrow 1$ *to* n **do**
  for $k \leftarrow 1$ *to* η **do**
    if $\mathcal{P}[i, k] > \theta[k]$ **then**
      $\lambda[i, k] \leftarrow' YES'$
    else
      $\lambda[i, k] \leftarrow' NO'$
  end
end
Step 6: Perform weighted majority voting to obtain $\delta_{i,WMV}$ using (3.9). E.g. $\delta_{i,WMV}$ for three BCs ($\eta = 3$)
for $i \leftarrow 1$ *to* n **do**
  if *λ[i,1]=='YES' ∧ λ[i,2]=='YES'* **then**
    $\delta[i, WMV] \leftarrow' YES'$
  else if *λ[i,1]=='YES' ∧ λ[i,3]=='YES'* **then**
    $\delta[i, WMV] \leftarrow' YES'$
  else if *λ[i,2]=='YES' ∧ λ[i,3]=='YES'* **then**
    $\delta[i, WMV] \leftarrow' YES'$
  else
    $\delta[i, WMV] \leftarrow' NO'$
  end
end
Step 7: Evaluate the performance of WMV in terms of Accuracy, FM, and MCC and also perform statistical tests for each pair of models to compare the results statistically

TABLE 3.4: Change in weights of base classifiers across each instance

Instances	BCs final weight calculation					#Wrong BC (Z)
	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	
Ini. weights	1	1	1	1	1	
1	1+2/5=1.4	1	1+2/5=1.4	1+2/5=1.4	1	2
2	1+2/5=1.4	1	1.4+3/5=2	1.4+3/5=2	1	3
3	1+2/5=1.4	1+2/5=1.4	2+2/5=2.4	2	1+2/5=1.4	2
4	1.4	1.4	2.4	2	1.4	0
5	1.4+2/5=1.8	1.4+2/5=1.8	2.4	2+2/5=2.4	1.4	2
6	1.8+1/5=2	1.8+1/5=2	2.4+1/5=2.6	2.4+1/5=2.6	1.4	1
F.weights ($\omega_{n,k}$)	2	2	2.6	2.6	1.4	

TABLE 3.5: Weighted probability $\mathcal{P}[i,k]$ calculation and original class label

Instances	Weighted probability of BCs					Original Class
	BC1	BC2	BC3	BC4	BC5	
1	0.8*2=1.6	0.3*2=0.6	0.7*2.6=1.82	0.7*2.6=1.82	0.4*1.4=0.56	YES
2	0.3*2=0.6	0.4*2=0.8	0.6*2.6=1.56	0.7*2.6=1.82	0.4*1.4=0.56	YES
3	0.7*2=1.4	0.3*2=0.6	0.3*2.6=0.78	0.6*2.6=1.56	0.4*1.4=0.56	NO
4	0.2*2=0.4	0.1*2=0.2	0.2*2.6=0.52	0.1*2.6=0.26	0.1*1.4=0.14	NO
5	0.4*2=0.8	0.4*2=0.8	0.6*2.6=1.56	0.3*2.6=0.78	0.7*1.4=0.98	NO
6	0.6*2=1.2	0.6*2=1.2	0.7*2.6=1.82	0.7*2.6=1.82	0.4*1.4=0.56	YES
$\max(\mathcal{P}_{i,k})$	1.6	1.2	1.82	1.82	0.98	
θ_k	0.8	0.6	0.91	0.91	0.49	

3.3 Experimental Design

The main contribution of WMV is 'calculating the weights 'of BCs based on their performance on the balanced software bug dataset (DS). In this section, we will explore the experimental setup necessary for conducting the experiments. We will delve into RQs in Section 3.3.1, datasets in Section 3.3.2, experimental baselines in Section 3.3.3, and performance measures in Section 3.3.4. To compare the performance of the proposed method with state-of-the-art (SOTA) techniques, we created RQs, which are explained in the following sections.

3.3.1 Research Questions (RQ)

RQ1: Are the results obtained by WMV technique comparable to those obtained by the BCs independently?

To address this RQ, we compared the performance of WMV with that of the five

Algorithm 2: Reward-based weight evaluation scheme

Input: Let $LT[\hat{y}_{i,k}, y_i]$ is a label table. Where, $\hat{y}_{i,k}$ is a predicted vector using $\phi_k(DS[x_i, y_i])$ and y_i is the original label vector.
Set of five BCs function ϕ_k={NB, SVM, KNN, RF, C50}
Output: $\omega_{n,k}$:Weight of each ϕ_k
Algorithmic Steps:
Step 1: Let $Z_i \leftarrow$ number of wrongly predicting BCs in an instance i
for $i \leftarrow 1$ *to* n **do**
  $nwbc \leftarrow 0$ //Initial no. of wrong predictors
  for $k \leftarrow 1$ *to* η **do**
    if $\hat{y}[i,k] \neq y[i]$ **then**
      $nwbc \leftarrow nwbc + 1$
  end
  $Z[i] \leftarrow nwbc$
end
Step 2: Assigning initial weight 1 to all BC
for $k \leftarrow 1$ *to* η **do**
  $\omega[0,k] \leftarrow 1$
end
Step 3: Evaluate the weight of each classifier
for $i \leftarrow 1$ *to* n **do**
  for $k \leftarrow 1$ *to* η **do**
    if $\hat{y}[i,k] == y[i]$ **then**
      $\omega[i+1,k] \leftarrow \omega[i,k] + Z[i]/\eta$
    else
      $\omega[i+1,k] \leftarrow \omega[i,k]$
  end
end

BCs (NB, SVM, KNN, RF and C50) used independently. The caret package [205] of the R library is used to implement these algorithms (BCs). If performance of WMV is comparable to that of BCs used independently, then it may be efficiently utilized to increase the reliability of software projects.

RQ2: Are the results obtained by WMV technique comparable to those of the existing SMV ensemble-based SOTA technique?
To address this RQ, we compared the performance of WMV with that of the SOTA SMV ensemble-based technique. If we achieve better results than SMV ensemble-based technique, then WMV technique will prove to be a better alternative than

SMV.

RQ3: Are the results obtained by WMV technique comparable to those obtained by best-performing SOTA advanced techniques?
To address this RQ, we compared WMV's performance with the recent research papers that have comparatively analyzed various existing SOTA techniques (Table 3.12 and Table 3.13).

RQ4: Is the ROSE sampling technique better than SMOTE and RUS for WMV?
To address this RQ, we first implemented our WMV technique with ROSE, SMOTE, and RUS. Thereafter, we compared their results.

3.3.2 Datasets (DSs)

The 28 standard datasets obtained from 5 repositories SOFTLAB [27], AEEEM [25, 28], NASA [178], ReLink [179], and MORPH [180] are shown in Table 2.3 and details given in Section 2.4.2.1. All of the 28 DSs are utilized to evaluate the WMV technique and compare it with the existing techniques.

3.3.3 Experimental Baseline

We implemented five supervised ML algorithms to compare the results of the proposed WMV and answer our four RQs.

- **Baseline 1:** As the first baseline methodology, we created 5 supervised algorithms (NB, SVM, KNN, RF, C50) on labeled data sets with a 10-FCV method. Fig. 3.1 shows the step-by-step block diagram of creating the WMV technique. Then, we compared the results obtained using conventional supervised algorithms independently with those obtained using the WMV technique. The findings from this comparison were utilized to address RQ1.

- **Baseline 2:** As the second baseline methodology, we built an SMV-based SBP model using the same five BCs as we have used in our proposed technique

WMV. We have compared the results of WMV with SMV. This comparison answers RQ2.

- **Baseline 3:** As the third baseline methodology, we selected results from recent research articles already published. These papers have comparatively analyzed the results of SOTA advanced methods to predict the bug on DSs. So, we compared the effectiveness of WMV with the results of best-performing algorithms in these papers on the same data set. This comparison answers RQ3.

- **Baseline 4:** As the fourth baseline methodology, we built WMV with ROSE, SMOTE, and RUS. We have compared the results of these three models to answer RQ4.

3.3.4 Performance Measures

The three performance parameters, viz. accuracy, FM, and MCC are utilized to compare the results of SBP models and details given in Section 2.4.3.1. Since the FM and MCC are particularly useful when dealing with imbalanced classes, it effectively addresses the issue of class imbalance. Additionally, it quantifies the bias in binary classifiers. Further, we have used boxplot to visualize the comparative performance of SBP models, and the boxplot attribute is provided in Section 2.4.3.3. For strong comparison, we employed two statistical tests to compare the various methods. Firstly, we utilized the Wilcoxon signed-rank test to conduct pairwise comparisons. Secondly, we employed Friedman test [193], followed by Nemenyi test [192] as recommended by Demsar [194]. The details about statistical tests are given in Section 2.4.4.

3.4 Experimental Results

This section summarises the results of our experiments by applying three distinct performance metrics (Subsection 2.4.3.1). Tables 3.6, 3.8, 3.10 presents the result

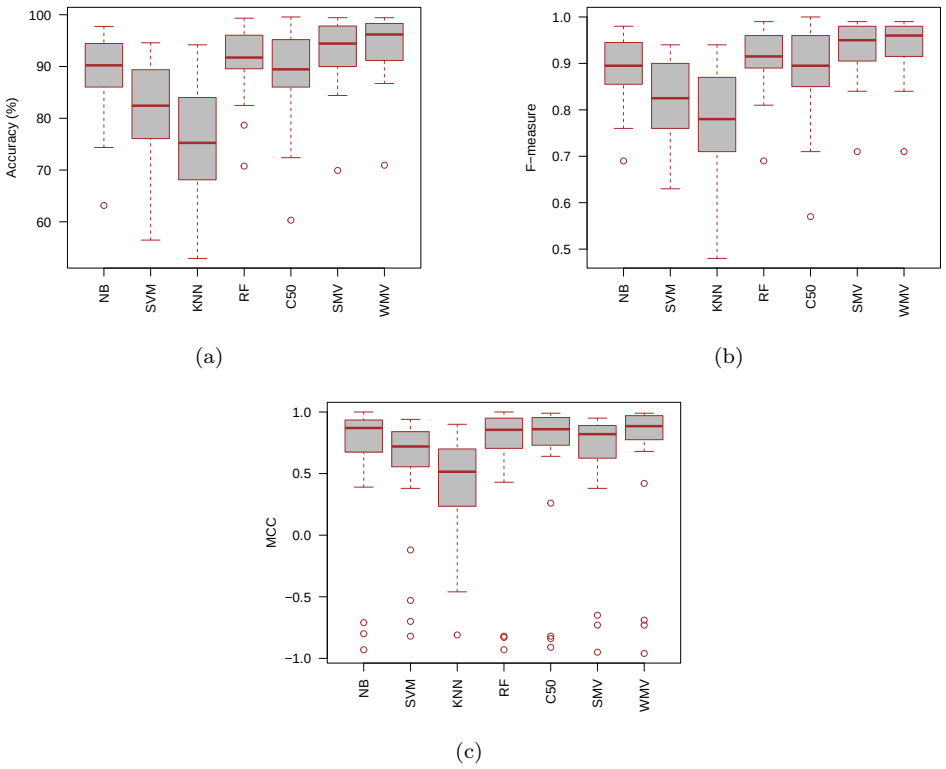

FIGURE 3.2: Comparative performance of different techniques are shown by boxplots with respect to (a) Accuracy, (b) FM, and (c) MCC.

of the WMV model to existing techniques in terms of accuracy, FM, and MCC across 28 data sets, respectively . The median values of the various techniques are shown in the last rows of each group of DSs. The median value of the WMV technique is indicated in bold wherever it is larger or equal to the other methods. The performance value of BCs, SMV, and WMV techniques are evaluated using 10-FCV. We observed that the performance of WMV is superior to that of BCs and SMV after extensive testing.

In terms of performance measure, the proposed method WMV is superior to the BCs viz. NB, SVM, KNN, RF, and C50, and an ensemble approach SMV are used independently. Among the BCs, the best performance is provided by RF in terms of accuracy and FM, whereas C50 performs the best in terms of MCC. SMV is also performing better than all BCs with respect to accuracy and FM but lags behind C50 in terms of MCC.

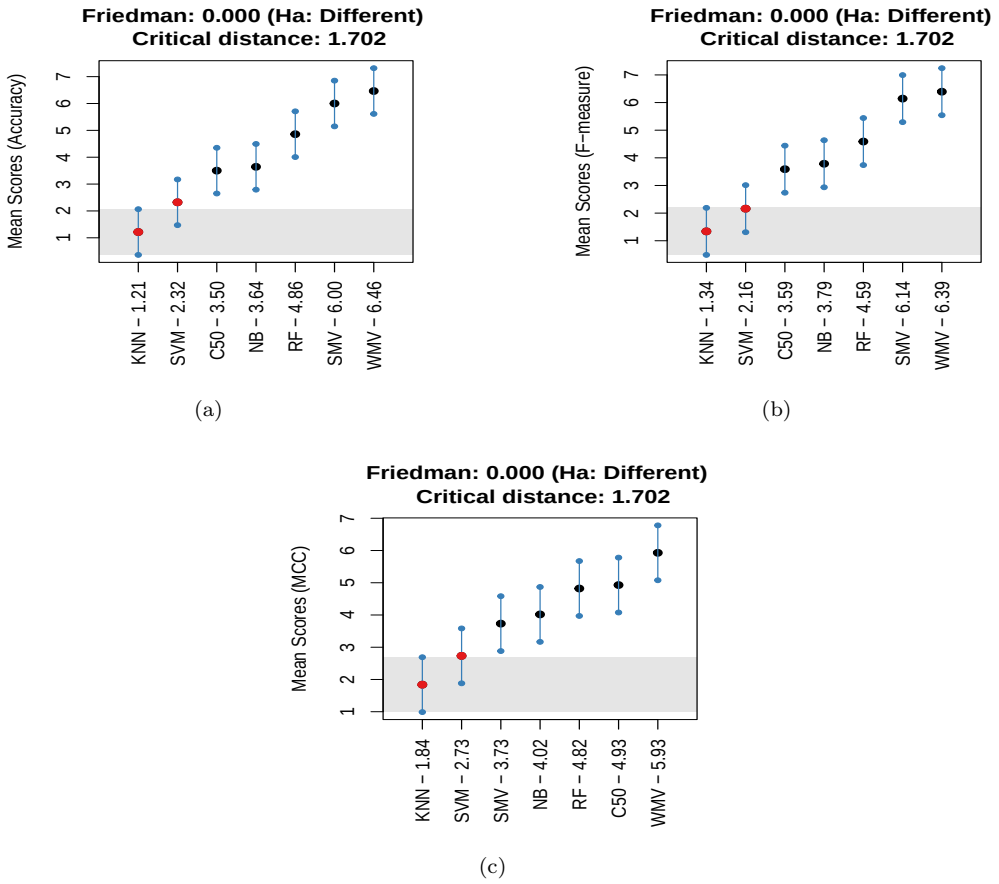

FIGURE 3.3: Critical diagram (using Nemenyi test) representation for the WMV and other techniques on 28 DSs with respect to (a) Accuracy, (b) FM, and (c) MCC.

Tables 3.7, 3.9, 3.11 displays the mean differences in terms of accuracy, f-measure and MCC, respectively, among the techniques, along with the corresponding p-values denoting statistical significance (when p-value < 0.05). The mean difference in these three Tables are the difference between the average results of the techniques across all DSs. The mean difference between column and row techniques is expressed by the left bottom triangle in these three Tables. The p-value, evaluated using the Wilcoxon signed-rank test, signifies the statistically significant difference between 2 techniques. The right upper triangle of these three Tables presents the p-values between the two techniques. If the p-value is greater than 0.05, there is no significant difference between the techniques. Conversely, if the p-value is less than 0.05, it indicates a significant difference between the two techniques [191].

The critical diagram generated by the Nemenyi test is displayed in Fig. 3.3. This diagram presents Friedman's p-value, indicating statistical significance (with hypothesis H_a as significant differences), along with the mean score of each technique. Additionally, the critical distance value (1.702) is indicated. The X-axis of the diagram shows the technique name followed by its corresponding mean score. The best-performing technique is situated on the right side of the diagram. In Fig. 3.3, the stick length of a technique represents its performance, with the length falling within plus/minus the critical distance (CD) from the mean score. The diagram visualizes the mean scores of different techniques using two colors: a blue stick with a red dot indicates comparatively poorer performance (with a shaded background) in contrast to a blue stick with a black dot (higher performance).

Note: If any two techniques have the same mean score value, then the performance of these two techniques is the same over all DSs. In Fig. 3.3a,b, and c, since no techniques have the same mean score value, so no techniques are performing the same over all DSs. If any two techniques with different mean score values belong to the same group, then their performance is significantly the same over all DSs. E.g., in Fig. 3.3a, C50 (3.50) and NB (3.64) have different mean scores but belong to the same group, so their performance is significantly the same over all DSs. The performance of the two techniques is considered significantly different if their mean scores differ by more than the calculated CD. E.g., in Fig. 3.3a, the minimum mean score is 1.21 (KNN). After adding the critical distance of 1.702, we will get 2.912. All techniques with a mean score greater than 2.912 will be marked in one group (black dot), and all the techniques with a mean score less than 2.912 will be marked in the other group (red dot). If any two techniques belong to different groups, their performance is significantly different over all DSs. E.g., in Fig. 3.3a, KNN (1.21) and NB (3.64) belong to different groups, so their performance is significantly different over all DSs.

3.4.1 Results with Respect to Accuracy

Table 3.6 presents the accuracy results. The accuracy % of WMV is superior or equal to SMV in all groups of DSs. However, SMV performs better in all groups of DSs as compared to all other BCs. RF is superior to NB, SVM, KNN, and C50 in 4 out of 5 groups of DSs. NB performance is the highest in SOFTLAB DSs among all the BCs.

The performance of WMV is significantly different from all of the BCs and SMV based on the p-values shown in Table 3.7. C50 outperforms NB with a slightly higher accuracy of 0.25%. However, the p-value of 0.63 (greater than 0.05) indicates that there is no significant difference between these two techniques. Also, mean performance (accuracy, FM, MCC) value of WMV is higher than mean performance value of all base classifiers (BCs) and SMV, as last row shown in Table 3.7 (higher values are highlighted in bold).

Using Fig. 3.2a, we infer that WMV surpasses all BCs and also SMV. SMV holds the second rank in performance by showing better results than BCs. RF is the best performer among all BCs. RF, also known as bagging method (type of homogeneous ensemble technique), shows that ensemble techniques are effective in SBP.

From Fig. 3.3a, based on the mean score of KNN (1.21) and SVM (2.32), we marked them as low performing group (shaded background), while WMV (6.46), along with C50 (3.50), NB (3.64), RF (4.86), and SMV (6.0) belongs to high performing group. Although the mean score of WMV exceeds the mean scores of C50, NB, RF, and SMV, all of these approaches belong to the same high-performing group. SVM, with a mean score of 2.32, falls on the boundary between the high and low-performing groups, making it challenging to ascertain its performance definitively.

TABLE 3.6: Performance of different SBP techniques with respect to accuracy

Groups	Projects	Accuracy (%)						
		NB	SVM	KNN	RF	C50	SMV	WMV
AEEEM	Equinox	88.39	91.43	86.95	92.68	88.81	95.06	98.44
	JDT	97.09	94.58	94.19	97.59	97.54	98.19	98.19
	Lucene	91.02	90.07	85.99	94.50	93.70	96.67	96.67
	Mylyn	83.11	90.05	67.70	90.33	90.15	92.11	92.11
	PDE	96.36	91.63	85.22	98.98	98.97	99.06	99.06
	MED	91.02	91.43	85.99	94.50	93.70	96.67	**98.19**
SOFTLAB	AR1	87.15	77.08	68.53	96.24	96.40	96.69	96.69
	AR3	97.74	89.14	82.81	90.52	82.38	93.65	93.65
	AR4	96.18	81.63	79.84	91.78	89.29	96.26	96.26
	AR5	89.83	81.80	80.50	78.67	77.08	91.67	98.89
	AR6	96.00	74.18	74.75	99.16	95.69	99.01	99.01
	MED	96.00	81.63	79.84	91.78	89.29	96.26	**96.69**
NASA	CM1	74.36	72.64	52.93	82.48	72.39	84.40	87.18
	MW1	81.78	77.72	58.96	89.90	88.35	89.33	92.72
	PC1	93.19	93.43	72.33	95.85	94.04	97.16	97.16
	PC3	92.20	83.85	89.90	95.01	94.43	96.10	96.10
	PC4	96.93	85.58	77.33	97.08	95.64	98.60	98.60
	MED	92.20	83.85	72.33	95.01	94.04	**96.10**	**96.10**
MORPH	Ant_1.3	91.79	89.18	78.90	89.91	87.37	91.20	91.20
	Arc	90.94	76.32	71.91	98.27	99.57	97.86	97.86
	Camel_1.0	91.50	83.20	69.31	90.08	87.52	93.81	93.81
	Poi_1.5	89.47	82.87	70.49	90.59	87.68	90.30	91.14
	Redaktor	90.62	72.65	56.90	99.33	99.44	99.43	99.43
	Skarbonka	87.75	56.47	60.10	94.90	94.75	97.78	98.89
	Tomcat	87.59	87.33	86.79	91.68	89.62	91.84	91.84
	Velocity_1.4	95.70	89.59	88.59	94.37	92.69	97.96	97.96
	Xalan_2.4	80.51	81.99	75.75	86.23	83.34	87.83	89.69
	Xerces_1.2	86.76	75.85	74.24	86.86	84.68	87.95	89.73
	MED	90.05	82.43	73.07	91.13	88.65	**92.82**	**92.82**
ReLink	Apache	85.30	74.85	57.35	89.22	88.98	89.69	89.69
	Safe	82.86	80.00	76.88	83.40	81.47	85.71	86.71
	Zxing	63.16	65.50	57.71	70.77	60.31	69.92	70.93
	MED	82.86	74.85	57.71	83.40	81.47	85.71	**86.71**

TABLE 3.7: Wilcoxon signed-rank test analysis (p<0.05): Mean difference (Column-Row) between the Accuracy of different approaches and p-value. Mean difference: Left bottom triangle; p-value: Right upper triangle

	Accuracy (%)						
	NB	SVM	KNN	RF	C50	SMV	WMV
NB	_	0.00	0.00	0.01	0.63	0.00	0.00
SVM	6.95	_	0.00	0.00	0.00	0.00	0.00
KNN	14.37	7.42	_	0.00	0.00	0.00	0.00
RF	-2.54	-9.49	-16.91	_	0.00	0.00	0.00
C50	-0.25	-7.20	-14.62	2.29	_	0.00	0.00
SMV	-4.28	-11.24	-18.66	-1.75	-4.03	_	0.01
WMV	**-5.15**	**-12.11**	**-19.53**	**-2.62**	**-4.90**	**-0.87**	_

3.4.2 Results with Respect to FM

Table 3.8 presents the FM performance on all DSs. WMV's FM is superior to or equivalent to other methods. In all DS groups, the FM of SMV outperforms all BC methods.

Table 3.9 concludes that WMV (in terms of FM) performs significantly better than all BCs and also SMV. The mean difference between C50 and NB is zero, which implies that they are not significantly different (p=0.98). However, on the basis of p-value, WMV performs significantly better than all BCs and SMV (shown in bold). P-value also indicates that SMV performs better than all the BCs.

As shown in Fig. 3.2b, WMV outperforms all BCs and SMV. Based on the boxplot, performance of SMV is also greater than all BCs.

Based on Fig. 3.3b, it can be observed that KNN, with a mean score of 1.34, falls into the low-performing group. Conversely, WMV, along with C50, NB, RF, and SMV, belongs to the high-performing group, as indicated by their mean scores. From the critical diagram shown in Fig. 3.3b, it is evident that WMV is a better performing model, followed by SMV. On the other hand, SVM, with a mean score of 2.16, lies on the boundary between the low and high-performing groups, making it challenging to ascertain its performance definitively.

TABLE 3.8: Performance of different SBP techniques with respect to F-measure

Groups	Projects	F-measure						
		NB	SVM	KNN	RF	C50	SMV	WMV
AEEEM	Equinox	0.88	0.91	0.87	0.92	0.89	0.95	0.98
	JDT	0.97	0.94	0.94	0.97	0.97	0.98	0.98
	Lucene	0.91	0.90	0.87	0.94	0.94	0.97	0.97
	Mylyn	0.85	0.90	0.73	0.90	0.90	0.92	0.96
	PDE	0.96	0.92	0.87	0.99	0.99	0.99	0.99
	MED	0.91	0.91	0.87	0.94	0.94	0.97	**0.98**
SOFTLAB	AR1	0.88	0.77	0.71	0.96	0.96	0.97	0.97
	AR3	0.98	0.91	0.88	0.93	0.85	0.95	0.95
	AR4	0.96	0.82	0.82	0.91	0.89	0.96	0.96
	AR5	0.90	0.87	0.84	0.82	0.82	0.92	0.99
	AR6	0.96	0.76	0.79	0.99	0.96	0.99	0.99
	MED	0.96	0.82	0.82	0.93	0.89	0.96	**0.97**
NASA	CM1	0.76	0.70	0.48	0.81	0.71	0.84	0.87
	MW1	0.82	0.77	0.58	0.89	0.87	0.89	0.89
	PC1	0.93	0.93	0.75	0.96	0.94	0.97	0.97
	PC3	0.92	0.83	0.89	0.95	0.94	0.96	0.96
	PC4	0.97	0.85	0.79	0.97	0.96	0.99	0.99
	MED	0.92	0.83	0.75	0.95	0.94	**0.96**	**0.96**
MORPH	Ant_1.3	0.91	0.88	0.80	0.89	0.85	0.91	0.91
	Arc	0.91	0.76	0.73	0.98	1.00	0.98	0.98
	Camel_1.0	0.91	0.83	0.73	0.89	0.87	0.94	0.94
	Poi_1.5	0.89	0.82	0.63	0.90	0.86	0.91	0.96
	Redaktor	0.89	0.74	0.49	0.99	0.99	0.99	0.99
	Skarbonka	0.86	0.71	0.71	0.96	0.96	0.98	0.98
	Tomcat	0.88	0.87	0.87	0.91	0.90	0.92	0.95
	Velocity_1.4	0.96	0.90	0.89	0.94	0.93	0.98	0.98
	Xalan_2.4	0.82	0.82	0.77	0.86	0.83	0.88	0.88
	Xerces_1.2	0.87	0.76	0.75	0.86	0.84	0.88	0.89
	MED	0.89	0.82	0.74	0.91	0.88	0.93	**0.96**
ReLink	Apache	0.84	0.73	0.57	0.89	0.89	0.90	0.90
	Safe	0.85	0.82	0.84	0.88	0.85	0.89	0.92
	Zxing	0.69	0.63	0.63	0.69	0.57	0.71	0.71
	MED	0.84	0.73	0.63	0.88	0.85	0.89	**0.90**

TABLE 3.9: Wilcoxon signed-rank test analysis (p<0.05): Mean difference (Column-Row) between the F-measure of different approaches and p-value. Mean difference: Left bottom triangle; p-value: Right upper triangle

	F-measure						
	NB	SVM	KNN	RF	C50	SMV	WMV
NB	_	0.00	0.00	0.01	0.98	0.00	0.00
SVM	0.07	_	0.00	0.00	0.00	0.00	0.00
KNN	0.13	0.07	_	0.00	0.00	0.00	0.00
RF	-0.02	-0.09	-0.15	_	0.00	0.00	0.00
C50	0.00	-0.07	-0.13	0.02	_	0.00	0.00
SMV	-0.04	-0.11	-0.18	-0.02	-0.04	_	0.02
WMV	**-0.05**	**-0.12**	**-0.18**	**-0.03**	**-0.05**	**-0.01**	_

3.4.3 Results with Respect to MCC

The results of different algorithms in terms of MCC are presented in Table 3.10. The MCC performance of WMV is better than all other BCs and SMV in 4 out of 5 groups of DSs. WMV outperforms SMV in all groups of DSs. The MCC value of NB is highest in SOFTLAB group of DSs. SMV is not performing superior in any group of DSs.

From Table 3.11, we determine that WMV performs better than all BCs and SMV and shows significantly different performance (in terms of MCC) due to a p-value less than 0.05 (shown in bold). The performance of C50 and RF, and C50 and NB are the same based on p-value (>0.05). However, the performance of SMV outperforms only SVM and KNN.

As shown in Fig. 3.2c, the performance of WMV is better than all BCs and SMV. While the performance of SMV is lower than RF and C50.

From Fig. 3.3c, the analysis reveals that KNN falls into the low-performing group, whereas WMV, along with SMV, NB, RF, and C50, belongs to the high-performing group. Meanwhile, SVM is situated on the boundary between the low and high-performing groups, making it challenging to determine its performance

TABLE 3.10: Performance of different SBP techniques with respect to MCC

Groups	Projects	MCC						
		NB	SVM	KNN	RF	C50	SMV	WMV
AEEEM	Equinox	0.85	0.87	0.76	0.88	0.87	0.88	0.89
	JDT	0.94	0.92	0.90	0.96	0.97	0.95	0.97
	Lucene	0.89	0.85	0.75	0.91	0.92	0.90	0.93
	Mylyn	0.79	0.84	0.40	0.82	0.83	0.80	0.84
	PDE	0.93	0.89	0.75	0.98	0.99	0.92	0.99
	MED	0.89	0.87	0.75	0.91	0.92	0.90	**0.93**
SOFTLAB	AR1	0.90	0.64	0.44	0.97	0.98	0.82	0.93
	AR3	0.97	0.84	0.72	0.88	0.64	0.88	0.88
	AR4	0.93	0.76	0.68	0.87	0.87	0.91	0.93
	AR5	1.00	0.76	0.60	0.60	0.78	0.85	0.84
	AR6	0.94	0.56	0.63	1.00	0.96	0.82	0.98
	MED	**0.94**	0.76	0.63	0.88	0.87	0.85	0.93
NASA	CM1	0.57	0.55	0.10	0.66	0.67	0.57	0.68
	MW1	0.65	0.62	0.22	0.84	0.81	0.66	0.89
	PC1	0.91	0.94	0.48	0.94	0.95	0.92	0.97
	PC3	0.89	0.79	0.82	0.91	0.92	0.91	0.93
	PC4	0.95	0.86	0.57	0.95	0.96	0.93	0.97
	MED	0.89	0.79	0.48	0.91	0.92	0.91	**0.93**
MORPH	Ant_1.3	0.84	0.79	0.67	0.84	0.84	0.84	0.83
	Arc	0.96	0.60	0.47	0.99	0.99	0.83	0.99
	Camel_1.0	0.84	0.75	0.45	0.83	0.85	0.82	0.88
	Poi_1.5	-0.80	-0.70	-0.46	-0.83	-0.82	-0.73	-0.73
	Redaktor	0.97	0.51	0.18	1.00	0.99	0.76	0.99
	Skarbonka	0.91	-0.12	0.25	0.95	0.91	0.42	0.97
	Tomcat	0.81	0.75	0.76	0.84	0.87	0.77	0.88
	Velocity_1.4	-0.93	-0.82	-0.81	-0.93	-0.91	-0.95	-0.96
	Xalan_2.4	0.70	0.69	0.53	0.75	0.76	0.71	0.75
	Xerces_1.2	0.77	0.60	0.50	0.77	0.76	0.69	0.80
	MED	0.82	0.60	0.46	0.83	0.84	0.74	**0.86**
ReLink	Apache	-0.71	-0.53	-0.21	-0.82	-0.84	-0.65	-0.69
	Safe	0.63	0.59	0.58	0.66	0.70	0.59	0.70
	Zxing	0.39	0.38	0.19	0.43	0.26	0.38	0.42
	MED	0.39	0.38	0.19	**0.43**	0.26	0.38	0.42

TABLE 3.11: Wilcoxon signed-rank test analysis (p<0.05): Mean difference (Column-Row) between the MCC of different approaches and p-value. Mean difference: Left bottom triangle; p-value: Right upper triangle

	MCC						
	NB	SVM	KNN	RF	C50	SMV	WMV
NB	_	0.01	0.00	0.09	0.16	0.03	0.00
SVM	0.12	_	0.01	0.01	0.01	0.00	0.00
KNN	0.23	0.12	_	0.00	0.00	0.00	0.00
RF	-0.01	-0.12	-0.24	_	0.68	0.02	0.00
C50	0.00	-0.12	-0.23	0.01	_	0.05	0.00
SMV	0.05	-0.07	-0.19	0.05	0.05	_	0.00
WMV	**-0.03**	**-0.15**	**-0.27**	**-0.03**	**-0.03**	**-0.08**	_

conclusively. However, it is evident that WMV outperforms all other BCs and SMV.

3.4.4 Response of Research Questions

RQ1: Are the results obtained by WMV technique comparable to those obtained by the BCs independently?

Response of RQ1: From the outcome shown in Tables 3.6, 3.8, 3.10 and Fig. 3.2 and 3.3 and also from the above discussion, we can conclude that WMV outperforms the BCs. The proposed reward-based ensemble learning approach WMV performs significantly better than the BCs in terms of performance measures viz. accuracy, FM, and MCC.

RQ2: Are the results obtained by WMV technique comparable to those of the existing SMV ensemble-based SOTA technique?

Response of RQ2: Based on the information presented in Tables 3.7, 3.9, 3.11, Fig. 3.2, Fig. 3.3, and the aforesaid discussion, it can be inferred that WMV's results are significantly superior to that of SMV. In general, WMV exhibits comparable or superior performance with respect to accuracy and FM while demonstrating significantly better results with respect to the more rigorous performance parameter, namely MCC.

RQ3: Are the results obtained by WMV technique comparable to those obtained by best-performing SOTA advanced techniques?
Response of RQ3: To compare the results of WMV with the results of the SOTA techniques, we implemented our approach with the same K-FCV on the same datasets (DSs) that are used in ten selected recent papers (2017 to 2022), and the result on each DS is shown in Table 3.12. The detailed insight of these papers is given in the related work Section 2.1.1 of Chapter 2. The performance value of state-of-the-art techniques and WMV on all the DSs is presented in Table 3.12 and the summary of Table 3.12 is given in Table 3.13. We found that FM is a common metric in all the techniques, so the Win-Tie-Loss value of WMV, when compared to the existing techniques, in terms of FM is shown in Table 3.13.

We have also applied Wilcoxon signed-rank test between WMV and existing techniques to evaluate the p-value. Table 3.13 concludes that WMV performs better than all existing techniques (except STACKING and LSSVM-P) based on mean difference (Mean diff.) performance. The mean difference between WMV and BPDET is 0.06 and the p-value is 0.08, which implies that they are not significantly different. However, on the basis of the p-value, WMV performs significantly better than majority of existing techniques (shown in bold).

The pairwise boxplot representation of the proposed approach WMV (blue boxplots) as compared to other recent methods (gray boxplots) is shown in Fig. 3.4. The best model should have a high median value and few outliers (small circles) for SBP. Fig. 3.4 infers that WMV is better (in terms of FM) than all the existing methods except STACKING and LSSVM-P.

TABLE 3.12: Results of state-of-the-art (SOTA) techniques and WMV on each dataset in terms of FM

Dataset	NDTF	WMV	Dataset	SDAEs-TSE	WMV	Dataset	LTSA-SVM	WMV	Dataset	DPDF	WMV	Dataset	WNB-ID	WMV	DSs	STAC-KING	WMV	Dataset	LSSV-M-P	WMV
ant-1.3	0.94	0.91	CM1	0.29	0.87	CM1	0.90	0.87	JW1	0.23	0.85	Ivy-1.1	0.79	0.84	PC1	0.96	0.97	AR1	1.00	0.97
ant-1.4	0.89	0.84	KC1	0.39	0.85	KC3	0.89	0.88	MC1	0.04	0.85	Lucen-2.2	0.74	0.91	CM1	0.95	0.87	AR3	0.98	0.95
ant-1.5	0.96	0.94	KC2	0.52	0.85	PC1	0.88	0.97	MC2	0.48	0.91	Lucen-2.4	0.79	0.96	KC1	0.97	0.85	AR4	0.99	0.96
ant-1.6	0.88	0.93	KC3	0.38	0.88	PC4	0.92	0.99	MW1	0.51	0.89	Poi-1.5	0.84	0.82	KC2	0.94	0.85	AR5	1.00	0.99
ant-1.7	0.90	0.91	MC1	0.22	0.85	MC2	0.83	0.91	PC1	0.17	0.97	Poi-2.5	0.87	0.85				AR6	0.97	0.99
arc	0.94	0.98	MC2	0.59	0.91	MW1	0.78	0.89	PC2	0.83	0.94	Veloc.-1.6	0.87	0.94	DSs	BPDET	WMV	CM1	0.96	0.87
berek	0.94	0.98	MW1	0.41	0.89				PC3	0.11	0.96	Xalan-2.6	0.77	0.83	CM1	0.85	0.87	KC1	0.93	0.85
camel-1.0	0.99	0.94	PC1	0.31	0.97	Dataset	NDTF	WMV	PC4	0.33	0.99	CM1	0.39	0.87	JM1	0.76	0.97	KC2	0.91	0.85
camel-1.2	0.78	0.80	PC2	0.22	0.94	skarbonka	0.88	0.98	PC5	0.46	0.95	MW1	0.46	0.89	KC1	0.83	0.85	KC3	0.99	0.88
camel-1.4	0.20	0.89	PC3	0.35	0.96	sklebagd	0.67	0.97	xalan-2.6	0.72	0.83	PC4	0.56	0.99	KC2	0.82	0.85	MC2	0.95	0.91
camel-1.6	0.89	0.87	PC4	0.55	0.99	synapse-1.0	0.98	0.79	ant-1.7	0.55	0.91				KC3	0.76	0.88	MW1	0.96	0.89
e-learning	0.97	0.94	JM1	0.32	0.97	synapse-1.1	0.89	0.76	camel-1.6	0.19	0.87	Dataset	VOT	WMV	MC1	0.97	0.85	PC1	0.98	0.97
intercafe	1.00	0.82				synapse-1.2	0.80	0.92	jedit-4.0	0.46	0.89	AR5	0.86	0.99	MC2	0.68	0.91	PC2	1.00	0.94
ivy-1.1	0.70	0.84	Dataset	NDTF	WMV	systemdata	0.97	0.93	log4j-1.0	0.48	0.92	AR6	0.82	0.99	PC1	0.93	0.97	PC3	0.97	0.96
ivy-1.4	0.98	0.97	lucen-2.2	0.51	0.91	szybkafucha	0.61	0.98	lucen-2.4	0.75	0.96	CM1	0.85	0.87	MW1	0.91	0.89	PC4	0.94	0.99
ivy-2.0	0.96	0.88	lucen-2.4	0.62	0.96	termoproj.	0.90	0.92	poi-3.0	0.83	0.90	JM1	0.77	0.97	PC2	0.99	0.94			
jedit-3.2	0.89	0.90	nierucho.	0.85	0.92	tomcat	0.96	0.95	tomcat	0.21	0.95	KC1	0.92	0.85	PC3	0.86	0.96	Dataset	SPE2	WMV
jedit-4.0	0.90	0.89	pdftransl.	0.84	0.95	velocity-1.5	0.61	0.97	LC	0.37	0.97	MC2	0.72	0.91	PC4	0.89	0.99	ant-1.7	0.60	0.91
jedit-4.1	0.90	0.97	prop-1	0.92	0.86	velocity-1.6	0.79	0.94	JDT	0.56	0.98	PC1	0.91	0.97				arc	0.33	0.98
jedit-4.2	0.95	0.94	prop-2	0.95	0.92	workflow	0.73	0.65	PDE	0.31	0.99	PC3	0.88	0.96				ivy-2.0	0.41	0.88
jedit-4.3	0.99	0.85	prop-3	0.94	0.95	wspomag.	1.00	0.87	EQ	0.75	0.98	intercafe	0.86	0.82				jedit-4.2	0.48	0.94
kalkulator	0.90	1.00	prop-4	0.95	0.94	xerces-1.2	0.92	0.89	ML	0.26	0.96	ivy-2.0	0.87	0.88				poi-2.0	0.38	0.96
log4j-1.0	0.91	0.92	prop-5	0.92	0.97	xerces-1.3	0.94	0.94	Apache	0.73	0.90	jedit-4.3	0.71	0.85				synp-1.1	0.60	0.76
log4j-1.1	0.88	0.91	prop-6	0.95	0.99	xerces-1.4	0.78	0.96	Safe	0.56	0.92	lucene-2.4	0.69	0.96				tomcat	0.40	0.95
log4j-1.2	0.77	1.00	redaktor	0.97	0.91	xerces-init	0.83	0.97	Zxing	0.29	0.71	serapion	0.72	0.87				velo.-1.6	0.63	0.94
lucene-2.0	0.70	0.94	serapion	0.93	0.87	zuzel	0.80	0.97				tomcat	0.89	0.95				xalan-2.4	0.46	0.88
												workflow	0.62	0.65				xerces-1.3	0.49	0.95

From Table 3.12, Table 3.13, and Fig. 3.4 we can conclude the following results:

- The average performance of WMV over 56 DSs using 10-FCV shows 0.92 FM and 93.75% accuracy (ACC), which is greater (0.06 FM) than the average performance of the existing approach Non-Linear Ensemble Decision Tree Forest **(NDTF)** which is showing 0.86 FM and 84.81% accuracy [55]. But, based on Win-Tie-Loss value, performance of WMV is better on 31 DSs, ties on 1 DS, and worse on 24 DSs as compared to NDTF.

- The performance of WMV (0.91 FM, 0.87 MCC), compared to Stacked Denoising Autoencoders Two-Stage Ensemble **(SDAEsTSE)** (0.38 FM, 0.27 MCC), shows a major increment of 0.53 FM over 12 NASA DSs. Based on Win-Tie-Loss, WMV performance wins over all DSs in terms of FM.

- The performance of WMV (0.92 FM, 92.38 ACC, 0.92 precision (PRS), 0.92 recall (RCL)), compared to Local Tangent Space Alignment Support Vector Machine **(LTSA-SVM)** (0.87 FM, 91.07 ACC, 0.84 PRS, 0.90 RCL), shows an increment of 0.05 FM over 6 NASA DSs using 10 FCV. Based on Win-Tie-Loss value, WMV wins on 4 DSs, ties on 1 DS, and gives worse result on 1 DS compared to LTSA-SVM.

- The performance of WMV (0.92 FM), compared to deep forest model to build the defect prediction model **(DPDF)** (0.45 FM) that uses deep learning and ensemble learning, shows a mean increment of 0.47 FM and win on all DSs. WMV outperforms DPDF in terms of all metrics.

- The performance of WMV (0.89 FM), compared to weighted Naive Bayes method based on information diffusion **(WNB-ID)** (0.71 FM), shows an increment of 0.18 FM and wins on 8 DSs. WMV outperforms WNB-ID in terms of FM, PRS, and RCL.

- The performance of WMV (0.90 FM), compared to majority voting **(VOT)** (0.81 FM) that combines two or more techniques, shows a mean increment of 0.09 FM. WMV wins on 13 DSs and loses on 2 NASA DSs.

- The performance of WMV, compared to Bug Prediction using Deep representation and ensemble learning **(BPDET)**, shows an increment of 0.06 FM and 0.59 MCC. WMV obtains better FM on 9 DSs, and worse on 3 DSs.

- The performance of WMV, compared to stacking ensemble **(STACKING)**, shows a decrement of 0.07 FM, also worse in terms of FM, ACC, PRS, and RCL. But based on Win-Tie-Loss, WMV is better on 1 DSs and worse on 3 DSs.

- The performance of WMV, compared to least square support vector machine using poly kernel **(LSSVM-P)**, shows a decrement of 0.04 FM. LSSVM-P wins in 14 DSs, worse in 1 DSs compared to WMV. Overall, LSSVM-P is providing the best results on these 15 DSs.

- WMV shows an increment of 0.44 mean FM and wins higher results in all DSs. WMV also outperforms Self-Paced Ensemble of Ensembles **(SPE2)** in terms of precision (PRS) and recall (RCL).

Overall, from the result shown in Table 3.12, 3.13 and Fig. 3.4 and also from the above conclusion, we can deduce that WMV outperforms the majority of SOTA techniques. There is no significant difference between WMV to LTSA-SVM, BPDET, and STACKING based on the p-value.

TABLE 3.13: Average performance in terms of different metrics, Win-Tie-Loss value of WMV as compared to SOTA techniques in terms of FM over all datasets, Mean_Difference (WMV-SOTA), and p-value

Existing SOTA	NDTF [55]	SDAEsTSE [45]	LTSA-SVM [146]	DPDF [37]	WNB-ID [136]	VOT [149]	BPDET [49]	STACKING [148]	LSSVM-P [145]	SPE2 [150]
Exist, WMV (FM)	0.86, **0.92**	0.38, **0.91**	0.87, **0.92**	0.45, **0.92**	0.71, **0.89**	0.81, **0.90**	0.85, **0.91**	**0.95**, 0.89	**0.97**, 0.93	0.48, **0.92**
Exist, WMV (ACC)	84.81, **93.75**	-	91.07, **92.38**	83.23, **92.76**	-	-	-	**95.83**, 92.34	**94.28**, 93.45	-
Exist, WMV (MCC)	-	0.27, **0.87**	-	-	-	-	0.29, **0.88**	-	-	-
Exist, WMV (PRS)	-	-	0.84, **0.92**	0.62, **0.94**	0.65, **0.91**	-	-	**0.95**, 0.88	-	0.38, **0.91**
Exist, WMV (RCL)	-	-	0.90, **0.92**	0.39, **0.92**	0.79, **0.89**	-	-	**0.97**, 0.91	-	0.70, **0.94**
#DS, K-FCV	56, 10	12, 5	6, 10	25, 2	10, 10	15, 10	12, 10	4, 10	15, 10	10, 5
Win-Tie-Loss (FM)	31-1-24	12-0-0	4-1-1	25-0-0	8-0-2	13-0-2	9-0-3	1-0-3	1-0-14	10-0-0
Mean_Diff. (FM)	**0.06**	**0.53**	0.05	**0.47**	**0.18**	**0.09**	0.06	-0.07	**-0.04**	**0.44**
p-value (FM)	0.01	0.00	0.08	0.00	0.02	0.00	0.08	0.09	0.00	0.00

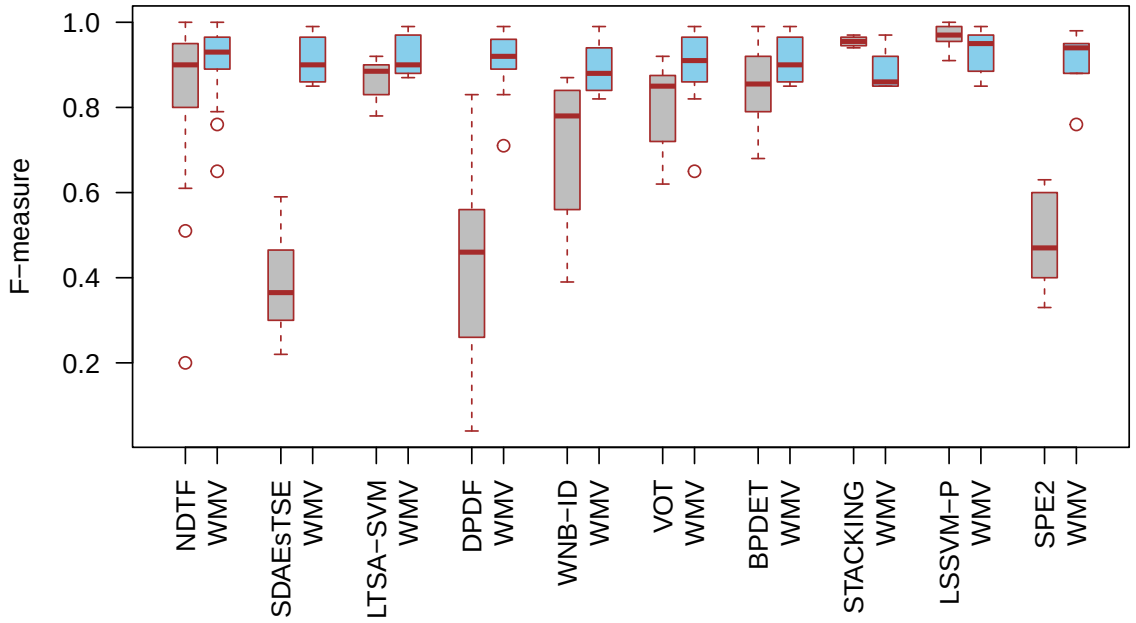

FIGURE 3.4: Pairwise boxplot comparison of different recent approaches with the proposed approach WMV

RQ4: Is the ROSE sampling technique better than SMOTE and RUS for WMV? Answer of RQ4: We have implemented WMV with ROSE, SMOTE, and RUS. We can conclude that WMV with ROSE is a better ensemble technique than WMV with SMOTE and WMV with RUS. WMV with ROSE produces better median results with 96.18% accuracy, 0.96 FM, and 0.89 MCC over all DSs. These results of WMV with ROSE show an increment of 4.8%, 0.05, and 0.06 as compared to WMV with SMOTE (91.38%, 0.91, 0.83) and an increment of 22.37%, 0.22, and 0.42 as compared to WMV with RUS (73.81%, 0.74, 0.47) in terms of median accuracy, FM, and MCC respectively.

Overall, the prediction results of SMV and WMV are significantly different, as presented in Fig. 3.2 and Fig. 3.3. The performance difference between SMV and WMV is considerable, with mean differences of 0.87% (accuracy), 0.01 (FM), and 0.08 (MCC). Additionally, the mean score differences between SMV and WMV are also notable, such as 6.0 vs. 6.46 (accuracy), 6.14 vs. 6.39 (FM), and 4.93 vs. 5.93 (MCC). It is evident that the mean score of WMV is significantly higher

than that of SMV, indicating a clear superiority of WMV over SMV. In WMV, a weight multiplier is required for SMV with the predicted probability to combine the predicted results of the BCs based on majority voting. SMV is a simple and straightforward ensemble as compared to WMV because it provides each BC with equal weights. Therefore, performance of SMV is worse in all the groups of DSs as compared to WMV. However, SMV performs equal to the WMV in terms of accuracy (96.10%) and FM (0.96) across NASA group of DSs, as shown in Tables 3.6, 3.8, 3.10. But, in case of MCC, WMV completely outperforms SMV in all groups of DSs. As a result, SMV is faster than WMV because it does not require weight calculation. For real-life situations, if a quick response is required, SMV should be used; if higher performance is required, WMV should be used.

Overall, in general, WMV performs significantly better and/or sometimes even equal to the BCs (RQ1), SMV (RQ2), and recent advanced techniques (RQ3). It should be noted that SMV does not need BCs to be weighted, whereas WMV does. As a consequence, WMV achieves significantly improved prediction results as compared to most of the state-of-the-art (SOTA) methods across various performance measures, including accuracy, FM, MCC, precision, and recall (Table 3.13). We have conducted Cohen's D test to calculate the effect size between WMV and other SBP models and results shown in Table B.6. We have implemented a cross-version SBP model of WMV and other baseline models. This means, we have trained these models on a version of a software project and tested them on the current version of the software project. The cross-version performance of different SBP models over 13 software projects is shown in Table B.9.

3.4.5 Result Discussion

The proposed approach, WMV is more effective than SMV. WMV also performs better than BCs used independently, and the best results were selected from the recent papers. The results produced by the proposed approach are completely dependent on the performance of the BCs used. We have chosen the BCs of different

natures, i.e., we have used different approaches to solve the problem. For example, Naive Bayes is conditional probability-based, KNN is distance-based, SVM is hyper-parameter-based, RF is based on the bagging ensemble technique, and C50 is based on the boosting ensemble technique. These BCs produce different results on the same DSs. However, combining them in weighted majority voting enhances the overall performance of WMV technique. The calculation of the weight of each BC effectively enhances the performance of WMV. Therefore, we can conclude that selecting the BCs and calculating the weight of each BC justifies the effective performance of the WMV. The performance of WMV is also dependent on an optimal value of decision threshold θ_k using (3.7). Hence, choosing the value of θ_k is an optimization problem and can be further tuned to analyze the performance of the WMV using different values of θ_k in the future.

Based on the answer given to the RQ3, we infer that WMV outperforms the majority of SOTA techniques proposed in the last five years. We conclude that WMV is performing better than all the existing techniques except LSSVM-P and STACKING. Win-Tie-Loss values in terms of common metric FM clearly indicate the comparative performance of WMV. We can infer that WMV performs greater on all DSs for SDAEsTSE, DPDF, and SPE2 and worse on some DS used in NDTF, LTSA-SVM, WNB-ID, VOT, BPDET, and STACKING. There are some datasets where WMV and SOTA techniques (NDTF, LTSA-SVM) perform equally up to two decimal places (Table 3.12, 3.13).

3.5 Threats to Validity

The proposed WMV approach may be associated with the following risks to validate the proposed SBP technique.

Internal validity: The WMV method was implemented using a total of 28 software project DSs collected from the public repository. So, the first risk may be

the consistency of the collected DSs. We have collected these DSs with utmost care and consistency, but we can not assert that the collected DSs are 100% accurate.

Construct validity: The proposed WMV only deals with binary (buggy or not buggy) classification problems. It does not consider predicting a precise bug count in each module of the software project. So, this limitation poses a risk in validating the WMV technique. BCs used in the WMV were developed with the default settings of the library used.

External validity: The SBP methods have been applied at different levels of granularity, including class, function, and file. However, it is believed that the proposed methodology can also be reliable for other programming paradigms. The WMV is developed using a given set of software metrics specific to a software project.

3.6 Conclusion and Future Work

It is difficult to develop a more reliable and universal SBP technique for newly developed software projects with conventional machine learning (ML) algorithms independently. To address the shortcomings of existing techniques, we proposed a reward-based majority voting ensemble approach WMV based on combining results of conventional ML algorithms. The tabular, boxplot, critical diagram, and statistical results show that WMV has a significant potential for SBP. Our experimental results show that WMV outperforms SMV, with 0.87% greater accuracy, 0.01 greater FM, and 0.08 greater MCC. The performance of WMV beats that of BCs with 2.62% to 19.53% higher accuracy, 0.05 to 0.18 higher FM and 0.03 to 0.27 higher MCC. The results of WMV and existing SOTA techniques, shown in Table 3.13 in terms of accuracy, FM, recall, precision, and Win-Tie-Loss, conclude that WMV is performing superior to all these techniques except LSSVM-P and STACKING. WMV shows an average FM gain of 0.05 to 0.53 as compared to SOTA techniques (Table 3.13). Boxplot (Fig. 3.4) clearly shows the better performance of WMV over the majority of the SOTA techniques.

Overall, we conclude that WMV is a more effective technique for developing SBP model over BCs, SMV, and the majority of SOTA techniques. Out of BCs, KNN is the lowest-performing algorithm, and RF is the highest-performing algorithm. Out of the SOTA techniques, LSSVM-P (0.97) and STACKING (0.95) are the highest-performing techniques in terms of mean FM. The lowest-performing existing techniques in terms of average FM are SDAEsTSE (0.38), DPDF (0.45), and SPE2 (0.48). The reported work can be useful for the research community and industry to build significantly effective SBP models. In the future, we plan to implement the proposed method on commercial datasets as part of our validation process in an industry setting.