SOME MACHINE LEARNING ASSISTED SOFTWARE BUG PREDICTION TECHNIQUES

The thesis submitted in partial fulfilment

for the Award of Degree

DOCTOR OF PHILOSOPHY

by

RAKESH KUMAR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY

(BANARAS HINDU UNIVERSITY)

VARANASI -221005

INDIA

Roll No.: 18071011 **Year: 2023**

CERTIFICATE

It is certified that the work contained in this thesis entitled "SOME MACHINE LEARNING ASSISTED SOFTWARE BUG PREDICTION TECHNIQUES" submitted by "RAKESH KUMAR" (Roll No.: 18071011) has been carried out under my supervision and that it has not been submitted elsewhere for a degree.

It is further certified that the student has fulfilled all the requirements of Comprehensive Examination, Candidacy and State-of-the-Art (SOTA) for the award of Ph.D. Degree.

Signature of Supervisor

Dr. AMRITA CHATURVEDI

Assistant Professor

Department of Computer Science and Engineering

Indian Institute of Technology

(Banaras Hindu University)

Varanasi-221005 , India

DECLARATION BY THE CANDIDATE

I, RAKESH KUMAR , certify that the work embodied in this thesis is my own bonafide work and carried out by me under the supervision of Dr. AMRITA CHATURVEDI from JULY-2018 to DECEMBER-2023, at the DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, Indian Institute of Technology (BHU) Varanasi. The matter embodied in this thesis has not been submitted for the award of any other degree/diploma. I declare that I have faithfully acknowledged and given credits to the research workers wherever their works have been cited in my work in this thesis. I further declare that I have not willfully lifted up any other's work, paragraphs, text, data, results, etc., reported in journals, books, magazines, reports, dissertations, theses, etc., or available on websites and have not included them in this thesis and have not cited as my own work.

Rakesh

Date : 22/12/2023

Place : Varanasi

Signature of the Student

(RAKESH KUMAR)

CERTIFICATE BY THE SUPERVISOR

It is certified that the above statement made by the student is correct to the best of my/our knowledge.

Amrita

Date : 22/12/2023

Place : Varanasi

Signature of Supervisor

(Dr. AMRITA CHATURVEDI)
Department of Computer Science and Engineering
Indian Institute of Technology
(Banaras Hindu University)
Varanasi-221005

Supervisor
गणक विज्ञान एवं अभियांत्रिकी विभाग
Department of Computer Sc. & Engg
भारतीय प्रौद्योगिकी संस्थान
Indian Institute of Technology
(काशी हिन्दू विश्वविद्यालय)
(Banaras Hindu University)
वाराणसी-221005
(U.P.)

Signature of Head of Department

(Prof. Sanjay Kumar Singh)

आचार्य व विभागाध्यक्ष
Professor & Head
गणक विज्ञान एवं अभियांत्रिकी विभाग
Department of Computer Sc. & Engg
भारतीय प्रौद्योगिकी संस्थान
Indian Institute of Technology
(काशी हिन्दू विश्वविद्यालय)
(Banaras Hindu University)

COPYRIGHT TRANSFER CERTIFICATE

Title of the Thesis: SOME MACHINE LEARNING ASSISTED SOFTWARE BUG PREDICTION TECHNIQUES

Name of the Student: RAKESH KUMAR

Copyright Transfer

The undersigned hereby assigns to the Indian Institute of Technology (Banaras Hindu University) Varanasi, all rights under copyright that may exist in and for the above thesis submitted for the award of the DOCTOR OF PHILOSOPHY.

Rakesh

Date : 22/12/2023

(RAKESH KUMAR)

Place : Varanasi

Note: However, the author may reproduce or authorize others to reproduce material extracted verbatim from the thesis or derivative of the thesis for author's personal use provided that the source and the Institute's copyright notice are indicated.

To

My Beloved Family

Friends, Teachers

and

Lord Shiva

ACKNOWLEDGEMENTS

"Knowledge is in the end based on acknowledgment." -Ludwig Wittgenstein

I want to take this opportunity to express my deep sense of gratitude to all who helped me directly or indirectly during this thesis work. Firstly, I would like to thank my supervisor, Dr. Amrita Chaturvedi, for being a great mentor and the best adviser I could ever have. Her advice, encouragement, and critics are a source of innovative ideas, inspiration, and causes behind the successful completion of this Thesis work. The confidence shown on me by her was the most significant source of inspiration for me. It has been a privilege working with her for several years. I am highly obliged to all the faculty members of the Computer Science and Engineering Department for their support and encouragement. I express my sincere thanks to RPEC members Dr. Lakshmanan Kailasam of the Department of Computer Science and Engineering and Dr. Ashok Ji Gupta, Department of Mathematical Sciences IIT (BHU), for providing continuous support, encouragement, and advice. I express my sincere thanks to all the Professors, office staff, supporting staff, and Ph.D. Research Scholars of Indian Institute of Technology (BHU) Varanasi India. I express my gratitude to Director, Registrars, Deans, Heads, and Student Alumni of the Indian Institute of Technology (BHU) Varanasi.

My memory of the study period at IIT (BHU) can never be complete without mentioning my fellow research scholars. Special thanks to Mr. Anurag Tiwari, Mr. Shashank Kumar Singh, Miss. Shruti Bajpai, Mr. Supriya Chanda, Ms. Naina Yadav, Manisha Singh, Miss. Divya Singh, Ms. Sheetal, Miss. Nitika Nigam, Mr. Sandeep Gautam, Mr. Shivansh Mishra, Mr. Ramakant Kumar, Dr. Santosh Tripathy, and Mr. Vinod Kumar for their great help and cooperation. I extend special thanks to the non-teaching staff in the department, particularly Mr. Ravi Kumar Bharti, Mr. Prakhar Kumar, and Mr. Shubham Pandey, for their consistent support.

My father, mother, and elder fathers/mothers, namely Mr. Umesh Yadav, Mr. Keshav Yadav, Mr. Vishnudhari Singh, Mr. Ram Niwas Singh, Ms. Sumitra Yadav, Ms. Surati Yadav, Ms. Savitri Yadav, and Lalmuni Yadav, who gave me the power and brain to work out on this research and their help at every level, made me see

this success. I owe thanks to my dearest wife, Ms. Archana Yadav for her continued and unfailing love, support and understanding during my pursuit of a Ph.D. degree that made the completion of my thesis possible. You all were always around at times when I thought that it is impossible to continue; you all helped me to keep things in perspective. I greatly value your contribution and deeply appreciate your belief in me.

Words are insufficient to express my profound sense of gratitude to my friends Rajkumar Yadav, Ashish Kumar, Sachin Mishra, Brijesh, Siddharth, Er. Satyendra Srivastava, Vikasanand, Dr. K.P. Yadav, Dr. R.H. Khan, Dr. Rajkumar Saini. My sisters and brothers Poonam Yadav, Sindhu, Bindu, Pooja, Anu, Tanu, Vimlesh, Chandrakesh, Krishan, Ajit, Akhilesh, Arjun, Kamlesh, Jitendra, Mukesh, Dinesh and Rajesh, Raju, whose encouragement gave me physical and moral strength throughout my career as well in the present research. I extend my thanks to my father/mother-in-law, Mr. Chhotak Yadav, Ms. Sarita Yadav, brother-in-law Harimohan, and daughter Gargi, who are part of my inspiration. Finally, I would like to wind up by paying my heartfelt thanks and prayers to the Almighty for their unbound love and grace.

- Rakesh Kumar

Contents

List of Figures

List of Tables

Abbreviations

ACL	**A**verage **C**lustering **L**abel
BCs	**B**ase **C**lassifiers
BRR	**B**ayesian **R**idge **R**egression
CLAMI	**C**lustering **L**abelling **M**etric selection **I**nstance selection
DTR	**D**ecision **T**ree **R**egression
DSs	**D**atasets
FCV	**F**old **C**ross **V**alidation
FM	**F**-**M**easure
FP	**F**unctional **P**aradigm
GA	**G**enetic **A**lgorithms
HC	**H**ierarchical **C**lustering
IVR	**I**nstance **V**iolation **R**atio
IVS	**I**nstance **V**iolation **S**core
ITV	**I**nstances **T**hreshold **V**iolation
KMS	**K**-means
LM	**L**inear **R**egression
MAE	**M**ean **A**bsolute **E**rror
MBC	**M**odel **B**ased **C**lustering
MCC	**M**atthew's **C**orrelation **C**oefficient
ML	**M**achine **L**earning
MLP	**M**ulti **L**ayer **P**erceptron
MTB	**M**etric selection **T**hreshold derivation **B**ug count vector

MTBP	**M**TB **P**lus
MTV	**M**etric **T**hreshold **V**iolation
MRE	**M**ean **R**elative **E**rror
MVS	**M**etric **V**iolation **S**core
NBR	**N**egative **B**inomial **R**egression
NGC	**N**eural **G**as **C**lustering
NB	**N**aive **B**ayes
OOP	**O**bject-**O**riented **P**aradigm
Pred(l)_Error	**P**rediction at level l Error
RD	**R**educed **D**ataset
RF	**R**andom **F**orest
ROSE	**R**andom **O**versampling **E**xamples
RQ	**R**esearch **Q**uestion
RUS	**R**andom **U**ndersampling
SD	**S**tandard **D**eviation
SM	**S**oftware **M**etrics
SMOTE	**S**ynthetic **M**inority **O**versampling **T**echnique
SMV	**S**imple **M**ajority **V**oting
SBCV	**S**oftware **B**ug **C**ount **V**ector
S[D, B, F]P	**S**oftware [**D**efect, **B**ug, **F**ault] **P**rediction
TCL	**T**hreshold **C**lustering **L**abelling
TCLP	**TCL P**lus
UMS	**U**nsupervised **M**etric **S**election
UMV	**U**nsupervised **M**ajority **V**oting

Symbols

ω_k	Weight of k^{th} classifier
η	Number of BCs
n	Number of instances
$p_{i,k}$	Prediction probability for i^{th} instance by k^{th} classifier
ϕ	BC function
m	Number of metrics
$\hat{y}_{i,k}$	Prediction label vector
$\delta_{i,SMV}$	Rresultant label vector using SMV method
i	For each instance i
β_i	Ratio of the number of wrong BC (Z) and η
Z_i	Number of wrong predictors for i^{th} instance
$\omega_{i,k}$	Weight of k^{th} BC after predicting label of the i^{th} instance
$\mathcal{P}_{i,k}$	Weighted probability
θ_k	Threshold
$\lambda_{i,k}$	Weighted label
$\delta_{i,WMV}$	Predicted class label using WMV
$ln(x)$	Log function
μ_m	Mean
Ω_m	Standard deviation
$T_m^{'}$	Symbolic metric threshold
T_m	Original threshold

$exp()$	Exponential function
X_k	Metric vector
δ_i	Minimum Euclidean distance
Ψ_i	Coverage measure
$\overline{\delta}$	Mean of δ_i
Φ	Effect size magnitude
γ_m	Skewness index
C	Number of clusters
U	n number of data points
$S(i)$	Silhouette coefficient

PREFACE

Software is now a part of every aspect of our daily lives. But, when we design, develop, and configure software, bugs are inevitable. These hidden bugs in the code snippet make the software insecure and unreliable. So, it's crucial to find and fix these bugs before we deliver the software to the end user. However, because software is getting larger and more complex, it's becoming a challenging task for software developers and testers to improve the software quality.

Software bugs occur when software doesn't meet the requirements provided in the software requirement specification (SRS) or end users' expectations (even if they aren't explicitly stated in SRS but are reasonable). A software bug prediction (SBP) model predicts bugs in each module of a software system. Once we predict where the bugs might be, the software quality assurance (SQA) team can optimally assign limited testing resources. This means more testers are required for the modules that have a high probability of containing bugs. This approach helps to save time and money by reducing testing efforts and software development costs.

Many software researchers have developed several SBP techniques to predict modules as buggy or non-buggy, utilizing statistical techniques, machine/deep learning, and ensemble learning approaches. These SBP techniques have many challenges, viz. missing values, data redundancy, irrelevant features, correlations, and class imbalance problems. The skewed distributions of software metrics in datasets may lead to issues like overfitting and dataset quality concerns.

The central aim of this thesis is to illuminate the advantages of software metrics (SMs) selection, machine learning (ML), or ensemble learning (EL) in predicting buggy/ non-buggy modules or bug counts in each software module using supervised and unsupervised SBP approaches. It also strives to mitigate class imbalance and

overfitting problems to produce unbiased results. Furthermore, building the supervised SBP model needs labeled datasets, so an alternate, an unsupervised SBP model is also proposed. Additionally, predicting only buggy and non-buggy modules may not help to categorize the modules and can not provide bug count information. So, to overcome these, an unsupervised bug count vector prediction model is proposed. The aforementioned supervised/unsupervised SBP models are employed on the object-oriented paradigm (OOP) or imperative programming datasets. These datasets are created from JAVA, C, and C++ software projects and are publicly available to software researchers. Additionally, we could not find any software bug dataset for Haskell and functional paradigm (FP). Therefore, to explore FP, four novel FP bug datasets are created using Haskell, which is an FP language. Further, a new classification-based unsupervised SBP model is proposed to be employed on novel FP datasets.

Based on the aforementioned discussion, the proposed work in this thesis is categorized into four contributing chapters. Chapter 3 adopts a classification-based supervised SBP approach. Then, in Chapter 4, we present a classification-based unsupervised SBP approach. Furthermore, in Chapter 5, we propose a regression-based unsupervised SBP approach to predict bug count vectors in each module. Additionally, Chapter 6 describes the creation of novel FP datasets and a novel classification-based unsupervised SBP model employed on FP datasets. The primary focus of this thesis is to explore the applications of ML techniques to address the specific challenges encountered in these four different SBP scenarios. The ultimate goal of this thesis is to enhance the overall performance of the existing state-of-the-art SBP models. The proposed four SBP approaches in this thesis are summarized as follows:

In order to enhance the performance of supervised ML/ ensemble learning SBP techniques and handle data imbalance problems, we present a novel classification-based SBP approach called reward-based weighted majority voting (WMV). In WMV, each base classifier's (BC's) performance is assessed, and a reward-based

mechanism is used to assign different weights to each classifier. BCs that make correct predictions receive rewards, and there are no penalties for wrong predictions. BCs that correctly predict more instances get higher weights in the ensemble. WMV surpasses the performance of the mentioned BCs, simple majority voting (SMV), and many recent state-of-the-art techniques in terms of accuracy, F-Measure, and Matthew's correlation coefficient (MCC).

The previously proposed SBP technique WMV employs on labeled datasets. To overcome the need for labeled datasets, handle the imbalanced datasets, and enhance the performance of unsupervised ML SBP techniques, we've developed a novel classification-based SBP approach called TCL/TCLP (Threshold Clustering Labelling/Threshold Clustering Labelling Plus). TCL/TCLP doesn't rely on the labeled dataset. This approach is based on the existing CLAMI technique but enhances it using logarithmic transformation and deriving metric thresholds. TCL labels instances into binary classes (buggy or non-buggy) based on the corresponding threshold of software metrics (SMs). TCLP extends TCL one step further by performing metrics/instances selection and ML training using a random forest algorithm to build an SBP model. Our empirical evaluation on 28 datasets from five software groups, varying in metrics and granularity, demonstrates that this unsupervised SBP method outperforms state-of-the-art techniques. It significantly improves accuracy, F-measure, and MCC compared to the majority of existing approaches.

The above proposed SBP technique WMV and TCL/TCLP can predict only binary classes, viz. buggy or non-buggy. So, to extend this study, we have proposed a Software Bug Count Vector (SBCV) prediction technique. It is a regression model that focuses on predicting the precise number of bugs in each module of a software system, which can help to optimize resource allocation and software maintenance. Most of the previous researches have relied on labeled datasets to predict SBCV using regression algorithms, but collecting and labeling the datasets are challenging tasks. To address this issue and the lack of an unsupervised regression model for bug count prediction, we introduce a novel unsupervised regression model (SBCV). Our

approach predicts SBCV on unlabeled data by determining independent thresholds for each software metric. Our method outperforms many standard supervised algorithms on 22 datasets in terms of mean absolute error (MAE), mean relative error (MRE), and Pred(l)_Error. Considered statistical tests confirm the significance of our approach.

Finally, the above-proposed techniques are employed only on OOP datasets. So, to overcome this issue, we explore SBP for Haskell, a functional programming language that hasn't been extensively investigated before. Our first objective is to create novel datasets by extracting source code metrics from Haskell functions. As the second objective, we utilize statistical analysis to determine the right-skewed nature of these metrics and hence apply three transformation techniques to reduce the skewness and calculate thresholds. Then, we propose an unsupervised majority voting (UMV) ensemble method to develop a classification-based SBP model and compare its performance. Our experimental results show that UMV performs well in Haskell, with high accuracy, f-measure, and MCC. This approach can be valuable for predicting software reliability in the absence of a labeled dataset.

In conclusion, this thesis addresses challenges in various SBP scenarios and proposes new approaches to enhance prediction performance using ML techniques. It extends the use of ML in software engineering and offers innovative solutions for SBP that benefit SQA activities.