BASIC CONCEPTS OF ORGANIC CHEMISTRY

Semester I

This textbook has been designed to meet the needs of B.Sc. First Semester students of Chemistry of Delhi University and Colleges as per the recommended National Education Policy 2020. This textbook explains the subject in the most student-friendly way and is designed to keep itself updated with the latest in research. Organic chemists think by constructing mental pictures of molecules and communicate with each other by drawing pictures. This book favors series of figures over long discussions in the text and covers important topics such as Fundamentals of Organic Chemistry, Reactive Intermediates and Rearrangement Reactions, Electrophilic addition reactions, Nucleophilic addition and substitution a reaction, Elimination reactions, Electrophilic substitution reactions and Stereochemistry.

Salient Features:

- An emphasis is placed on presenting the material pictorially
- Fundamental concepts of organic chemistry like reactive intermediates, rearrangement reactions and types of organic reactions are discussed broadly
- Stereochemistry includes a summary of isomers, different projections and relative versus absolute configurations
- The book also provides an overview of various laboratory techniques and their applications

S. CHAND PUBLISHING
A division of S Chand And Company Limited
(ISO 9001 Certified Company)
E-mail: info@schandpublishing.com
Customer care (toll free) No.: 1800-1031926
www.schandpublishing.com

9 789355 016218
1122
₹ 225.00

LPSPE BASIC CONCEPTS OF ORGANIC CHEMISTRY Semester I

Dr. P. P. Singh

Low Priced Students' Paperback Edition

For the University of Delhi

Semester I

BASIC CONCEPTS OF ORGANIC CHEMISTRY

S. CHAND

2023

Springer Series in Reliability Engineering

Vijay Kumar
Hoang Pham *Editors*

Predictive Analytics in System Reliability

Springer

Springer Series in Reliability Engineering

Series Editor

Hoang Pham, Department of Industrial and Systems Engineering, Rutgers University, Piscataway, NJ, USA

Vijay Kumar · Hoang Pham
Editors

Predictive Analytics in System Reliability

Springer

Editors
Vijay Kumar
Department of Mathematics
Amity Institute of Applied Sciences
Amity University Uttar Pradesh
Noida, Uttar Pradesh, India

Hoang Pham
Department of Industrial and Systems Engineering
Rutgers University
Piscataway, NJ, USA

ISSN 1614-7839 ISSN 2196-999X (electronic)
Springer Series in Reliability Engineering
ISBN 978-3-031-05346-7 ISBN 978-3-031-05347-4 (eBook)
https://doi.org/10.1007/978-3-031-05347-4

Contents

Machine Learning Based Software Defect Categorization Using Crowd Labeling

Sushil Kumar, Meera Sharma, S. K. Muttoo, and V. B. Singh

Abstract Defect categorization is an important task which helps in software maintenance. It also helps in prioritizing the defects, resource allocation, etc. Standard machine learning techniques can be used to automate the categorization of defects. Labeled data is needed for learning models. The expert is required for obtaining the labeled data. Sometimes, it is costly or expert is not available. So, to overcome this dependency, crowd labeled data is used to train a model. Crowd (a set of novices) is asked to assign a category as defined by IBM's Orthogonal Defect Classification (ODC) to the defect reports. Obtaining categories through crowd can be inaccurate or noisy. Inferencing ground truth is a challenge in crowd labeling. Support Vector Machine, k Nearest Neighbor and Gaussian Naive Bayes classifier, are learnt effectively using new methodology from data labeled by a set of novices. In this chapter, we have proposed a learning model which learns effectively to predict the impact category of software defects using the expectation maximization algorithm and shows the better performance according to the various types of metrics by improving the existing technique by 8% and 11% accuracy for Compendium and Mozilla datasets respectively.

Keywords Crowd labeling · Naive Bayes classifier · Categorization · Expectation maximization

S. Kumar
Department of Computer Science, Shyam Lal College, University of Delhi, Delhi, India

M. Sharma
Department of Computer Science, Swami Shraddhanand College, University of Delhi, Delhi, India
e-mail: meerasharma@ss.du.ac.in

S. K. Muttoo
Department of Computer Science, University of Delhi, Delhi, India
e-mail: skmuttoo@cs.du.ac.in

V. B. Singh (✉)
School of Computer and Systems Sciences, Jawaharlal Nehru University, Delhi, India
e-mail: vbsingh@mail.jnu.ac.in

1 Introduction

Defect categorization is an important task that improves the efforts needed for software maintenance during software development process [1]. Defect categorization is a time consuming task which is done manually by the experts and involves high cost. Machine learning techniques like Supervised learning algorithms [2] can be used to categorize the defects as these algorithms need labeled dataset to train a classifier. To get the labelled dataset in case of unavailability of experts is a very challenging task. Crowd labeling can be used to obtain the labeled data. It is a process to get the data labeled by a set of novices or nonexperts. It is possible to learn to classify from this type of labeled data [3]. The number and quality of these novices influence the learning of classifier from this data set. The major concern of learning form crowd labeled data is the reliability. This chapter addresses the realiability issue of non-experts through expectation maximization algorithm.

Data set contains the category that is defined by the Orthogonal Defect Classification (ODC). The ODC was initially specified in [4] for defect classification to improve software development process. The main motive behind ODC was to extract defect information and to know the relation between cause and effect. The ODC permits software developers to distinguish defects based on their impact on customer. Categorization of software defects provides valuable information which is very useful to prioritize and fix the defects. It can also be helpful in prediction of defects and assigning defects to software developers.

This chapter explores the possibility and proposes a methodology to learn an accurate classifier for predicting the impact of software defects from the crowd labeled data using expectation maximization algorithm. The process of defect categorization is shown in Fig. 1. A methodology similar to [21] is used to integrate the labels to find the ground truth to train three classifiers, namely Naïve Bayes, k Nearest Neighbor and Support Vector machine.

The main contribution of this chapter is defect categorization from unstructured text from summary and description and analysis of subjective labeling assigned to the defect report by non-experts using Expectation–Maximization algorithm. Training of classifier has been done by taking into account the reliability of each non-expert. The performance has analysed based on majority voting and Expectation–Maximization algorithm.

This chapter is organized in following sections. Section 2 discusses the related work, ODC and Crowdsourcing. Datasets, Classifier and Expectation Maximization are explained in Sect. 3. Section 4 explains the methodology. The experimental work and metrics are explained in Sect. 5. Results are presented and discussed in Sect. 6.

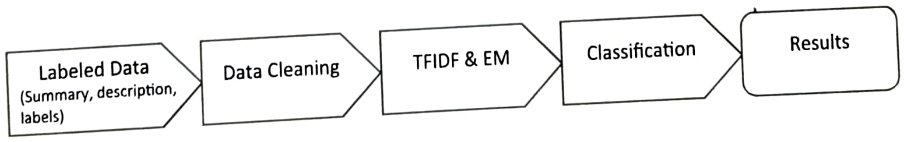


Fig. 1 Process of defect categorization

Section 7 discusses the threats to validity. Section 8 concludes the chapter with future work.

2 Related Work

Learning from crowd (a set of non-experts) is a new Supervised learning paradigm in which the real or true labels of examples or instances are unavailable. However, each instance is provided with a set of noisy class labels, each indicating the class-membership of the instance according to the subjective opinion of an annotator. Many research works have been carried out in recent years. Most of them focus on labeling techniques and on the quality of the labels. Snow et al. [5] evaluated that the knowledge of four annotators is equal to the one expert. Sheng et al. [6] in his study proposed the idea of relabeling and also compared advantages of it. The idea of weak labeling was proposed by Benaran-Munoz et al. [7] in which every annotator provides more than one label for each instance. GLAD Whitehill et al. [8] proposes different levels of expertise and difficulty of examples. Donmez et al. [9] proposes a novel method of repeated trials to get the knowledge about a label and as well as about a labeller. Welinder and Perona [10] distinguished between a reliable and unreliable labeller. In case of unreliable labeler, more labels need to be asked. On the other hand for a reliable labeler, acquired label is a true label. The probability of getting true category/label follows a Bernoulli distribution by Yan et al. [11]. Gonzalez et al. [12] proposes to learn a classifier using five novices with k Means clustering and EM method. Dermartini et al. [13] proposed the method based on probabilistic reasoning and crowdsourcing. Furthermore, severity prediction of defect reports based on the textual description of defects using machine learning algorithms has been performed. Chaturvedi and Singh [14], proposed a severity prediction method which classifies the severity of the defect reports using supervised machine learning algortihms, namely Multinonmial Naïve Bayes, Support Vector Machine, k-Nearest Neighbor, Naïve Bayes, J48 and RIPPER. To carry out the experimental work, the authors collected the two bug reports data sets from NASA and PROMISE repository. Text mining techniques are applied on bug description to extract the relevant features. Liu et al. [15] present a ranking-based technique to improve the feature selection algorithms and also propose an ensemble feature selection algorithm. To evaluate the performace, the authors collect bug reports from two projects, namely Eclipse and Mozilla. They improve the existing methods by 54.76% in terms of f-measure. In [16], authors present a severity prediction technique using textual features of bug reports from three projects Eclipse, Mozilla and Gnome. They were able to achieve 67% accuracy using adaboost classifer. Yang et al. [17] present a severity prediction approach based on emotion similarity of the reporter by calculating the emotion similarity probability. To validate the proposed approach, they collected the bug reports from five projects: GNU, JBoss, Mozilla, Eclipse and Wireshark.

Table 1 Defect impact category and their definition

Impact	Definition
Capability	The ability of the product/system to perform its intended functions and satisfy the customer's functional requirements
Usability	The ability to use and utilize functions of a system by the user
Performance	The speed and responsiveness of the product/system as perceived by the customer
Reliability	The ability of the product/system to consistently perform its intended functions without unplanned interruption
Installability	The ability to easily install a product
Maintainability	The ease with which a failure can be diagnosed and the product/system can be upgraded to apply corrective fixes without impacting the customer's data and operations
Documentation	The ability of a system to provide user manuals and documentation to its user to understand a system easily
Migration	The ease and degree to which the product/system can be upgraded to the newer release without impacting the customer's data and/or operations
Standards	The degree to which the product/system conforms to established pertinent standards
Integrity/security	The degree to which the product/system is protected from inadvertent or malicious destruction, modification, or disclosure
Capacity	The loss of capability when configured at full capacity
Serviceability	The capacity to diagnose faults and failures easily

2.1 *Orthogonal Defect Classification (ODC)*

Orthogonal Defect Classification (ODC) is a precise system for Software Defect Classification created by IBM in the mid of 1990s [4]. ODC empowers in-process input to designers by separating marks on the improvement procedure from defects. The 13-classification ODC enables engineers to isolate absconds relying upon their effect. It is especially appropriate for open-source ventures. The impact category and the definition are provided in Table 1. The program structure involved in defect can be indicated by ODC [18].

2.2 *Crowdsourcing and Learning from Crowd*

The author distributed an article in the wired magazine in 2006 [19]. In this article, He profoundly broke down the effect of a rising miniaturized scale outsourcing through Internet on current business conditions and the term crowdsourcing was first presented. Crowdsourcing has become an important strategy to manage issues at any phase of Software Development Life Cycle (SDLC) from software requirements to maintenance [20, 21]. It is a way of solving a problem with collective efforts [22, 23]. There are various online platforms such as Amazon's MTurk and crowdFlower where a problem can be posted. Crowdsourcing is very helpful in decision making to a software development team. The enthusiasm for the learning from crowd is because of getting large amount of data labeled at very cheap cost through web.

Learning form labeled data by crowd is challenging as each instance of a dataset is assigned a category by non-expert. These non-experts are of obscure trustfulness. The low reliability of these non-experts is another challenge. There are various strategies proposed in past literature. However, in such cases where there is no ground truth and trustworthiness of each non-expert is doubtful, a classifier can be learnt by combining the opinion of each non-expert. Snow et al. [5] estimated the contribution of the non-expert annotators: they recommend that the blend of four non-expert explanations coordinates the information of domain expert.

The following research question has been addressed in this chapter:

Research question: Can we predict more accurately the impact of software defect by estimating the reliability of each non-expert?

The chapter in address to the above question, studied the two datasets Compendium and Mozilla that covers the entire product in both the datasets. To do further analysis, three classifiers Naïve Bayes, Support Vector Machine and k Nearest Neighbor are trained. EM based technique similar to [24, 25] are used. The technique uses the subjective opinion of all the non-expert and estimates the reliability of each non-expert.

3 Datasets and Methods

3.1 Datasets

Two datasets Compendium and Mozilla have been used directly from [12]. The Compendium dataset is taken from http://compendium.open.ac.uk/bugzilla/ which is a software tool. All issues reported in August 2014 are considered. Total 846 defects were obtained. Another dataset Mozilla has 598 defects. Mozilla is an open source application. For both the datasets, two fields summary and description are considered. Figures 2 and 3 show the number of labels assigned by the non-experts (labelers) according to the impact categories defined by ODC for the Compendium and mozilla datasets respectively. Usability, requirement and Capability are the most assigned categories for Compendium dataset.

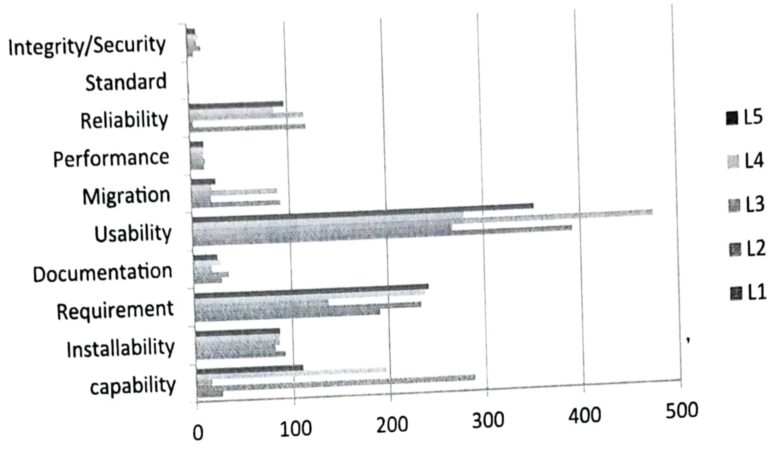


Fig. 2 Number of labels assigned by five labellers for Compendium dataset

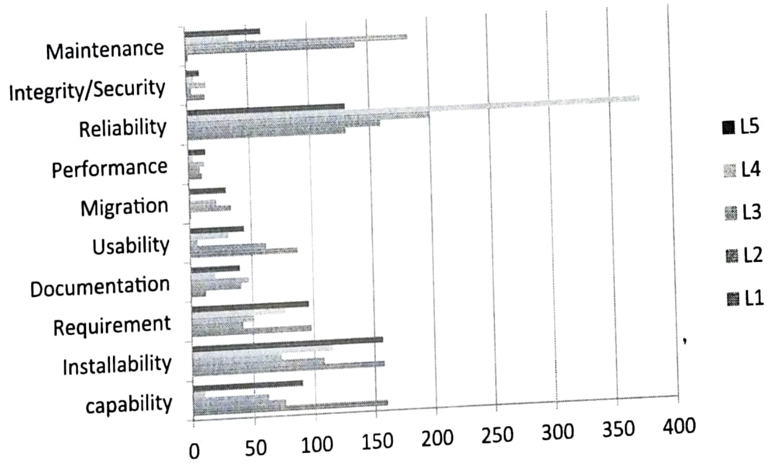


Fig. 3 Number of labels assigned by five labellers for moziila dataset

3.2 *Expectation–Maximization*

Expectation Maximization algorithm has been widely used [8, 10, 23–25]. The Methods based on expectation maximization is not new in crowd learning methods. The EM based technique proposed similar to [25] for multidimensional learning from crowd labeling is used to categorize the software defects.

Let N be the number of defect reports (instance or examples) in a dataset. Let n_d^l be the number of times, a defect instance d is labeled with label l. Let a function b_i^l is defined as, $b_i^l = 1$, if the assigned label is same as the true label (i.e. $l = l'$) and 0 otherwise. We assume that the labels are assigned independently by the labelers (non-experts). By the definition of multinomial distribution we can define the probability of a observed (assigned) label while the true label l' (most voted label) is known using Eq. (1) as

$$p\left(l|l',d\right) \propto \prod_{l=1}^{L} p(l'|l)^{n_d^{l'}} \quad (1)$$

Let each defect example 'd' of a dataset D is labeled independently, then we can rewrite the above equation as (2) for each category 'c' for all the defect examples.

$$p(l,l'_d) \propto \prod_{d=1}^{N}\prod_{c=1}^{C}(p(l)\prod_{l=1}^{L} p\left(l'|l\right)^{n_d^{l'}})^{B_i^l} \quad (2)$$

Algorithm: EM (D, n, ∈)

- $D =< (d_i, l_i) >$ where $1 \le i \le n$
- $D = < (d_1,l_1),(d_2,l_2),\ldots,(d_n,l_n) >$

1. Initialization

$$\text{Calculate } E\left[b_i^l\right] = \frac{n_d^l}{\sum_l n_d^l} \quad (3)$$

2. M step: Select the value for (4) and (5)

$$p(l'|l) = \frac{\sum_i b_i^l \cdot n_d^l}{\sum_1^C \sum_i b_i^l \cdot n_d^l} \quad (4)$$

$$\text{and } p(l) = \frac{1}{N}\sum b_i^l \quad (5)$$

to maximize the likelihood.

3. E step: Estimate the reliability of each non-expert as

$$E[b_i^l|\mathrm{D}] = p(b_i^l = 1|D) = \prod_1^C p(l'|l)^{n_d^l} \cdot p(l) \quad (6)$$

Repeat Steps (2) and (3) until i reaches to Maximum Iteration N or differnces between iterations<0.001.

The EM system enables us to consolidate the estimation of each non-expert that display the dependability of every labeler and the learning of the model utilizing the labels assigned by these non-experts. The initial value is calculated using Eq. (1) as the ratio by counting the frequency of a specific label to the total number of labels assigned. In our technique, the Expectation step calculates the expectation for each non-expert to estimate the reliability using Eq. (6), by integerating the probability of a label with the probability of true label when observed label is given for each category and thus calculate the estimated posterior probabilities.The Maximization step, the model parameters are re-evaluated with the end goal that the probability is augmented given the information and the loads assessed in the Maximization step using Eqs. (4) and (5) which are the maximum likelihood estimators of $p(l'|l)$ and $p(l)$. $p(l'|l)$ is the likelihood of true label when observed label is given and $p(l)$ is the probability of observed label. Iteratively, the steps 2 and 3 are rehashed until the likelihood converges to a local maxima or the maximum number of iterations is reached.

3.3 *Classification Model*

The model uses the summary and description field to predict the impact category of a defects reported in the two datasets of Compendium and Mozilla. The Naïve Bayes (NB), Support Vector Machine (SVM) and k-Nearest Neighbor (k-NN) classification algorithms are used to categories the defects. Naive Bayes classifier [26] is one of the most effective classifier because of its performance with other competitive classifiers. It learns by computing the probability of an attribute x_i given the class y_i where $(x_i, y_i) \in D$ i.e. training data. Naïve Bayes classifier makes strong assumption that all the attributes x_i are independent.

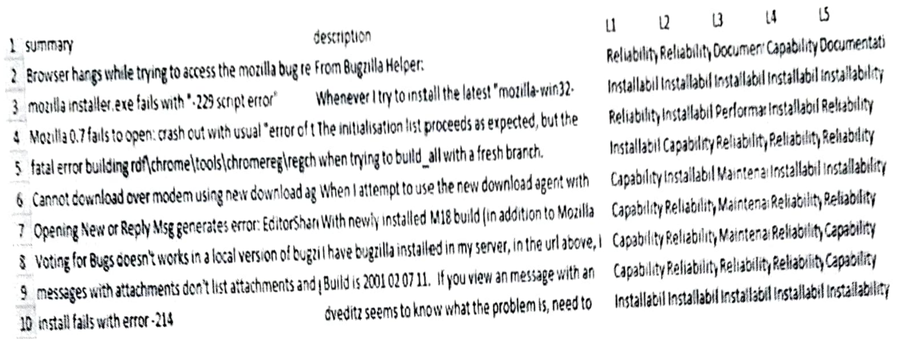
Support Vector Machine (SVM) [27] is a supervised learning technique which is initially used for dividing hyperplane. Its capability to generalize and better performance for multiclass problem makes it suitable for categorizing software defects. k Nearest Neighbor (k-NN) is a very simple supervised technique. It is suitable for large datasets and assigns a category to new object by finding its nearest neighbor.

4 Methodology

As specified by [12] a group of five non-experts are asked to provide category to each example of the defect reports of compendium and mozilla. The categories assigned by each non-experts were processed along with the summary and description fields as in Fig. 4.

The graphical representation of the whole learning process is shown in Fig. 5.

The text provided in summary and description field are combined as summary and pre-processed using Natural Language Processing Tool Kit (NLTK) implemented in python. Figure 6 depicts all the steps of data processing. The relevant and important words from the summary field were extracted so that it can be easily used by machine learning techniques [28, 29]. Stop words were removed by downloading the stop words from nltk and by importing ***nltk.corpus.reader*** package written in python. The Text in summary field was converted to lowercase using the in-built lower () function. Porter stemmer [28] was used for stemming the words and also the tokens were formed. A bag of words was created by extracting features and ***countVectorizer*** is improted to count the frequency of a word. The value for max-features parameter of



1	summary	description	L1	L2	L3	L4	L5
2	Browser hangs while trying to access the mozilla bug re	From Bugzilla Helper:	Reliability	Reliability	Documen	Capability	Documentati
3	mozilla installer.exe fails with "-229 script error"	Whenever I try to install the latest "mozilla-win32-	Installabil	Installabil	Installabil	Installabil	Installability
4	Mozilla 0.7 fails to open: crash out with usual "error of t	The initialisation list proceeds as expected, but the	Reliability	Installabil	Performa	Installabil	Reliability
5	fatal error building rdf\chrome\tools\chromereg\regch when trying to build_all with a fresh branch.		Installabil	Capability	Reliability	Reliability	Reliability
6	Cannot download over modem using new download ag	When I attempt to use the new download agent with	Capability	Installabil	Maintena	Installabil	Installability
7	Opening New or Reply Msg generates error: EditorShar	With newly installed M18 build (in addition to Mozilla	Capability	Reliability	Maintena	Reliability	Reliability
8	Voting for Bugs doesn't works in a local version of bugzi	I have bugzilla installed in my server, in the url above, I	Capability	Reliability	Maintena	Reliability	Capability
9	messages with attachments don't list attachments and	Build is 2001 02 07 11. If you view an message with an	Capability	Reliability	Reliability	Reliability	Capability
10	install fails with error -214	dveditz seems to know what the problem is, need to	Installabil	Installabil	Installabil	Installabil	Installability

Fig. 4 Defect report of Compendium and labeled assigned by five non-experts

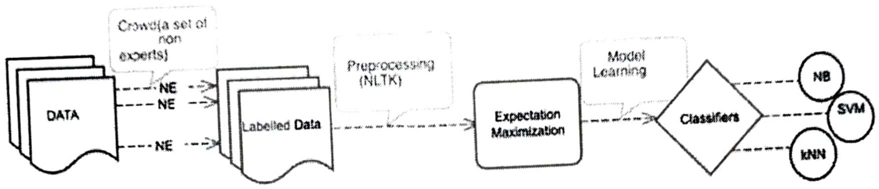


Fig. 5 Graphical representation of the process

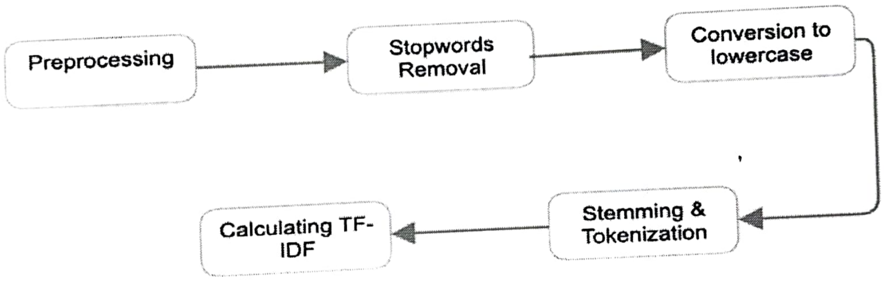


Fig. 6 Steps involved in Data Processing

countVectorizer function was set to 900. Term frequency inverse document frequency (TF-IDF) was calculated for every word. The value for parameter in EM based technique was set to 0.001. The number of iteration was set to 400. This helps to learn a classifier more accurately by using the crowd learning approaches. We have used tenfold cross validation to split the datasets. All the classifiers learn from the same dataset.

5 Experimental Framework

The various experiments on two datasets have been performed. The capabilities of EM based techniques have been explored. The different metrics are used to check the

Table 2 Different Metrics used to measure the performance of classifiers

$accuracy = \frac{\#instances\ correctly\ classified}{total\#instances}$	$Precision = \frac{\#instances\ correctly\ classified\ as\ class\ A}{total\#instances\ as\ class\ A}$
$recall = \frac{\#instances\ correctly\ classified\ as\ Class\ A}{total\#instances\ labelled\ as\ class\ A}$	$f\ measure = \frac{2*precision*recall}{precision+recall}$
$Max\ recall = max(recall)$	$min\ recall = min(recall)$

Table 3 Results of three classifiers learnt from compendium dataset and different metric based on EM

Classifier	Accuracy	Precision	Recall	F measure	Max recall	Min recall	Majority voting
NB	**0.6212**	0.5412	**0.5310**	0.4880	0.5310	0.3201	0.4016
SVM	0.5819	0.5762	0.3901	0.4645			0.4932
kNN	0.3941	0.3209	0.3821	0.4248			0.4417

Table 4 Results of three classifiers learnt from mozilla dataset and different metric based on EM

Classifier	Accuracy	Precision	Recall	F measure	Max recall	Min recall	Majority voting
NB	**0.6541**	0.6216	**0.6152**	0.4914	0.6152	0.3170	0.3754
SVM	0.5991	0.5804	0.5770	0.4032			0.3762
kNN	0.5946	0.5709	0.5610	0.4566			0.3912

capabilities of our proposed approach. The metrics used to measure the performance of classifier are shown in Table 2.

The performance of classifiers using the different metrics described in Table 2 is shown in Tables 3 and 4. Table 3 shows the accuracy, precision, recall and F measure for Naïve Bayes, SVM and kNN classifiers on the Compendium dataset. whereas the performance of these classifiers are shown in Table 4 on the Mozilla dataset.

6 Results and Discussion

The labels assigned by different non-experts are compared for both compendium and mozilaa dataset. We have used the same datasets as of [12]. For compendium datasets, we considered installability, Requirement, Usability and other. Other is a new label which is assigned to rest of the labels. The classfiers Naïve Bayes, SVM and kNN are learnt using these four categories. For Mozilla dataset, installability, maintenance, reliability and other (new label) are used to train a classifier.

So as to give a total overview of the performance of classifiers, namely Naïve Bayes, SVM and kNN, the results are presented in Tables 3 and 4. The performance of the classifiers learnt using Compendium dataset are shown in Table 3. Table 4

Table 5 Comparison of proposed approach to the Hernanedez Gonzalez [12] in terms of accuracy based on EM algorithm

Dataset	Approch	Accuracy (%)
Compendium	Reference [12]	**54**
	Proposed	62
Mozilla	Reference [12]	**54**
	Proposed	65

presents the result of classifiers learnt using Mozilla dataset. The results provide in Tables 3 and 4 measure the performance of classifiers using the same metric for both the dataset. The metrics accuracy, precision, recall, F measure and maximum and minimum recall are used in this chapter. The definition of metrics to evaluate the performance of classifiers is provided in Table 2.

Columns of Tables 3 and 4 show the majority voting, EM based method and different metrics accuracy, precision, recall, F measure, maximum and minimum recall. Whereas row represents the experiment values for each classifier. The best value for each classifier is represented in bold. The differences between minimum and maximum recall values are related to the accuracy and f measure. The high difference indicates the large values of accuracy while low difference contributes to high f measure values. Hence the performance of the classifiers can be assessed from these values across all labels.

We have also compared our results on the same dataset compendium and mozilla used by Hernández- Gonzalez's et al. [12]. They have classified their dataset by using naïve bayes, 2DB (Dependence Bayesian) and TAN (Tree Augmented Naïve Bayes). By analyzing the results, we can observe that maximum 62% accuracy is achieved in case of compendium dataset when naïve bayes classifier is learnt. An accuracy of 65% is achieved when navie bayes classifer is learnt using Mozilla dataset as shown in Table 5.

Figures 7 and 8 shows the comparision of accuracy for Compendium and Mozilla respectively.

The results comparision with the previous approaches are shown in Table 6.

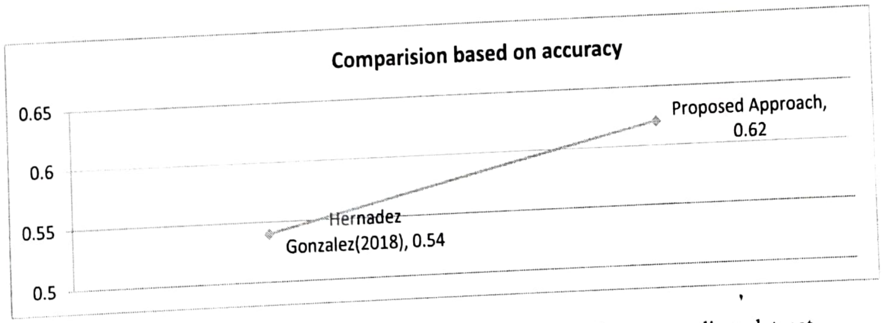


Fig. 7 Comparision based on accuracy between two approaches for compendium dataset

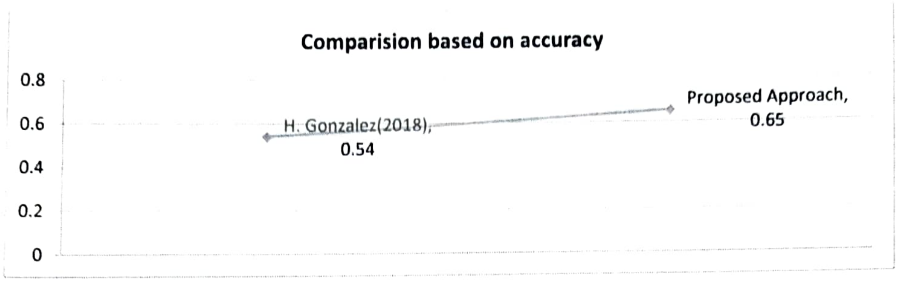


Fig. 8 Comparision based on accuracy between two approaches for mozilla dataset

Table 6 Comparision with other approaches

Approach	Accuracy (%)	Precision (%)	Categories	# Defect reports
Thung et al. [2]	77.8	69	Control and data flow, structural and non-functional	500
Thung et al. [31]	-	65.1	Control and data flow, structural and non-functional	500
Liu et al. [32]	79	75	Data, computational, interface, control/logic	1174
Hunag et al. [33]	80.7–82.9	–	ODC impact attributes	1653
Gonzalez et al. [12]	62–64	–	ODC impact attributes	1444

7 Threats to Validity

In this section we have discussed various threats to validity to our study.

7.1 Threats to Construct Validity

Threats to construct validity refer to the selection of measures and measurement tools. We have used four measures to evaluate the performance of our proposed model. These four measures are accuracy, precision, recall and f-measure. These measures are commonly used. So we can believe that there is minimal threat to construct validity.

7.2 *Threats to Internal Validity*

Threats to internal validity refer to the biasness of the experimenter. These defect reports are labeled manually by five people having no expertise. The distribution of classes/labels for both the datasets is not uniform. The performance of classifiers are different for both the datasets. It can due to the text describing the defects and preprocessing the textual description. As we are only using the unstructured textual defect reports, it can influence the result of the categorization.

7.3 *Threats to External Validity*

We have used 1444 defect reports from two projects. The number of defect reports may not be enough to generalize the results. Manual labeling of defect reports according to one of the ODC attributes is a difficult and lengthy task and the limitation to obtain a large dataset. Generalizability of the result is one of the threats to external validity.

8 Conclusion and Future Work

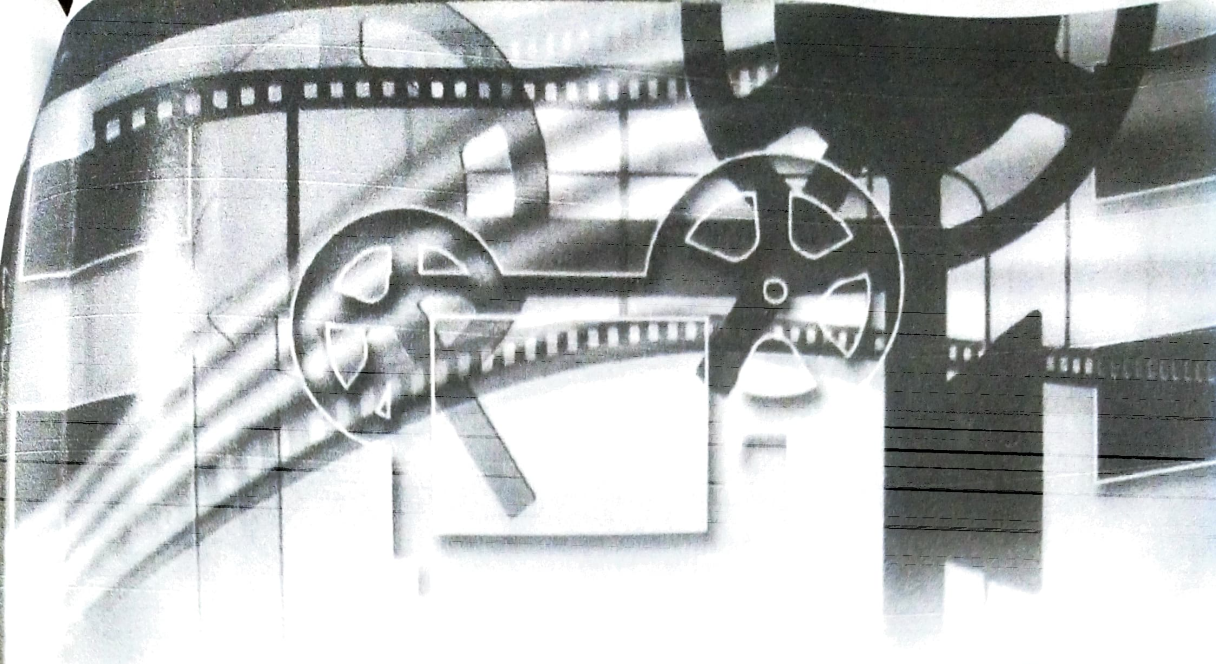
The chapter proposed a defect categorization approach based on EM algorithm through crowd labeled data. Two datasets from compendium and mozilla have been used to test the proposed methodology. EM method applied to learn three classifiers naive Bayes, support vector machine and k-NN. The experiment results show the performance of these classifiers. The EM-based method calculates the reliability of each non-expert. It models the problem of multiclass using multinomial distribution and maximum likelihood. Thus classifiers are learnt from the best possible configuration. The proposed approach shows the better performance as compare to exiting approach by 8 and 11% accuracy. There are various issues which can be fixed in future. To combine the knowledge of each non-experts, retrieving ground truth from crowd labeled data are such issues which must be addressed.

References

1. Boehm B, Basili VR (2005) Software defect reduction top 10 list. In: Boehm B, Rombach HD, Zelkowitz MV (eds) Foundations of empirical software engineering: the legacy of Victor R. Springer, Basili, pp 426–431
2. Thung F, Lo D, Jiang L (2012) Automatic defect categorization. In: Proceedings of 19th working conference on reverse engineering, pp 205–214
3. Hernández-González J, Inza I, Lozano JA (2015) Multidimensional learning fromcrowds: usefulness and application of expertise detection. Int J Intell Syst 30(3):326–354

4. Chillarege R, Bhandari I, Chaar J, Halliday M, Moebus D, Ray B, Wong M-Y (1992) Orthogonal defect classification—a concept for in-process measurements. IEEE Trans Softw Eng 18(11):943–956
5. Snow R, O'Connor B, Jurafsky D, Ng AY (2008) Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In: Proceedings of conference on empirical methods in NLP; Honolulu, Hawaii, USA, pp 254–263
6. Sheng VS, Provost FJ, Ipeirotis PG (2008) Get another label? Improving data quality and data mining using multiple, noisy labelers. In: Proceedings of 14th international conference on knowledge discovery and data mining (ACM SIGKDD), Las Vegas, Nevada, USA, pp 614–622
7. Beñaran-Muñoz I, Hernández-González J, Pérez A (2018) Weak labeling for crowd learning
8. Whitehill J, Ruvolo P, Wu T, Bergsma J, Movellan JR (2009) Whose vote should count more: optimal integration of labels from labelers of unknown expertise. In: Proceedings of advances neural information processing systems 22 (NIPS), Vancouver, Canada, pp 2035–2043
9. Donmez P, Carbonell JG, Schneider J (2009) Efficiently learning the accuracy of labeling sources for selective sampling. In: Proceedings of the 15th international conference on knowledge discovery and data mining (KDD), pp 259–268
10. Welinder P, Branson S, Belongie S, Perona P (2010) The multidimensional wisdom of crowds. In: Proceedings of advances neural information processing systems 23 (NIPS), Vancouver, Canada, pp 2424–2432
11. Yan T, Kumar V, Ganesan D (2010a) Crowdsearch: exploiting crowds for accurate real-time image search on mobile phones. In: Proceedings of the 8th international conference on mobile systems, applications, and services. ACM, pp 77–90
12. Hernández-González J, Rodriguez D, Inza I, Harrison R, Lozano JA (2018) Learning to classify software defects from crowds: a novel approach. Appl Soft Comput 62:579–591
13. Dermatini G (2012) ZenCrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking
14. Chaturvedi KK, Singh VB (2012) Determining bug severity using machine learning techniques. In: 2012 CSI sixth international conference on software engineering (CONSEG) 2012 Sep 5, pp 1–6. IEEE
15. Liu W, Wang S, Chen X, Jiang H (2018) Predicting the severity of bug reports based on feature selection. Int J Softw Eng Knowl Eng 28(04):537–558
16. Otoom AF, Al-Shdaifat D, Hammad M, Abdallah EE, Aljammal A (2019) Automated labelling and severity prediction of software bug reports. Int J Comput Sci Eng 19(3):334–342
17. Yang G, Zhang T, Lee B (2018) An emotion similarity based severity prediction of software bugs: a case study of open source projects. IEICE Trans Inf Syst 101(8):2015–2026
18. Catolino G et al (2019) Not all bugs are the same: understanding, characterizing, and classifying bug types. J Syst Softw
19. Howe J (2006) The rise of crowdsourcing. Wired Mag 15(6):1–4
20. Mao K et al (2017) A survey of the use of crowdsourcing in software engineering. J Syst Softw 126:57–84
21. Sarı A, Tosun A, Alptekin GI (2019) A systematic literature review on crowdsourcing in software engineering. J Syst Softw 153(2019):200–221
22. Rodrigo EG, Aledo JA, Gámez JA (2019) Machine learning from crowds: a systematic review of its applications. Wiley Interdisc Rev: Data Min Knowl Discov 9(2):e1288
23. Raykar VC, Yu S, Zhao LH, Valadez GH, Florin C, Bogoni L, Moy L (2010) Learning from crowds. J Mach Learn Res 11:1297–1322
24. Dawid AP, Skene AM (1979) Maximum likelihood estimation of observer error-rates using the EM algorithm J. R Stat Soc Ser C: Appl Stat 28(1):20–28
25. Smyth P, Fayyad U, Burl M, Perona P, Baldi P (1994) Inferring ground truth from subjective labelling of venus images. In: Proceedings of advances in neural information processing systems (NIPS); Denver, Colorado, USA, pp 1085–1092
26. Friedmen N, Gieger D, Goldszmidt M (1997) Bayesian network classifier. Mach Learn 29:131–163

27. Hsu C-W, Lin C-J (2002) A comparison of methods for multiclass support vector machines. IEEE Trans Neural Netw 13(2):415–425
28. Vedula RMS, Bhadoria RS, Dixit M (2021) Integrating blockchain with AI. In: Multi-disciplinary functions of blockchain technology in AI and IoT applications, pp 1–25. IGI Global
29. Samanta S, Pal M, Mahapatra R, Das K, Bhadoria RS (2021) A study on semi-directed graphs for social media networks. Int J Comput Intell Syst 14(1):1034–1041
30. van Rijsbergen CJ, Robertson SE, Porter MF (1980) New models in probabilistic information retrieval. British Library, London
31. Thung F, Le XBD, Lo D (2015) Active semi-supervised defect categorization. In: 2015 IEEE 23rd international conference on program comprehension. IEEE
32. Liu et al (2015) An ast-based approach to classifying defects. In: 2015 IEEE international conference on software quality, reliability and security-companion. IEEE
33. Huang et al (2015) AutoODC: automated generation of orthogonal defect classifications. Autom Softw Eng 22(1):3–46



साहित्य संस्कृति और सिनेमा

डॉ. प्रतिभा राणा
डॉ. ममता कुमारी

प्रकाशक
श्री नटराज प्रकाशन
ए-507/12, साउथ गामड़ी एक्स.
दिल्ली-110053
फोन : 011-22941694
Email : shreenatrajprakashan@gmail.com

शाखा
19/154, शांतिपुरम् कालोनी
नियर ए.डी.ए. पुलिस चौकी,
आगरा रोड, अलीगढ़
उत्तर प्रदेश–202001
फोन : 0571-2417643

ISBN : 978-93-90915-30-9

© द्विलेखिका

प्रथम संस्करण : 2023

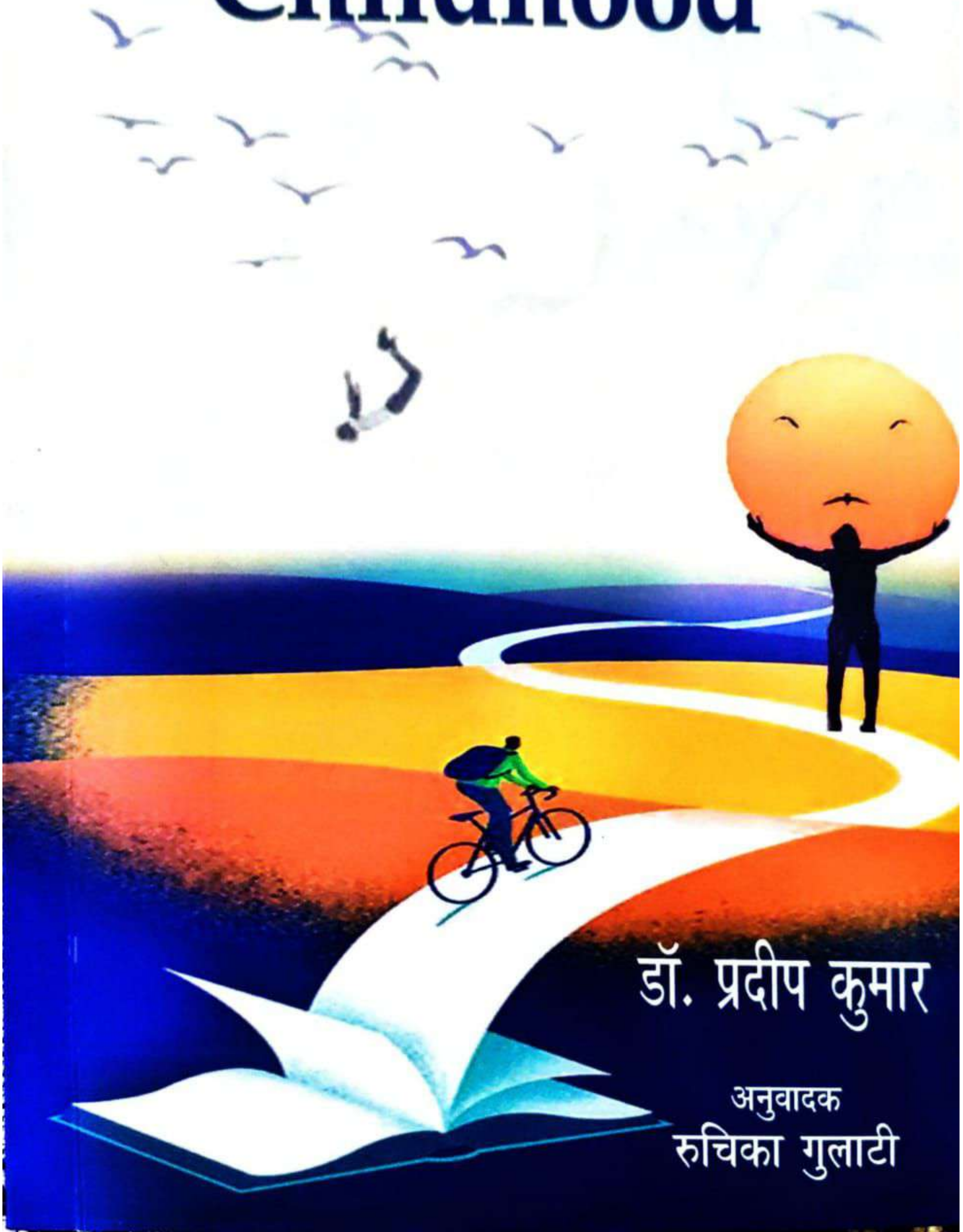
मूल्य : 100.00 रुपये

शब्द संयोजन : श्री नटराज प्रकाशन दिल्ली-110053
मुद्रक : पूजा आफसेट, दिल्ली-110053
आवरण चित्र : शशिकांत सिंह

भारत में मुद्रित : SAHITYA SANSKRITI AUR CINEMA
by Dr. Pratibha Rana, Dr. Mamta Kumari

श्री नटराज प्रकाशन, ए-507/12, साउथ गामड़ी एक्स. दिल्ली-110053 से टी.एस. राघव द्वारा प्रकाशित, श्री नटराज प्रकाशन द्वारा पृष्ठ सज्जा, शशि सिंह द्वारा आवरण सज्जा तथा पूजा ऑफसेट, दिल्ली-110053 द्वारा मुद्रित।

बचपन
Childhood



डॉ. प्रदीप कुमार

अनुवादक
रुचिका गुलाटी

अनामिका पब्लिशर्स एंड डिस्ट्रीब्यूटर्स (प्रा.) लिमिटेड
4697/3, 21-ए, अंसारी रोड, दरियागंज, नई दिल्ली 110002
फोन : 011-23281655, 011-43708938
E-mail: anamikapublishers@yahoo.co.in

प्रथम संस्करण : 2023

© डॉ. प्रदीप कुमार

आई.एस.बी.एन. 978-93-95404-49-5

भारत में मुद्रित

अनामिका पब्लिशर्स एंड डिस्ट्रीब्यूटर्स (प्रा.) लिमिटेड, 4697/3, 21-ए, अंसारी रोड, दरियागंज, नई दिल्ली 110002 द्वारा प्रकाशित। शिवानी कम्प्यूटर्स, दिल्ली 110093 द्वारा शब्दांकित एवं विकास कंप्यूटर एंड प्रिंटर्स, ट्रॉनिका सिटी, गाजियाबाद में मुद्रित।