ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

Optimization of Maintenance Cost for Software Repositories by using Support Vector Machine Based Mining

Parwinder Kaur, Gaurav Kumar

Abstract*— The nature of software maintenance is to manage the progressions that happen amid the product development. To viably oversee and control these progressions, programming archives are utilized to record data about these progressions. Mining the product component features the most recent exploration way from a long while. The report has demonstrated that selecting the best programming for re-architect is observed to be exceptionally bigger task. The utilization of support vector machines and code measurements for mining the product repositories are overlooked in the existing literature yet at the same time fundamentally improvement can be possible. Keeping in mind the end goal to overthrow the constraints with the sooner work another upgraded technique is proposed in this work. The proposed strategy utilizes the unsupervised filtering based Support vector machine to deal with extracting the required data from every repository. The evaluation has plainly shown that proposed system for mining information enhances adequacy of programming upkeep tasks.

***Index Terms*— Maintenance, SVM, Topic Model, Unsupervised filtering, Data Mining.**

I. INTRODUCTION

Maintenance has been perceived as the most troublesome, immoderate and work concentrated action in the product improvement life cycle. Successfully supporting programming upkeep is fundamental to give a dependable brilliant advancement of programming frameworks. Programming upkeep is an extremely expansive action that incorporates mistake corrections, upgrades of abilities, erasure of out-dated capacities, and streamlining. So any work done to change the product after it is in operation is thought to be upkeep work. The purpose is to protect the value of software over the time. Effective programming upkeep is performed utilizing procedures particular to maintenance.

A. Mining Software Repositories

Information mining, the extraction of appropriate data from huge databases, is an intense new innovation with awesome potential to concentrate on the most essential data in their information distribution centers. The Mining Software Repositories (MSR) field examines and cross-link the rich information accessible in these stores to reveal significant data about programming frameworks. By changing these stores from static record-keeping ones into dynamic archives, we can direct choice processes in current software projects. The product building group investigated the archives to perform some product upkeep undertakings, for instance, bug forecast, testing, sway examination, and so on. Some archives are:

- **Source control stores:** These archives record the data of the advancement history of a task. They track all the progressions to the source code alongside the meta-information connected with every change, for instance, who and when performed the change and a short message portraying the change.
- **Bug Repositories:** These storehouses track the determination history of bug reports or highlight asks for that are accounted for by clients and designers of the ventures.
- **Correspondence Archives:** These stores track talks and interchanges about the task over its lifetime. Mailing records and messages have a place with the correspondence documents.

B. Data Mining Software-WEKA

Waikato Environment for Knowledge Analysis (Weka) is a well-known suite of machine learning programming written in Java. These days, WEKA is perceived as a milestone framework in information mining and machine learning.

The key components in charge of Weka's success are:

- It gives a wide range of calculations to information mining and machine learning.
- It is open source and freely accessible.
- It is stage independent.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

- It is effortlessly used by individuals who are not information mining authorities.
- It gives adaptable facilities to scripting tests.

Weka is an accumulation of machine learning calculations for information mining assignments. The calculations can either be connected straightforwardly to a dataset or called from your own Java code. Weka contains tools for information pre-handling, grouping, relapse, bunching, affiliation guidelines, and representation. It is likewise appropriate for growing new machine learning plans.

Datasets in Weka: Every entry in a dataset is an instance of the java class. Every occurrence comprises of various qualities. ARFF Documents are the outer representation of an Instance class.

weka.core.Instance

Classifiers in Weka: Learning calculations in Weka are from the dynamic class. Classifiers in WEKA are models for predicting numeric amounts.

weka.classifiers.Classifier

C. Approach MSR4SM

MSR4SM utilizes topic model to encourage pre-handling the software repositories to separate the fundamental data significant to product maintenance. In view of this approach, some unimportant data from each of the storehouses is expelled. We will first discuss the topic model.

Topic Models: Topic models are generative probabilistic models which have been connected to data recovery to consequently sort out and give structure to content corpus. These models find subjects in the corpus, which speak to real world ideas by habitually co-occuring words. As of late, analysts observed topics to be more useful for organizing different programming artifacts, for example, source code, prerequisites archives, and bug reports. One of the normally utilized theme models as a part of programming designing group is Latent Dirichlet Allocation (LDA).

It gives a powerful approach to utilize programming storehouses in backing of programming support errands. It is accomplished by preprocessing the data in programming storehouses. In particular, we just concentrate the data significant to the upkeep solicitation and current programming from each of the storehouses.

The procedure of MSR4SM is a preprocess to earlier programming archives based procedures. Given an upkeep request, we have to investigate the significant data in programming archives to help cognizance and execution of this solicitation. The information store of MSR4SM incorporates a support demand, programming stores and current programming. We extricate current programming from programming stores because the present programming more often needs some fundamental changes to achieve the change demand. For a product upkeep demand, it is made out of a textual depiction, which should be tokenized, i.e., the solicitation is transformed into tokens and some unessential and insignificant words ought to be expelled. For other information source, as they can be seen as unstructured content, we utilize LDA to extricate the dormant subjects in them. Before investigating the delicate product stores, they require some preprocessing operations for viable utilization of LDA. Initially, some unessential and insignificant words ought to be evacuated. After this, we utilize LDA to produce the subjects for every storehouse. At that point, we lead recurrence investigation and comparability examination among this preprocessed information. At long last, the important data in every product storehouse can be acquired in view of the input from the recurrence examination and comparability investigation. Therefore, designers can get the pertinent information from every product archive which is identified with the maintenance request and current programming.

MSR4SM plans to enhance the adequacy of traditional product storehouses based programming support assignments by preprocessing the product archives. In the connection, it goes for tending to the accompanying two exploration questions (EQs):

- EQ1: Does MSR4SM enhance the viability of earlier programming stores based component location method?
- EQ2: Does MSR4SM enhance the viability of conventional programming archives based change impact examination way?

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

D. Support Vector Machine

SVMs are among the most prominent characterization schemes in the field of machine learning. The key thought of this idea is to recognize the hyper plane isolating both classes such that the instigated separation between the hyper plane and the preparation examples is maximal. The kernel functions are one reason for the achievement of support vector machines. In the meantime, designs existing in the passageway (impelled by the edge and the examples) are penalized. For a learning setting, there are information from two sorts of items alongside related class marks. Much of the time, every information thing, likewise called pattern, is shown by arrangement of genuine esteemed components portraying the objects.

These elements and the class marks constitute the training set. As a rule, master information is required to educate the machines how to consequently perform a task. Such a showing procedure is generally in view of master learning as far as names. This sort of errand is called supervised learning. SVMs have a place with the class of administered learning strategies since they just make utilization of marked information for producing a suitable choice hyper plane.

Unsupervised Filtering based Support Vector Machine: For the unsupervised learning, one is just given the arrangement of unlabeled examples and the objective comprises in finding a segment of these examples into two classes such that a consequent use of a standard support vector machine yields the best general result.

At early phases of the learning procedure, no marks at all are typically given for the examples. For such settings, one is just given an unlabeled preparing set and the actuated learning undertakings have a place with unsupervised learning situations. The unsupervised analogon for classification is unsupervised classification. Here, the objective is to parcel the examples into gatherings such that examples inside the same gathering show comparative properties and those being in different bunch display divergent ones. Nonetheless, the sought classification and grouping models don't as a matter of course agree.

E. Confusion Matrix

It contains data about real and predicted characterizations done by an order framework. Execution of such frameworks is normally assessed utilizing the information as a part of the matrix. The confusion matrix for a parallel grouping issue (which has just two classes positive and negative), is appeared in Table 1.

Table 1: Confusion Matrix

	Predicted Class	
True Class	Positive	Negative
Positive	TP	FN
Negative	FP	TN

In general, Positive = recognized and negative = discarded. Therefore:

- True positive = correctly recognized
- False positive = incorrectly recognized
- True negative = correctly discarded
- False negative = incorrectly discarded

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

II. METHODOLOGY

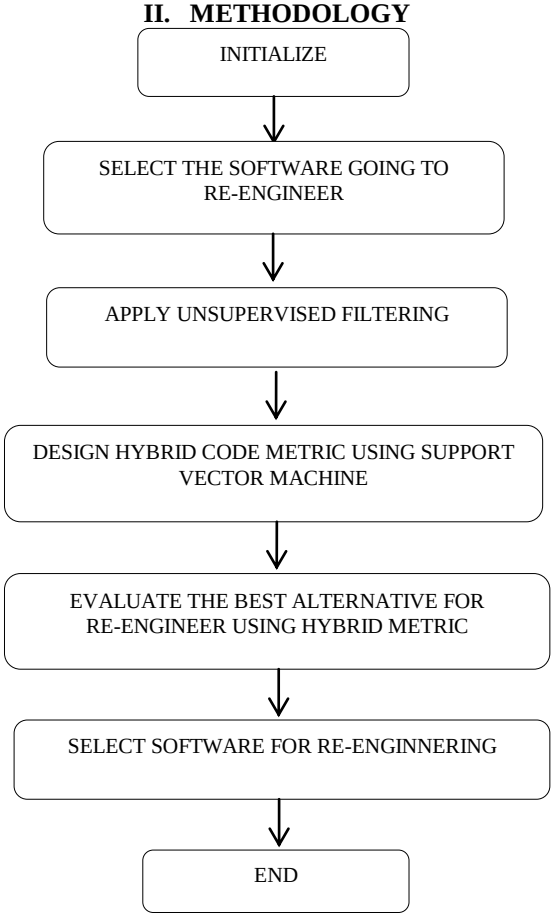

Fig. 1 Steps of proposed work

Step1: In the initial step we introduce the best option selection for software re-building.
Step2: Select the best program which requires re-building.
Step3: Apply unsupervised filtering to decrease the undesirable information from programming storehouses.
Step4: Design hybrid code metric, union and coupling metric of the given programming.
Step 5: Now among the accessible choices discover best option as per the hybrid metric.
Step 6: Select alternative and End.

III. RESULTS AND DISCUSSIONS

For experimental setup, our proposed algorithm is a combination of Support Vector Machine and also Unsupervised Filtering. By using these algorithms our main motive in our thesis is to improve the accuracy of software repositories by extracting only the relevant information through data mining procedure. For accomplishing the work, we have used WEKA software tool for data mining and MATLABR2010a programming language for displaying the results and graphs.

The results demonstrate the proposed solution provides improvement over previous approaches by improving effectiveness of data mining. After the results, we compared the proposed solution against the current procedures.

A. PERFORMANCE ANALYSIS

I. Evaluation of Precision, Recall and F-Measure Parameters

The values of existing and proposed work are given below. Here Table 2 depicts the performance measures of Precision, Recall and F-Measure parameter metrics of different algorithms. The results show that the proposed (PROP) algorithm provides more accuracy, thus enhancing the effectiveness of tasks done.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

Table 2: Precision, Recall and F-Measure Evaluation

ALGORITHM	Precision	Recall	F-Measure
ADT	0.743	0.722	0.718
RF	0.802	0.778	0.774
RT	0.666	0.625	0.602
DT	0.709	0.708	0.708
HP	0.754	0.75	0.75
ABM1	0.711	0.681	0.672
RoF	0.764	0.75	0.748
D	0.745	0.694	0.681
KS	0.775	0.764	0.762
SMO	0.771	0.708	0.693
NB	0.762	0.694	0.677
BN	0.813	0.806	0.804
BFT	0.762	0.736	0.731
D S	0.802	0.792	0.789
LWL	0.752	0.75	0.749
PROP	0.825	0.821	0.822

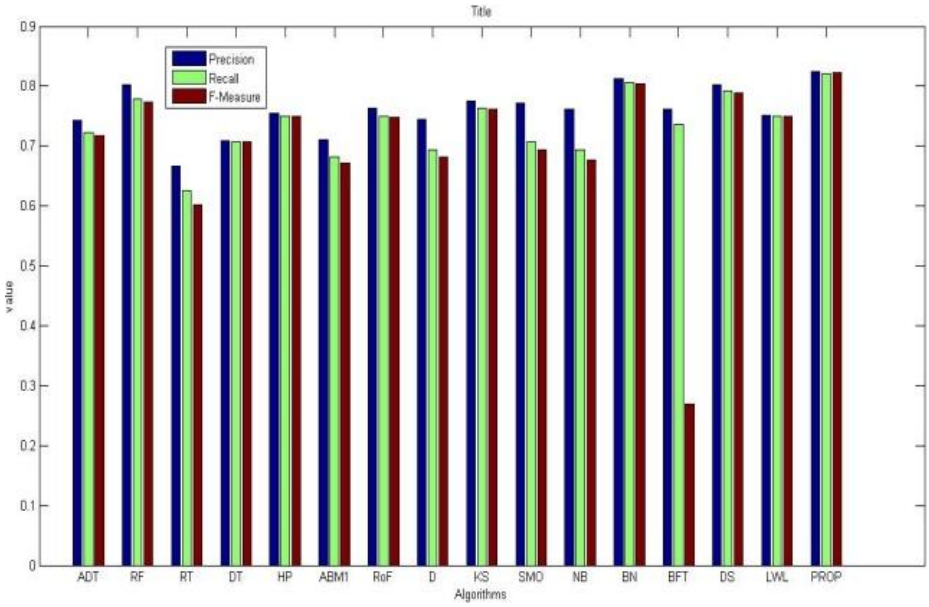

Fig. 2 Precision, Recall and F-Measure Graph

Fig. 2 shows the comparison of Precision, Recall and F-Measure over different algorithms that includes both previous and proposed algorithms. Bar graph below clearly shows the values of algorithms from which it is easily known that PROP is better than all existing algorithms. As seen, for PROP Precision, Recall and F-Measure increases in the bar graph below.

II. Evaluation of Sensitivity and Specificity

Table 3 clearly represents the values of different earlier and now used algorithm on the basis of Sensitivity and Specificity parameters. Here also, it can be seen that PROP algorithm gives better results.

Table 3: Sensitivity and Specificity Predictions

ALGORITHM	Sensitivity	Specificity
ADT	0.8148	0.6666
RF	0.8888	0.7111
RT	0.75	0.714
DT	0.7222	0.6944

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

HP	0.7878	0.7179
ABM1	0.7916	0.625
RoF	0.8275	0.6976
D	0.8571	0.6274
KS	0.8333	0.7142
SMO	0.9	0.6346
NB	0.8947	0.6226
BN	0.7674	0.8620
BFT	0.8461	0.6739
D S	0.75	0.8517
LWL	0.7317	0.7741
PROP	0.7575	0.8734

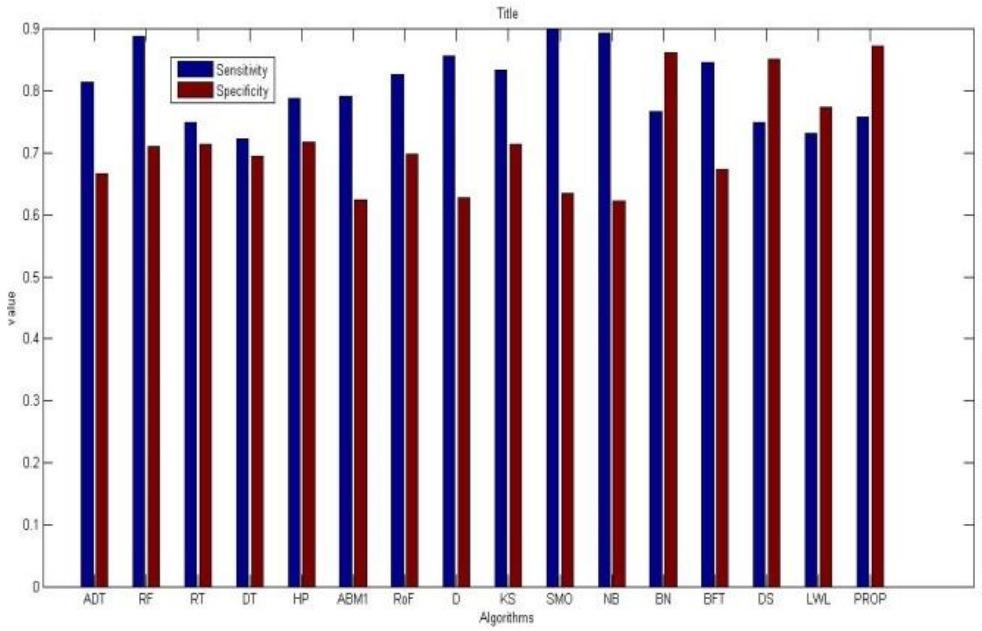

Fig. 3 Sensitivity and Specificity parameters

Fig.3 exhibits comparison among existing and proposed technique with respect to different parameters. The values in the bar graph reveal that PROP algorithm is better than others.

III. Evaluation of Accuracy, RRSE and Bit Error Rate

Table 4 demonstrates the predicted values existing and proposed approach on the basis of Accuracy, RRSE and Bit Error Rate parameters. It defines that algorithm (PROP) used by the proposed technique is much better than the existing algorithms. Also, the Bit Error is reduced to a greater extent.

Table 4: Values of Accuracy, RRSE and Bit Error Rate

ALGORITHM	Accuracy	Root Relative Squared Error	Bit Error Rate
ADT	72.2222	85.569	27.7778
RF	77.7778	75.4346	22.2222
RT	62.5	114.2279	37.5
DT	70.8333	82.574	29.1667
HP	75	83.8882	25

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

ABM1	68.0556	84.4703	31.9444
RoF	75	75.7905	25
D	69.4444	81.688	30.5556
KS	76.3889	97.8923	23.6111
SMO	70.8333	100.7396	29.1667
NB	69.4444	102.5657	30.5556
BN	80.5556	82.2532	19.4444
BFT	73.6111	89.6968	26.3889
D S	79.1667	84.4594	20.8333
LWL	75	85.1722	25
PROP	82.069	85.9774	17.931

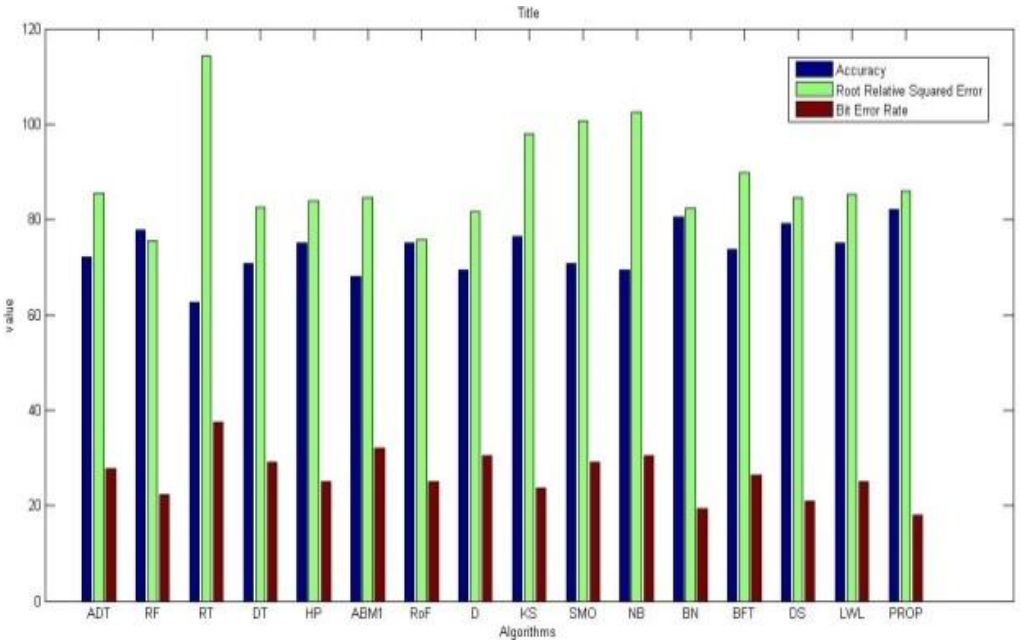

Fig. 4 Bar Graph of Accuracy, RRSE and Bit Error Rate Values

Fig. 4 reports clear values of Accuracy, RRSE and Bit Error Rate from which it can be known that proposed values are far better than the existing algorithms values. Moreover, most of the Bit error rate value is decreased which is a good sign of improvement in the algorithm. Hence from the all the above results, it is confirmed that the proposed approach enhances the efficiency of results on applying to the different parameters.

IV. CONCLUSION AND FUTURE WORK

Mining programming vaults has risen as an exploration course over the previous decade, making considerable progress in both research and practice to support different programming upkeep errands. Successfully supporting programming upkeep is fundamental to give a solid superb development of programming frameworks. Programming stores incorporate bug archives, correspondence files, source control storehouse, and so forth. At the point when utilizing these archives for programming support, consideration of unessential data in every archive can prompt diminished adequacy or even wrong results.

This research work proposed unsupervised filtering based Support vector machine approach to get required data. The comparison between the various algorithms are shown in this research work by using various parameters such as: precision, recall, f- measure, sensitivity, specificity, accuracy, RRSE, BER has shown that the proposed algorithm proved better results as compared to the existing algorithm.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 9, Issue 1, January 2020

Future studies could incorporate upgrading more than one parameter of proposed calculation utilizing concurrent improvement procedures like co-operative PSO and multi objective PSO (Particle Swarm Optimization). Studies should likewise be possible where observational investigation is done on a wide assortment of information set particularly industrial tasks.

REFERENCES

[1] Ghafoori, Zahra, Sutharshan Rajasegarar, Sarah M. Erfani, Shanika Karunasekera, and Christopher A. Leckie, "Unsupervised Parameter Estimation for One-Class Support Vector Machines", In Advances in Knowledge Discovery and Data Mining, Springer International Publishing, pp. 183-195, 2016.

[2] Jiang, Heling, An Yang, Fengyun Yan, and Hong Miao, "Research on Pattern Analysis and Data Classification Methodology for Data Mining and Knowledge Discovery", International Journal of Hybrid Information Technology, vol. 9, no. 3, pp. 179-188, 2016.

[3] X. Sun, B. Li, H. Leung, Bin Li, Y. Li, "MSR4SM: Using Topic Models to Effectively Mining Software Repositories for Software Maintenance Tasks", Information and Software Technology, 2015.

[4] Kalpana Rangra, Bansal, "Comparative Study of Data Mining Tools", IJARCSSE, Volume 4, Issue 6, pp. 216-223, June 2014.

[5] Chaturvedi, Krishna Kumar, V. B. Sing, and Prashant Singh, "Tools in Mining Software Repositories", IEEE 13th International Conference In ICCSA, pp. 89-98., 2013.

[6] Gieseke, Fabian. "From supervised to unsupervised support vector machines and applications in astronomy", KI-Künstliche Intelligent, no. 3, pp. 281-285, 2013.

[7] S.W. Thomas, B. Adams, A.E. Hassan, D. Blostein, "Studying software evolution using topic models", Science of Computer Programming, 2012.

[8] Kumar, D., & Bhardwaj, D., "Rise of data mining: Current and future application areas", IJCSI International Journal of Computer Science, Issue 8, pp. 256–260, 2011.

[9] Reddy, Dr Lokanatha C., "A Review on Data mining from Past to the Future", International Journal of Computer Applications, pp. 0975-8887, 2011.

[10] Cong Jin and Jin-An Liu, "Applications of support vector machine and unsupervised learning for predicting maintainability using object-oriented metrics", Multimedia and Information Technology (MMIT), 2010 Second International Conference, IEEE, 2010.

[11] Anbalagan, Prasanth, and Mladen Vouk, "On mining data across software repositories", In Mining Software Repositories, MSR'09 6th IEEE International Working Conference, IEEE, pp. 171-174, 2009.

[12] Frank, Eibe, Mark Hall, Geoffrey Holmes, Richard Kirkby, Bernhard Pfahringer, Ian H. Witten, and Len Trigg, "Weka-a machine learning workbench for data mining", In Data mining and knowledge discovery handbook, Springer US, pp. 1269-1277, 2009.

[13] Maskeri Girish, Santonu Sarkar, and Kenneth Heafield, "Mining business topics in source code using latent dirichlet allocation" Proceedings of the 1st India software engineering conference, ACM, 2008.

[14] V. Rubin, C.W.Günther, W. M. P. van der Aalst , E. Kindler, B. F. van Dongen , and W. Schäfer, "Process mining framework for software processes", Software Process Dynamics and Agility, LNCS, Springer, pp. 169-181, 2007.

[15] M. Burch, Stephan Diehl, and Peter Weibgerber, "Visual data mining in software archives", Proceedings of the 2005 ACM symposium on Software visualization, ACM, 2005.