ENHANCED BUG TRAIGING WITH DATA TRUNCATION

KRISHNAVENI V.L,KARTHIK.M

.

Abstract- *Software bugs being an inevitable substance ,which is growing up daily as a distractor for the software field. Assigning an efficient and well experienced developer for the bugs are a hazardous task. Bug traiging is a processing that is defined in a manner that to properly select a developer who can handle and fix the bug in an effective manner. Historical dataset plays a major role as a supporter in tariging process, that is we could use it as a reference when a new bug arrives. As per the traditional mannerism, the human traiger does the work of assigning the developer,but it overpowered with automatic startegy. Automatic bug traiging with truncated data source could pave the way much more easier and thus could select the accurate developer for the processing.The Instance and Feature Selection combination help in the case of reduction data.Large amount of data can be truncated throughout.Finally the classifier is demanded for the classification.The most efficient binary classifier(Naive Baye's),moved on to Multi Classifier(Random Forest)is used for classification.Finally it will lead to the assignment of most accurate developer for the present software bug.*

Keywords:- Data, Features, Bug,Instance, Traiging .

1. INTRODUCTION

Generally software companies spend a gud sum of amount for the repairement of bug or to be fixed it with.. Multi or vast software projects have bug repository that keeps all the information related to the bugs and is very well maintained for further processing. In the case of bug repository, each software bug has its own report and is also known by the name bug data. It keeps track of textual information of the bug and the updates on the basis of the status of bug fixing, which is available in historical bug data set accordingly.

Researchers proposed various techniques to have a look into the bug-report analysis. Certain open source software projects keep on updating about about the defects and issues facing in the field as a record, also keep a suggestion about possible modification with enhancement. Due to the araising of the everyday bug,the maintenance is getting much tougher.

Two challenges that comes face to face is that namely the huge amount of data and less quality.Manual bug triaging by a human bug triager is a vast process and the same time it may find difficulty due to the errors which occurs by the arrival of huge amount of bug data and less availability of developers who has an accurate knowledge of the bug to be fixed with. In some former methods , the manual assigning process is does,whenever the bug is found as such.Those developers status is stored in historical recored as item assigned .

The data truncation holds for the betterment of traiging. The mingled techniques such as instance selection along with that of feature selection is used to truncate the dimensions of bug and the data. The new version consist with similar meaning of old one with less data amount.

The application of the processing is done in its evident way is that for the truncation purpose and the quality maintenance afford the feature and instance process of selectiong.prediction is done further and for that classifier is used.Here both Naïve and Random techniques are used. Random forest is also used for predcition, because Rf itself is a multiclassifier and provide more accurate and better result, when it comes to the case of assigning developer. comparitive result is also provided.

2. LITERATURE SURVEY

In [1] they had a deeper look on to the malfunctioning of the web scripts as well as the crashes that occurs along with the functioning. A derived solution for that serious approach was Test generation in dynamic for Automatic test is done, then captures logical on inputs , reduction in test failure .At the same time it is simple and safe too. An Apollo tool with PHP Programming is specified.

- *Krishnaveni V.l is currently pursuing masters degree program in computer science and engineering in Mohandas College of Engineering, Kerala University,Kerala, India, PH-+918281890776. E-mail: anjalishivsai@gmail.com*
- *Karthik.M is working as Assistant Professor of computer science and engineering in Mohandas College of Engineering, Kerala University,Kerala, India, PH-+919496327831. E-mail: karthikcpta@gmail.com*

In [2] they mentioned bug triaging and also gave a deep focusing look into the matter as a whole, which is of "Bug Triage and Resolution Practices", mostly the analysis is does on the basis of the root cause analysis of reassignment and among those bugs which were re-opened .length variation occurs in data, it is printed and splitted.. They also planned to perform a sorting with card, keeping in touch with response of interview as a medium to lessen those level and also keep focusing to identify the common themes in the participant's feedback methodology.

In [3], Bug tossing history approach is mostly concentrated over here by markov chain methodization. Primarly it gives clear idea about networking structure. It essentializes the probability of tossing between developers from the history which is available in bug tracking systems .

In [4] study of optimization and the accuracy is does and it leads to a solution which leads to the recommendation .First, reformulate the problem as an optimization problem of both accuracy and cost then Second, CBF is coping up with the currently using one with a (CF) was the best-chosed option, and modify the recommendation quality of either approach alone.

In[5], the approach does is semi-supervised which is far better when compared to the previous methods , The mixed usage of naive classifier and the expectation in the maximization has been considered , both labeled and unlabeled bug reports . The techniques offer some contributions such as bug triaging by semi- supervised approach and weight recommendation list is maintained.

In [6] they addressed the data truncation and quality. The triaging of the bug lights on to the matter of assignment of developer who is well capable of handling it so . For e.g., the new bug reports or will-not-fix bug reports.

The mingled techniques of instance s election with feature selectioning for the job of reduction in data scaling on the behalf dimension of bug and data is given for reference. To getting the order of applying selections, the accurate extracted information of datas from the historically maintaining data is considered and create a model for prediction.

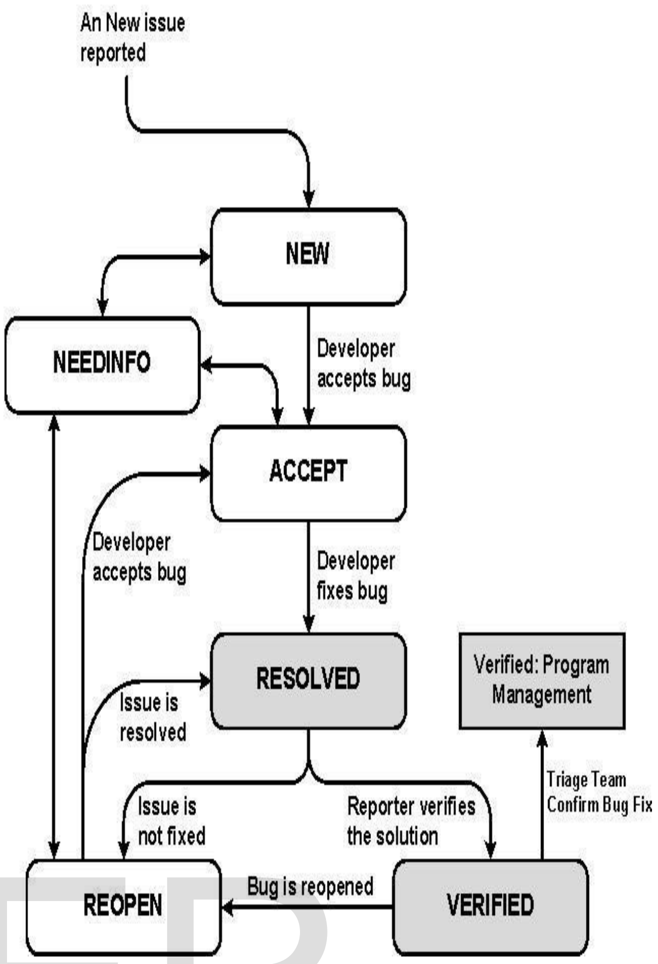


Fig 1. BUG LIFE CYCLE

3. PROPOSED SYSTEM / MODULE DESCRIPTION

Manual Bug fixing is not worth of time usage and never provides accurate result. To come up with the difficulties this proposed system is been introduced. There is problem of getting an exact bug solution according to the domain as per. According to the previous approach, the automatic triaging is done with reduced set but the naïve bayes classifier couldn't fulfil the multi classification and accuracy in developer assignment. To pay attention towards a precise predication ,existing system instance selection and feature selection for reducing bug dataset is done. Furtherly does clustering in bug data inorder of reporter, developer etc. Finally classifies with Random forest algorithm in predction , for the improvement of data reduction quality by the means of result when compares to that of existing system and get domain wise bug solution in automatic bug traiging.

A.Architecture

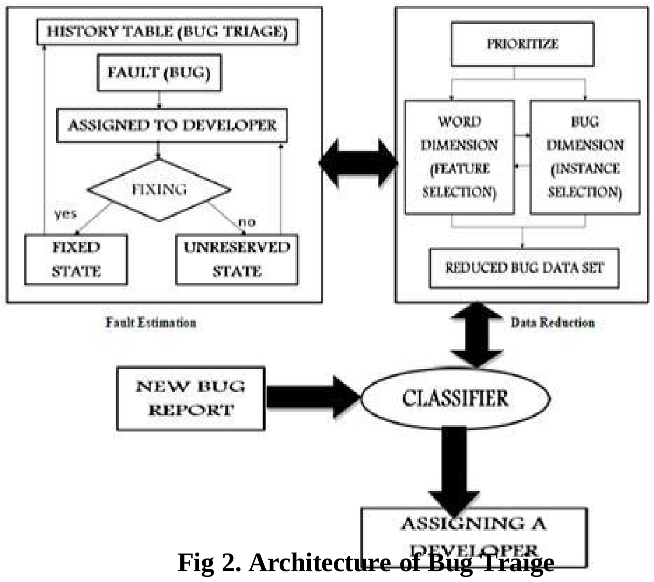


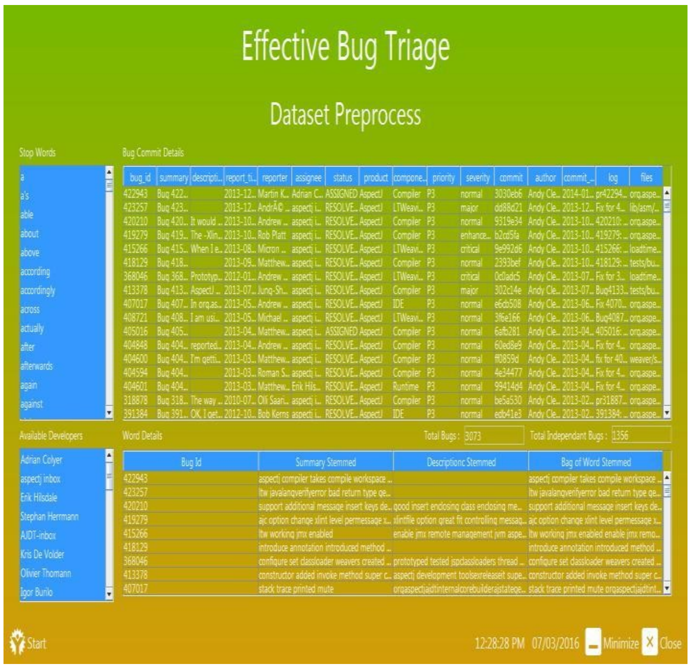
Fig 2. Architecture of Bug Traige

The architecture is divided into 3 parts as Historical data repository as primary stage,next moves on with the selection processing and prediction orders.Ends with the classifier application for developer assignment. Both data truncation as well as the redundancy removal is paid with more concentration too.

a)Assignment of a apt developer for the newly reported bug in maintenace is generally mentioned as traiging of bug.The Repository will provide the status whether it could be rectified or not and also in bug repository previous bug data details are also present for the reference purpose[5].

b) Quality improvement of data is does by applying instance as well as the feature selectioning. Finally we will be awarded with atmost quality data that too a truncated one.

c)original data set is classified with the upcoming new bug report and this job is carried out by none other than the classifier. According to the performance in solving the bugs, the priority of the developer status is also maintained. Classifier is applied to the data which is available with us that was truncated before itself through the Is and Fs techniques[9] . If the expert solved only very few bugs and remains idle then the expert is removed from that priority list too.



Effective Bug Triage

Dataset Preprocess

Stop Words

a
a's
able
about
above
according
accordingly
across
actually
after
afterwards
again
against

Bug Commit Details

bug_id	summary	descripti...	report_ti...	reporter	assignee	status	product	compone...	priority	severity	commit	author	commit_...	log	files
422943	Bug 422...		2013-12...	Martin K...	Adrian C...	ASSIGNED	AspectJ	Compiler	P3	normal	3030eb6	Andy Cle...	2014-01...	pr42294...	org.aspe...
423257	Bug 423...		2013-12...	AndrÃ© ...	aspectj i...	RESOLVE...	AspectJ	LTWeavi...	P3	major	dd88d21	Andy Cle...	2013-12...	Fix for 4...	lib/asm/...
420210	Bug 420...	It would ...	2013-10...	Andrew ...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	normal	9319e34	Andy Cle...	2013-10...	420210: ...	org.aspe...
419279	Bug 419...	The -Xlin...	2013-10...	Rob Platt	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	enhance...	b2cd5fa	Andy Cle...	2013-10...	419279: ...	org.aspe...
415266	Bug 415...	When I e...	2013-08...	Micron ...	aspectj i...	RESOLVE...	AspectJ	LTWeavi...	P3	critical	9e992d6	Andy Cle...	2013-10...	415266: ...	loadtime...
418129	Bug 418...		2013-09...	Matthew...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	normal	2393bef	Andy Cle...	2013-10...	418129: ...	tests/bu...
368046	Bug 368...	Prototyp...	2012-01...	Andrew ...	aspectj i...	RESOLVE...	AspectJ	LTWeavi...	P3	critical	0d0adc5	Andy Cle...	2013-07...	Fix for 3...	loadtime...
413378	Bug 413...	AspectJ ...	2013-07...	Jung-Sh...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	major	302c14e	Andy Cle...	2013-07...	Bug4133...	tests/bu...
407017	Bug 407...	In org.as...	2013-05...	Andrew ...	aspectj i...	RESOLVE...	AspectJ	IDE	P3	normal	e6cb508	Andy Cle...	2013-06...	Fix 4070...	org.aspe...
408721	Bug 408...	I am usi...	2013-05...	Michael ...	aspectj i...	RESOLVE...	AspectJ	LTWeavi...	P3	normal	3f6e166	Andy Cle...	2013-06...	Bug4087...	org.aspe...
405016	Bug 405...		2013-04...	Matthew...	aspectj i...	ASSIGNED	AspectJ	Compiler	P3	normal	6afb281	Andy Cle...	2013-04...	405016: ...	org.aspe...
404848	Bug 404...	reported...	2013-04...	Andrew ...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	normal	60ed8e9	Andy Cle...	2013-04...	Fix for 4...	org.aspe...
404600	Bug 404...	I'm getti...	2013-03...	Matthew...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	normal	ff0859d	Andy Cle...	2013-04...	fix for 40...	weaver/s...
404594	Bug 404...		2013-03...	Roman S...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	normal	4e34477	Andy Cle...	2013-04...	Fix for 4...	org.aspe...
404601	Bug 404...		2013-03...	Matthew...	Erik Hils...	RESOLVE...	AspectJ	Runtime	P3	normal	99414d4	Andy Cle...	2013-04...	Fix for 4...	org.aspe...
318878	Bug 318...	The way ...	2010-07...	Olli Saari...	aspectj i...	RESOLVE...	AspectJ	Compiler	P3	normal	be5a530	Andy Cle...	2013-02...	pr31887...	org.aspe...
391384	Bug 391...	OK, I get...	2012-10...	Bob Kerns	aspectj i...	RESOLVE...	AspectJ	IDE	P3	normal	edb41e3	Andy Cle...	2013-02...	391384: ...	org.aspe...

Available Developers

Adrian Colyer
aspectj inbox
Erik Hilsdale
Stephan Herrmann
AJDT-inbox
Kris De Volder
Olivier Thomann
Igor Burilo

Word Details

Total Bugs : 3073 Total Independant Bugs : 1356

Bug Id	Summary Stemmed	Descriptionc Stemmed	Bag of Word Stemmed
422943	aspectj compiler takes compile workspace ...		aspectj compiler takes compile workspace ...
423257	ltw javalangverifyerror bad return type ge...		ltw javalangverifyerror bad return type ge...
420210	support additional message insert keys de...	good insert enclosing class enclosing me...	support additional message insert keys de...
419279	ajc option change xlint level permessage x...	xlintfile option great fit controling messag...	ajc option change xlint level permessage x...
415266	ltw working jmx enabled	enable jmx remote management jvm aspe...	ltw working jmx enabled enable jmx remo...
418129	introduce annotation introduced method ...		introduce annotation introduced method ...
368046	configure set classloader weavers created ...	prototyped tested jspclassloaders thread ...	configure set classloader weavers created ...
413378	constructor added invoke method super c...	aspectj development toolsexreleaseit supe...	constructor added invoke method super c...
407017	stack trace printed mute	orgaspectjajdtinternalcorebuilderajstatege...	stack trace printed mute orgaspectjajdtint...

Fig 3. Refined Data by IS and FS Algorithm

B. ***Algorithm***

```
Algorithm 1. Data reduction based on FS → IS
Input:   training set $\mathcal{T}$ with $n$ words and $m$ bug reports,
         reduction order FS→IS
         final number $n_F$ of words,
         final number $m_I$ of bug reports,
Output:  reduced data set $\mathcal{T}_{FI}$ for bug triage
1) apply FS to $n$ words of $\mathcal{T}$ and calculate objective values
   for all the words;
2) select the top $n_F$ words of $\mathcal{T}$ and generate a training
   set $\mathcal{T}_F$;
3) apply IS to $m_I$ bug reports of $\mathcal{T}_F$;
4) terminate IS when the number of bug reports is equal to
   or less than $m_I$ and generate the final training set $\mathcal{T}_{FI}$.
```

• **IS(Instance selection**) – process of developing much needed subset of instances removing stem words, bag of words etc (i.e., bug reports filteration as well bug data) .
• **FS(feature selection**) which focus towards the miniature form of features(i.e.,filtering by words).

IS and FS is mingled so that we could achieve both quality data in a miniature form.Before application was done separately but it never fulfilled neither accuracy nor truncation.Thus here it is used together to achieve best. Reduction order is an important criteria for prediction.

4. SYSTEM ANALYSIS/ EXPERIMENTAL RESULTS

The processing is divided into 4 sections:-

1. Once a new bugset is provided,we can have a look on to the developer status from bug repository.If that kind of bug has never been handles with any of the developer then we shall move further.

2. In the 2nd section,we can apply Is and Fs for redundancy check and as well as the dataset truncation.At the end of this processing we will be awarded with truncated data with a good quality.

3. Next we can identify through clustering, how many bugs are rectified by the developer's. It will give complete status about the bugs to the admin or the reporter and there by we will get knowledge on the bug status.Move on to prediction orders too.

4. Finally we shall apply classifier for prediction orders with the help of historical data set and the data which is available with us after those earlier processing.Both Naive and Random forest algorithm is used and does the comparison too.random forest is applied to get more precision while assigning the developer.

A. NAIVE BAYE'S

A well familiar one used for features/predictors in a learning problem. Maximum processing or training is does expression of closewardness while evaluvating. It probably leap, instead of approximating with high costage like other types of classifiers as such.

1. **Learning Phase:-** Given a training set of **F** features and **L** classes.

2. **Testing Phase:-** Look up tables to assign the label

B. RANDOM FOREST

Random forest technique pves the individual tree class output mode. Mainly a approach of tree with decision processing is does.

1. Create training set:-

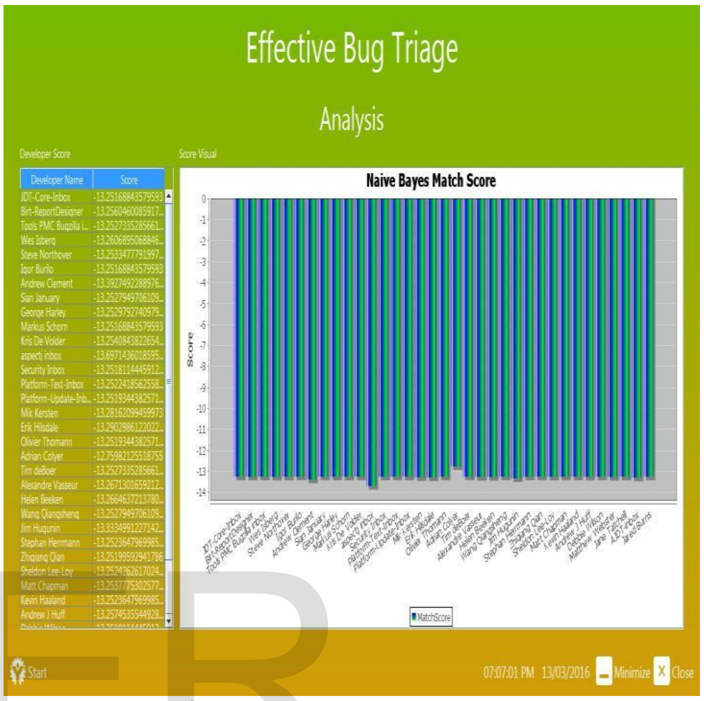
Random n sample is selected with replacement of N available sample tainee.

2. Create decision rule at node of each tree:-

○ Randomly process and select m features for the decision at that node.

○ Select feature (from m available) that maximizes information gain in the set of training samples.

3. Tree are grown completely and not at all pruned.



Developer Name	Score
JDT-Core-Inbox	-13.25168843579593
Birt-ReportDesigner	-13.2560460085917...
Tools PMC Bugzilla i...	-13.2527335285661...
Wes Isberg	-13.2606895068846...
Steve Northover	-13.2533477791997...
Igor Burilo	-13.25168843579593
Andrew Clement	-13.3927492288976...
Sian January	-13.2527949706109...
George Harley	-13.2529792740979...
Markus Schorn	-13.25168843579593
Kris De Volder	-13.2540843822654...
aspectj inbox	-13.6971436018595...
Security Inbox	-13.2518114445912...
Platform-Text-Inbox	-13.2522418562558...
Platform-Update-Inb...	-13.2519344382571...
Mik Kersten	-13.28162099459973
Erik Hilsdale	-13.2902986122022...
Olivier Thomann	-13.2519344382571...
Adrian Colyer	-12.75982125518755
Tim deBoer	-13.2527335285661...
Alexandre Vasseur	-13.2671301659212...
Helen Beeken	-13.2664637213780...
Wang Qiangsheng	-13.2527949706109...
Jim Hugunin	-13.3334991227142...
Stephan Herrmann	-13.2523647969985...
Zhiqiang Qian	-13.25199592941786
Sheldon Lee-Loy	-13.2524262617024...
Matt Chapman	-13.2537775302577...
Kevin Haaland	-13.2523647969985...
Andrew J Huff	-13.2574535544928...

Fig 4. Naïve Baye's Output

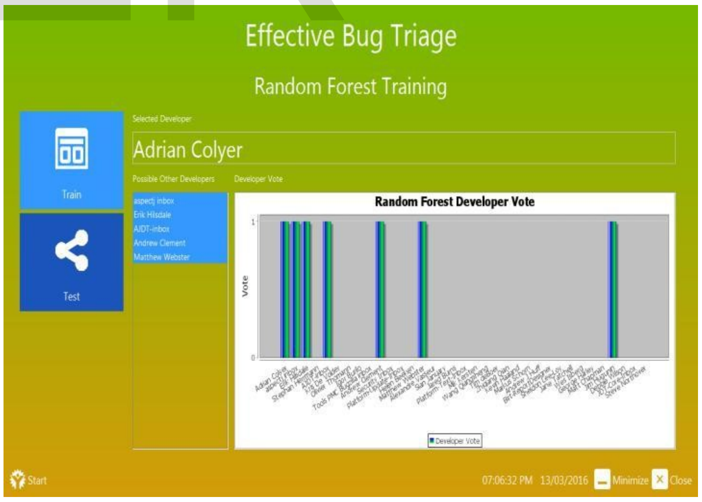
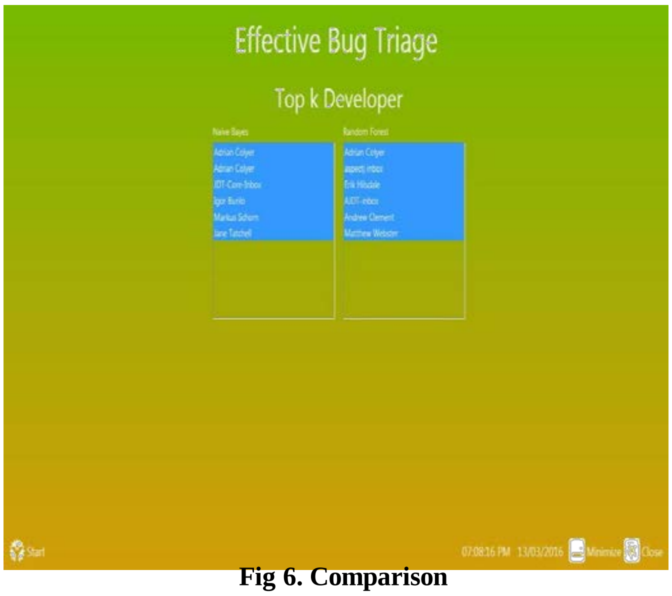


Fig 5. Random Forest Output



Naive Bayes	Random Forest
Adrian Colyer	Adrian Colyer
Adrian Colyer	aspectj inbox
JDT-Core-Inbox	Erik Hilsdale
Igor Burilo	AJDT-inbox
Markus Schorn	Andrew Clement
Jane Tatchell	Matthew Webster

Fig 6. Comparison

5. CONCLUSION

Software world faces much difficulty in the case of bug maintenance,thereby it led to drastic loss in time and money.mingled usage of feature and instance selection pave the benefit in providing quality in data and the truncation of it.Innovative usage of Classifier for prediction order award with more accuracy and precision in assigning the developer for a newly reported bug.

References

[1] S. Artzi, A. Kie_ zun, J. Dolby, F. Tip, D. Dig, A.Paradkar, and M. D. Erns t, “Finding bugs in web applications us ing dynamic tes t generation and explicit-state model checking,” IEEE Softw., vol. 36,no. 4, pp. 474–494, Jul./Aug 2010.

[2] Olga Bays al, Reid Holmes , and Michael W. Godfrey David R. Cheriton ,”Revis iting Bug Triage and Res olution Practices ”, School of Computer Science Univers ity of Waterloo Waterloo, ON,july 2009.

[3] G. Jeong, S. Kim, and T. Zimmermann, “Improving training s et reduction for bug triage,” in Proc. 35th bug triage with tos s ing graphs ,” in Proc. Joint Meeting 12th Eur. Softw. Eng.Conf. 17th ACM SIGSOFT Symp. Found. Softw. Eng., Aug. 2009, pp. 111–120.

[4] J. W. Park, M. W. Lee, J. Kim, S. W. Hwang, and S. Kim,“Cos triage: A cos t-aware triage algorithm for bug reporting s ystems ,” in Proc. 25th Conf. Artif. Intell., Aug. 2011, pp. 139–144.

[5] Katja Kevic, Sebas tian C. Muller, Thomas Fritz, and Harald C. Gall “Collaborative Bug Triaging using Textual Similarities and Change Set Analys is ”, Department of Informatics Univers ity of Zurich, Switzerland.

[6] J. Xuan, H. Jiang, Z. Ren, J. Yan, and Z. Luo, “Automatic bug triage us ing s emi-s upervis ed text class ification,” in Proc. 22nd Int. Conf. Softw. Eng. Knowl. Eng., Jul. 2010, pp. 209– 214.

[7] D.Cubrani_c and G. C. Murphy, “Automatic bug triage us ing text categorization,” in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng.,Jun. 2004, pp. 92– 97.

[8] Towards Effective Bug Traige With Software Data Reduction Techniques ,Jifeng Xuan, He Jiang, Member, IEEE, Yan Hu, Zhilei Ren, Weiqin Zou,Zhongxuan Luo, and Xindong Wu, Fellow, IEEE,IEEE trans action on Dta Knowledge and Engineering.

[9] X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun, “An approach to detecting duplicate bug reports us ing natural language and execution information,” in Proc.30th Int. Conf. Softw. Eng., May 2008,pp. 461–470.

[10] J. Xuan, H. Jiang, Z. Ren, and Z. Luo, “Solving the large s cale next release problem with a backbone bas ed multilevel algorithm,”IEEE Trans . Softw. Eng., vol. 38, no. 5, pp. 1195–1212, Sept./Oct.2012.

[11] J. Xuan, H. Jiang, Z. Ren, and W. Zou, “Developer prioritization in bug repos itories ,” in Proc. 34th Int. Conf. Softw. Eng., 2012, pp. 25–35.

[12] Y. Yang and J. Peders en, “A comparative s tudy on feature s election in text categorization,” in Proc. Int. Conf. Mach. Learn., 1997, pp. 412–420.

[13] H. Zhang, L. Gong, and S. Vers teeg, “Predicting bug-Fixing time:An empirical s tudy of commercial s oftware projects ,” in Proc. 35th Int. Conf. Software En g.,May 2013, pp. 1042–1051.

[14] W. Zou, Y. Hu, J. Xuan, and H. Jiang, “Towards Annu. IEEE Int. Comput. Soft. Appl. Conf., Jul. 2011, pp. 576–581.

[15] A. K. Farahat, A. Ghodsi, M. S. Kamel, “Efficient greedy feature selection for unsupervised learning,” Knowl. Inform. Syst., vol. 35,no. 2, pp. 285–310, May 2013.