1. INTRODUCTION:
Bug triage aims to assign an appropriate developer to fix a new bug, i.e., to determine who should fix a bug. A bug repository (a typical software repository, for storing details of bugs), plays an important role in managing software bugs. Software bugs are inevitable and fixing bugs is expensive in software development. Software companies spend over 45 percent of cost in fixing bugs. Large software projects deploy bug repositories (also called bug tracking systems) to support information collection and to assist developers to handle bugs. In a bug repository, a bug is maintained as a bug report, which records the textual description of reproducing the bug and updates according to the status of bug fixing. A bug repository provides a data platform to support many types of tasks on bugs, e.g., fault prediction, bug localization, and reopened bug analysis. Bug reports in a bug repository are called bug data.

Most open source software developments incorporate an open bug repository that allows both developers and users to post problems encountered with the software, suggest possible enhancements, and comment upon existing bug reports. One potential advantage of an open bug repository is that it may allow more bugs to be identified and solved, improving the quality of the software produced.

2. RELATED WORK:
Our approach is used to triage the bug particular user and assign the bug, also bug reduction so performance of the user, developer are increased reduction order and each attribute is helpful to the prediction.

The primary contributions of this paper are as follows:
1) We address the problem of data reduction for bug triage.
2) We propose a combination approach to addressing the problem of data reduction. This can be viewed as an application of instance selection and feature selection in bug repositories.
3) We build a binary classifier to predict the order of applying instance selection and feature selection. For that we use Fs->Is and Is->Fs algorithm

Problem Statement:
Address and assign the bug to the developer depending on the history of developer bug which bug they solve and how effectively.

3. SYSTEM IMPLEMENTATION:
System Implementation consist of various parts described as follows:

We are implementing our project by using Java Technology and MySQL database.
We are going to implement following modules for achieving our propose system:

Assign bug to the developer:
Is a process to assign a priority to each developer in a bug repository and to rank all the contributions of developers to assist software tasks.

Feature selection:
Feature selection returns a subset of the features. Feature selection techniques are often used in domains where there are many features and comparatively few samples (or data points).

Instance selection:
Instance selection returns a subset of relevant instances is a technique to reduce the number of instances by removing noisy and redundant instances.

Algorithm:-
Data reduction based on FS->IS

INPUT:
1) Training set T with n words and m bug report
2] Reduction order feature selection then instance selection

3] final number n of words

4] final number m of bug reports

OUTPUT:
Step1] :
• Reduce dataset TFI for bug triage for all the words
• Apply FS to n words of T and calculate objective values

Step2] :
• Select the top nF words of T and generate a training set TF

Step3] :
• apply IS to mI bug reports of TF ;

Step4] :
• terminate IS when the number of bug reports is equal to or less than mI and generate the final training set TFI

Where
F=Feature selection
I=Instance selection
TFI=final data after feature and instance selection

4. MODULE FLOW:

<table>
<thead>
<tr>
<th>Table 1. Fixed Bug file report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug File</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>File 1</td>
</tr>
<tr>
<td>File 2</td>
</tr>
<tr>
<td>File 3</td>
</tr>
</tbody>
</table>

This module shows four parts as follow:

1) Firstly it will show how many bug files are not assigned to any developer. It will give complete status about the bugs to the admin so that he will come to know which bugs are not assigned yet.

2) Secondly it will show how many bugs are not assigned to any developer. It will give complete status about the bugs to the admin so that he will come to know which bugs are assigned.

3) Thirdly it will show how many bugs are rectified by the developers. It will give complete status about the bugs to the admin so that he will come to know which bugs are rectified completely.

4) Fourthly it will show how many bugs are not rectified by the developers. It will give complete status about the bugs to the admin so that he will come to know which bugs are not rectified yet.

5. CONCLUSION:
Bug triage is an expensive step of software maintenance in both labor cost and time cost. In this paper, we combine feature selection with instance selection to reduce the scale of bug data sets as well as improve the data quality. We empirically investigate the data reduction for bug triage in bug repositories of two large open source projects, namely Netbeans and Mozilla.

In future work, we plan on improving the results of data reduction in bug triage to explore how to prepare a high-quality bug data set and tackle a domain-specific software task. For predicting reduction orders, we plan to pay efforts to find out the potential relationship between the attributes of bug data sets and the reduction orders.

6. REFERENCES:
4. J. Arvink, L. Hiew, and G. C. Murphy, “Who should fix this bug?” in Proc. 28th Int.