When Would This Bug Get Reported?

Presented By:-

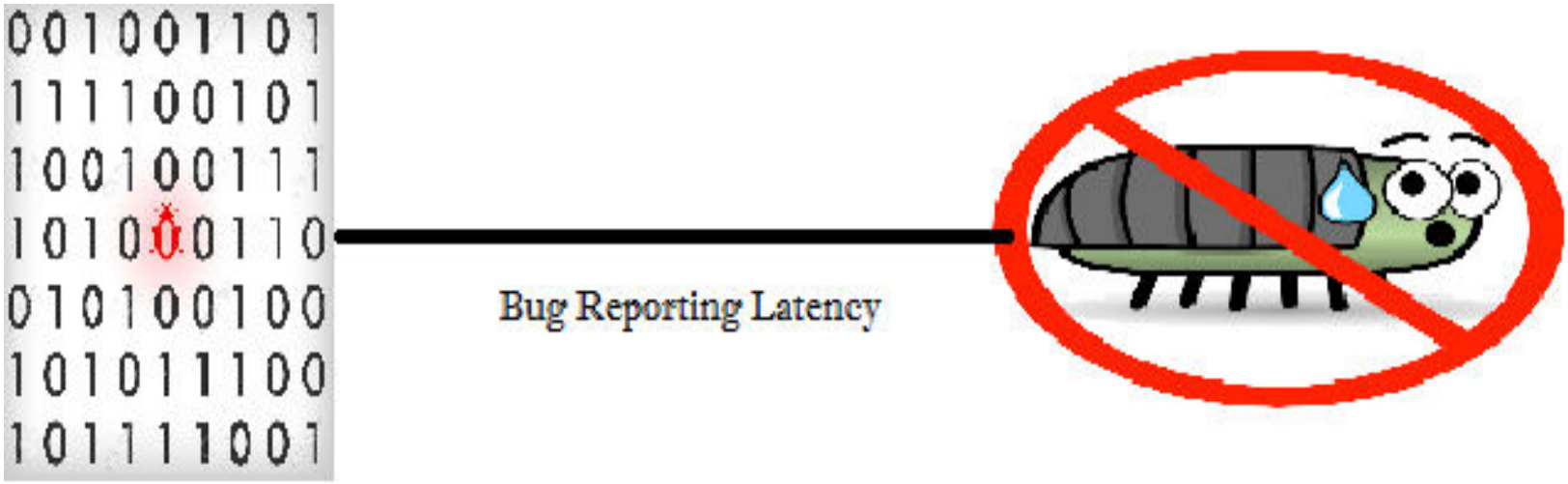
Sheenam Sharma

Main Idea

The author creates a classifier which identifies a bug under “early” or “late” label using bug reporting latency and feature extraction. This prediction model was validated using AUC.

190 bugs from 3 Java software systems were analyzed: AspectJ, Rhino, Lucene using iBugs and JIRA systems.

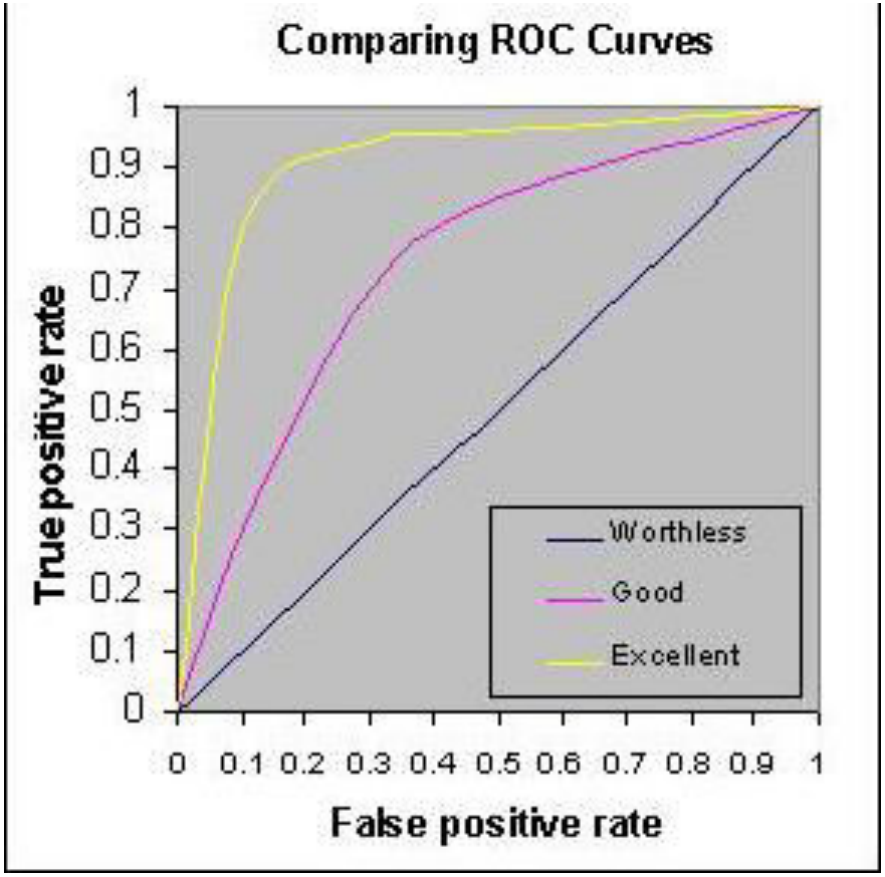
Bug Reporting Latency



Feature Extraction

It is a process of extracting features which are informative and non redundant which facilitates subsequent learning for building discriminative models.

AUC


Comparing ROC Curves
True positive rate
False positive rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Worthless
Good
Excellent

What is the need of such classifier?

Basically, help the developers in decision making. That is, which bugs should be fixed before and after the release.

Bugs with a “late” label can be fixed after the release and the ones with “early” label are preferred to be fixed before the release.

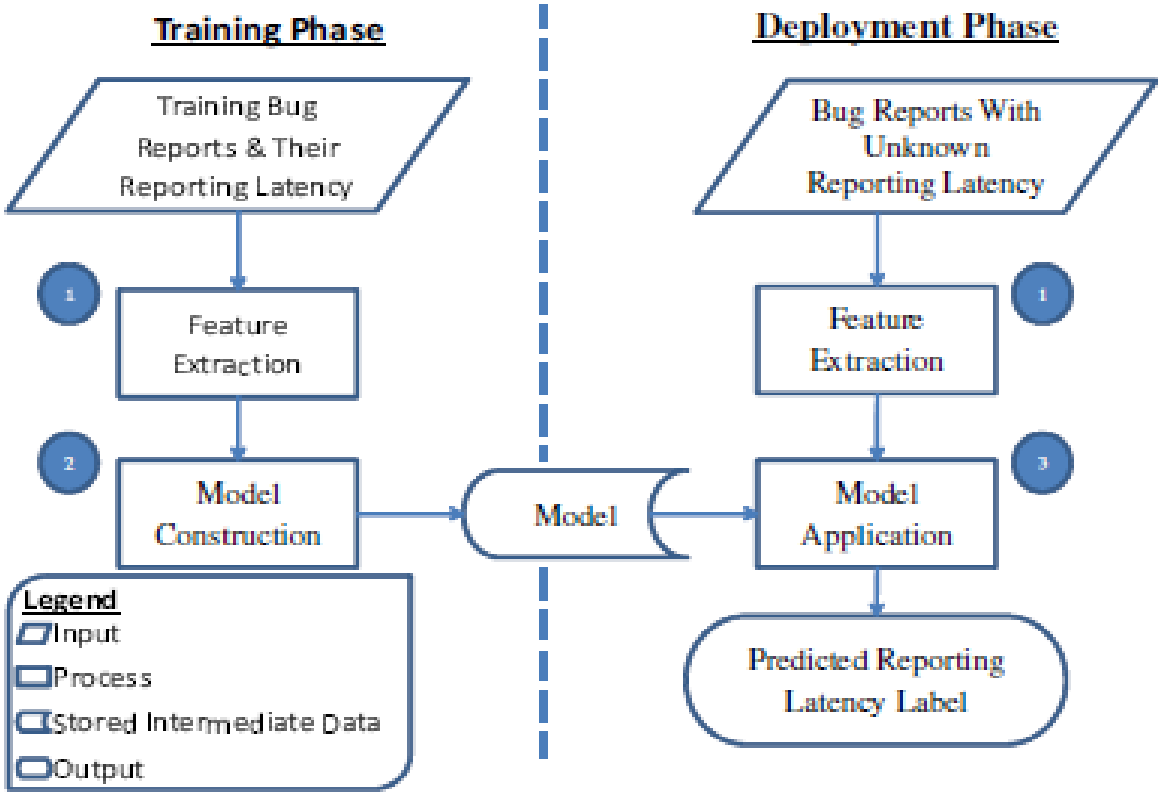
How their work is different?

- Prioritize bugs before the release and not after the release
- Bug reporting latencies are taken into account
- Investigate the buggy code and not focus on the bug reports

Approach followed

Supervised learning is used to build a classifier for prediction. Training set is used to train the SVM classifier. Then the model generated is used for testing new bugs.

The author divides the full process as “training” and “deployment” phase.


Training Phase
Deployment Phase
Training Bug Reports & Their Reporting Latency
Bug Reports With Unknown Reporting Latency
1
Feature Extraction
1
Feature Extraction
2
Model Construction
Model
3
Model Application
Predicted Reporting Latency Label
Legend
Input
Process
Stored Intermediate Data
Output

- Faulty lines are found manually using bug report messages.
- Time for reporting bugs is also stored from the bug report messages.
- Using git blame and annotate, time when the bug was induced in the code is stored.
- Latency time is calculated.
- Features are extracted for each bug and stored as vectors.
- The above mentioned vectors are labelled as early or late.

- SVM is trained using these vectors.
- Data is trained and tested using 10 fold cross validation.

Note:

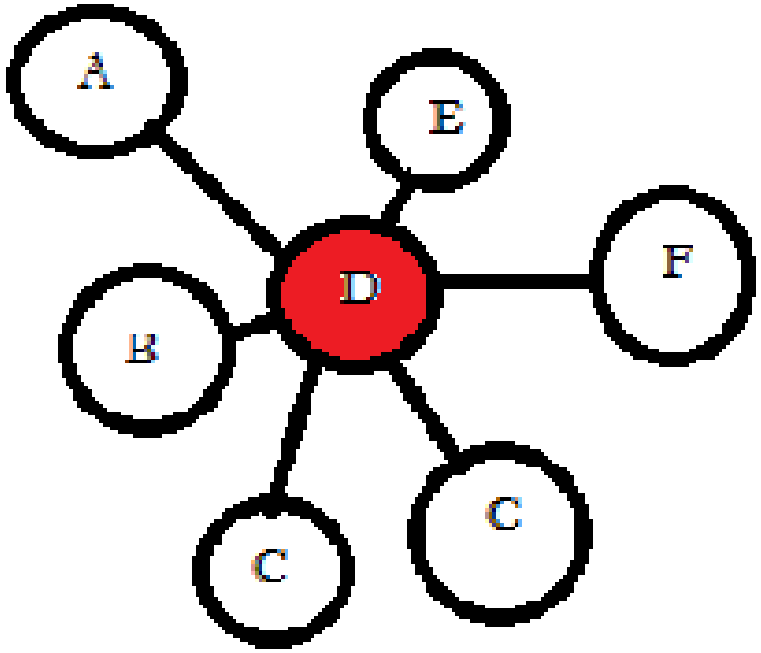
label “short” -> less than or equal to 30 days

label “long” -> more than 30 days

Feature extraction key points

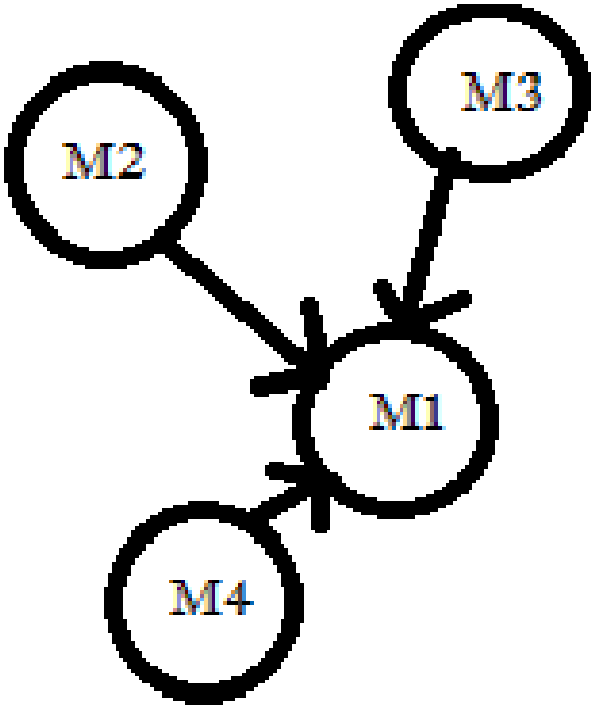
- Program element count: Count for methodDeclarationNode, methodInvocationNode, conditionNode, loopNode, assignmentNode, returnNode, and tryNode is taken into consideration.
- Betweenness centrality: To calculate the importance of a piece of code.
- Page rank: To calculate how many times a piece of code is called.

Betweenness Centrality



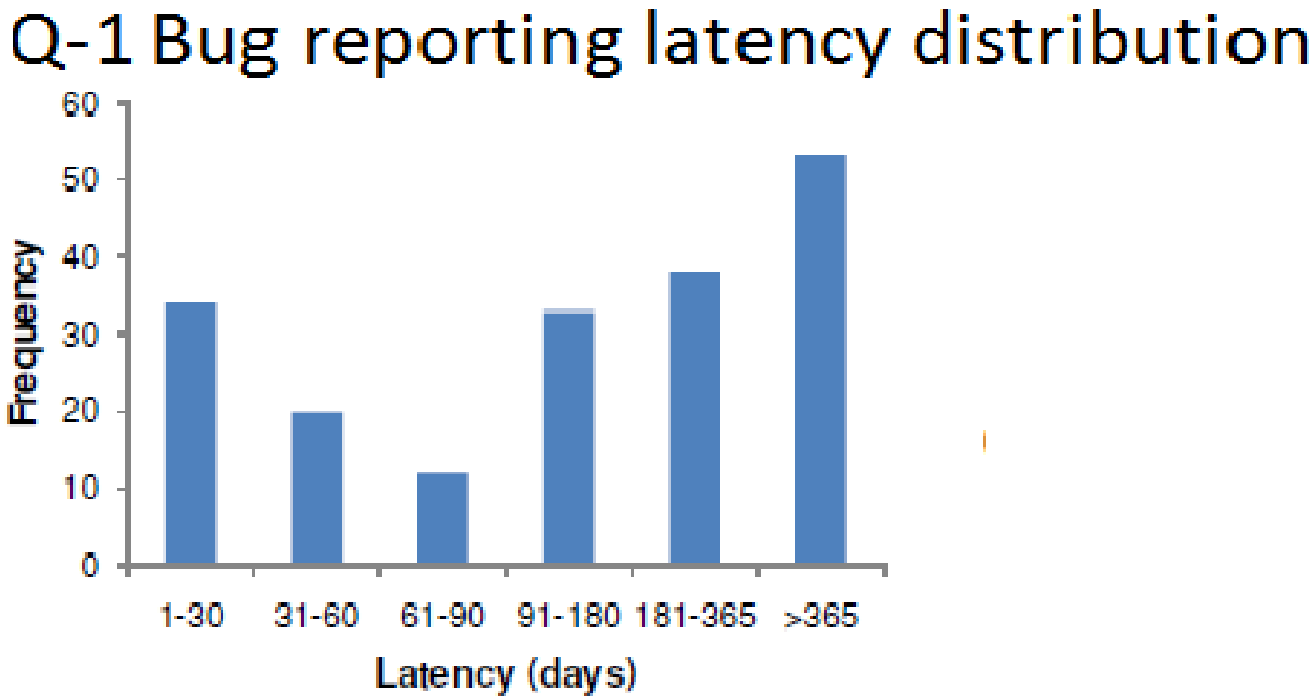
D node depicts the faulty node

Page Rank



Page Rank for method "M1" is 3 in this example as it is called by M2, M3, and M4

Research questions to validate the work



Q-2 Bug reporting latency versus severity

Table II
DISTRIBUTION OF SEVERITY PER LATENCY GROUP

Group	Blocker	Critical	Major	Normal	Minor	Trivial
1-30	1	1	7	19	5	1
31-60	0	1	7	9	3	0
61-90	0	0	0	7	4	1
91-180	0	4	7	19	3	0
181-365	1	2	4	31	0	0
>365	2	3	4	36	6	2

Table III
DISTRIBUTION OF SEVERITY PER LATENCY GROUP

Group	Blocker	Critical	Major	Normal	Minor	Trivial
$\leq$ 30 days	1	1	7	19	5	1
$>$ 30 days	3	10	22	102	16	3

Q-3 Bug report latency prediction results

- As the data is skewed, AUC is used to evaluate the prediction performance. An AUC of 70.869% is achieved for the experiments performed by the authors.
- Most discriminative feature is obtained by the fisher score.
- Best classification algorithm is achieved by checking the AUC score.

Proposed future work by the author

- Extract semantic and functionality relations from the code.
- Better classification by increasing the amount of labels.
- Work on larger set of data so as to have bugs more than 190.

Proposed future work by the presenter

- Rather then using supervised learning, it will be interesting to see the results by using active learning.
- The author used oversampling to remove class imbalance problem but it does not help much in SVM. Rather we should used SMOTE technique to synthetically generate vectors.
- Use RUS boost instead of SVM algorithm.

Discussion

- In which other areas we can use classifiers.
- How the work presented in the paper could be modified?
- How is this work related to our course?


Thank
You