IEEE Indicon 2017, 15-17 Dec, 2017, IIT Rorkee, India, Paper No. 324

Aging Related Bug Prediction using Extreme Learning Machines

Lov Kumar (lovkumar505@gmail.com)[1] and Ashish Sureka (ashish.sureka@ieee.org)[2]
Thapar University[1] and Ashoka University[2]

Abstract:

1) Aging-Related Bugs (ARBs): Error-conditions associated with software aging such as memory leakage or unreleased files and locks.
2) Approach:
   i. Source code metrics and machine learning techniques used to predict aging related bugs.
   ii. Five different feature selection techniques are considered for dimensionality reduction.
   iii. SMOTE method to counter the effect of class imbalance.
   iv. Extreme Learning Machines (ELM) with three different kernels are considered for model building.

Introduction:

1) Hard to uncover aging related bugs during system testing.
2) Recent research shows that static source code metrics can be used as predictors for aging related bugs.
3) Challenges in building predictive models
   i. Imbalanced Data.
   ii. High-Dimensional Data.

Research questions:

RQ1: Is there a statistically significant difference in the prediction performance of five different feature ranking techniques? Which feature ranking technique(s) yields the best result?

RQ2: Is there any statistically significant difference in the prediction performance of the predictive model after applying SMOTE method to address the class imbalance problem?

RQ3: What is the relative performance of the ELM with three different types of kernels?

Experimental Dataset:

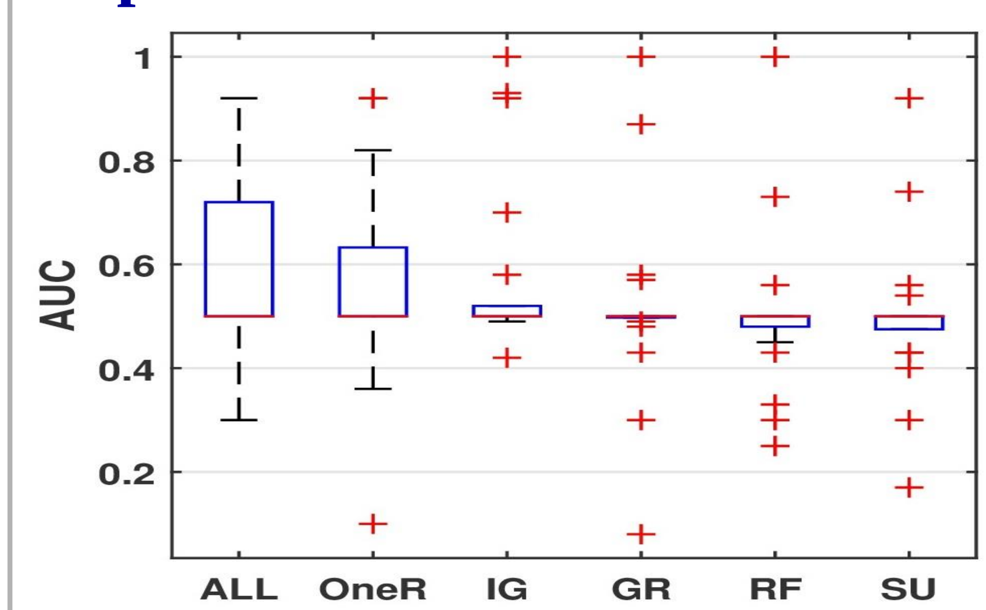
Publicly available dataset, tera PROMISE.

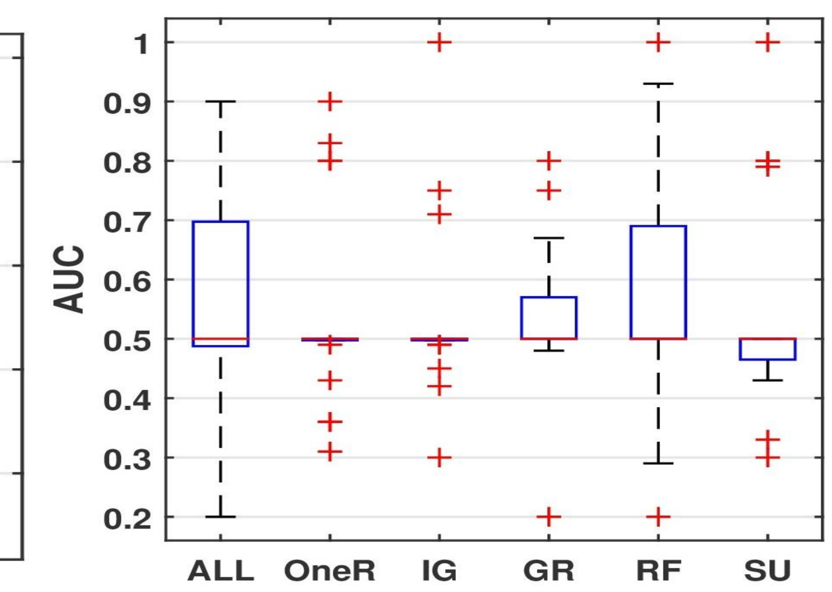
Name	# Records	Majority Class	Minority Class
Linux driver net	2292	2283	9
Linux driver scsi	962	958	4
Linux ext3	29	24	5
Linux ipv4	117	115	2

Research Framework and Proposed Technique:

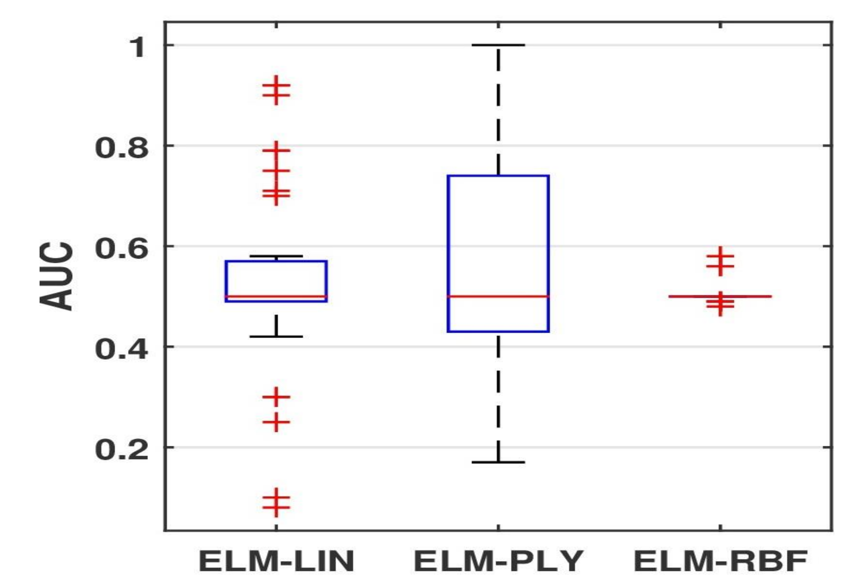
1) Feature Ranking and Selection
2) Learning from Imbalanced Dataset
3) Learning Algorithms
4) Performance Evaluation

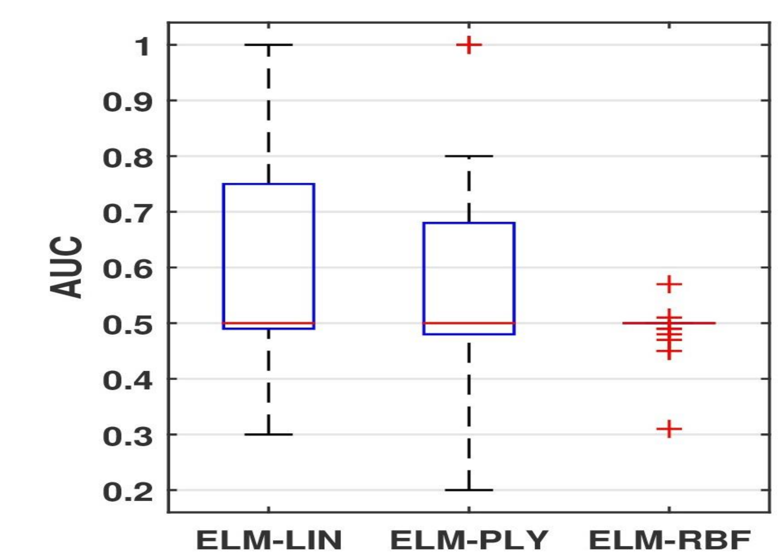
Experimental Results:





Metrics Sets: Boxplot Showing the Degree of Dispersion in the AUC Value without and with SMOTE





Kernel: Boxplot Showing the Degree of Dispersion in the AUC Value without and with SMOTE

Descriptive Statistics of the Relative Performance of SMOTE and Without SMOTE in-terms of AUC

	AUC						
	Min	Max	Mean	Median	Std Dev	Q1	Q3
Without Smote	0.080	1.000	0.541	0.500	0.169	0.500	0.540
With Somte	0.200	1.000	0.546	0.500	0.159	0.500	0.510

t-test

(a) t-test: Among Different Kernels

	p-value		
	ELM-LIN	ELM-PLY	ELM-RBF
ELM-LIN	NaN	0.63	0.00
ELM-PLY	0.63	NaN	0.00
ELM-RBF	0.00	0.00	NaN

(b) t-test: Among Different Sets of Metrics

	p-value					
	ALL	OneR	IG	GR	RF	SU
ALL	NaN	0.55	0.52	0.15	0.36	0.11
OneR	0.55	NaN	1.00	0.43	0.72	0.25
IG	0.52	1.00	NaN	0.40	0.67	0.39
GR	0.15	0.43	0.40	NaN	0.65	0.89
RF	0.36	0.72	0.67	0.65	NaN	0.64
SU	0.11	0.25	0.39	0.89	0.64	NaN

Conclusion:

1) There are several metrics which can be extracted from the software system, not all are equally important as several metrics are redundant and irrelevant.
2) Feature ranking techniques improve the performance of the predictive model in comparison to all metrics but there is no significant difference in their performance.
3) SMOTE improves the performance for ELM linear kernel but not polynomial or RBF kernel.
4) Performance of 3 different ELM kernel functions are significantly different according.

References:

1) Is software aging related to software metrics?- D. Cotroneo, R. Natella, and R. Pietrantuono, Software Aging and Rejuvenation (WoSAR), 2010.
2) datasets: from sampling to classifiers- T. R. Hoens and N. V. Chawla, Imbalanced Learning: Foundations, Algorithms, and Applications, 2013.