Data-Mining Accuracy Boost Technique for Software Deformity Prophecy Datasets Model

Maaz Rasheed Malik[1] Liu Yining[2] and Salahuddin Shaikh[3]

[1] Guilin University of Electronic Technology, Guilin, China
[2] Guilin University of Electronic Technology, Guilin, China
[3] North China Electric Power University, Beijing, China
dr.maazmalik@outlook.com
ynliu@guet.edu.cn
engineersalahuddin@gmail.com

Abstract. Data Mining is characterized as data mining or an endeavor to extricate significant and valuable data on a huge database. Data mining investigation can be utilized to settle on business choices that would improve cost, income and operational productivity of human services industry while keeping up elevated levels of patient consideration. In machine learning area, Software Deformity Prophecy datasets model is actuated on the arrangement of preparing information which capacities dependent on a lot of rules to separate among defected and non-defected datasets model. In this manner, the capacity of these Software Deformity Prophecy datasets model to classify a module is fundamentally a component of the nature of preparing dataset. In order to boost the accuracy of classification model for Software Deformity Prophecy datasets model, the most proper methods in each progression of preprocessing is Discretization. A data-preprocessing technique called Discretization we have used in our research for software deformity prophecy datasets model. This is our classification accuracy boost technique for our software deformity prophecy datasets model. For observation and analysis, we have used multiple classifiers for getting the boost accuracy with the help of discretization preprocess method. All experiments analysis clearly showed that all classifiers cannot be perfect for the accuracy and efficiency in software deformity prophecy datasets models. In case of correctly classified instances where we easily can judge the improvement of every classifiers that decision stump, hoeffding tree and lmt, their efficiency and accuracy is not very good but increased as compare to use these without discretization way. The position of stacking is quite bad and not good to use in these experiments because their efficiency and accuracy have not increased but seems to be worst in all case.

Keywords: Data Mining, Defect-prone, Classification, Machine Learning, Software-defect, Bug; Software-model, Data-Pre-process.

1 Introduction

The enormous measure of data that are put away in databases contains important shrouded knowledge which causes the client to improve the presentation of basic leadership process. Data mining is a multidisciplinary field of science, depicting work territories that incorporate database innovation, machine learning, insights, knowledge-based systems, pattern recognition, superior processing, data recovery, artificial intelligence, artificial neural networks and data perception. Data mining is characterized as data mining or an endeavor to extricate significant and valuable data on a huge database. Data mining investigation can be utilized to settle on business choices that would improve cost, income and operational productivity of human services industry while keeping up elevated levels of patient consideration. There are a few significant data mining systems have been created and utilized in data mining. Data mining methods are utilized in social insurance the executives for, diagnosis and treatment, healthcare resource management, customer relationship management and fraud and anomaly detection. Data mining strategies can assist physicians with distinguishing powerful medicines and best practices, and patients get better and progressively reasonable social insurance administrations. Data mining is the procedure of semi-consequently investigating huge databases to discover patterns. Data mining is about design of difficulty by look at and decide the data that previously existed in the databases. It finds the significant data covered up in huge volumes of data. Data mining is likewise portrayed as the arrangement of activity of revealing patterns in data.
The conceivable utilization of data mining strategy characterizes that the methodology wherein an archive of data can be used may extend a long way past what was see when the data was at first assembled. Bunches of utilizations in machine learning to data mining as appeared in the comprehension. The significant knowledge structures that are picked up, the central clarification, are at any rate as essential, and often especially increasingly generous contrast with the ability to achieve well on new models.
In machine learning area, software deformity prophecy datasets model is actuated on the arrangement of preparing information which capacities dependent on a lot of rules to separate among defected and non-defected datasets model. In this manner, the capacity of these software deformity prophecy datasets model to classify a module is fundamentally a component of the nature of preparing dataset. Be that as it may, information accumulation accompanies a few difficulties in grouping, preparing, putting away, and recovery. These procedures as per have a higher capability of causing information quality troubles, for example, include repetition and superfluity, information model clash, class imbalance, the nearness of commotion, and anomaly.
Having the option to software deformity prophecy datasets model which parts are bound to be imperfection inclined backings better focused on resting assets and along these lines improved productivity. Lamentably, classification stays a generally unsolved issue. So as to address this, analysts have been utilizing progressively software deformity prophecy datasets model plans that incorporate data-preprocessing, attribute selector and learning algorithms. The primary issue is how to choose the best software deformity prophecy datasets model plan, as per explicit dataset? In reality, doesn't exist a recommendation that utilizations genetic algorithm with the target to

choose the best software deformity prophecy datasets model plan setup utilizing a particular dataset. The software deformity prophecy datasets model plans have a significant issue, it is the means by which to choose a right mix of data-preprocessing, attribute choice and learning algorithm for a specific dataset. Different deliberates on software deformity prophecy datasets model have been accounted for by numerous scientists. A significant examination displayed a model that can anticipate the flawed module based on intricacy metric like size multifaceted nature. The examination proposed an immediate connection between the multifaceted nature of code and imperfections. In spite of the fact that the analyst's investigations were not able present an exact model, it brought up many research issues. The investigations additionally reasoned that better metric for software deformity prophecy datasets model should be recognized and a standard procedure ought to be advanced, that can be coordinated with the software improvement procedure to guarantee a superior software imperfection expectation at beginning times of software advancement.
Various data mining methods have been proposed for software deformity analysis in the past, although, not many of them figure out how to manage the entirety of the software deformity issues. Many data classifications models' evaluations are hard to understand and furthermore give the precise number of imperfections which is excessively risky, especially in the start of a task when excessively little data is accessible. Then again classification models that foresee possible faultiness can be specific, yet less useful to provide insight about the real number of faults. Numerous researchers used numerous techniques with various dataset that anticipate faultiness. In any case, there are such huge numbers of classification algorithms that can be compelling to foresee faultiness. Every one of these issues motivates to our research in these field of software deformity prophecy.

2 Related Work

Software faults basically because of programming coding bugs keep on disquieting the software business with deplorable effect, particularly in the undertaking application software classification. Recognizing the amount of these imperfections are explicitly because of software programming coding blunders is a difficult issue. Current software designing requests experts and analysts to proactively and all in all work towards investigating and testing suitable and significant instruments so as to remove a wide range of degenerative bugs, security openings, and potential deviations at the underlying stage. A short evaluation roughly ongoing inquiries about is demonstrated here. In 2018, one analyst name HanWu proposed a novel model dependent on data mining strategies for predicting two sort diabetes mellitus. The principle goal of his examination study is to improve the exactness of the prediction model and to more than one dataset model is made versatile in nature. Proposed model included two sections dependent on a progression of pre-processing methodology. These two sections are improved K-means algorithm and the logistic regression algorithm. According to performed tests, it is inferred that proposed model show better exactness when contrasted with different techniques and furthermore give the adequate dataset quality. So as to assess the presentation of the model it is applied to different diabetes dataset, in which great execution is appeared by both the strategies.

Chandrasekharan, Puranik and Deshpande also revealed linear regression technique to calculate a software fault-prone datasets model. A combination of multiple regression technique and linear regression technique performed by these researches to locate the well-fitting curve for the defect-prone datasets model. This combination techniques basically were not proved suitable to get the consequence results for software deformity prophecy datasets model. Another one research study presented by Xu et al. to improve the software deformity prophecy model. Xu et al. struggled for improvement and observed that with the help of high dimensionality in training data model, the fault forecast datsets model is affected. For ensure their hypothesis, an extremely wide research was led to observed the influence of 32 feature selection techniques of fault forecast datsets model for overall numerous models. This observation showed that these multiple feature selection techniques are timely fluctuates expressively over diverse datasets model and also observed the good effectiveness on these datasets model. These consequences of experiments did not only prove their privileges but also prove that the excellent of software metric which is very important to enhanced the efficiency and accuracy for software deformity prophecy datasets model. In 2014, another examination name Ying Zhang et al, was extricate valuable data and he utilized data mining consolidate to the machine learning, perception, and factual methodologies.

There are bunches of methods are embraced to gather the data for making dataset for investigation like surveys, feedback structure, talk with, discourse. In the wake of gathering all data make a dataset as per the chose apparatus for investigation and afterward apply a few methods for examination like Naive Bayes, Decision Tree, Classification, Linear Regression, K-mean, Clustering and Support Vector Machine. One of the best forecast strategy given by Sivaji et al. (2015) for software deformity prophecy and that strategy basically used for generating the atmosphere for extreme proficiency of software deformity prophecy. The need of this strategy also must be a very high precise model. They evaluate the improvement percentage of each feature and choose the most useful features from the software datasets model. During the analysis of experiments, the extreme accomplishes got which showed that the all datasets model has increased from 4.1 to 12.52% in every feature datasets model produce the very impressive consequences for all datasets model software fault forecast. Two famous researchers who also worked on software deformity prophecy, they discussed the very important issue in 2007. They argued that every fault is not ensure to present in defected software component or how we can understand that software module is defected with faults or not? They established the way with the help of logistic regression models in defected models to examined the forecast model.

The soonest considers in Software Deformity Prophecy datasets model concentrated on setting up connections between software multifaceted nature, typically estimated in lines of code, and imperfections. Broadly realized metrics presented during 70s is Halstead's hypothesis and McCabe's cyclomatic unpredictability. The typical downside of unpredictability metrics is that they show software size as the main indicator of issues. In this way, in 80s and a while later research has attempted to relate software intricacy to sets of various metrics, determining multivariate regression models. Regression models then again introduced the weakness of giving outcomes hard to decipher that disregarded causal impacts. During the 90s classification models were embraced to tackle this issue. Clustering, logistic regression and Bayesian nets are applied for the estimation of deficiency inclination. A large portion of the above investigations gauge potential issue inclination of software segments without giving specific fault numbers. Fenton and O'Neil gave a basic survey of writing and proposed a hypothetical structure dependent on Bayesian systems that could tackle the issues distinguished.

3 Classification Accuracy Boost way

In order to boost the accuracy of classification model for Software Deformity Prophecy datasets model, the most proper methods in each progression of preprocessing is Discretization. Since Data preprocessing is an indispensable advance in data mining and preprocessing settle different kinds of data errors experienced in enormous databases so as to deliver quality data for the mining task. Discretization is the way toward changing over a nonstop attribute into an ordinal attribute. A conceivably limitless number of qualities are mapped into few classes. Discretization is a procedure that changes quantitative data into subjective data and quantitative data are normally engaged with data mining applications. Discretization is usually utilized in classification and numerous classification algorithms have good results if both the free and ward variables have just a couple of qualities. Discretization of constant highlights or attributes, assume a significant job in the Machine learning data preprocessing stage. Many machine learning algorithms even play out their own discretization system for classification and arranging the qualities from classes and compute data in every conceivable split. When pick the split that limits data, it does exclude breakpoints between qualities having a place with a similar class since this will build data. Discretization consistently apply the equivalent to the subsequent interims until some halting foundation is fulfilled.

3.1 Datasets Model

We have used open-source datasets model for our research experiments, which are very familiarly famous as NASA repository datasets model. These are basically public datasets models and freely available as NASA MDP datasets model. These datasets models were used by many researches same for software deformity prophecy issues. We have chosen 17 datasets models which are mentioned in table 1. The main reason for use of these datasets models, basically these models are divided in two categories class, one is defective datasets model which belongs to class(y) and another one is non-defective class which is belongs to class (Y).

Table1, basically offers some basic information. Every datasets model is contained number of software datasets model. These software datasets models are in shape of buggy, faulty and non-buggy, non-faulty model and also have attributes and total number of models.

Table 1. NASA MDP DATASETS MODEL

S.NO	Datasets	Attribute	Models	Defective	Non-Defective
1	JM1	22	7782	1672	6110
2	AR1	30	121	9	112
3	KC3	40	194	36	158
4	AR6	30	101	15	86
5	CM1	38	327	42	285
6	KC2	22	522	107	415
7	MC1	39	1988	46	1942
8	MC2	40	125	44	81
9	PC5	39	17186	516	16670
10	AR3	30	63	8	55
11	PC1	38	705	61	644
12	AR4	30	107	20	87
13	MW1	38	253	27	226
14	AR5	30	36	8	28
15	PC3	38	1077	134	942
16	PC4	38	1458	158	1289
17	PC2	37	745	16	729

3.2 Evaluation Measure

In data-mining classification, we have used different classifiers for our datasets model. All these classifiers we have measures or evaluate by data evaluation measure. Because data evaluations measure is very easy to use and can understand the efficiency and accuracy in datasets model. All the experiments are basically analyzed by evaluation measures. The most useful evaluation measures we have used in our paper that are tp-rate, f-measure positive accuracy, area under curve and correctly classified instances. We have focused on class (y) model.

4 Classification Accuracy Frame-work Model & Experiments

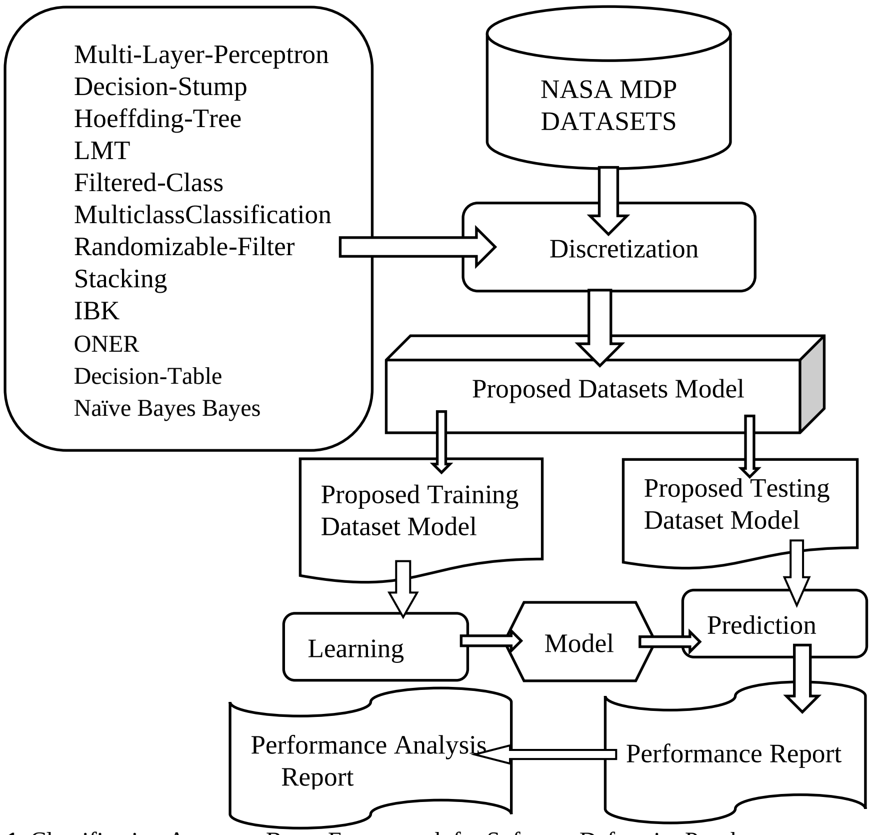

Fig. 1. Classification Accuracy Boost Frame-work for Software Deformity Prophecy

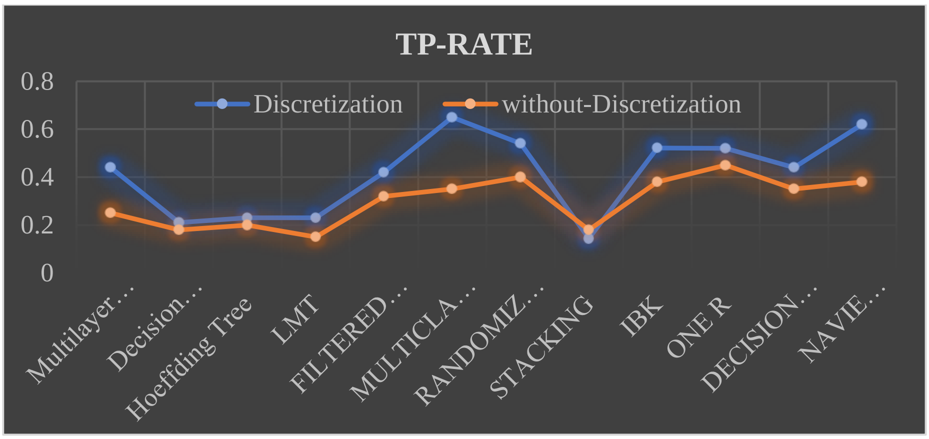

Fig. 2. TP-RATE Accuracy Graph

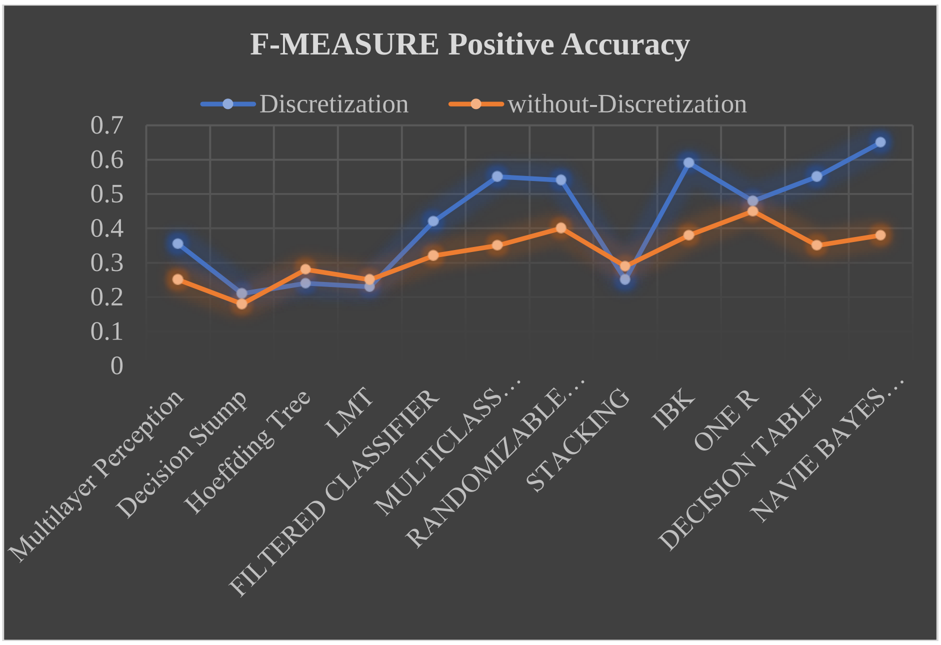

Fig. 3. F-MEASURE Positive Accuracy Graph

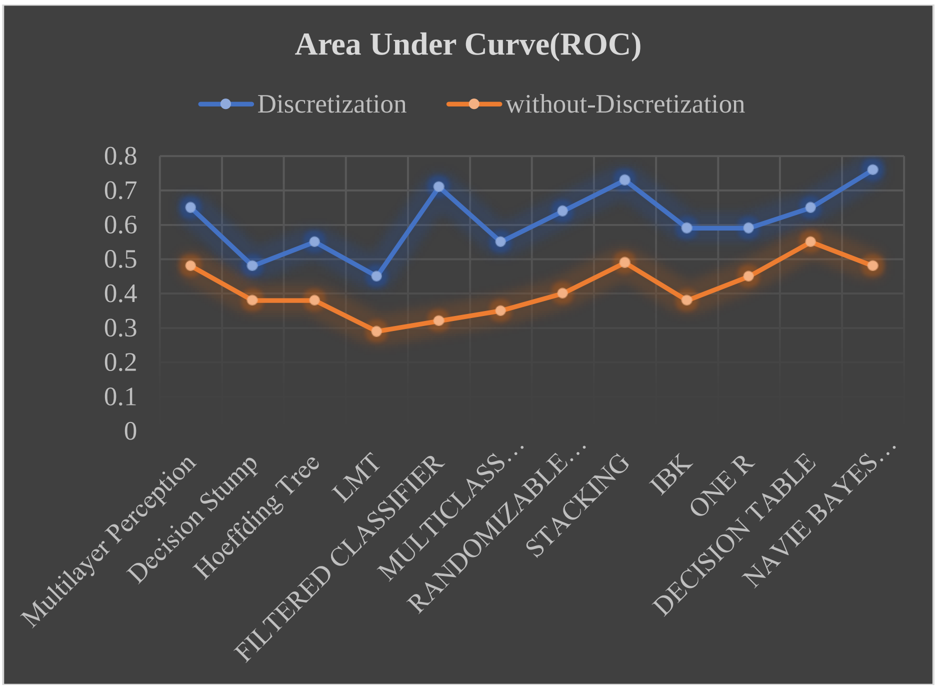

Fig. 4. Area Under Curve (AUC) Accuracy Graph

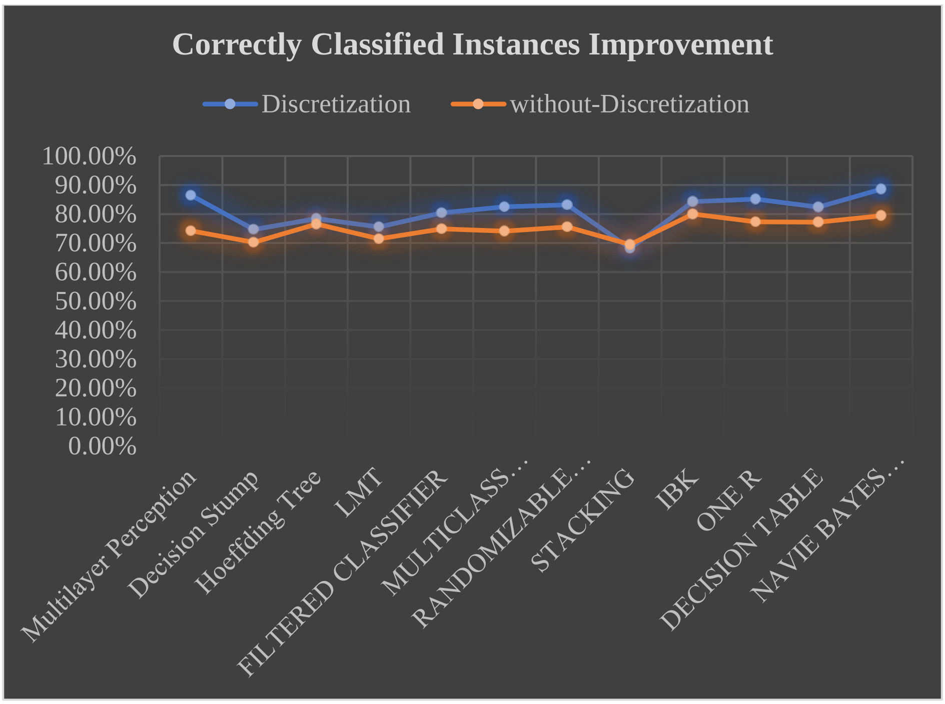

Fig. 5. Correctly Classified Instances Accuracy Graph

Table 2. Correctly Classified Instances Improvement.

Classifiers	Discretization	Without-Discretization	Improvement
Multilayer-Perceptron	86.45%	74.25%	12.20%
Decision-Stump	74.66%	70.20%	4.46%
Hoeffding-Tree	78.36%	76.49%	1.87%
LMT	75.55%	71.50%	4.45%
Filtered-Classifier	80.35%	74.86%	5.51%
Multiclass-Classification	82.45%	74.15%	8.30%
Randomizable-Filtered	83.15%	75.50%	7.65%
Stacking	68.15%	69.50%	-1.35
Ibk	84.25%	80.00%	4.25
One-r	85.10%	77.33%	7.77%
Decision-table	82.35%	77.15%	5.20%
Naïve-bayes-updateable	88.56%	79.40%	9.16%

Table 3. TP-RATE, F-MEASURE & AUC Accuracy.

Classi-fiers	TP-RATE		F-MEASURE		AREA-UNDER CURVE	
	Dicreti-zation	W-Dicreti-zation	Discreti-zation	W- Dis-cretiza-tion	Discreti-zation	W- Dis-cretiza-tion
Multlaer-Percep	0.44	0.25	0.35	0.25	0.65	0.48
Decsion Stump	0.21	0.18	0.21	0.18	0.48	0.38
Hoeffd-ing-Tree	0.23	0.2	0.24	0.28	0.55	0.38
LMT	0.23	0.15	0.23	0.25	0.45	0.29
Filtered-Classifier	0.42	0.32	0.42	0.32	0.71	0.32
Mutclass -Classif	0.65	0.35	0.55	0.35	0.55	0.35
Random-izable-Fil	0.54	0.4	0.54	0.40	0.64	0.41
Stacking	0.144	0.18	0.25	0.29	0.73	0.49
Ibk	0.521	0.38	0.59	0.38	0.59	0.38
One-r	0.52	0.45	0.48	0.45	0.59	0.45
Decsion-table	0.44	0.35	0.55	0.35	0.65	0.55
Naïve-bayes	0.62	0.38	0.65	0.38	0.76	0.48

Table 1 shows that the datasets model which we have used in our experiments. These datasets models belong to software deformity prophecy where class of interest is defect-prone and non-defect-prone models. We have used multiple classifier for classification these class. Our evaluation measure is tp-rate, f-measure, area under curve and correctly classified instances. We have used one preprocess technique called Discretization for boosting the accuracy of these classifiers. From fig2 to fig5, we illustrated that mostly classifiers accuracy and efficiency are improved or boost with help of discretization method. In case of TP-RATE analysis, we have observed that Naive Bayes Updateable, MultiClass, Randomizable, Ibk and oneR teir tp-rate is increased. But few classifiers as like decision stump, hoeffdding tree, lmt their tp-rate couldn't increase very well. In case of positive accuracy and area under curve we have observed that filtered class, multilayer perceptron and naive bayes update their improvement is quite good and boost very well. In case of correctly classified instances where we easily can judge the improvement of every classifiers that decision stump, hoeffding tree and lmt, their efficiency and accuracy is not very good but increased as compare to use these without discretization way. Ibk, oneR, Randomizable, multiclass and Naive bayes their progress efficiency and accuracy have boost very well in all case. The position of stacking is quite bad and not good to use in these experiments because their efficiency and accuracy have not increased but seems to be worst in all case. All experiments analysis clearly showed that all classifiers cannot be perfect for the accuracy and efficiency in software deformity prophecy datasets models.

5 Conclusion

A data-preprocessing technique called Discretization we have used in our research for software deformity prophecy datasets model. This is our classification accuracy boost technique for our software deformity prophecy datasets model. For observation and analysis, we have used multiple classifiers for getting the boost accuracy with the help of discretization preprocess method. All experiments analysis clearly showed that all classifiers cannot be perfect for the accuracy and efficiency in software deformity prophecy datasets models. In case of correctly classified instances where we easily can judge the improvement of every classifiers that decision stump, hoeffding tree and lmt, their efficiency and accuracy is not very good but increased as compare to use these without discretization way. The position of stacking is quite bad and not good to use in these experiments because their efficiency and accuracy have not increased but seems to be worst in all case.
Author Contributions: One of the best techniques known as Discretization used for software deformity prophecy datasets model. reported. Data-sets are analyzed by evaluation measures, where we have used tp-rate, f-meausre, area under curve and correctly classified instance. Data is collected by WEKA 3.9.3, in which different 12 classifiers we have used and did classification for knowing the highest accuracy and efficiency in between these classifiers. All the experiments we have performed two times and verify it. It took too long time to get results due to the cashed storage full in the system. A data-mining book also we have read and deeply know about the classification of data. Multiple researched papers we have also read and know the nature of data for software deformity prophecy datasets model.
Funding: “This research received no external funding”.
Conflicts of Interest: “The authors declare no conflict of interest.”

References

1. P. Singh and S. Verma, "An Investigation of the Effect of Discretization on Defect Prediction Using Static Measures," 2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies, Trivandrum, Kerala, 2009, pp.837-839.
2. Ishani Arora a, Vivek Tetarwala, Anju Sahaa, Open issues in software defect prediction, Elsevier, 2015

3. Ahmed H. Yousef, extracting software static defect models using data mining, Elsevier ,2015.
4. Wang, Shuo, and Xin Yao. "Using class imbalance learning for software defect prediction." IEEE Transactions on Reliability 62.2 (2013): 434-443.
5. Ren, J., Qin, K., Ma, Y., & Luo, G. (2014). On software defect prediction using machine learning. Journal of Applied Mathematics,2014.
6. Rish, Irina. "An empirical study of the naive Bayes classifier." IJCAI 2001 workshop on empirical methods in artificial intelligence. Vol. 3. No. 22. IBM New York, 2001.
7. Garcia, Salvador, et al. "A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning." IEEE Transactions on Knowledge and Data Engineering 25.4 (2013): 734-750. S.
8. Kotsiantis, D. Kanellopoulos Discretization techniques: A recent survey GESTS Int. Trans. Computer Sci. Eng., 32 (2006), pp. 47–58
9. A.D. Bakar, A. Sultan, H. Zulzalil and J. Din,2014. Predicting Maintainability of Objecto-riented Software Using Metric Threshold. Information Technology Journal, 13: 1540-1547.
10. Metz, CE (October 1978). "Basic principles of ROC analysis" (PDF). Semin Nucl Med. 8(4): 283–98.
11. Kaya, Fatih. "Discretizing continuous features for naïve Bayes and C4. 5 classifiers."University of Maryland publications: College Park, MD, USA (2008).
12. Kohavi, Ron, and Mehran Sahami. "Error-Based and Entropy-Based Discretization of Continuous Features." KDD. 1996.
13. Rish, Irina. "An empirical study of the naive Bayes classifier." IJCAI 2001 workshop on empirical methods in artificial intelligence. Vol. 3. No. 22. IBM New York, 2001. http://promisedata.googlecode.com
14. P. Singh and S. Verma, "An Investigation of the Effect of Discretization on Defect Prediction Using Static Measures," 2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies, Trivandrum, Kerala, 2009, pp.837-839.
15. J. Munson and T. Khoshgoftaar, “Regression modelling of software quality: empirical investigation,” Information and Software Technology, vol. 32, no. 2, pp. 106–114, 1990.
16. N. B. Ebrahimi, “On the statistical analysis of the number of errors remaining in a software design document after inspection,” Software Engineering, IEEE Transactions on, vol. 23, no. 8, pp. 529–532, 1997.
17. Q. Song, M. Shepperd, M. Cartwright, and C. Mair, “Software defect association mining and defect correction effort prediction,” Software Engineering, IEEE Transactions on, vol. 32, no. 2, pp. 69–82, Feb 2006.
18. R. Malhotra, “Comparative analysis of statistical and machine learning methods for predicting faulty modules,” Applied Soft Computing, vol. 21, pp. 286–297, 2014.
19. T. McCabe, “A complexity measure,” IEEE Transactions on Software Engineering, vol. 2, no. 4, pp. 308–320, December 1976.
20. M.J Catalas, J.Paul A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks, Journal of Software Engineering, April2015.
21. M. Halstead, Elements of Software Science. Elsevier, 1977.
22. T. Menzies, J. Greenwald, and A. Frank, “Data mining static code attributes to learn defect predictors,” Software Engineering, IEEE Transactions on, vol. 33, no. 1, pp. 2–13, Jan 2007.

23. F. Rahman and P. Devanbu, “How, and why, process metrics are better,” in Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, 2013, pp. 432–441.
24. R. Malhotra and A. Jain, “Fault prediction using statistical and machine learning methods for improving software quality.” JIPS, vol. 8, no. 2, pp. 241–262, 2012.
25. Laetitia Jourdan. Encyclopedia of Data Warehousing and Mining, Second Edition 2009 (pp. 823-828).
26. Wilfred Ng and Mark Levene. Data Warehousing and Web Engineering 2002 (pp. 285-311).
27. Marwa Manaa, Thouraya Sakouhi and Jalel Akaichi. Emerging Perspectives in Big Data Warehousing 2019 (pp. 83-104).
28. Z. Xu, J. Liu, Z. Yang, G. An, X. Jia, The impact of feature selection on defect prediction performance: An empirical comparison, in 27th International Symposium on Software Reliability Engineering (ISSRE), Ottawa, Canada, 2016, pp. 309–320.
29. S. Wang, X. Yao, using class imbalance learning for software defect prediction, IEEE Trans. Rel. 62(2) (2013), 434–443.
30. J.A. Sáez, M. Galar, J. Luengo, F. Herrera, INFFC: An iterative class noise filter based on the fusion of classifiers with noise sensitivity control, Inf Fusion. 27 (2016), 19–32.
31. T.M. Khoshgoftaar, K. Gao, N. Seliya, Attribute selection and imbalanced data: Problems in software defect prediction, in 22nd IEEE International Conference on Tools with Artificial Intelligence, Arras, France, 2010, pp. 137–144.
32. A.A. Shanab, T.M. Khoshgoftaar, R. Wald, J. Van Hulse, Comparison of approaches to alleviate problems with high-dimensional and class-imbalanced data, in IEEE International Conference on Information Reuse and Integration, Las Vegas, USA, 2011, pp. 234–239.
33. T.M. Khoshgoftaar, K. Gao, A. Napolitano, R. Wald, A comparative study of iterative and non-iterative feature selection techniques for software defect prediction, Inform. Syst. Front. 16(5) (2014), 801–822.
34. H. Liu, H. Motoda, L. Yu, A selective sampling approach to active feature selection, Artif. Intell. 159(2) (2004), 49–74.[2].
35. Basili, V.R., Briand, L.C., Melo, W.L.: A validation of object-oriented design metrics as quality indicators. IEEE Transactions on Software Engineering 22(10) (1996) 751–761
36. Ohlsson, N., Alberg, H.: Predicting fault-prone software modules in telephone switches. IEEE Transactions on Software Engineering 22(12) (1996) 886–894
37. Subramanyam, R., Krishnan, M.S.: Empirical analysis of ck metrics for object-oriented design complexity: Implications for software defects. IEEE Transactions on Software Engineering 29(4) (2003) 297–310
38. Gyim´othy, T., Ferenc, R., Siket, I.: Empirical validation of object-oriented metrics on open source software for fault prediction. IEEE Transactions on Software Engineering 31(10) (2005) 897–910
39. Nagappan, N., Ball, T., Zeller, A.: Mining metrics to predict component failures. In: Proceedings of the ICSE 2006 (28th International Conference on Software Engineering), ACM (2006) 452–461
40. Nagappan, N., Ball, T.: Use of relative code churn measures to predict system defect density. In: Proceedings of ICSE 2005 (27th International Conference on Software Engineering), ACM (2005) 284–292
41. Moser, R., Pedrycz, W., Succi, G.: A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction. In: Proceedings of ICSE 2008 (30th International Conference on Software Engineering). (2008) 181–190.