Plagiarism Checker X Originality Report

Similarity Found: 14%

Date: Sunday, May 31, 2020
Statistics: 565 words Plagiarized / 4101 Total words
Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

Combining Integreted Sampling Technique with Feature Selection for Software Defect Prediction Sukmawati Anggraeni Putri STMIK Nusa Mandiri, Information System Program Jakarta, Indonesia sukmawati@nusamandiri.ac.id Frieyadie AMIK BSI Jakarta, Management Informatic Program Jakarta, Indonesia frieyadie@bsi.ac.id Abstract —Good quality software is a supporting factor that is important in any line of work in of society.

But the software component defective or damaged resulting in reduced performance of the work, and can increase the cost of development and maintenance. An accurate prediction on software module prone defects as part of efforts to reduce the increasing cost of development and maintenance of software. An accurate prediction on software module prone defects as part of efforts to reduce the increasing cost of development and maintenance of software.

From the results of these studies are known, there are two problems that can decrease performance prediction of classifiers such imbalances in the distribution of the class and irrelevant of the attributes that exist in the dataset. So as to handle both of these issues, we conducted this research using integrated a sample technique with feature selection method.

Based on research done previously, there are two methods of samples including random under sampling and SMOTE for random over sampling. While on feature selection method such as chi square, information gain and relief methods. After doing the research process, integration SMOTE technique with relief method used on Naïve Bayes classifiers, the result of the predicted value better than any other method that is 82%. Keywords —imbalance class, feature selection, software defect prediction I.

INTRODUCTION In the development of the use of software to support the activities and the work increases, certainly the quality of the software must be considered. But the software component defective or damaged resulting in a decrease in customer satisfaction, as well as an increase in the cost of development and maintenance [1].

An accurate prediction on software module software defects as part of effort to reduce the increasing cost of development and maintenance of software that has been done by previous researchers [2]. In this study focuses on 1) estimating the amount of defect in the software, 2) find the relationship of software defects, 3) classifying defect software components, which defect module and non defect module [3].

While the software defect prediction research that has been done by previous research such as Naïve Bayes classifier [4] produce a good performance with an average probability of 71%. Naïve Bayesian is a simple classification [5] with a time of learning process is faster than any other machine learning [4]. Additionally it has a good reputation on the accuracy of prediction [6]. However, this method is not optimal in the case of having an unbalanced dataset [7].

The predicted performance of this method gets worse when the dataset has an irrelevant attribute [8]. While NASA MPD dataset [9] which have been used by previous researchers on software predictions have unbalanced defect datasets with attributes that are not all usable.

To deal with unbalanced datasets there are three approaches that can be used, including data level (sample technique), algorithm level and ensamble method [10]. In general, the sample technique is divided into two types, including over sampling method is Random Over Sampling [11]. While the under sampling method is Random Under Sampling [12] and Resample method [13].

As for solving the problem of attributes that are irrelevant using attribute selection methods such as Information Gain, Chi Square, and Relief [14]. In this study, we propose to integrate the sample technique with feature selection method to handle imbalance class and attribute irrelevant to the Naïve Bayesian classification to produce a better accuracy in the software defect prediction.

There are several steps done in this study, First, sample technique to handle the imbalance class. Then, approaching the selection attributes thrown clasifiying for software defect predicition. Then calculating the validation and evaluation technique to determine the proposed method is wheter it better or not with the existing methods. II.

RELATED WORK Research on the software defect prediction are one of the research that has been done by previous researches. From these studies it is known state of the art about software defect prediction research that discusses the imbalance class. As research done by Chawla [15] who proposed the use of Synthetic Minority Oversampling Technique to handle the class imbalance by using a Naïve Bayesian classifier, implemented in eight different dataset from the UCI repository.

The results showed for all the data using SMOTE technique on balancing process has a greater potential to improve the performance of Naïve Bayesian and C.45 classifier use in the classification process. While the research done by Riquelme [13] which states that the dataset in software engineering is very unbalanced. Therefore to balance using two technique, including SMOTE and Weka Randomly Resampling using J48 and Naïve Bayesian classifier is applied to the five datasets form PROMISE repository. The results show the approach SMOTE able to increase the average AUC value of 11.6%.

Based on these results, balancing techniques can better classify minority classes. While the research done by Putri, Wahono [16] states that NASA MDP dataset has unbalanced classes and attribute not relevant. Using the balancing class SMOTE and feature selection information gain, can improve the prediction results are better than Riquelme research only to rebalance the dataset class. Furthermore the study done by Gao using one of the feature selection algorithm which Relief that have been used in the research done by Kira.

The research Gao shows that the Relief method as well as the Information Gain [17]. Therefore in this study implement an approach sample techniques which Synthetic Minority Over-sampling Technique (SMOTE) to reduce the influence of class imbalance and improve the ability to predict the minority class. Relief algorithm as well as for selection the relevant attribute.

It also use Naïve Bayes algorithm used in the classification process. III. METHODE 3.1. Sample Technique The sample approach is one approach to solve the problem of class imbalance in a dataset. The commonly used sample approaches are over-sampling and under-sampling techniques [10]. a.

Over-Sampling Technique Over-sampling causes excessive duplication in the positive class cause over-fitting. Moreover, over-sampling can increase the number of training dataset, thus causing excessive computational costs [15]. Nevertheless, in research carried out by Chawla [15] found Synthetic Minority Over-sampling Technique (SMOTE) which produces artificially interpolated data on the over-sampling in the minority.

The algorithm is simulated by finding k nearest to each minority sample, and then for each neighbor, randomly pick a point on the line connecting neighbors and sample itself. Finally, the data at that point is entered as an example of the new minority. By adding new minority sample into training data, is expected to over-fitting can be resolved [15]. b.

Under-sampling Technique Under-sampling approaches have been reported to outperform over-sampling approaches in previous literatures. However, the under-sampling approach reduces the majority class, perhaps losing useful information. This results in less accurate predictions [11]. Sampling is done randomly, so the majority of the sample is as large as the number of minority samples.

Meanwhile, the sample used in the under-skilled approach is the majority sample that is under the sample [18] . We implemented our proposed Random Under- Sampling and SMOTE in the WEKA tool. 3.2. Feature Selection At dataset software defects, attributes represent software metrics taken from the source code of the software used in the learning process.

However, some attributes that are not relevant to require the removal to improve the accuracy of software defects prediction. There are two algorithms used in the selection of attributes that wrapper and a filter [17]. In the wrapper algorithm using feedback from learning algorithm. While on the filter algorithm, the training data are analyzed using methods that do not require learning algorithms to determine the most relevant attributes [17].

In this study only uses algorithms to filter the selection attribute. Such as, chi-square (CS), information gain (IG), and Relief algorithm (RLF) [17]. a. Chi Square (CS) CS can evaluate attribute values by calculating the statistical value related to the class. Statistical CS (also symbolized as 2) is a nonparametric statistical techniques by using nominal data (category) with the test frequency.

(1) where 2 is the test statistic is asymptotically approaching the 2 distribution, O i is the observed frequencies, and Ei is the expected frequency. n is the number of possible outcomes of each event. b. Information Gain (IG) In the IG is able to assess the importance atibut by measuring the information gain associated with the class.

Generally IG estimates that the change in entropy of information before the state took some information. I (CsAttriute)(Css)-HIass|Attriute) (were etientopy ore pecfca,sppos tha sthe et ll ttriutesacs ttriutesb ? la efn hvle fsecficexap o aba ? ? ? elem n e et .Gtaba ? .A ia fllow: (3) c.

Relief (RLF) For a given sample R, Relief find the nearest neighbor of the same class or different, which is called the 'nearest hit H' and 'nearest miss M'. It will be updated estimate of the quality of W [A] for all attributes A depending on their values for R, F, and H. The process is repeated in accordance with the value of m, where m is determined by the user.

Diff function (Attribute, Instance1, Instance2) clearly defined in accordance with the type attribute. For discrate attributes are defined as follows: The underlying hypothesis is that the relevant attributes are able to distinguish between things of different classes and showed no difference between instances of the same class. 3.3.

Naive Bayesian (NB) Classifier Naïve Bayes assumes that the impact of a certain class attribute value is independent of the values of other attributes. This assumption is called the independent class conditional. This is done to simplify the calculation involved, and in this sense it is considered naive. Naïve Bayes allows representation of dependencies among a subset of attributes [19].

By mathematical calculation as follows: (The probability P(X 1 |C 1 2 |C j),... (n |C i) can be easily estimated from the training set. Given that Xk refers to the attribute values for the sample X. a. If Ak is a category, then P(X k |C j) is number of tuples in D class Cj has a value Xk to attribute Ak, divided from |C 1, |, number of class Cj tuples in D. b.

If Ak is a continuous value, it is usually assumed that the values have a Gaussian distribution with mean (μ) and standard deviation (s), can be defined as follows: ((7) We need to calculate and , where the mean and standard deviation of the value attribute A k for training samples of class C j . 3.4. Validation Technique In this study using validation techniques 10 fold cross validation, with resulting confusion matrix [20] which are described in Table 1.

In the confusion matrix, TN is true negative results are classified (true negative). FN is a positive result that is not properly classified as negative. TP is a positive result correctly classified (true positive). FP is the negative results are not correctly classified as positive (false positive). TABLE 1. CONFUSION MATRIX Confusion matrix of values will produce the ROC curve (Receive Operationg Characteristics) whose task is to evaluate the performance of the classifier algorithm. Then the Area Under the ROC as a reference for evaluating which provides a summary of the performance of the classifier algorithm [20].

Area Under the ROC (Receive Operating Characteristic) (AUC) is a single value measurements are derived from signal detection. AUC values range from 0 to 1. The

ROC curve is used to characterize the trade-offs between true positive rate (TPR) and false positive rate (FPR). A classifier that provides a large area under the curve is more of a classifier with a smaller area under the curve [21]. 3.5.

Evaluation Technique In the statistical evaluation consisted of testing parametric and non-parametric test. As for testing the significant difference of the classifier algorithm performance using the non-parametric tests, such as tests friedman [22]. Friedman test is a non-parametric test that is equivalent to the ANOVA parametric test.

In the Friedman test ranking algorithm for each data set separately, the algorithm performance is good to be ranked first, while for the second- best given. Friedman test carried out by the appropriate post hoc test for comparison of more than one classifier with multiple datasets [22]. Below will be shown on the test friedman formula: 8) Where =ha va-l ofo-a sqa fedma N mu fsple K b roupsaples 1, 12 = cta Class Initial Value True False Predicition Value True TP FP False FN TN IV. EXPERIMENT RESULT 4.1.

Dataset In this study, using a dataset of software metric (National Aeronautics and Space Administration) MDP repository. They are public datasets used by previous researchers in the field of software defects prediction. NASA dataset MDP can be obtained via the official website Wikispaces (http://nasa-softwaredefectdatasets.wikispaces.com/).

Dateset used in this study consisted of CM1, MW1, PC1 and PC4 are described in Table 2. TABLE 2. NASA MDP DATASET As shown in Table 2, that each dataset consists of several software modules, along with the number of errors and attributes characteristic code. NASA dataset preprocessing MDP has 38 attributes plus one attribute disabled or not disabled (defective?).

The attribute consists of an attribute type Halstead, McCabe, Line of Code (LOC) and miscellaneous attributes [23]. The dataset was obtained from NASA MDP software matrices which are described in Table 3, as follows: TABLE 3. SPECIFICATIONS AND ATTRIBUTES NASA MDP 4.2. Implementation and Experiment Results In this study using Naive Bayesian classifier algorithm at 4 dateset NASA MDP (CM1, MW1. PC1 and PC4).

Classifier algorithm will be applied on the integration sample technique with a selection attribute method. Like, NB classifier with SMOTE and CS, NB classifier with SMOTE and IG, NB classifier with SMOTE and RLF, NB classifier with RUS + CS, NB classifier with RUS and IG, and NB classifier with RUS and RLF. TABLE 4.

AUC VALUE Clasification CM1 MW1 PC1 PC4 NB 0,694 0,727 0,768 0,825 NB with

SMOTE and CS 0,766 0,759 0,734 0,856 NB with RUS and CS 0,752 0,722 0,79 0,859 NB with SMOTE and IG 0,751 0,767 0,817 0,856 NB with RUS and IG 0,753 0,722 0,79 0,859 NB with SMOTE and RLF 0,761 0,779 0,821 0,86 NB with RUS and RLF 0,755 0,747 0,793 0,878 In Table 4 shows the results AUC values were well on the use of models NB with SMOTE and RLF on two datasets (MW1, PC1).

As for the CM1 dataset shows AUC good value on NB with SMOTE and CS models. And for PC4 dataset shows AUC good value on NB with RUS and RLF model. 4.3. Comparison Between Previous Models To know that the proposed model has increased the accuracy after the optimized use of integration between sampling technique and feature selection algorithm, then do a comparison between the proposed model and a model that has been proposed by Menzies [4], Requilme [13] and Putri [24].

NASA MDP Dataset CM1 MW1 PC1 PC4 LOC Count LOC_total X X X X LOC_blank X X X X LOC_code_and_com ment X X X X LOC_comment X X X X LOC_executable X X X X Number_of_lines X X X X Halstead Attributes Content X X X X Difficulity X X X X Effort X X X X Error_est X X X X Length X X X X Level X X X X Prog_time X X X X Volume X X X X Num_operands X X X X Num_operators X X X X Num_unique_operan ds X X X X Num_unique_operat ors X X X X McCabe Attriutes Cyclomatic_complex ity X X X X Cyclomatic_density X X X X Design_complexity X X X X Essential_complexity X X X X Miscellan eous Attributes (another) Branch_count X X X X Call_pairs X X X X Condition_count X X X X Decision_count X X X X Decision_density X X X X Design_density X X X X Edge_count X X X X Essential_density X X X X Parameter_count X X X X Maitenance_s everity X X X X Modified_condition_ count X X X X Multiple_condition_ count X X X X Global_data_comple xity Global_data_density Normalized_cycloma tic_compl X X X X Precent_comments X X X X Node_count X X X X Number of code attribute 37 37 37 37 Number of Modul 342 266 759 1399 Number of defect modul 41 28 61 178 System Language Program Dataset LOC Instruments a spacecraft C CM1 17K Database C MW1 8K Flight software for satellites orbiting the Earth C PC1 26K PC4 30K TABLE 5.

AUC OF COMPARISON BETWEEN PREVIOUS MODELS Model CM1 MW1 PC1 PC4 Menzies (2011), NB 0,694 0,727 0,768 0,825 Requilme (2008), NB with SMOTE 0,739 0,751 0,793 0,858 Putri, Wahono (2015), NB with SMOTE and IG 0,751 0,767 0,817 0,856 Propose Model, NB with SMOTE and RLF 0,761 0,779 0,821 0,86 Results of the experiments are shown in Table 5 to produce the best classification model in the dataset displayed in bold. Shows the proposed model produces increased AUC values compared to other models.

While in Figure 1 describes the a comparison chart of AUC values for the four models of

the four datasets NASA MDP. Figure 1. Chart of Comparison AUC values between Previous Model To know the difference any proposed model, then do a comparison using the non-parametric statistical calculations used for the computation of the classifier algorithm. Like, friedman test.

AUC values model of NB, NB with SMOTE, NB with SMOTE and IG, and NB with SMOTE and RLF compared using friedman test described in Table 6. TABLE 6. THE P VALUE OF AUC COMPARISON FRIEDMAN TEST NB NB with SMOTE NB with SMOTE and IG NB with SMOTE and RLF NB (Menzies, 2011) 1 0,046 (Sig) 0,046 (Sig) 0,046 (Sig) NB with SMOTE (Riquelme, 2008) 0,046 (Sig) 1 0,317 (No Sig) 0,046 (Sig) NB with SMOTE and IG (Putri, 2015) 0,046 (Sig) 0,317 (No Sig) 1 0,046 (Sig) NB with SMOTE and RLF (Proposed Model) 0,046 (Sig) 0,046 (Sig) 0,046 (Sig) 1 As shown in Table 6 shows the proposed model of model NB with SMOTE and RLF having P value 0.046, then P < a (0.05). So the NB model with SMOTE and RLF has significant differences with the pure NB model.

The model of his study also has significant differences with NB, with each P value for NB with SMOTE is 0.046, while P value NB with SMOTE and IG is 0.046. From these results, the SMOTE and RLF model applied to the Naive Bayesian classification has better calculation performance than the model that has been proposed with previous researchers. V.

CONCLUSION From the results of calculations on research application of integration of sample method with the selection attribute of SMOTE and RLF in Naive Bayes classification yields better AUC value compared to the other model. SMOTE and RLF model is superior to the two datasets of the four datasets used, with a value of 78% in the MW1 and 82% datasets on the PC1 dataset.

Whereas when compared with models that have been proposed by previous researchers, such as Naive Bayesian, SMOTE on Naive Bayesian, SMOTE and IG on Naive Bayesian. From the results of research the value of AUC SMOTE and RLF on Naive Bayesian better performance than the model in all dataset used in the other research. This result can be concluded from comparison result using friedman test, where P value is 0,046, which means P < a (0,05).

But from these results, the use of sample techniques and attribute selection algorithms in software defect prediction research can be done in the next research development, including: 1. For the selection of attributes in future studies may use techniques wrapper on attribute selection methods. 2. In further research can use a combination of sample technique with ensemble algorithm to improve the performance of the classifier. 3.

In further research can use other classifiers, such as Logistic Regression, Neural Networks and SVM. ACKNOWLEDGMENT We should like to express our gratitude to RSW (Romi Satria Wahono) Intelligent Research Group for warm discussion about this research. Also for PPPM STMIK Nusa Mandiri Jakarta and PPPM AMIK BSI Jakarta, which has supported us to do this research. REFERENCES [1] A. B. de Carvalho, A. Pozo, and S.

R. Vergilio, "A symbolic fault- prediction model based on multiobjective particle swarm optimization," J. Syst. Softw. , vol. 83, no. 5, pp. 868 –882, May 2010. [2] C. Catal, "Software fault prediction: A literature review and current trends," Expert Syst. Appl. , vol. 38, no. 4, pp. 4626 –4636, Apr. 2011. [3] Q. Song, Z. Jia, M. Shepperd, S. Ying, and J. Liu, "A General Software Defect-Proneness Prediction Framework," IEEE Trans. Softw. Eng., vol. 37, no. 3, pp. 356 –370, May 2011.

[4] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," IEEE Trans. Softw. Eng. , vol. 33, no. 1, pp. 2 –13, Jan. 2007. [5] P. Domingos, "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss," Mach. Learn. , vol. 29, no. 2 –3, pp. 103–130, 1997. [6] B. Turhan and A. Bener, "Analysis of Naive Bayes' assumptions on software fault data: An empirical study," Data Knowl. Eng. , vol. 68, no. 2, pp. 278 –290, Feb.

2009. [7] C. Andersson, "A replicated empirical study of a selection method for software reliability growth models," Empir. Softw. Eng. , vol. 12, no. 2, pp. 161 –182, Oct. 2006. [8] T. M. Khoshgoftaar and K. Gao, "Feature Selection with Imbalanced Data for Software Defect Prediction," 2009 Int. Conf. Mach. Learn. Appl. , pp. 235–240, Dec. 2009. [9] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data Quality ? : Some Comments on the NASA Software Defect Data Sets," Softw. Eng.

IEEE Trans., vol. 39, no. 9, pp. 1 –13, 2013. [10] B. W. Yap, K. A. Rani, H. Aryani, A. Rahman, S. Fong, Z. Khairudin, and N. N. Abdul lah, "An Application of Oversampling, Undersampling, Bagging and Boosting in Handling Imbalanced Datasets," Proc. First Int. Conf. Adv. Data Inf. Eng. , vol. 285, pp. 13–23, 2014. [11] Y. Liu, X. Yu, J. X. Huang, and A. An, "Combining integrated sampling with SVM ensembles for learning from imbalanced datasets," Inf.

Process. Manag. , vol. 47, no. 4, pp. 617 –631, Jul. 2011. [12] K. Gao and T. M. Khoshgoftaar, "Software Defect Prediction for High-Dimensional and Class-I mbalanced Data," Proc. 23rd Int. Conf. Softw. Eng. Knowl. Eng. , no. 2, 2011. [13] J. C. Riquelme, R. Ruiz, and J. Moreno, "Finding Defective Modules from Highly Unbalanced Datasets," Engineering , vol. 2, no. 1, pp. 67 –74, 2008. [14] K. Gao, T.

M. Khoshgoftaar, H. Wang, and N. Seliya, “Choosing software metrics for defect prediction ? : an investigation on feature selection techniques,” Softw. Pract. Exp. , vol. 41, no. 5, pp. 579 – 606, 2011. [15] N. V Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE ? : Synthetic Minority Over-sampling Technique,” J. Artif. Intell., vol. 16, pp. 321 –357, 2002. [16] S. A. Putri and R. S.

Wahono, “Integrasi SMOTE dan Information Gain pada Naive Bayes untuk Prediksi Cacat Software,” J. Softw. Eng., vol. 1, no. 2, pp. 86 –91, 2015. [17] K. Gao and T. M. Khoshgoftaar, “Software Defect Prediction fo r High-Dimensional and Class-Imbalanced Data,” Conf. Proc. 23rd Int. Conf. Softw. Eng. Knowl. Eng. , no. 2, 2011. [18] N. Japkowicz, “The Class Imbalance Problem ? : Significance and Strategies,” Proc. the2000 Int. Conf. Artif. Intell. Spec. Track Inductive Learn. Vegas , 2000.

[19] M. Jain and V. Richariya, “An Improved Techniques Based on Naive Bayesian for Attack Detection,” Int. J. Emerg. Technol. Adv. Eng., vol. 2, no. 1, pp. 324 –331, 2012. [20] C. X. Ling, “Using AUC and Accuracy in Evaluating Learning Algorithms,” pp. 1– 31, 2003. [21] C. X. Ling and H. Zhang, “AUC: a statistically consistent and more discriminating measure than accuracy,” Proc. 18th Int. Jt. Conf. Artif. Intell., 2003. [22] J.

Demsar, “Statistical Comparisons of Classifiers over Multiple Data Sets,” J. Mach. Learn. Res. , vol. 7, pp. 1 –30, 2006. [23] S. Lessmann, S. Member, B. Baesens, C. Mues, and S. Pietsch, “Benchmarking Classification Models for Software Defect Prediction ? : A Proposed Framework and Novel Findings,” IEEE Trans. Softw. Eng. , vol. 34, no. 4, pp. 485 –496, 2008. [24] S. A. Putri and Wahono, “Integrasi SMOTE dan Information Gain pada Naive Bayes untuk Prediksi Cacat Software,” J.

Softw. Eng. , vol. 1, no. 2, pp. 86 –91, 2015.

INTERNET SOURCES:

<1% - https://ifs.host.cs.st-andrews.ac.uk/Resources/Notes/Evolution/SWReeng.pdf
<1% - https://research-methodology.net/sampling-in-primary-data-collection/stratified-sampling/
<1% - https://www.sciencedirect.com/science/article/pii/S1532046418301400
<1% - https://www.sciencedirect.com/science/article/abs/pii/S1568494615002720
<1% - http://www.mtc.gov/uploadedFiles/Multistate_Tax_Commission/Audit_Program/Links/auditsamplingmanuals.pdf
<1% - https://www.sciencedirect.com/science/article/pii/S1568494616302484
<1% - https://www.sciencedirect.com/science/article/pii/S0888327017300468
<1% - https://github.com/topics/uci-machine-learning
<1% - https://www.researchgate.net/publication/221390127_Software_Defect_Prediction_for_High-Dimensional_and_Class-Imbalanced_Data
<1% - https://www.sciencedirect.com/science/article/pii/S093336571830681X
<1% - https://www.researchgate.net/publication/236651567_Class_Imbalance_Problem_in_Data_Mining_Review
<1% - https://whatis.techtarget.com/definition/over-sampling-and-under-sampling
<1% - https://www.researchgate.net/publication/325789071_SMOTE_for_Learning_from_Imbalanced_Data_Progress_and_Challenges_Marking_the_15-year_Anniversary
<1% - https://www.researchgate.net/publication/257365846_An_Application_of_Oversampling_Undersampling_Bagging_and_Boosting_in_Handling_Imbalanced_Dataset
<1% - https://www.sciencedirect.com/science/article/pii/S0950584914001591
<1% - https://www.researchgate.net/publication/4046596_Application_of_an_Attribute_Selection_Method_to_CBR-Based_Software_Quality_Classification
<1% - http://uregina.ca/~gingrich/ch10.pdf
<1% - https://www.mathgoodies.com/lessons/vol6/intro_probability
<1% - https://www.sciencedirect.com/science/article/pii/S0045793020301225
<1% - https://www.researchgate.net/publication/235333385_A_Decision_Tree_Classification_Model_for_University_Admission_System
<1% - https://link.springer.com/content/pdf/10.1023%2FA%3A1025667309714.pdf
<1% - https://www.geeksforgeeks.org/static-methods-vs-instance-methods-java/
<1% - https://arxiv.org/pdf/1009.4964
<1% - http://infolab.stanford.edu/pub/gio/1990/walkerDiscovery.html

<1% -
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
<1% - https://jcheminf.biomedcentral.com/articles/10.1186/s13321-014-0047-1
<1% -
https://towardsdatascience.com/confusion-matrix-and-class-statistics-68b79f4f510b
<1% - https://www.sciencedirect.com/science/article/pii/S0950705111001286
<1% -
https://machinelearningmastery.com/assessing-comparing-classifier-performance-roc-curves-2/
<1% - https://www.medcalc.org/manual/roc-curves.php
<1% - https://neptune.ai/blog/f1-score-accuracy-roc-auc-pr-auc
<1% - https://en.wikipedia.org/wiki/Mann-Whitney-Wilcoxon
<1% - https://www.betterevaluation.org/en/evaluation-options/nonparametricinferential
<1% - https://accendoreliability.com/non-parametric-friedman-test/
<1% - https://www.sciencedirect.com/science/article/pii/S0957417417302397
<1% - https://dl.acm.org/citation.cfm?id=3018329
<1% - https://repository.bsi.ac.id/index.php/repo/viewitem/20238
<1% - https://www.sciencedirect.com/science/article/pii/S0950584919301466
<1% - https://www.hindawi.com/journals/cin/2018/1067350/
1% -
https://www.researchgate.net/publication/323771404_Integrasi_SMOTE_dan_Information_Gain_pada_Naive_Bayes_untuk_Prediksi_Cacat_Software
<1% - https://educacion.unmsm.edu.pe/pdf_temporales/2020/unayoe(25-05-2020).pdf
<1% - https://www.scribd.com/document/156321808/v01-Analysis-of-Variance
<1% -
https://translational-medicine.biomedcentral.com/articles/10.1186/s12967-018-1758-2
<1% -
https://www.isixsigma.com/tools-templates/hypothesis-testing/making-sense-mann-whitney-test-median-comparison/
<1% - https://www.sciencedirect.com/science/article/pii/S0031320319304479
<1% -
https://www.researchgate.net/publication/273722148_A_mapping_study_of_the_Brazilian_SBSE_community
<1% - https://www.sciencedirect.com/science/article/pii/S0164121218301213
<1% -
https://www.researchgate.net/publication/220069857_A_General_Software_Defect-Proneness_Prediction_Framework
<1% - https://www.worldscientific.com/doi/abs/10.1142/S0218194018500110
<1% - https://link.springer.com/article/10.1007/s10462-007-9052-3

<1% -
https://www.researchgate.net/publication/325075522_Semi-Supervised_Deep_Fuzzy_C-Mean_Clustering_for_Software_Fault_Prediction
<1% - https://dl.acm.org/doi/10.1016/j.jss.2014.12.029
<1% -
https://www.researchgate.net/publication/327131466_The_Empirical_study_of_Semi-Supervised_Deep_Fuzzy_C-Mean_Clustering_for_Software_Fault_Prediction
<1% - http://ijain.org/index.php/IJAIN/article/view/350
<1% - https://www.hindawi.com/journals/acisc/2016/7658207/
<1% -
https://www.researchgate.net/publication/337732652_A_Framework_for_Software_Defect_Prediction_Using_Feature_Selection_and_Ensemble_Learning_Techniques
<1% - https://link.springer.com/chapter/10.1007/978-3-7091-1538-1_6
<1% - https://link.springer.com/chapter/10.1007%2F978-3-030-29908-8_2
<1% - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4981579/
<1% - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4057383/
<1% - https://link.springer.com/article/10.1186/s40965-018-0056-5
1% -
https://www.researchgate.net/publication/269510338_A_Comparison_Framework_of_Classification_Models_for_Software_Defect_Prediction