p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416

Website : jurnal.umj.ac.id/index.php/semnastek

LINEAR REGRESSION DENGAN PEMBOBOTAN ATRIBUT DENGAN METODE PSO UNTUK *SOFTWARE DEFECT PREDICTION*

Muhammad Rizki Fahdia[1*], Richardus Eko Indrajit[2]
[*1]Ilmu Komputer, STMIK Nusamandiri, Jakarta, Indonesia
Jalan Kramat Raya No 25
[2]ABFI Institute Perbanas, Jakarta, Indonesia[1]
Jl. HR Rasuna Said, RT.6/RW.7, Karet Kuningan, Kecamatan Setiabudi
[*]E-mail : mrfahdia@gmail.com

ABSTRAK

Kualitas *software* sudah menjadi bagian yang penting dalam proses pengembangan. Karena semakin kompleksnya sebuah *software* dan tingginya ekspektasi dari pelanggan. Maka saat ini biaya pengembangan *software* juga semakin tinggi. Oleh karena itu dibutuhkan efisiensi untuk menekan biaya pengembangan *software.* Salah satu cara yang bisa dilakukan yaitu dengan *software defect prediction.* Dengan *sotware defect prediction* maka dapat diketahui proyek software mana yang butuh pengecekan lebih intens. Tim test software dapat mengalokasikan waktu dan biaya lebih efektif berdasarkan hasil dari model algoritma. Metode pada riset ini menggunakan p*repocessing* dengan mengoptimalkan bobot atribut dengan menggunakan metode PSO yang merupakan algoritma pencarian berbasis populasi dan yang diinisialisi dengan populasi solusi acak yang disebut partikel. Berdasarkan hasil pengolaha data dengan metode preprocessing terhadap dataset NASA MDP CM1. Maka didapatkan metode preprocessing dengan pembobotan atribut dengan metode PSO memiliki peningkatan akurasi menjadi 86.37% dari sebelumnya 85.54% dan AUC menjadi 0.827 dari sebelumnya 0.762.

Kata kunci: prediksi cacat software, linier regression, feature selection, optimize weight.

ABSTRACT

Software quality has become an important part of the development process. Due to the complexity of a software and the high expectations of customers. So now the cost of software development is also higher. Therefore it takes efficiency to reduce the cost of software development. One way that can be done is with software defect prediction. With sotware defect prediction it can be known which software project needs more intense checking. The software test team can allocate time and cost more effectively based on the results of the algorithm model. The method in this research uses prepocessing by optimizing the attribute weights by using the PSO method which is a population-based search algorithm and which is initialized with a population of random solutions called particles. Based on the result of data processing with preprocessing method to NASA MDP CM1 dataset. Then the method of preprocessing with attribute weighting with PSO method has increased accuracy to 86.37% from 85.54% and AUC to 0.827 from 0.762.

Keywords : *software defect prediction, linear regression, feature selection, optimize weight.*

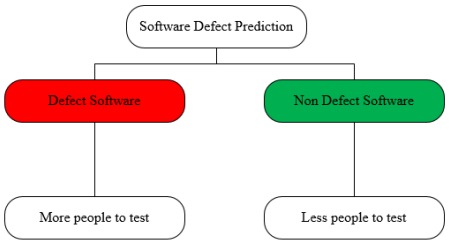
1. PENDAHULUAN

Kualitas software mulai diperhatikan akhir akhir ini seiring dengan semakin kompleksnya software dan tingginya ekspektasi dari pelanggan. Bahkan saat ini sudah ada standar internasional untuk mengevaluasi kualitas software yaitu ISO/IEC 9126. Kualitas *software* seakan akan sudah menjadi bagian yang penting dalam pengembangan *software.* Karena semakin kompleksnya sebuah *software* dan tinginya ekspektasi dari pelanggan, maka saat ini biaya pengembangan *software* juga semakin tinggi. Oleh karena itu dibutuhkan efisiensi untuk menekan biaya pengembangan *software.* Salah satu car yang bisa dilakukan yaitu dengan *software defect prediction.* Dengan mendeteksi kecacatan software sejak dini, maka dapat mengalokasikan sumber daya dalam melakukan *testing.*

p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416
Website : jurnal.umj.ac.id/index.php/semnastek

Deteksi cacat *software* ini diperkenalkan sudah lebih dari 30 tahun [1]. Penelitian selama ini fokus kepada: 1. Memperkirakan jumlah cacat *software* pada sebuah sistem, 2. Menemukan asosiasi dari cacat *software*, 3. Mengklasifikasikan cacat atau tidaknya sebuah *software*. Dengan membangun sebuah sistem untuk memprediksi cacat *software* yang baik maka dapat pula mengurangi biaya pengembangan *software* [2].

Sudah lebih dari dua dekade model *software defect prediction* menjadi perhatian para peneliti. Model model algoritma digunakan untuk memprediksi cacat dari modul software sebelum melakukan test lebih lanjut. Dengan *sotware defect prediction* maka dapat diketahui proyek software mana yang butuh pengecekan lebih intens. Tim test software dapat mengalokasikan waktu dan biaya lebih efektif berdasarkan hasil dari model algoritma. Modul dengan prediksi cacat lebih membutuhkan fokus dari pada modul dengan prediksi tidak cacat. Gambar 1 menggambarkan bagaimana pembagian kerja pada tim software test.



Gambar 1. Pembagian kerja pada tim software test

Makalah ini akan mengaplikasikan metode *preprocessing* dengan pengoptimalan bobot dengan metode PSO. PSO merupakan algoritma pencarian berbasis populasi dan yang diinisialisi dengan populasi solusi acak yang disebut partikel [18]. Tidak seperti sistem teknik yang lain, di dalam metode PSO juga memperhatikan kecepatan, setiap partikel terbang melalui ruang pencarian dengan kecepatan yang dinamis berdasarkan histori kebiasaan mereka. Maka dari itu, setiap partikel memiliki kecenderungan untuk terbang menuju ke daerah pencarian yang lebih baik dan lebih baik lagi selama proses pencarian berlangsung [18].

Accuracy, Area Under Curve (AUC) adalah indikator utama dalam menentukan algoritma yang terbaik dalam pengklasifikasian. Untuk mencegah timbulnya hasil yang sangat beragam, dataset diacak dengan metode 10 fold cross-validation.

Makalah ini disusun sebagai berikut: pada bagian 2, penjelasan dari penelitian sebelumnya. Pada bagian 3, menampilkan metode yang diusulkan. Hasil eksperimen dari model yang diusulkan akan ditampilkan pada bagian 4. Dan pada bagian akhir, akan ditampilkan rangkuman dari makalah ini.

2. KAJIAN TEORI

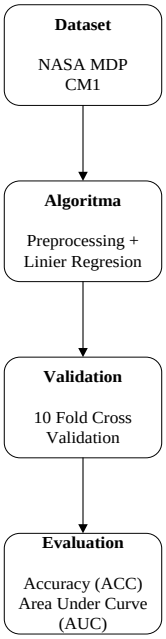
Pengembangan software bebas cacat merupakan tantangan tersendiri bagi para pengembang software. Maka dari itu telah banyak dilakukan penelitian untuk deteksi software terhadap kecacatan, sehingga software dapat di perbaiki sebelum di buka untuk umum atau dijual.

Penelitian software defect prediction telah banyak dilakukan, antara lain pada tahun 2000, G.Denaro mencoba untuk mengaplikasikan algoritma Logistic Regression untuk penelitiannya [7]. T. M. Khoshgoftaar, N. Seliya, dan K. Gao pada tahun 2005 membandingkan algoritma Decision Tree, discriminant analysis dan case base reasoning dalam penelitiannya [8]. Pada tahun yang sama, Gary D. Boetticher mencoba mengaplikasikan k-NN untuk software defect prediction pada penelitiannya [9]. Kemudian algoritma Naïve Bayes di teliti untuk software defect prediction pada tahun 2007 oleh T. Menzies, J. Greenwald, dan A. Frank [10]. Pada tahun 2008, S.Bibi et.al mencoba untuk menerapkan *regression via classification* untuk *software defect prediction* [19]. Pada tahun yang sama, Lesmann et.al mencoba mengkomparasi beberapa metode algoritma untuk menemukan akurasi yang terbaik [20]. Pada tahun 2010, Jun Zheng telah mencoba untuk mengaplilasikan algoritma *Neural Network* pada *software defect prediction* [21]. Algoritma Random Forest juga telah diujicoba pada tahun 2014 oleh Ishani Aroraa, Vivek Tetarwala dan Anju Sahaa [11].

3. METODE

Kerangka kerja yang diusulkan dapat dilihat pada gambar 2. Kerangka kerja terdiri dari 1) Dataset 2) Algoritma yang diusulkan 3) Metode validasi 4) Metode evaluasi

p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416
Website : jurnal.umj.ac.id/index.php/semnastek



Gambar 2 Kerangka kerja yang diusulkan

3.1 DATASET

Pada makalah ini, akan digunakan dataset public, sehingga penelitian ini dapat diulang, di ujicoba dan diverifikasi [12]. Dataset yang digunakan pada makalah ini diambil dari NASA MDP repository. NASA MDP Repository merupakan database yang menyimpan masalah, produk, dan data matrik dari software [13].

Pada setiap dataset NASA terdapat beberapa modul software, jumlah kesalahannya dan atribut kode karakteristik. Selain atribut Line of Code (LOC), dataset NASA MDP juga terdapat beberapa atribut Halstead [14] dan tingkat kompleksitas McCabe [15]. Peneliti terdahulu memperkirakan kompleksitas dari sebuah software dari banyaknya operator dan operan dari modul software.

Pada makalah ini akan digunakan dataset CM1 dari NASA MDP. Atribut atribut dari dataset, deskripsi dan nilainya dapat dilihat di tabel 1. Dataset yang disajikan mempunyai bahasa pemrograman C.

Tabel 1 Karakteristik dataset NASA MDP

Code Attributes		NASA MDP Dataset
		CM1
LOC counts	LOC_total	v
	LOC_blank	v
	LOC_code_and_comment	v
	LOC_comments	v
	LOC_executable	v
	number_of_lines	v
Halstead	content	v
	difficulty	v
	effort	v
	error_est	v
	length	v
	level	v
	prog_time	v
	volume	v
	num_operands	v
	num_operators	v
	num_unique_operands	v
	num_unique_operators	v
McCabe	cyclomatic_complexity	v

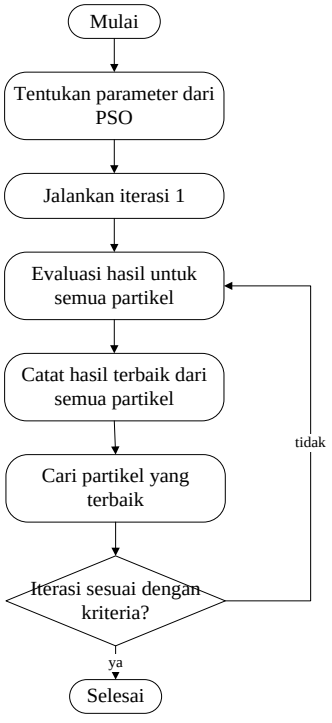
p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416
Website : jurnal.umj.ac.id/index.php/semnastek

	cyclomatic_density	v
	design_complexity	v
	essential_complexity	v
Misc.	branch_count	v
	call_pairs	v
	condition_count	v
	decision_count	v
	decision_density	v
	edge_count	v
	essential_density	v
	parameter_count	v
	maintenance_severity	v
	modified_condition_count	v
	multiple_condition_count	v
	normalized_cyclomatic_complexity	v
	percent_comments	v
	node_count	v
Programming Language		C

3.2 Algoritma Yang Diusulkan

Pada makalah ini akan menggunakan metode *prprocessing* dengan mengoptimalkan bobot atribut dengan menggunakan metode PSO. PSO diperkenalkan oleh Kennedy dan Eberhart pada tahun 1995. PSO merupakan algoritma pencarian berbasis populasi dan yang diinisialisi dengan populasi solusi acak yang disebut partikel [18]. Tidak seperti sistem teknik yang lain, di dalam metode PSO juga memperhatikan kecepatan, setiap partikel terbang melalui ruang pencarian dengan kecepatan yang dinamis berdasarkan histori kebiasaan mereka. Maka dari itu, setiap partikel memiliki kecenderungan untuk terbang menuju ke daerah pencarian yang lebih baik dan lebih baik lagi selama proses pencarian berlangsung [18]. Sehingga diharapkan akan mendapatkan hasil akurasi yang lebih baik.

Pada gambar 3 dapat dilihat flowchart dari algoritma yang diusulkan.



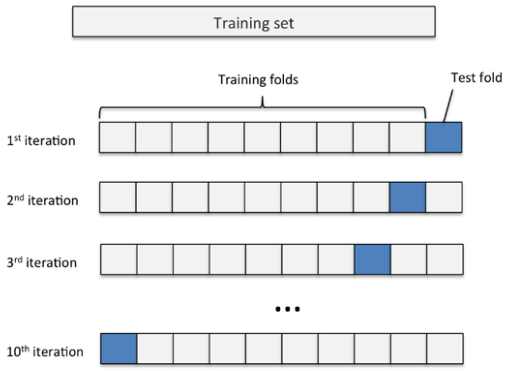
Gambar 3. Flowchart PSO

3.3 Metode Validasi

Validasi untuk data learning dan testing pada makalah ini menggunakan stratified 10-fold cross-validation. Validasi ini berarti keseluruhan data dibagi menjadi 10 bagian

p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416
Website : jurnal.umj.ac.id/index.php/semnastek

sama besar dan kemudian dilakukan proses learning sebanyak 10 kali. Pada gambar 4 bisa dilihat bahwa saat salah satu bagian dijadikan data testing, maka ke sembilan bagian data lainnya akan dijadikan sebagai data learning. Setelah itu dihitung rata rata akurasi dari masing masing iterasi untuk mendapatkan akurasinya. 10-fold cross-validation ini sudah menjadi standard dari penelitian akhir akhir ini, dan beberapa penelitian juga didapatkan bahwa penggunaan stratifikasi dapat meningkatkan hasil yang lebih tidak beragam [16].



Gambar 4. Stratified 10-fold cross-validation

3.4 Metode Evaluasi

Pada makalah ini akan digunakan Area under Curve (AUC) untuk mengukur performa akurasi dari algoritma, karena AUC dapat merepresentasikan yang paling baik dan indicator yang objektif dari akurasi prediksi [17]. Pada umumnya, algoritma yang memiliki nilai AUC diatas 0.6, mempunyai performa yang cukup efektif untuk mengenali modul software yang cacat. Pada tabel 3 dapat dilihat interpretasi dari masing masing nilai AUC.

AUC Value	Meaning
0.90 – 1.00	Excellent classification
0.80 – 0.90	Good classification
0.70 – 0.80	Fair classification
0.60 – 0.70	Poor classification
< 0.60	Failure

Tabel 3. Nilai AUC dan interpretasinya

4. HASIL DAN PEMBAHASAN

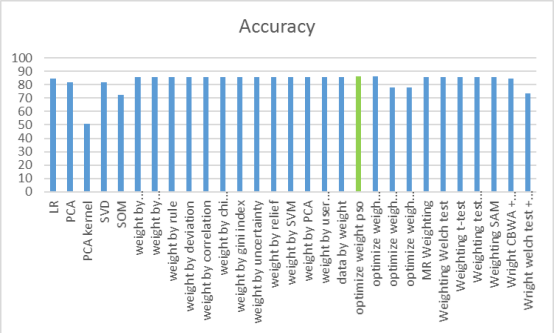
Eksperimen dilakukan pada laptop berbasis Core2Duo 1.66GHz CPU, 4 GB RAM dan sistem operasi Windows 10 Professional 64-bit. Aplikasi yang digunakan adalah RapidMiner 7.2 library.

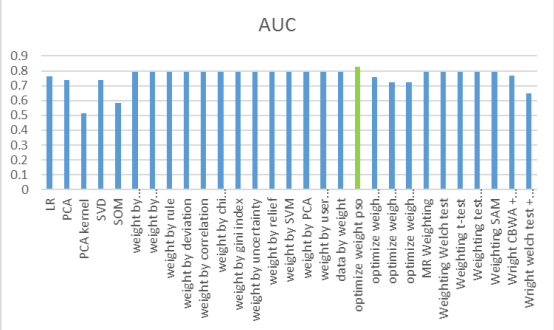
Pada makalah ini digunakan dataset CM1 software defect dari NASA MDP. Pada Tabel 1 dapat dilihat atribut atribut yang terdapat pada dataset.

Pada tabel 4 dapat dilihat nilai AUC dan ACC pada algoritma yang diujicoba.

No	Algoritma	ACC	AUC
1	LR	84.75	0.762
2	LR+PCA	81.51	0.739
3	LR+PCA kernel	50.71	0.515
4	LR+SVD	81.51	0.739
5	LR+SOM	72.29	0.584
6	LR+weight by information gain	85.54	0.794
7	LR+weight by information gain ratio	85.54	0.794
8	LR+weight by rule	85.54	0.794
9	LR+weight by deviation	85.54	0.794
10	LR+weight by correlation	85.54	0.794
11	LR+weight by chi squared statistic	85.54	0.794
12	LR+weight by gini index	85.54	0.794
13	LR+weight by uncertainty	85.54	0.794
14	LR+weight by relief	85.54	0.794
15	LR+weight by SVM	85.54	0.794
16	LR+weight by PCA	85.54	0.794
17	LR+weight by user specification	85.54	0.794
18	LR+data by weight	85.54	0.794
19	**LR+optimize weight pso**	**86.37**	**0.827**
20	LR+optimize weight forward + optimize selection weight guided	86.15	0.756
21	LR+optimize weight pso + optimize selection weight guided	78.1	0.722
22	LR+optimize weight pso + optimize selection weight guided	78.1	0.722
23	LR+MR Weighting	85.54	0.794
24	LR+Weighting Welch test	85.54	0.794
25	LR+Weighting t-test	85.54	0.794
26	LR+Weighting test significance	85.54	0.794
27	LR+Weighting SAM	85.54	0.794
28	LR+Wright CBWA + Optimize selection weight guided	84.36	0.766
29	LR+Wright welch test + Optimize selection weight guided	73.51	0.651

p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416
Website : jurnal.umj.ac.id/index.php/semnastek





5. KESIMPULAN

Berdasarkan penelitian perbandingan beberapa metode preprocessing terhadap dataset NASA MDP CM1, dengan model validasi 10 fold cross , maka didapatkan metode preprocessing dengan pembobotan atribut dengan metode PSO memiliki peningkatan akurasi menjadi 86.37% dari sebelumnya 85.54% dan AUC menjadi 0.827 dari sebelumnya 0.762.

DAFTAR PUSTAKA

[1] Q. Song, Z. Jia, M. Shepperd, S. Ying, J. Liu, A general software defect-pronenessprediction framework, IEEE Trans. Softw. Eng. 37 (3) (2011) 356–370.

[2] O.F. Arar, K. Ayan, Software defect prediction using cost-sensitive neuralnetwork, Appl. Soft Comput. 33 (C) (2015) 263–277.

[3] P. Michaels, Faulty Software Can Lead to Astronomic Costs, 2008, http://www.computerweekly.com/opinion/Faulty-software-can-lead-to-astronomic-costs, ComputerWeekly.com (retrieved 23.02.14).

[4] S. Dick, A. Meeks, M. Last, H. Bunke, A. Kandel, Data mining in software metricsdatabases Fuzzy Sets Syst. 145 (1) (2004) 81–110.

[5] L. Pelayo, S. Dick, Applying novel resampling strategies to software defect pre-diction, in: IEEE Fuzzy Information Processing Society, NAFIPS'07, San Diego,USA, June 24–27, 2007, pp. 69–72.

[6] J.D Lovelock, IT Spending Forecast, 2Q16 Update Gartner http://www.gartner.com/technology/research/it-spending-forecast/

[7] G. Denaro, "Estimating software fault-proneness for tuning testing activities," in *Proceedings of the 22nd International Conference on Software engineering - ICSE '00*, 2000, pp. 704–706.

[8] T. M. Khoshgoftaar, N. Seliya, and K. Gao, "Assessment of a New Three-Group Software Quality Classification Technique: An Empirical Case Study," *Empir. Softw. Eng.*, vol. 10, no. 2, pp. 183–218, Apr. 2005.

[9] Gary D. Boetticher, "Nearest Neighbor Sampling for Better Defect Prediction", ACM SIGSOFT Software Engineering Notes, vol 30, page 1-6, July 2005

[10] 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.

[11] Ishani Aroraa, Vivek Tetarwala,Anju Sahaa, "Open Issues in Software Defect Prediction", Procedia Computer Science 46, 906 – 912, 2015

[12] C. Catal and B. Diri, "Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem," *Inf. Sci. (Ny).*, vol. 179, no. 8, pp. 1040–1058, Mar. 2009.

[13] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, "Reflections on the NASA MDP data sets," *IET Softw.*, vol. 6, no. 6, p. 549, 2012.

[14] M. H. Halstead, *Elements of Software Science*, vol. 7. Elsevier, 1977, p. 127.

[15] T. J. McCabe, "A Complexity Measure," *IEEE Trans. Softw. Eng.*, vol. SE-2, no. 4, pp. 308–320, 1976.

[16] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining Third Edition*. Elsevier Inc., 2011.

p- ISSN : 2407 – 1846
e-ISSN : 2460 – 8416
Website : jurnal.umj.ac.id/index.php/semnastek

[17] Lessmann, S., Baesens, B., Mues, C., & Pietsch, S. "Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings." IEEE Transactions on Software Engineering, 34(4), 485–496, 2008

[18] A. Abraham, C. Grosan and V. Ramos, Swarm Intelligence In Data Mining, Verlag Berlin Heidelberg: Springer, 2006.

[19] S. Bibi and others, 'Regression via Classification Applied on Software Defect Estimation', *Expert Systems with Applications*, 34.3 (2008), 2091–2101.

[20] Stefan Lessmann and others, 'Benchmarking Classification Models for Software Defect Prediction: A Proposed Framework and Novel Findings', *IEEE Transactions on Software Engineering*, 34.4 (2008), 485–96.

[21] Jun Zheng, 'Cost-Sensitive Boosting Neural Networks for Software Defect Prediction', *Expert Systems with Applications*, 37.6 (2010), 4537–43.

Muhammad Rizki Fahdia, Lahir 17 November 1991, Bekerja sebagai System Engineer di PT Berlian Sistem Informasi (*BSI*). Pendidikan S2 di STIMIK Nusa Mandiri Jakarta jurusan Pasca sarjana Ilmu Komputer dengan fokus studi Data mining, pendidikan S1 di STIMIK Nusa Mandiri jurusan Sistem Informasi.

Prof. Richardus Eko Indrajit, Lahir di Jakarta, Indonesia, 24 Januari 1969. Lulus dari Institut Teknologi Surabaya sebagai Insinyur Komputer pada tahun 1992 dan mendapat beasiswa penuh dari Pertamina Oil Company untuk menyelesaikan studinya sebagai Master of Applied Computer Science di Harvard University, Massachusetts. , Amerika Serikat. Beliau juga merupakan pemegang Master of Business Administration dari Leicester University, Inggris, Master of Communication dari London School of Public Relasions - Jakarta, dan Master of Philosophy dari Masstricht School of Management, Belanda. Gelar Doktor Administrasi Bisnisnya berasal dari Pamantasan ng Lungsod ng Maynila (Universitas Kota Manila), Filipina.