JURNAL RESPONSIF, Vol. 2 No.2 Agustus 2020, pp. 198~203
E-ISSN**:** 2685-6964

PENERAPAN TEKNIK PSO OVER SAMPLING DAN ADABOOST J48 UNTUK MEMPREDIKSI CACAT *SOFTWARE*

Sugiono[1], Andi Taufik[2], Ricky Faizal Amir[3]

[1]Universitas Bina Sarana Informatika
e-mail: sugiono.sgx@bsi.ac.id

[2]STMIK Nusa Mandiri
e-mail: andi.iuf@nusamandiri.ac.id

[3]Universitas Bina Sarana Informatika
e-mail: richky.cfj@bsi.ac.id

Abstrak

Perangkat lunak yang bermutu ditentukan oleh jumlah cacat yang ditemukan pada saat proses pengujian. Proses perbaikan perangkat lunak setelah terdistribusi memiliki resiko yang lebih tinggi. Beberapa metode telah diujikan untuk memprediksi cacat pada perangkat lunak. Secara umum *dataset software metrics* telah digunakan sebagai acuan. *Dataset software metrics* bersifat tidak seimbang sehingga berpengaruh terhadap tingkat akurasi pemrediksi cacat perangkat lunak. Pada tahapan pra pemrosesan, digunakan metode *Particle Swarm optimization* (PSO) untuk mengatasi masalah polusi data serta metode *Random Over Sampling* (ROS) untuk menangani ketidak seimbangan kelas pada dataset. Metode yang diusulkan pada penelitian ini yaitu algoritma *decision tree* J48 yang dioptimalkan dengan teknik adaboost. *Dataset software metrics* yang digunakan pada penelitian ini bersumber pada *dataset* PROMISE *repository*. Hasil penelitian menunjukan bahwa penggunaan teknik adaboost pada algoritma *decision tree* J48 layak digunakan sebagai metode untuk memprediksi cacat pada perangkat lunak dengan nilai akurasi mencapai 93,507% dan nilai AUC mencapai 0,935.

Kata Kunci: Ketidakseimbangan Kelas, Adaboost, Decision Tree J48.

Abstract

Quality software is determined by the number of defects found during the testing process. The process of repairing software after distribution has a higher risk. Several methods have been tested to predict defects in software. In general, the metrics software dataset has been used as a reference. The metrics software dataset is unbalanced so it affects the accuracy of the prediction of software defects. At the pre-processing stage, the Particle Swarm optimization (PSO) method is used to overcome the problem of data pollution and the Random Over Sampling (ROS) method to deal with class imbalance in the dataset. The method proposed in this research is J48 decision tree algorithm which is optimized with adaboost technique. The metrics software dataset used in this study is sourced from the PROMISE repository dataset. The results showed that the use of the adaboost technique on the J48 decision tree algorithm is feasible as a method for predicting defects in software with an accuracy value reaching 93.507% and AUC values reaching 0.935

Keywords*: Imbalancing Class, Adaboost, Decision Tree J48.*

Naskah diterima 17 Juni 2020; direvisi 08 Juli 2020; diterbitkan 31 Agustus 2020

1. Pendahuluan

Kualitas suatu software dapat ditemukan pada saat fase pemeriksaan dan fase pengujian (Fitriani & Wahono, 2015). Cacat pada perangkat lunak merupakan kesalahan, kegagalan atau kesalahan pada sistem komputer tersebut sehingga menghasilkan kesalahan lainnya yang tak terduga serta dapat menurunkan kualitas suatu perangkat lunak (Wahono, Dian, Semarang, & Suryana, 2013). Perbaikan bug perangkat lunak setelah proses pemasangan 100 kali lebih mahal dari pada memperbaikinya selama proses pengembangan (Faruk, 2015). Hal tersebut di atas menunjukan betapa pentingnya proses pemrediksian cacat software agar dapat mengurangi segala resiko cacat yang dapat ditemukan.

Kajian prediksi cacat software dalam *software engineering* terutama dalam mengatasi masalah efektifitas dan efisiensi di luar pengujian perangkat lunak dan review, sehingga ketepatan prediksi cacat *software* dapat mempermudah pengujian, mengurangi biaya dan dapat meningkatkan kualitas *software* (Wahono, Suryana, & Ahmad, 2014). Selain itu prediksi cacat *software* dapat memungkinkan pengembang *software* mengalokasikan sumber daya yang terbatas sesuai dengan anggaran dan efisiensi waktu (Zheng, 2010). Rekayasa terhadap perangkat lunak memberikan kontribusi besar terhadap penghematan waktu dalam proses produksi perangkat lunak (Askari & Bardsiri, 2014).

Prediksi kesalahan pada perangkat lunak merupakan langkah penting untuk meningkatkan kualitas perangkat lunak. Kesalahan yang terdapat pada perangkat lunak di masa depan merupakan kesalahan pada data yang salah sebelumnya (Sathyaraj & Prabu, 2015). Oleh karena itu diperlukan adanya penelitian yang matang dalam memprediksi cacat pada perangkat lunak.

Banyak metode yang dapat digunakan untuk memprediksi modul cacat software dari potensi rawan kegagalan, salah satunya yang efektif adalah dengan menggunakan reka teknik data mining yang diaplikasikan pada *software metrics* yang diperoleh selama proses pengembangan *software* (Khoshgoftaar, 2010).

Pada penelitian sebelumnya Saifudin dan Wahono menawarkan metode penggabungan teknik *ensemble* adaboost dan *bagging* pada algoritma *naïve bayes* (Saifudin & Wahono, 2015). *Dataset* yang digunakan berupa 10 *dataset* NASA MDP *repository* dan 10 *dataset* PROMISE *repository*. Hasil evaluasi pada penelitian ini yaitu Model *bagging* dan *naïve bayes* meningkat secara signifikan terhadap model *naïve bayes* untuk pengukuran sensitivitas dan G-Mean, sedangkan model *adaboost* dan *naïve bayes* tidak memberikan peningkatan secara signifikan.

Pada penelitian yang dilakukan oleh Sathyaraj dan Prabu digunakan metode naïve bayes, J48, K*, dan Random forest (Sathyaraj & Prabu, 2015). Dataset yang digunakan yaitu NASA MDP *repository*. Hasil pengujian pada penelitian ini menyimpulkan algoritma naïve bayes cocok untuk *dataset* kecil dan algoritma *random forest* cocok untuk *dataset* besar.

Pada penelitian ini akan digunakan teknik PSO dan ROS pada tahap preprocessing data. Sedangkan pada tahapan pengolahan data digunakan teknik ensemble adaboost yang diintegrasikan dengan algoritma *decision tree* J48. Tujuan penelitian ini untuk mengetahui pengaruh keseimbangan kelas terhadap tingkat akurasi algoritma serta kemampuan teknik *adaboost* dalam meningkatkan kinerja algoritma *decision tree* J48.

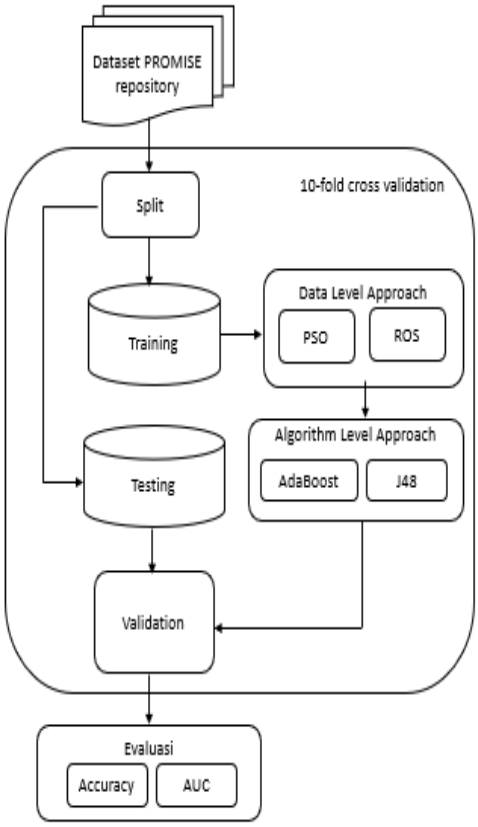
2. Metode Penelitian

Salah satu data metrik perangkat lunak yang populer digunakan dalam studi pengembangan perangkat lunak adalah *dataset* NASA. *Dataset* NASA tersedia dari dua sumber, yaitu NASA MDP (*Metrics Data Program*) *repository* dan *PROMISE* (*Predictor Models in Software Engineering*) *Repository* (Gray, Bowes, Davey, Sun, & Christianson, 2011). NASA *Metrics* Data Program (MDP) *Repository* saat ini terdiri dari 13 *dataset* secara eksplisit ditujukan untuk penelitian metrik perangkat lunak (Gray et al., 2011). Setiap kumpulan data merupakan sistem/ subsistem software NASA dan berisi metrik kode statis, dan data kesalahan dari setiap modul. Metrik kode statis yang dicatat meliputi ukuran LoC-count seperti jumlah baris kode dan komentar. ukuran Halstead seperti hitungan operan dan operator yang unik serta ukuran McCabe, seperti kompleksitas *cyclomatic*.

Objek data pada penelitian ini menggunakan dataset PROMISE repository bentuk pertama yang dapat diunduh dari PROMISE *repository* pada halaman http://promise.site.uottawa.ca/SERepository.

Pada penelitian ini teknik pengolahan data terbagi menjadi dua tahapan yakni tahapan *pre-processing* dan tahapan processing. Pada tahapan pre-processing

dilakukan pendekatan pada level data dengan menggunakan teknik PSO dan ROS. Teknik PSO digunakan untuk mengurangi polusi data dan teknik ROS digunakan untuk mengatasi ketidak seimbangan kelas pada *dataset software metrics*. Pada tahapan *processing* data digunakan teknik *ensemble adaboost* untuk mengoptimalkan kinerja algoritma *decision tree* J48. Pengujian dilakukan dengan menggunakan *10-fold cross validation* dengan mengambil 10% *dataset* untuk dijadikan sebagai data *testing* dan dilakukan secara berulang sehingga didapatkan tingkat akurasi dan AUC kinerja model.



Gambar 1. Kerangka Pemikiran Penelitian

3. Hasil dan Pembahasan

Dataset PROMISE *repository* merupakan data bersih yang sudah terbebas dari *inkonsistensi* serta *anomaly* sehingga tidak perlu untuk dilakukan proses pengolahan data. Dataset yang digunakan pada penelitan ini yaitu *dataset* CM1, KC1, dan PC1. Tahap *preporcessing* dilakukan dengan melihat beberapa ketentuan awal yang perlu menjadi perhatian diantaranya jumlah atribut, jumlah modul dan jumlah cacat pada setiap dataset. Spesifikasi dataset yang digunakan dalam penelitian ini sebagai berikut:

Tabel 1. Spesifikasi *dataset* yang digunakan.

Dataset	Atribut	Modul	Cacat	Tidak Cacat
CM1	22	498	49	449
KC1	22	2109	326	1783
PC1	22	1109	77	1032

Berdasarkan spesifikasi *dataset* yang telah dideskripsikan di atas dapat kita simpulkan bahwa *dataset* tersebut tidak seimbang atau *imbalanced* hal tersebut dikarenakan adanya kesenjangan antara jumlah modul cacat dengan jumlah modul tidak cacat. Hal tersebut dapat mempengaruhi kinerja model pengklasifikasian, untuk itu diperlukan suatu model yang dapat menangani ketidak seimbangan tersebut. Setelah proses penyeimbangan data, tahap selanjutnya adalah melakukan uji coba pengklasifikasian menggunakan algoritma decision tree J48 yang hasilnya dapat dibandingkan dengan model usulan sehingga terlihat pengaruh model usulan terhadap pengklasifikasian tersebut.

Kita ketahui bahwa semua dataset yang digunakan untuk proses pengujian terindikasi tidak seimbang yaitu terjadi perbedaan antara kelas yang berlabel “Y” dengan kelas yang berlabel “N” pada *atribut defect*. Hal tersebut perlu diperhatikan dalam prose pengklasifikasian supaya data minoritas dapat diklasifikasikan dengan baik, untuk itu metrik yang tepat digunakan sebagai tolak ukur pengklasifikasian adalah *Are Under ROC Curve (AUC), F-Measure, G-Means*, dan akurasi.

Untuk menghitung akurasi pada kelas minoritas dapat digunakan metrik *TPrate/recall (sensitivitas). G-Mean* dan *AUC* merupakan evaluasi prediktor yang lebih komprehensif dalam konteks ketidak seimbangan. Sedangkan F-*Measure* mengkombinasikan *recall/sensitivitas* dan *precision* sehingga menghasilkan metrik yang efektif untuk pencarian kembali informasi dalam himpunan yang mengandung masalah ketidak seimbangan. Secara prinsip, F-*Measure* merepresentasikan rata-rata harmonis antara *recall* dan *precision*.

Pengujian dilakukan dengan melakukan *preprocessing* awal terhadap *dataset* dengan menggunakan teknik PSO. Teknik ini digunakan untuk menghilangkan polusi atau *noise* yang terdapat pada *dateset*.

Tahap *preprocessing* data berikutnya dengan menghilangkan ketidak seimbangan kelas pada *dataset*. Pada tahap ini dilakukan

imbalancing dengan menggunakan teknik ROS. Teknik ini menduplikasi kelas minoritas sehingga jumlahnya seimbang dengan kelas mayoritas.

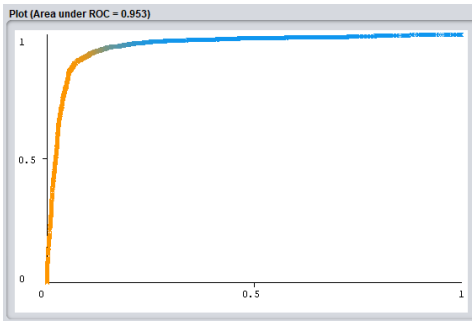
Pada tahapan pengolahan data digunakan algoritma *decision tree* J48 sebagai algoritma pengklasifikasi. Untuk lebih mengoptimalkan kinerja model algoritma *decision tree* J48 diintegrasikan dengan teknik *ensemble adaboost*.

Pengujian dilakukan dengan teknik 10-*fold cross validation*. Teknik pengujian ini dilakukan dengan membagi 10% *dataset* sebagai data *testing*. Pengujian dilakukan secara berulang terhadap semua *dataset*.

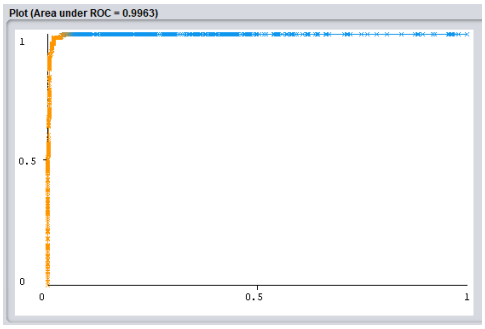
Hasil pengujian berupa tabel *confusion matrix* yang terdiri dari hasil *True Positif* (TP), *False Negative* (FN), *False Positive* (FP) dan *True Negative* (TN). Selanjutnya dilakukan penghitungan untuk mencari akurasi, *sensitivitas*/ *recall*/ *TPrate*, *specificity*/ *TNrate*, *FPrate*, *FNrate*, *Precision*/ PPV, F-*Measure*, G-*Means* dan AUC. *Confusion matrix* dari hasil pengujian tersebut terdapat pada tabel 2.

Tabel 2. *Confusion matriks* hasil pengujian

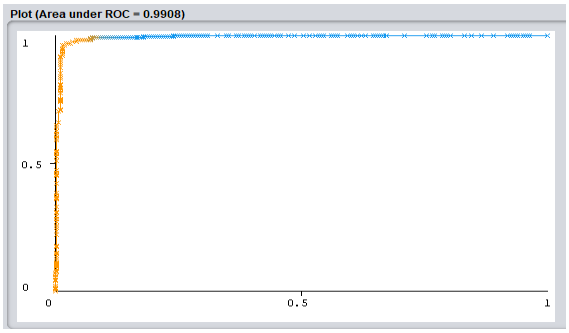
Dataset	TP	FN	FP	TN
KC1	978	76	118	936
PC1	554	0	27	527
CM1	247	2	21	228



Gambar 2. Diagram AUC *dataset* KC1



Gambar 3. Diagram AUC Dataset PC1



Gambar 4. Diagram AUC *dataset* CM1

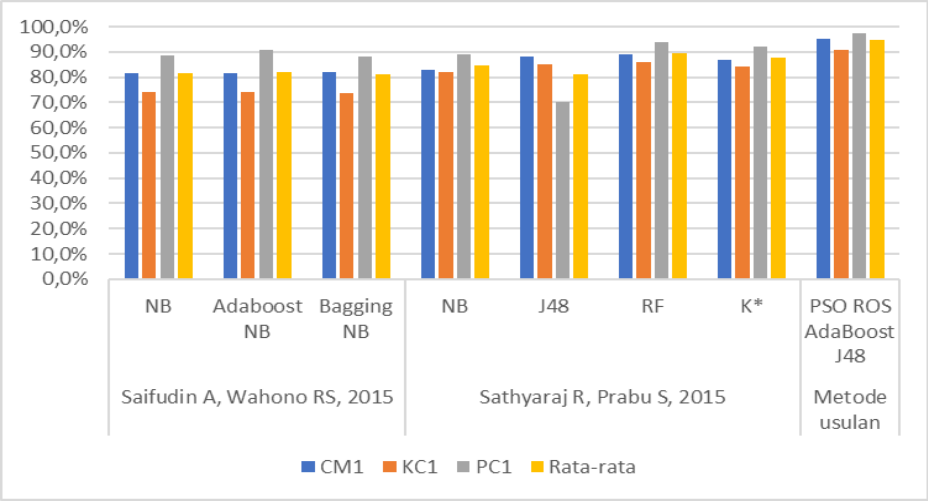
Tabel 3. Hasil pengujian model

Dataset	KC1	PC1	CM1	Rata-rata
Akurasi	90,1%	97,6%	95,4%	93,5%
Sensitivity	0,928	1,000	0,992	0,964
Specificity	0,888	0,951	0,916	0,906
FP-rate	0,112	0,049	0,084	0,094
FN-rate	0,072	0,000	0,008	0,036
Precision	0,892	0,954	0,922	0,911
F-Measure	0,910	0,976	0,956	0,937
G-Means	0,908	0,975	0,953	0,935
AUC	0,908	0,976	0,954	0,935

Dapat kita lihat pada tabel diatas bahwa nilai akurasi tertinggi yang dihasilkan dari integrase model PSO ROS AdaBoost dan J48 terdapat pada *dataset* PC1 dengan nilai akurasi mencapai 97,6% dan nilai AUC tertinggi juga ditunjukan pada dataset PC1 dengan nilai AUC mencapai 0,976. Berikut ini grafik perbandingan kinerja model terhadap dataset CM1, KC1 dan PC1.

Tabel 4. Perbandingan pengujian model dengan penelitian terkait

Penelitian	METODE	CM1	KC1	PC1
Saifudin A, Wahono RS, 2015	NB	81,7	74,1	88,8
	Adaboost NB	81,7	74,1	90,9
	Bagging NB	82,0	73,8	88,2
Sathyaraj R, Prabu S, 2015	NB	83,0	82,0	89,0
	J48	88,0	85,0	70,0
	RF	89,0	86,0	94,0
	K*	87,0	84,0	92,0
Metode usulan	PSO ROS AdaBoost J48	95,4	90,8	97,6



Gambar 5. Diagram perbandingan hasil pengujian model

Hasil komparasi model usulan dengan penelitian sebelumnya yang terkait dengan pemrediksi cacat *software* menunjukan bahwa model pemrediksi cacat *software* yang diusulkan memiliki kinerja yang lebih unggul dibandingkan dengan model pemrediksi cacat software yang diusulkan pada penelitian sebelumnya. Keunggulan kinerja ditunjukan pada model pemrediksi cacat *software* berbasis PSO ROS *AdaBoost* J48 dengan nilai rata-rata akurasi mencapai 94,6%.

4. Kesimpulan

Pada tahapan prepocessing dalam pemprediksi cacat *software* digunakan metode *Particle Swarm Optimization* (PSO) untuk mengatasi permasalahan *noise* pada data training. Untuk mengatasi permasalahan ketidak seimbangan kelas pada *dataset software metrics* dapat digunakan metode *resampling* yaitu *Random Over Sampling* (ROS). Kinerja algoritma J48 dapat lebih ditingkatkan lagi melalui kombinasi dengan metode AdaBoost. Hasil pengujian menunjukan metode dengan integrasi seleksi fitur PSO, pendekatan level data ROS, pendekatan level algoritma *AdaBoost* dan algoritma pengklasifikasi J48 menunjukan kinerja pengklasifikasian yang tinggi dengan tingkat rata-rata akurasi 94% dan AUC 0,935. Sehingga metode PSO ROS *AdaBoost* J48 layak digunakan sebagai metode pemrediksian cacat *software*.

Referensi

Askari, M. M., & Bardsiri, V. K. (2014). *Software Defect Prediction using a High Performance Neural Network*. *8*(12), 177–188.

Faruk, Ö. (2015). Software defect prediction using cost-sensitive neural network. *Elsevier*, *33*, 263–277. https://doi.org/10.1016/j.asoc.2015.04.045

Fitriani, & Wahono, R. S. (2015). Integrasi Bagging dan Greedy Forward Selection pada Prediksi Cacat Software dengan Menggunakan Naïve Bayes. *Journal of Software Engineering*, *1*(2), 101–108.

Gray, D., Bowes, D., Davey, N., Sun, Y., & Christianson, B. (2011). *The Misuse of the NASA Metrics Data Program Data Sets for Automated Software Defect Prediction*. 96–103.

Khoshgoftaar, T. M. (2010). *Attribute Selection and Imbalanced Data : Problems in Software Defect Prediction*. https://doi.org/10.1109/ICTAI.2010.27

Saifudin, A., & Wahono, R. S. (2015). *Penerapan Teknik Ensemble untuk Menangani Ketidakseimbangan Kelas pada Prediksi Cacat Software*. *1*(1).

Sathyaraj, R., & Prabu, S. (2015). *An Approach for Software Fault Prediction to Measure the Quality of Diferent Prediction Methodologies using Software Metrics*. *8*(December). https://doi.org/10.17485/ijst/2015/v8i35/73717

Wahono, R. S., Dian, U., Semarang, N., & Suryana, N. (2013). *Combining Particle*

Swarm Optimization based Feature Selection and Bagging Technique for Software Defect Prediction Combining Particle Swarm Optimization based Feature Selection and Bagging Technique for Software Defect Prediction. (September). https://doi.org/10.14257/ijseia.2013.7.5.16

Wahono, R. S., Suryana, N., & Ahmad, S. (2014). *Metaheuristic Optimization based Feature Selection for Software Defect Prediction*. *9*(5), 1324–1333. https://doi.org/10.4304/jsw.9.5.1324-1333

Zheng, J. (2010). Expert Systems with Applications Cost-sensitive boosting neural networks for software defect prediction. *Expert Systems With Applications*, *37*(6), 4537–4543. https://doi.org/10.1016/j.eswa.2009.12.056