STRATEGI *RESAMPLING* BERBASIS *CENTROID* UNTUK MENANGANI KETIDAKSEIMBANGAN KELAS PADA PREDIKSI CACAT PERANGKAT LUNAK

Utomo Pujianto

Abstrak: Dataset yang digunakan pada penelitian prediksi cacat perangkat lunak umumnya bersifat tidak seimbang, sehingga dapat menurunkan kinerja model prediksi cacat perangkat lunak. Ketidakseimbangan kelas dapat ditangani dengan dua pendekatan, yaitu pada aras data dan aras algoritma. Pendekatan aras data ditujukan untuk memperbaiki keseimbangan kelas, sedangkan pendekatan aras algoritma ditujukan untuk memperbaiki algoritma pengklasifikasi agar lebih sensitif terhadap kelas minoritas. Pada penelitian ini diusulkan pendekatan aras data dengan metode *resampling* yang menggunakan jarak terhadap centroid kelas minoritas sebagai dasar pemilihan data sampel baru. Algoritma klasifikasi yang digunakan adalah Naïve Bayes. Hasil penelitian menunjukkan bahwa kombinasi metode resampling baru dan Naïve Bayes, yang kemudian disebut dengan model farCentro+NB merupakan model pendekatan yang lebih baik untuk memprediksi cacat perangkat lunak karena nilai Akurasi, Sensitivitas, Fmeasure, dan *Area Under Curve (AUC)* model farCentro+NB meningkat secara signifikan, sedangkan model RUS+NB tidak meningkat secara signifikan.

Kata Kunci: Ketidakseimbangan Kelas, Klasifikasi, Resampling, Cacat, Perangkat Lunak

Salah satu ukuran kualitas perangkat lunak adalah jumlah cacat yang ada pada produk yang dihasilkan (Turhan & Bener, 2007, p. 244). Perangkat lunak dikatakan berkualitas tinggi jika tidak ditemukan cacat selama pemeriksaan dan pengujian, serta dapat memberikan nilai kepada pengguna dan memenuhi harapan mereka (McDonald, Musson, & Smith, 2008, pp. 4-6). Perangkat lunak yang cacat akan menyebabkan biaya pengembangan, perawatan dan estimasi menjadi tinggi (Gayatri, Nickolas, Reddy, & Chitra, 2009, p. 393).

Cara yang paling umum dilakukan untuk mencari cacat perangkat lunak adalah debugging, yaitu pencarian dengan melibatkan seluruh source code, berjalannya, keadaannya, dan riwayatnya (Weiss, Premraj, Zimmermann, & Zeller, 2007, p. 1). Fakta bahwa potensi cacat tertinggi terjadi pada tahap pengkodean (Jones, 2013, p. 3) menjadi dasar pertimbangan ditempuhnya cara ini.

Cara ini membutuhkan sumber daya yang besar dan cenderung kurang efisien. Pengujian merupakan tahapan pengembangan perangkat lunak yang paling mahal dan banyak memakan waktu, karena sekitar 50% dari jadwal proyek digunakan untuk pengujian (Fakhrahmad & Sami, 2009, p. 206). Untuk itu diperlukan strategi pengujian yang efektif dengan menggunakan sumber daya yang efisien untuk mengurangi kebutuhan waktu dan biaya pengembangan perangkat lunak.

Di antara berbagai macam cara yang dapat digunakan untuk menangani cacat perangkat lunak, pencegahan cacat adalah pendekatan yang terbaik. Hal ini dikarenakan manfaat dari upaya pencegahannya dapat diterapkan ulang di kemudian hari (McDonald, Musson, & Smith, 2008, p. 4). Pencegahan cacat dilakukan dengan memprediksi kemungkinan terjadinya cacat pada komponen-komponen perangkat lunak.

Utomo Pujianto adalah Dosen Jurusan Teknik Elektro Universitas Negeri Malang

Sampai saat ini prediksi cacat perangkat lunak memusatkan perhatian pada: 1) bagaimana memperkirakan jumlah cacat dalam perangkat, 2) bagaimana menemukan hubungan cacat, dan 3) bagaimana mengklasifikasikan kerawanan cacat dari komponen perangkat lunak, ke dalam kelompok rawan dan tidak rawan (Song, Jia, Shepperd, Ying, & Liu, 2011, p. 356). Klasifikasi adalah pendekatan yang populer untuk memprediksi cacat software (Lessmann, Baesens, Mues, & Pietsch, 2008, p. 485).

Salah satu masalah yang dihadapi dalam prediksi cacat perangkat lunak adalah ketidakseimbangan kelas. Hal ini terjadi secara alamiah, karena data modul perangkat lunak yang teridentifikasi tidak cacat mendominasi dalam jumlah terhadap data modul perangkat lunak yang teridentifikasi memiliki cacat.

Ada dua pendekatan umum yang dipakai untuk menangani permasalahan ketidakseimbangan kelas, yaitu pendekatan aras data, dan pendekatan aras algoritma. Metode-metode yang termasuk dalam pendekatan aras data antara lain Random Under Sampling, Random Over Sampling, dan SMOTE. Sedangkan metode-metode yang termasuk dalam pendekatan aras algoritma antara lain Bagging, Boosting, dan Random Forest.

Penelitian ini menggunakan pendekatan pada aras data untuk menangani masalah ketidakseimbangan kelas pada dataset prediksi cacat perangkat lunak. Pendekatan level data yang dimaksud adalah metode resampling yang mengadopsi sebagian prinsip yang digunakan pada metode Random Under Sampling.

METODE

Penelitian ini dilakukan dengan mengusulkan model, menerapkannya pada dataset yang tersedia secara publik, kemudian mengukur kinerjanya.

Dataset yang digunakan dalam penelitian ini berasal dari NASA (National Aeronautics and Space Administration) MDP (Metrics Data Program) repository. Dataset ini tersedia secara publik dan sudah umum digunakan para peneliti dalam penelitian Software Engineering (Hall et al., 2011). Data NASA MDP dikhususkan untuk topik penelitian cacat perangkat lunak dan kegagalan perangkat lunak. Penelitian ini menggunakan data NASA dari MDP yang sudah diperbaiki oleh Martin Shepperd dkk dengan melakukan preprocessing terhadap data yang bernilai null atau data yang kosong (Liebchen & Shepperd, 2008). Dataset Nasa MDP Repository ditunjukan pada Tabel 1.

Langkah pertama pada penelitian ini adalah normalisasi terhadap data. Hal ini dilakukan dengan pertimbangan bahwa atribut-atribut metrik perangkat lunak yang digunakan memiliki rentang nilai yang tidak seragam.

Langkah berikutnya adalah penentuan centroid kelas minoritas dalam dataset. Persamaan yang digunakan untuk menghitung lokasi centroid masing-masing kelas ditunjukkan oleh persamaan (1).

Langkah ketiga adalah melakukan resampling berdasarkan jarak masing-masing data pada kelas mayoritas terhadap centroid kelas minoritas. Algoritma yang diusulkan dalam penelitian ini mengadopsi prinsip *under-sampling*, yaitu mengurangi populasi kelas mayoritas sedemikian hingga seimbang dalam jumlah dengan kelas minoritas. Data dari kelas mayoritas yang akan dipilih untuk digunakan kembali sebagai data latih adalah data-data yang jaraknya paling jauh terhadap centroid data-data kelas minoritas. Pertimbangan pemilihan tersebut adalah bahwa data-data tersebut diasumsikan membentuk gugus tersendiri sedemikian rupa sehingga dapat

secara efektif dibedakan dengan data-data kelas minoritas.

Langkah keempat dalam penelitian ini adalah menggunakan hasil *resampling* yang didapat pada langkah ketiga sebagai data masukan untuk model prediksi berbasis Naïve Bayes. Kinerja dari model prediksi kemudian dievaluasi menggunakan 10-fold cross validation.

Metode resampling yang digunakan sebagai pembanding terhadap metode yang diusulkan adalah random undersampling (RUS). Pada perangkat lunak WEKA, metode ini diterapkan dengan mengaplikasikan filter SpreadSubSample pada data yang telah dinormalisasi.

Perangkat lunak yang digunakan untuk membangun model adalah WEKA dan Microsoft Excel.

Untuk mengukur kinerja model digunakan confusion matrix, karena confusion matrix merupakan alat yang berguna untuk menganalisa seberapa baik pengklasifikasi dapat mengenali tupel/fitur dari kelas yang berbeda (Han, Kamber, & Pei, 2011, p. 365). Confusion matrix dapat membantu menunjukkan rincian kinerja pengklasifikasi dengan memberikan informasi jumlah fitur suatu kelas yang diklasifikasikan dengan tepat dan tidak tepat (Bramer, 2007, p. 89). Confusion matrix memberikan penilaian kinerja model klasifikasi berdasarkan jumlah objek yang diprediksi dengan benar dan salah (Gorunescu, 2011, p. 319). Confusion matrix merupakan matrik 2 dimensi yang menggambarkan perbandingan antara hasil prediksi dengan kenyataan.

Untuk data tidak seimbang, akurasi lebih didominasi oleh ketepatan pada data kelas minoritas, maka metrik yang tepat adalah AUC (Area Under the ROC Curve), F-Measure, GMean, akurasi keseluruhan, dan akurasi untuk kelas minoritas (Zhang & Wang, 2011, p. 85). Akurasi kelas minoritas dapat menggunakan metrik TPrate/recall (sensitivitas).

Tabel 1. NASA MDP Datasets dan Atribut

Code Attributes		**NASA MDP Dataset**				
		CM1	**KC3**	**MC2**	**PC1**	**PC3**
LOC counts	LOC_total	√	√	√	√	√
	LOC_blank	√	√	√	√	√
	LOC_code_and_comment	√	√	√	√	√
	LOC_comments	√	√	√	√	√
	LOC_executable	√	√	√	√	√
	number_of_lines	√	√	√	√	√
Halstead	content	√	√	√	√	√
	difficulty	√	√	√	√	√
	effort	√	√	√	√	√
	error_est	√	√	√	√	√
	length	√	√	√	√	√
	level	√	√	√	√	√
	prog_time	√	√	√	√	√
	volume	√	√	√	√	√
	num_operands	√	√	√	√	√
	num_operators	√	√	√	√	√
	num_unique_operands	√	√	√	√	√
	num_unique_operators	√	√	√	√	√
McCabe	cyclomatic_complexity	√	√	√	√	√
	cyclomatic_density	√	√	√	√	√
	design_complexity	√	√	√	√	√
	essential_complexity	√	√	√	√	√
Misc	branch_count	√	√	√	√	√
	call_pairs	√	√	√	√	√
	condition_count	√	√	√	√	√
	decision_count	√	√	√	√	√
	decision_density	√	√	√	√	√
	edge_count	√	√	√	√	√
	essential_density	√	√	√	√	√
	parameter_count	√	√	√	√	√
	maintenance_severity	√	√	√	√	√
	modified_condition_count	√	√	√	√	√
	multiple_condition_count	√	√	√	√	√
	global_data_complexity		√	√		
	global_data_density		√	√		
	normalized_cyclo_complx	√	√	√	√	√
	percent_comments	√	√	√	√	√
	node_count	√	√	√	√	√

G-Mean dan AUC merupakan evaluasi prediktor yang lebih komprehensif

dalam konteks ketidakseimbangan (Wang & Yao, 2013, p. 438). Rumus-rumus yang digunakan untuk melakukan penghitungannya adalah (Gorunescu, 2011, pp. 320-322):

$$akurasi = \frac{TP + TN}{TP + TN + FP + FN}$$

$$sensitifitas = TP_{rate} = \frac{TP}{TP + FN}$$

$$F\ measure = \frac{2 * recall * precision}{recall + precision}$$

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2}$$

F-Measure adalah metrik evaluasi yang populer untuk masalah ketidakseimbangan. F-Measure mengkombinasikan recall/sensitivitas dan precision sehingga menghasilkan metrik yang efektif untuk pencarian kembali informasi dalam himpunan yang mengandung masalah ketidakseimbangan.

Area Under the ROC (Receiver Operating Characteristic) Curve (AUROC atau AUC) adalah ukuran numerik untuk membedakan kinerja model, dan menunjukkan seberapa sukses dan benar peringkat model dengan memisahkan pengamatan positif dan negatif (Attenberg & Ertekin, 2013, p. 114). AUC menyediakan ukuran tunggal dari kinerja pengklasifikasi untuk menilai model mana yang lebih baik secara rata-rata (López, Fernández, & Herrera, 2014, p. 4).

AUC merangkum informasi kinerja pengklasifikasi ke dalam satu angka yang mempermudah perbandingan model ketika tidak ada kurva ROC yang mendominasi (Weiss, 2013, p. 27). AUC merupakan cara yang baik untuk mendapatkan nilai kinerja pengklasifikasi secara umum dan untuk membandingkannya dengan pengklasifikasi yang lain (Japkowicz, 2013, p. 202). AUC adalah ukuran kinerja yang populer dalam ketidakseimbangan kelas, nilai AUC yang tinggi menunjukkan kinerja yang lebih baik (Liu & Zhou, 2013, p. 75). Sehingga untuk memilih model mana yang terbaik, dapat dilakukan dengan menganalisa nilai AUC.

HASIL

Eksperimen yang dilakukan dalam penelitian ini menggunakan 5 dataset NASA MDP (CM1, KC3, MC2, PC1, dan PC3). Yang dikategorikan sebagai dataset D' pada repository Shepperd. Metode yang diuji adalah metode pengklasifikasi farCentro+NB dan RUS+NB. Hasil eksperimen disajikan pada Tabel 2 sampai dengan Tabel 5, dengan informasi yang disajikan secara berturut-turut adalah akurasi, sensitivity (recall), F-Measure, dan AUC.

Tabel 2 Akurasi Model

Data Set	farCentro+NB (%)	RUS+NB (%)
CM1	79,82	63,04
MC2	63,14	61,08
KC3	77,39	57,80
PC1	86,13	63,13
PC3	57,43	52,36

Tabel 3 Sensitifitas Model (true positive rate)

Data Set	farCentro+NB	RUS+NB
CM1	0,71	0,87
MC2	0,84	0,87
KC3	0,80	0,79
PC1	0,90	0,88
PC3	0,22	0,13

Tabel 4 F-Measure Model

Data Set	farCentro+NB	RUS+NB
CM1	0,76	0,70
MC2	0,69	0,69
KC3	0,77	0,65

PC1	0,87	0,71
PC3	0,33	0,16

Tabel 5 AUC Model

Data Set	farCentro+NB	RUS+NB
CM1	0,88	0,76
MC2	0,82	0,69
KC3	0,79	0,61
PC1	0,90	0,72
PC3	0,89	0,77

PEMBAHASAN

Hasil uji akurasi seperti diperlihatkan pada Tabel 3 menunjukan rata-rata akurasi terhadap lima dataset menggunakan metode farCentro+NB adalah 72,78% sedangkan rata-rata akurasi metode RUS+NB adalah 59,48%.

Rata-rata f-measure dan AUC juga memperlihatkan nilai yang lebih baik pada metode farCentro+NB dibandingkan dengan metode RUS+NB. Hanya pada rerata sensitivitas nilai kinerja farCentro+NB tidak lebih baik dari metode RUS+NB. Walaupun demikian, kinerja metode farCentro+NB pada uji sensitivitas relatif cukup kompetitif mengingat hasil pengujian terhadap empat dari limadataset menghasilkan nilai yang lebih baik.

KESIMPULAN

Dua metode sampling, yaitu Random Under Sampling dan farCentro, telah diusulkan untuk meningkatkan kinerja pengklasifikasi Naïve Bayes dan dilakukan pengukuran kinerjanya. Selanjutnya dilakukan uji statistik terhadap hasil pengukuran.

Berdasarkan hasil eksperimen yang telah dilakukan menunjukkan bahwa model yang mengintegrasikan metode sampling farCentro dengan pengklasifikasi Naïve Bayes (farCentro+NB) memberikan hasil yang lebih baik dibandingkan dengan metode RUS+NB (Random Under Sampling – Naïve Bayes). Secara umum kemampuan memprediksi kelas minoritas meningkat secara signifikan berdasarkan pengukuran Akurasi, Sensitivitas, $F_{measure}$ dan Area Under Curve (AUC). Sehingga disimpulkan bahwa teknik sampling farCentro lebih baik daripada teknik Random Under Sampling ketika diintegrasikan dengan pengklasifikasi Naïve Bayes pada prediksi cacat perangkat lunak.

DAFTAR RUJUKAN

Turhan, B., & Bener, A. (2007). Software Defect Prediction: Heuristics for Weighted Naive Bayes. Proceedings of the 2nd International Conference on Software and Data Technologies (ICSOFT'07), (pp. 244-249).

McDonald, M., Musson, R., & Smith, R. (2008). The Practical Guide to Defect Prevention. Washington: Microsoft Press.

Gayatri, N., Nickolas, S., Reddy, A., & Chitra, R. (2009). Performance Analysis Of Data Mining Algorithms for Software Quality Prediction. International Conference on Advances in Recent Technologies in Communication and Computing (pp. 393-395). Kottayam: IEEE Computer Society.

Weiss, C., Premraj, R., Zimmermann, T., & Zeller, A. (2007). How Long will it Take to Fix This Bug? Fourth International Workshop on Mining Software Repositories (MSR'07) (pp. 1-8). Washington: IEEE Computer Society.

Jones, C. (2013). Software Defect Origins and Removal Methods. Namcook Analytics.

Fakhrahmad, S. M., & Sami, A. (2009). Effective Estimation of Modules' Metrics in Software Defect Prediction.

Proceedings of the World Congress on Engineering (pp. 206-211). London: Newswood Limited.

Lehtinen, T. O., Mäntylä, M. V., Vanhanen, J., Itkonen, J., & Lassenius, C. (2014). Perceived Causes of Software Project Failures - An Analysis of Their Relationships. Information and Software Technology, 623–643.

In, H. P., Baik, J., Kim, S., Yang, Y., & Boehm, B. (2006, December). A Quality-Based Cost Estimation Model for the Product Line Life Cycle. Communications of the ACM, 49(12), 85-88. doi:10.1145/1183236.1183273

Song, Q., Jia, Z., Shepperd, M., Ying, S., & Liu, J. (2011). A General Software Defect-Proneness Prediction Framework. IEEE Transactions on Software Engineering, 356-370.

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

Hall, T., Beecham, S., Bowes, D., Gray, D., & Counsell, S. (2011). A Systematic Literature Review on Fault Prediction Performance in Software Engineering. IEEE Transactions on Software Engineering, Accepted for publication - available online, 1-31.

Liebchen, G. a., & Shepperd, M. (2008). Data sets and data quality in software engineering. Proceedings of the 4th International Workshop on Predictor Models in Software Engineering.

Han, J., Kamber, M., & Pei, J. (2011). Data Mining: Concepts and Techniques (3rd ed.). San Francisco: Morgan Kaufmann Publishers Inc.

Bramer, M. (2007). Principles of Data Mining. London: Springer

Gorunescu, F. (2011). Data Mining: Concepts, Models and Techniques. Berlin: Springer-Verlag.

Zhang, H., & Wang, Z. (2011). A Normal Distribution-Based Over-Sampling Approach to Imbalanced Data Classification. Advanced Data Mining and Applications - 7th International Conference (pp. 83-96). Beijing: Springer.

Wang, S., & Yao, X. (2013). Using Class Imbalance Learning for Software Defect Prediction. IEEE Transactions on Reliability, 434-443.

Attenberg, J., & Ertekin, S. (2013). Class Imbalance and Active Learning. In H. He, & Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications (pp. 101-149). New Jersey: John Wiley & Sons.

López, V., Fernández, A., & Herrera, F. (2014). On the Importance of the Validation Technique for Classification with Imbalanced Datasets: Addressing Covariate Shift when Data is Skewed. Information Sciences, 1–13. doi:10.1016/j.ins.2013.09.038

Weiss, G. M. (2013). Foundations of Imbalanced Learning. In H. He, & Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications (pp. 13-41). New Jersey: John Wiley & Sons.

Japkowicz, N. (2013). Assessment Metrics for Imbalanced Learning. In H. He, & Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications (pp. 187-206). New Jersey: John Wiley & Sons.

Liu, X.-Y., & Zhou, Z.-H. (2013). Ensemble Methods for Class Imbalance Learning. In H. He, & Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications (pp. 61-82). New Jersey: John Wiley & Sons.