Penerapan Greedy Forward Selection dan Bagging pada Logistic Regression untuk Prediksi Cacat Perangkat Lunak

Rakhmat Purnomo
Program Studi Teknik Informatika, Fakultas Teknik, Universitas Bhayangkara Jaya
Kampus II, Jl. Raya Perjuangan Kel. Marga Mulya, Bekasi Utara, Kota bekasi Jawa 17121
rakhmat.purnomo@dsn.ubharjaya.ac.id

ABSTRACT

Software defects are errors or failures in software. Software defect detection manually can only produce 60% of the total existing defects. Defect prediction method using probability to find up to 71% better than the method used by the industry. One of the best methods for prediction of software defects is Logistic Regression. Logistic Regression is a linear classifier that has been shown to produce a powerful classification with statistical probabilities and handle multi-class classification problem. The main weakness of Logistic Regression algorithm is a class imbalance in high-dimensional datasets. Dataset software metric used is NASA dataset MDP. The dataset is generally unbalanced and experiencing problems with redundant data. This paper proposed a greedy forward selection method to solving the problem of redundant data and bagging technic to solving the class imbalance. The algorithm used is the Logistic Regression. Results of the experiments in this study scored the highest accuracy in the dataset PC2 at 0,990, up 0.19% compared with logistic regression method without GFS and bagging. While the highest AUC value of 0.995 at PC2, an increase of 7.94% compared to logistic regression method without GFS and bagging.

Keywords: *Greedy Forward Selection, Bagging, Logistic Regression, dataset NASA MDP, Software Defect Prediction.*

ABSTRAK

Cacat perangkat lunak merupakan kesalahan atau kegagalan pada perangkat lunak. Pemeriksaan cacat perangkat lunak secara manual hanya dapat menghasilkan 60% dari total cacat yang ada. Metode prediksi cacat menggunakan probabilitas dapat menemukan sampai 71% lebih baik dari metode yang digunakan oleh industri. Salah satu metode terbaik untuk prediksi cacat perangkat lunak adalah Logistic Regression. Logistic Regression merupakan pengklasifikasi linier yang telah terbukti menghasilkan klasifikasi yang powerfull dengan statistik probabilitas dan menangani masalah klasifikasi multi kelas. Kelemahan utama algoritma Logistic Regression adalah ketidakseimbangan kelas pada dataset berdimensi tinggi. Dataset sofware metric yang digunakan adalah dataset NASA MDP. Dataset ini umumnya bersifat tidak seimbang dan mengalami masalah pada redundant data. Penelitian ini mengusulkan metode greedy forward selection untuk mengatasi masalah redundant data dan metode bagging untuk mengatasi ketidakseimbangan kelas. Algoritma yang digunakan adalah Logistic Regression. Hasil percobaan pada penelitian ini mendapatkan nilai akurasi tertinggi sebesar 0,990 pada dataset PC2, naik 0,19% dibandingkan dengan metode logistic regression tanpa GFS dan bagging. Sedangkan nilai AUC tertinggi sebesar 0,995 pada PC2, naik 7,94% dibandingkan dengan metode logistic regression tanpa GFS dan bagging.

Kata Kunci: *Greedy Forward Selection, Bagging, Logistic Regression, dataset NASA MDP, Prediksi Cacat Perangkat Lunak.*

I. PENDAHULUAN

Perusahan atau organisasi memerlukan perangkat lunak untuk meningkatkan efisiensi dan efektifitas pada proses bisnisnya. Perangkat lunak yang digunakan umumnya dibuat dan disesuaikan dengan kebutuhan perusahaan. Pembuatan perangkat lunak yang berkualitas tinggi selama proses pengembangan dan perawatan memerlukan biaya yang sangat mahal (Wahono dan Herman, 2014), Pemeriksaan cacat software secara manual

hanya dapat menghasilkan 60% dari total cacat yang ada (Wahono dan Herman, 2014). Sehingga perlu dicarikan alternatif lain.

Menurut (Naik dan Tripathy, 2008), cacat perangkat lunak merupakan kesalahan atau kegagalan pada perangkat lunak. Perangkat lunak disebut gagal jika tidak dapat menjalankan fungsi sebagaimana yang telah dipersyaratkan sebelumnya. Indikator perangkat lunak yang berkualitas tinggi adalah tidak ditemukan cacat selama pengembangan dan pengujian (McDonald, Musson, dan Smith, 2008). Cacat perangkat lunak dapat juga dihasilkan oleh pengembang perangkat lunak sebagai akibat dari pengkodean program yang salah selama program dibuat. Pada beberapa kasus, perangkat lunak yang berkualitas tinggi banyak dinilai dari pengguna, sehingga perlu diukur dari sudut pandang pengguna.

Pengujian perangkat lunak biasanya memerlukan waktu sekitar 50% dari jadwal keseluruhan waktu pengerjaan (Fakhrahmad dan Sami, 2009). Hal ini menunjukan pentingnya proses pengujian untuk menghasilkan perangkat lunak yang berkualitas tinggi.

Untuk meningkatkan kualitas dan menjamin sebuah perangkat lunak berkualitas tinggi diperlukan program untuk prediksi cacat software (Chang, Mu, dan Zhang, 2011). Prediksi cacat perangkat lunak digunakan untuk memprediksi modul perangkat lunak yang rawan cacat dengan yang tidak rawan cacat sehingga dapat di rilis perangkat lunak yang berkualitas tinggi. Metode prediksi cacat menggunakan probabilitas dapat menemukan sampai 71% (Menzies, Greenwald, dan Frank, 2007), lebih baik dari metode yang digunakan oleh industri. Hasil tersebut membuktikan bahwa menggunakan probabilitas merupakan metode terbaik untuk menemukan cacat perangkat lunak.

Menurut (Song, Jia, Shepperd, Ying, dan Liu, 2011), penelitian prediksi cacat perangkat lunak berfokus pada 1) memperkirakan jumlah cacat software, 2) menemukan hubungan antar cacat, dan 3) mengklasifikasikan kerawanan cacat dari modul perangkat lunak yang dikelompokan menjadi rawan atau tidak rawan cacat. Klasifikasi merupakan pendekatan yang paling banyak digunakan untuk prediksi cacat perangkat lunak (Lessmann, Baesens, Mues, dan Pietsch, 2008). Dengan dideteksinya sejak awal cacat perangkat lunak dapat membantu pengembang menghasilkan perangkat lunak yang berkualitas.

Menurut (Hall, Beecham, Bowes, Gray, dan Counsell, 2011), metode algoritma terbaik untuk prediksi cacat perangkat lunak adalah Naive Bayes atau Logistic Regresion. Naive bayes merupakan pengklasifikasi probabilitas sederhana dengan penggunaan yang mudah dan nyaman karena tidak memerlukan perkiraan parameter yang rumit. Naive bayes juga efektif digunakan pada dataset yang besar dan dapat menyajikan hasil klasisfikasi kepada pengguna dengan mudah tanpa harus memiliki pengetahuan teknologi klasifikasi terlebih dahulu. Tetapi Naive Bayes berasumsi pada setiap atribut dataset sama penting dan tidak berhubungan satu sama lain. Logistic regresion merupakan model klasifikasi statistik probabilitas. Keuntungan Logistic Regression adalah algoritma ini telah dipelajari secar ekstensif (Hosmer, 2000). Sedangkan kelemahan Logistic Regression adalah rentan terhadap *underfitting* dan memiliki akurasi yang rendah (Harrington, 2012).

Logistic Regression merupakan pengklasifikasi linier yang telah terbukti menghasilkan klasifikasi yang powerfull dengan statistik probabilitas dan menangani masalah klasifikasi multi kelas (Canu, 2006). Masalah utama pada algoritma Logistic Regression adalah ketidakseimbangan kelas pada dataset berdimensi tinggi (Lin, Weng, dan Keerthi, 2007). Padalah untuk melakukan klasifikasi diperlukan data berupa dataset.

Data yang digunakan untuk mendeteksi modul perangkat lunak apakah rawan cacat atau tidak disebut *software matrics* (Chiş, 2008). Menurut (Khoshgoftaar, Gao, dan Seliya, 2010), penggunaan data mining sangat efektif untuk mengidentifikasi modul perangkat lunak dari potensi rawan cacat jika diterapkan pada *software metrics* yang dikumpulkan selama proses pengembangan perangkat lunak. *Software metric* yang digunakan selama pengembangan perangkat lunak disimpan dalam bentuk dataset.

Dataset NASA merupakan data metrik perangkat lunak yang paling banyak digunakan dalam pengembangan model prediksi cacat perangakt lunak, karena 62 penelitian dari 208 penelitian telah menggunakan dataset NASA (Hall et al., 2011). Dataset NASA tersedia secara online, sehingga para peniliti dapat menganalisa secara empiris untuk mengevaluasi penelitian sebelumnya. Penggunaan dataset NASA merupakan pilihan terbaik karena mudah diperoleh dan kinerja dari model yang digunakan menjadi mudah untuk dibandingkan dengan penelitian sebelumnya.

Software metrics berupa dataset yang umum digunakan untuk prediksi perangkat lunak bersifat tidak seimbang, karena umumnya cacat perangkat

lunak ditemukan dalam persentase yang kecil dari modul perangkat lunak (Seiffert, Khoshgoftaar, Van Hulse, dan Folleco, 2014). Ketidakseimbangan data pada dataset yang rawan cacat dan tidak rawan cacat akan mengakibatkan model prediksi cacat perangkat lunak tidak efektif karena hasil prediksi cenderung menghasilkan kelas mayoritas.

Selain masalah data tidak seimbang, prediksi cacat perangkat lunak mengalami masalah pada *redundant* data, korelasi, fitur yang tidak relevan, *missing sample*, dan masalah ini dapat membuat dataset yang tidak seimbang tersebut menjadi sulit untuk memprediksi cacat pada modul perangkat lunak (Laradji, Alshayeb, dan Ghouti, 2015). Seleksi fitur dapat menangani masalah *redudant* data dan fitur yang tidak relevan.

Greedy forward selection merupakan salah satu metode seleksi fitur yang sangat efisien, sederhana dan tidak seperti teknik seleksi fitur yang membutuhkan waktu lama dalam prosesnya. Dengan metode greedy forward selection, fitur yang dipilih hanya yang berkontribusi sehingga dapat meningkatkan performa klasifikasi (Laradji et al., 2015). Metode Seleksi fitur ini yang akan digunakan untuk yang menyelesaikan masalah *redudant* data.

Untuk masalah data tidak seimbang, salah satu cara untuk menangani data tidak seimbang adalah dengan berbagai teknik *resampling* dan sintesis data untuk memperbaiki kecondongan distribusi kelas data latih (Saifudin dan Wahono 2015). Penggunaan teknik *boosting* dan *bagging* dapat meningkatkan pengklasifikasi.

Pada penelitian ini diguankan seleksi fitur *greedy forward selection* untuk menangani masalah redundant data dan teknik *bagging* untuk menangani ketidakseimbangan kelas. Pengklasifikasi yang digunakan adalah algoritma *logistic regression*, dan menggunakan dataset NASA (*National Aeronautics and Space Administration*) MDP (*Metrics Data Program*).

II. LANDASAN TEORI

2.1 Tinjauan Studi

Tinjauan studi melingkupi penelitian terkait yang membahas ketidakseimbangan kelas dan redundant data pada Logistic Regression. Kajian ini perlu dilakukan agar dapat mengetahui metode apa saja yang digunakan, data seperti apa yang diproses, dan metode apa yang dihasilkan.

(Komarek dan Moore, 2005) melakukan penelitian untuk meningkatkan akurasi prediksi model Logistic Regression dengan menerapkan 3 metode yaitu 1) *Iteratively r-weighted least squares* (IRLS), 2) *Truncated Regularized* IRLS (TR-IRLS), dan 3) *Generic Likelihood Maximization*. *Logistic Regression* merupakan algoritma klasifikasi dalam data mining yang memiliki performa tinggi. Dalam beberapa implementasi data mining Logistic Regression dapat mengungguli algoritma lain seperti Naive Bayes, Support Vector Machine, dan K-Nearest Neighbor. Dalam penelitian ini menggunakan dataset yang dibagi dalam tiga kategori yaitu 1) pendeteksian link (citeseer, imdb), 2) dataset penelitian (ds2, ds1, ds1.100, ds1.10) dan 3) klasifikasi teks (modapte.sub). Hasil penelitian menunjukan matrik *Area Under Curve* (AUC) yaitu TR-IRLS 0.94 citeseer, 0.94 imdb, 0.72 ds2, 0.94 ds1, 0.91 ds1.00, dan 0.84 ds1.10.

(Lin et al., 2007) melakukan penelitian dengan mengusulkan metode *Trust Region Newton* (TRON) pada Logistic Regression. Penelitian ini menunjukan hasil konvergensi yang lebih cepat dari pada metode *quasi Newton*. Metode ini telah diakui mampu menangani dataset berdimensi tinggi namun tidak sepenuhnya digunakan. Penelitian ini merupakan turunan penggunaan model Newton yang menggunakan dataset berdimensi tinggi yaitu a9a dengan 32561 instance, real-sim dengan 72309 instance, news20 dengan 19996 instance, yahoo-japan dengan 176203 instance, rev1 dengan 677399 instance dan yahoo-korea dengan 460554 instance. Penerapan metode *trust Regiaon Newton* (TRON) memperlihatkan hasil komputasi 50% lebih cepat dibandingkan metode *limited memory quasi Newton*.

(Maalouf dan Trafalis, 2011) melakukan penelitian dengan menerapkan model *rare event re-wighted kernel logistic regression* (RE-WKLR). Penelitian mampu menangani dataset yang seimbang dan peristiwa langka. Dataset yang digunakan adalah UCI Machine Learning dan kejadian nyata angin tornado. Performa dari metode RE-WKLR menunjukan hasil klasifikasi yang lebih tinggi dari pada SVM. Secara keseluruhan hasil akurasi dari penelitian ini dengan menggunakan pengukuran komparasi paired t-test 0.017. Kesimpulan dari penelitian ini menunjukan bahwa algoritma RE-WKLR sangat mudah diimplementasikan dan kuat dalam menangani data seimbang dan peristiwa langka.

(Wahono dan Suryana, 2013) menerapkan optimasi metaheuristik untuk menemukan solusi optimal dalam seleksi fitur. Penelitian ini secara signifikan dapat mencari solusi berkualitas tinggi dengan jangka waktu yang wajar. Metode yang diusulkan dalam penelitian ini adalah optimasi

metaheuristik (*algoritma genetika* dan *particle swarm optimization* (PSO)) dan teknnik Bagging untuk meningkatkan kinerja prediksi cacat perangkat lunak. Bagging baik digunakan untuk model klasifikasi dan regresi. Bagging merupakan algoritma pembelajaran yang stabil pada dataset berdimensi tinggi dan atributnya masih mengandung noise. Dalam penelitian ini menggunakan 9 dataset NASA MDP dan 10 algoritma pengklasifikasi yang dikelompokan dalam 5 tipe yaitu klasifikasi statistic tradisional (Logistic Regression (LR), Linear Discriminant Analysis (LDA), dan Naive Bayes (NB)), Nearest Neghbors (k-Nearest Neighbor (k-NN) dan K*), Neural Network (Back propagation (BP), Support Vector Machine (SVM)), dan Decision Tree (C4.5, Classification *and* Regression Tree (CART), dan Random Forest (RF)). Hasil penelitian ini menunjukan bahwa metode yang diusulkan menunjukan peningkatan kinerja model prediksi cacat perangkat lunak. Kesimpulan penelitian ini adalah tidak ada perbedaan signifikan dalam penggunaan optimasi PSO dan algoritma genetik saat digunakan pada seleksi fitur.

2.2 Tinjauan Pustaka

2.2.1 Dataset

Dataset NASA yang telah ada untuk umum merupakan data metrik perangkat lunak yang sangat banyak digunakan untuk pengembangan model prediksi cacat perangkat lunak, karena 62 penelitian dari 208 penelitian telah menggunakan dataset NASA (Hall et al., 2011). Menurut (Wahono, 2015), sebanyak 64,79% penelitian prediksi cacat perangkat lunak menggunakan dataset publik. Hal ini dikarenakan mudah untuk didapatkan dan diuji kembali.

Dataset NASA MDP saat ini terdiri dari 13 dataset yang secara eksplisit ditujukan untuk penelitian metrik perangkat lunak (Gray et al., 2011). Metrik kode statis yang dicatat meliputi :

1. Ukuran *LOC-count*, merupakan jumlah baris kode program dan komentar.
2. Ukuran *Helstead*, merupakan perhitungan operan dan operator unik yang digunakan
3. Ukuran *McCabe*, merupakan kompleksitas dari *cyclometric*.

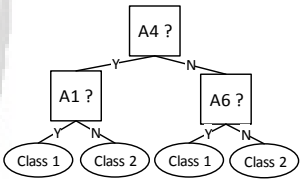
Dataset NASA yang asli masih mengandung *noise*, yaitu data yang tidak konsisten. *Noise* dapat disebut data yang salah (Saifudin dan Wahono, 2015). Oleh karena itu perlu dilakukan pemprosesan awal untuk membersihkan data dari noise.

2.2.2 Greedy Forward Selection

Menurut (Gorunescu, 2012) algoritma greedy merupakan algoritma yang menggunakan metaheuristic untuk menyelesaikan permasalahan dengan mengidentifikan lokal optimum dan pendekatannya mengarah pada global optimum. Algoritma greedy merupakan metode yang populer untuk memecahkan persoalan optimasi atau mencari solusi optimum.

Pada Gambar 1. Dijelaskan mengenai metode seleksi fitur dengan greedy untuk mencari atribut terbaik atau terburuk (Han, Kamber, dan Pei, 2012).

Menurut (Han et al., 2012) metode seleksi atribut greedy terdapat beberapa pendekatan seperti : greedy forward selection, greedy backward elimination dan kombinasi forward selection dan backward elimination. Greedy forward selection merupakan salah satu pendekatan dalam algoritma greedy pada metode seleksi fitur.

Forward Selection	Backward Selection	Decision Tree Induction
Atribut awal : {A1, A2, A3, A4, A5, A6} Pengurangan atribut awal : {} ⇨ {A1} ⇨ {A1, A4} ⇨ Atribut yang dikurangan : {A1, A2, A6}	Atribut awal : {A1, A2, A3, A4, A5, A6} ⇨ {A1 A3, A4, A5, A6} ⇨ {A1, A4, A5, A6} ⇨ Atribut yang dikurangan : {A1, A4, A6}	Atribut awal {A1, A2, A3, A4, A5, A6  Atribu yang dikurangi : {A1, A3, A6}

Gambar 1 Metode Heuristik (Greedy) untuk Seleksi Atribut Subset

2.2.3 Bagging

Bagging merupakan kependekan dari *bootstrap aggregating*, yaitu metode belajar yang sederhana dan efektif. Menurut (Laradji et al., 2015), bagging merupakan salah satu teknik ensemble yang berhasil menangani dataset yang tidak seimbang meskipun tidak secara khusus dirancang untuk masalah tersebut. (Wahono dan Suryana, 2013) menjelaskan, teknik bagging merupakan salah satu teknik penggabungan yang pada klasifikasi memisahkan data training ke dalam

beberapa data training baru dengan random sampling dan membangun model berbasis data training baru.

2.3 Logistic Regresion

Logistic Regression dipresentasikan untuk prediksi dengan menggunakan lebih dari satu Linear Regression (Karsmakers, Pelckmans, dan Suykens, 2007). Sedangkan menurut (Witten, Frank, dan Hall, 2011), model Logistic Regression merupakan probabilitas dari beberapa peristiwa, metode ini menggunakan fungsi linear untuk perhitungan prediksi pada beberapa variabel. Logistic Regression menampilkan persamaan linear yang saling berhubungan antara beberapa variabel acak, dimana variabel yang tergantung adalah variabel yang berkelanjutan. Metode ini merupakan pengembangan metode Linear Regression yang menggunakan lebih dari satu variabel. Dalam beberapa kasus, variabel yang tergantung menunjuk kepada dua nilai atau kategori tidak dapat menggunakan Linear Regression, tetapi dapat melakukan pendekatan yang serupa yang dapat juga Multiple Linear Logistic Regression.

Linier Regression sangat berguna untuk dataset linier dan aplikasi yang memadai (Komarek dan Moore, 2005). Namun untuk dataset yang kontinue (0,1) atau dengan dataset bernilai biner (0,1) tidak mungkin tepat menggunakan linear regression. Logistic Regression adalah model yang tepat digunakan untuk dataset bertipe seperti ini. Logistic regression menggunakan variabel yang sudah ditentukan atau variabel yang dikategorikan menjadi 2 variabel. Seperti pada prediksi sukses atau gagal, hidup atau mati, sakit atau tidak sakit, dan yang lainnya.

Notasi dalam Logistic Regression dimana variabel $X \in Re^{Nxd}$ adalah data matrik dimana N adalah jumlah sampel, d adalah jumlah dari parameter atau attribut, dan variabel y yang merupakan variabel hasil bersifat biner. Untuk setiap baris dalam data sampel $X_i \in R^d$ dimana X adalah vektor dari tiap baris data sampel, dimana i = 1 .. N dengan hasil dari $y_i = 1$ masuk dalam class positif dan sampel yang menghasilak $y_i = 0$ adalah masuk dalam class negatif. Tujuan dari klasifikasi seluruh data sampel X_i adalah nilai positif atau negatif. Data sampel dapat menggunakan metode percobaan Bernouli dengan harapan hasil nilai E (y_i) atau probabilitas p_i. Formula yang umum digunakan Logistic Regression untuk model masing-masing data sampel X_i menganut pada fungsi dari (Hosmer, 2000) sebagai barikut :

$$E[y_i|x_i|\beta] = P_i = \frac{e^{x_i\beta}}{1+e^{x_i\beta}} \quad \text{..... (2.1)}$$

Dimana β adalah parameter vektor dengan asumsi bahwa $x_{i0} = 1$ sehingga β_0 adalah variabel konstan dengan nilai 1.

III. METODE PENELITIAN

Penelitian merupakan kegiatan terencana yang bertujuan untuk memberikan kontribusi kepada pengetahuan (Dawson, 2009). Penelitian ini dilakukan dengan mengusulkan metode, menguji metode yang diusulkan menggunakan rapidminer 6. Penelitian ini menggunakan eksperimen dengan tahapan :

3.1 Pengumpulan Data

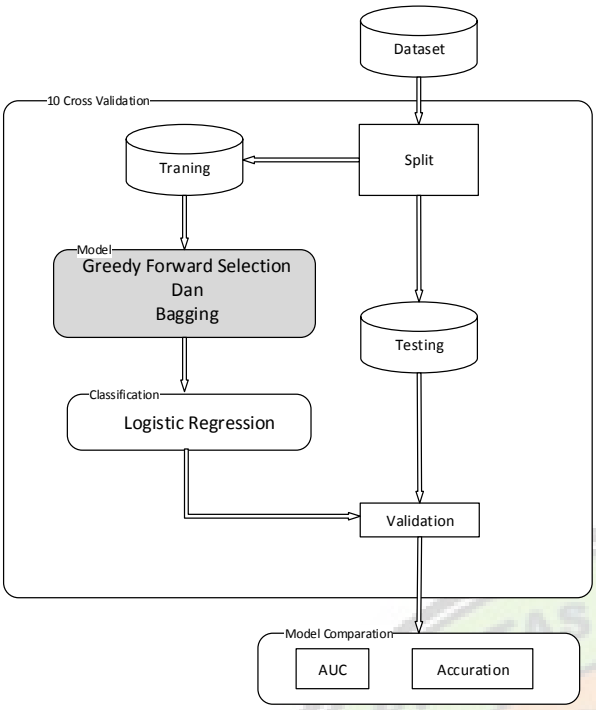
Penelitian ini menggunakan dataset NASA MDP yang diperoleh dari PROMISE (*Predictor Models in Software Engineering*). Dataset ini mudah diperoleh dan tersedia untuk umum dan banyak digunakan oleh peneliti dalam bidang rekayasa perangkat lunak (Hall et al., 2011). Pada penelitian ini menggunakan dataset yang NASA MDP dengan 9 dataset cacat perangkat lunak dari NASA MDP (Wahono dan Suryana, 2013). Dataset NASA MDP dan atributnya ditunjukan pada Tabel 1.

3.2 Pengolahan Data Awal

Penelitian ini menggunakan 9 dataset NASA MDP yang sudah dilakukan pembersihan sehingga data sudah tidak mengandung *noise*. Dataset ini dapat langsung diproses menggunakan metode yang diusulkan.

3.3 Metode Yang Diusulkan

Metode yang diusulkan adalah penerapan greedy forward selection dan bagging pada logistic regression pada prediksi cacat perangkat lunak. Gambar 3.1 menunjukan pembagian dataset dengan metode 10 cross validation yaitu data latih dan data uji, kemudian data latih diproses menggunakan greedy forward selection. Kemudian dilakukan proses keseimbangan kelas dengan metode bagging. Setelah itu dilakukan proses klasifikasi dengan logistic regression. Hasil metode ini diukur dengan akurasi dan Area Under the ROC Curve (AUC).



Gambar 3.1 Metode yang diusulkan

3.4 Eksperimen dan Pengujian Metode

Tahapan eksperimen dan pengujian metode pada proses penelitian ini adalah :

1. Menyiapkan dataset untuk eksperimen.
2. Mendesain arsitektur logistic regression
3. Melakukan training dan testing terhadap model logistic regression dan mencatat hasilnya.
4. Mendesain arsitektur logistic regression dan bagging.
5. Melakukan training dan testing terhadap model logistic regression dan bagging dan mencatat hasilnya.
6. Mendesain arsitektur logistic regression dengan greedy forward dan bagging.
7. Melakukan training dan testing terhadap model logistic regression dengan greedy forward selection dan bagging, kemudian mencatat hasilnya.
8. Membandingkan nilai Akurasi dan AUC dari metode yang diusulkan, membuat grafik dan menentukan persentase kenaikannya.

3.5 Evaluasi dan Validasi Hasil

Evaluasi dan validasi metode yang diusulkan dimulai dari pembagian dataset dengan metode 10-*fold cross validation* yaitu membagi dataset menjadi dua bagian, bagian pertama digunakan sebagai data latih dan bagian kedua digunakan sebagai data uji untuk memvalidasi model (Witten et al., 2011). Setelah itu diterapkan tahapan evaluasi menggunakan *Area Under Curve* (AUC) untuk mengukur hasil akurasi indikator dari performa model prediksi. Hasil akurasi dapat dilihat dengan dilakukan perbandingan klasifikasi menggunakan curva *Receiver Operating Characteristic* (ROC) dari hasil *confusion matrix*. ROC menghasilkan dua garis dengan bentuk *true positives* sebagai garis vertikal dan *false positive* sebagai garis horisontal (Vercellis, 2009). Kurva ROC adalah grafik antara sensitivitas (*true positive rate)* pada sumbu Y dengan sumbu X. Pengukuran dengan *confusion matrix* dapat dilihat pada tabel 1.

Tabel 1. *Confusion Matrix*

Class		**Actual**	
		TRUE	FALSE
Prediction	TRUE	True Positive (TP)	False Negative (FN)
	FALSE	False Positive (FP)	True Negative (TN)

Menurut (Zhang dan Wang, 2011) untuk mengukur akurasi keseluruhan dan akurasi kelas minoritas pada data tidak seimbang menggunakan metrik yang tepat adalah AUC (Area Under the ROC Curve), F-Measure, G-Mean.

Dalam pengklasifiksi keakuratan dari tes diagnostik menggunakan Area under Curve (AUC) dapat dijelaskan melalui Tabel 2 (Gorunescu, 2012).

Tabel 2. Nilai AUC, Keterangan dan Simbol

Nilai AUC	**Klasifikasi**	**Simbol**
0,90 – 1,00	*Excellent classification*	
080 – 0,90	*Good classification*	
0,70 – 0,80	*Fair Classification*	
0,60 – 0,70	*Poor classification*	
< 0,60	*Failure*	

Menurut (Gorunescu, 2012), rumus-rumus yang digunakan untuk perhitungannya adalah

$$\boldsymbol{Akurasi} = \frac{\boldsymbol{TP + TN}}{\boldsymbol{TP + TN + FP + FN}}$$

$$\boldsymbol{Sensitivitas = Recall = TP_{rate} = \frac{TN}{TP + FN}}$$

$$\boldsymbol{Spesificity = TN_{rate} = \frac{TN}{TN + FP}}$$

$$\boldsymbol{FPrate = NPV = \frac{FP}{FP + TN}}$$

$$\boldsymbol{Precision = PPV = \frac{TP}{TP + FP}}$$

$$\boldsymbol{F - Measure = \frac{2\,x\,recall\,x\,precision}{recall\,x\,precision}}$$

$$\boldsymbol{G - Mean = \sqrt{Sensivitas\,x\,Specificity}}$$

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 mana yang yang lebih baik secara rata-rata (Attenberg, 2013). Menganalisa nilai AUC merupakan cara paling tepat untuk memilih metode mana yang terbaik.

IV. PEMBAHASAN

Eksperimen dilakukan menggunakan laptop HP Pavilion G series dengan prosesor Intel(R) Core(TM) i3-2310 CPU @ 2,10 GHz, memori RAM 4 GB, dan sistem operasi windows 7 ultimate 32 bit. Sedangkan perangkat lunak untuk experimen menggunakan rapid miner 6.

Eksperimen dimulai dengan menyiapkan dataset NASA MDP. Tabel 3 menunjukan dataset yang digunakan yaitu yaitu CM1, KC1, KC3, MC2, MW1, PC1, PC2, PC3, PC3 beserta atributnya yaitu *LOC counts*, *Halstead*, *McCabe*, *dan Misc.*

Tabel 3. Dataset NASA MDP dan Atributnya

Code Attributes		NASA MDP Dataset								
		CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
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	√		√	√	√	√	√	√	√
Programming Language		*C*	*C++*	*Java*	*C*	*C*	*C*	*C*	*C*	*C*
Number of Code Attributes		*37*	*21*	*39*	*39*	*37*	*37*	*36*	*37*	*37*
Number of Modules		*344*	*2096*	*200*	*127*	*264*	*759*	*1585*	*1125*	*1399*
Number of fp Modules		*42*	*325*	*36*	*44*	*27*	*61*	*16*	*140*	*178*
Percentage of fp Modules		*12.21*	*15.51*	*18*	*34.65*	*10.23*	*8.04*	*1.01*	*12.44*	*12.72*

Selanjutnya, dataset yang tersedia diterapkan pada algoritmya Logistic Regression (LR). Tabel 4 ditampilkan rekap pengukuran akurasi model Logistic Regression (LR). Informasi yang disajikan antara lain akurasi, *sensitivity (recall)*, *spesificity*, *Positive Predictive Value* (PPV) atau *Precision*, *Negative Predictive Value* (NPV) atau FP_{rate} *F-Measure*, *G-Mean* dan AUC. Pada percobaan ini Logistic Regression menghasilkan performa *excellent* pada Akurasi di PC1, PC2, dan PC4. Sedangkan pada AUC, performa *excellent* pada CM1, KC1, dan PC2.

Tabel 4. Hasil Pengukuran LR

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	299	2	3	40	0,875	0,882	0,400	0,990	0,070	0,550	0,594	0,906
KC1	1757	48	14	277	0,861	0,864	0,774	0,992	0,048	0,817	0,818	0,908
KC3	158	4	6	32	0,810	0,832	0,400	0,963	0,158	0,540	0,577	0,837
MC2	79	14	4	30	0,732	0,725	0,778	0,952	0,118	0,750	0,751	0,804
MW1	231	6	6	21	0,898	0,917	0,500	0,975	0,222	0,647	0,677	0,847
PC1	691	9	7	52	0,922	0,930	0,563	0,990	0,119	0,701	0,723	0,906
PC2	1566	0	3	16	0,988	0,990	0,000	0,998	0,158	0,000	0,000	0,916
PC3	973	11	12	129	0,875	0,883	0,478	0,988	0,085	0,620	0,650	0,899
PC4	79	1195	99	26	0,911	0,752	0,923	0,444	0,792	0,829	0,834	0,894
	Rata - rata				0,875	0,864	0,535	0,921	0,197	0,606	0,625	0,880

Percobaan berikutnya dengan menerapkan Greedy Forward Selection (GFS) pada Logistic Regression (LR). Tabel 5 menunjukan hasil percobaannya. Informasi yang disajikan antara lain akurasi, *sensitivity (recall)*, *spesificity*, *Positive Predictive Value* (PPV) atau *Precision*, *Negative*

Predictive Value (NPV) atau FP_{rate} *F-Measure*, *G-Mean* dan AUC. Pada percobaan ini menghasilkan performa *excellent* pada Akurasi di MW1, PC2, dan PC4. Sedangkan pada AUC, performa *excellent* pada CM1, KC1, MW1, PC1, PC, dan PC3.

Tabel 5. Hasil Pengukuran LR + GFS

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	301	2	1	40	0,881	0,883	0,667	0,997	0,024	0,760	0,767	0,929
KC1	1755	45	16	280	0,859	0,862	0,738	0,991	0,054	0,795	0,798	0,904
KC3	162	6	2	30	0,840	0,844	0,750	0,988	0,063	0,794	0,795	0,891
MC2	81	14	2	30	0,748	0,730	0,875	0,976	0,063	0,796	0,799	0,834
MW1	237	6	0	21	0,920	0,919	1,000	1,000	0,000	0,958	0,958	0,959
PC1	687	6	1	55	0,925	0,926	0,857	0,999	0,018	0,890	0,891	0,954
PC2	1569	0	0	16	0,990	0,990	0,000	1,000	0,000	0,000	0,000	0,995
PC3	984	5	1	135	0,879	0,879	0,833	0,999	0,007	0,856	0,856	0,936
PC4	73	1200	105	21	0,910	0,777	0,920	0,410	0,833	0,842	0,845	0,881
	Rata - rata				**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,920**

Percobaan selanjutnya dengan menggunakan teknik Bagging (BG) pada Logistic Regression (LR). Tabel 6. menunjukan hasil percobaannya. Informasi yang disajikan antara lain akurasi, *sensitivity (recall), spesificity, Positive Predictive Value* (PPV) atau *Precision, Negative Predictive Value* (NPV) atau FP_{rate} *F-Measure, G-Mean* dan AUC. Pada percobaan ini menghasilkan performa *excellent* pada Akurasi di MW1, PC1, PC2, dan PC4. Sedangkan pada AUC, performa *excellent* pada CM1, KC1, PC1, PC2, PC3, dan PC4.

Tabel 6. Hasil Pengukuran LR + BG

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	299	2	3	40	0,875	0,882	0,400	0,990	0,070	0,550	0,594	0,906
KC1	1756	47	15	278	0,860	0,863	0,758	0,992	0,051	0,807	0,809	0,906
KC3	159	4	5	32	0,815	0,832	0,444	0,970	0,135	0,579	0,608	0,849
MC2	78	15	5	29	0,732	0,729	0,750	0,940	0,147	0,739	0,739	0,791
MW1	232	7	5	20	0,905	0,921	0,583	0,979	0,200	0,714	0,733	0,860
PC1	693	9	5	52	0,925	0,930	0,643	0,993	0,088	0,760	0,773	0,921
PC2	1566	0	3	16	0,988	0,990	0,000	0,998	0,158	0,000	0,000	0,916
PC3	974	11	11	129	0,876	0,883	0,500	0,989	0,079	0,638	0,664	0,902
PC4	84	1192	94	29	0,912	0,743	0,927	0,472	0,764	0,825	0,830	0,912
	Rata - rata				**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,885**

Percobaan berikutnya (*propose method*) dengan menggabungkan Greedy Forward Selection (GFS) dan Bagging (BG) pada Logistic Regression (LR). Tabel 7. menunjukan hasil percobaaannya. Informasi yang disajikan antara lain akurasi, *sensitivity (recall), spesificity, Positive Predictive Value* (PPV) atau *Precision, Negative Predictive Value* (NPV) atau FP_{rate} *F-Measure, G-Mean* dan AUC. Pada percobaan ini menghasilkan performa excellent pada Akurasi di MW1, PC1, PC2, dan PC4. Sedangkan pada AUC, performa excellent pada CM1, KC1, MW1, PC1, PC2, dan PC3.

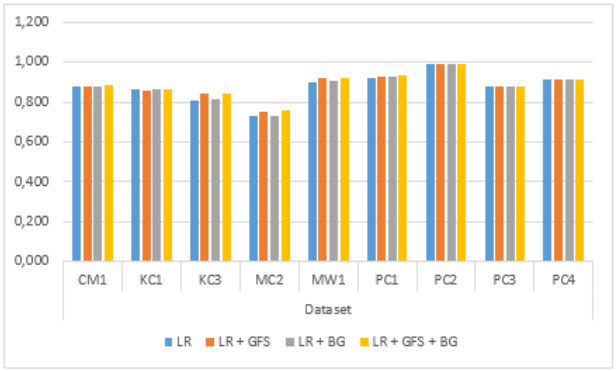
Tabel 7. Hasil Pengukuran LR + GFS + Bagging

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	302	2	0	40	0,884	0,883	1,000	1,000	0,000	0,938	0,940	0,942
KC1	1759	47	12	278	0,862	0,864	0,797	0,993	0,041	0,829	0,829	0,911
KC3	163	6	1	30	0,845	0,845	0,857	0,994	0,032	0,851	0,851	0,906
MC2	82	14	1	30	0,756	0,732	0,933	0,988	0,032	0,821	0,827	0,850
MW1	236	6	1	21	0,917	0,918	0,857	0,996	0,045	0,887	0,887	0,936
PC1	697	10	1	51	0,931	0,932	0,909	0,999	0,019	0,920	0,920	0,956
PC2	1569	0	0	16	0,990	0,990	0,000	1,000	0,000	0,000	0,000	0,995
PC3	984	5	1	135	0,879	0,879	0,833	0,999	0,007	0,856	0,856	0,936
PC4	69	1206	109	15	0,911	0,821	0,917	0,388	0,879	0,867	0,868	0,885
	Rata - rata				**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,875**	**0,924**

Terakhir, untuk melihat perbandingan hasil percobaan, Tabel 8 dan Gambar 2 menampilkan rekap pengukuran akurasi metode dan grafiknya metode Logistic Regression (LR), Logistic Regression (LR) dan Greedy Forward Selection (GFS), Logistic Regression (LR) dan Bagging (BG), serta Logistic Regression (LR), Greedy Forward Selection (GFS) dan Bagging (BG) pada prediksi cacat perangkat lunak. Hasilnya metode LR+GFS+BG *excellent* pada dataset MW1, PC1, PC2, dan PC4. Hasil *good* pada dataset CM1, KC1, KC3, dan PC3. Hasil *fair* pada dataset MC2.

Tabel 8. Hasil Pengukuran Akurasi LR, LR + GFS, LR + BG, dan LR + GFS + BG

Metode	Dataset								
	CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
LR	0,875	**0,861**	0,810	0,732	0,898	0,922	0,988	0,875	0,911
LR + GFS	**0,881**	0,859	**0,840**	**0,748**	**0,920**	0,925	**0,990**	**0,879**	0,910
LR + BG	0,875	0,860	0,815	0,732	0,905	0,925	0,988	0,876	**0,912**
LR + GFS + BG	**0,884**	**0,862**	**0,845**	**0,756**	**0,917**	**0,931**	**0,990**	**0,879**	**0,911**

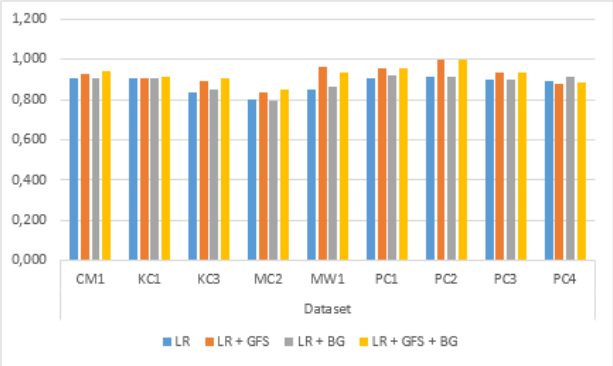


Gambar 2 Perbandingan Akurasi dari metode LR, LR+GFS, LR+BG, dan LR+GFS+BG

Sedangkan pada Tabel 9 dan Gambar 3 menunjukan hasil perbandingan nilai *Area Under Curve* (AUC) dan grafiknya metode Logistic Regression (LR), Logistic Regression (LR) dan Greedy Forward Selection (GFS), Logistic Regression (LR) dan Bagging (BG), serta Logistic Regression (LR), Greedy Forward Selection (GFS) dan Bagging (BG) pada prediski cacat perangkat lunak. Hasilnya metode LR+GFS+BG excellent pada CM1, KC1, KC3, MW1, PC1, PC2, dan PC3. Hasil good pada MC2, tetapi tidak terjadi peningkatan pada PC4.

Tabel 9 Hasil Pengukuran AUC LR, LR + GFS, LR + BG, dan LR + GFS + BG

Metode	Dataset								
	CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
LR	0,906	**0,908**	0,837	0,804	0,847	0,906	0,916	0,899	**0,894**
LR + GFS	**0,929**	0,904	**0,891**	**0,834**	**0,959**	**0,954**	**0,995**	**0,936**	0,881
LR + BG	0,906	0,906	0,849	0,791	0,860	0,921	0,916	0,902	**0,912**
LR + GFS + BG	**0,942**	**0,911**	**0,906**	**0,850**	**0,936**	**0,956**	**0,995**	**0,936**	0,885



Gambar 3 Perbandingan AUC dari metode LR, LR+GFS, LR+BG, dan LR+GFS+BG

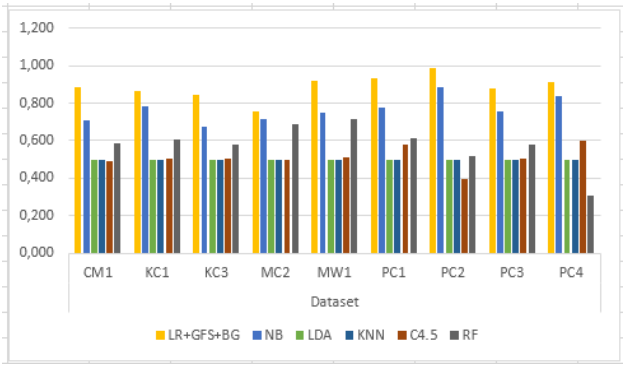
Berdasarkan hasil percobaan diatas maka dapat dilihat bahwa metode LR+GFS+BG mampu menangani ketidakseimbangan kelas dan *redundant* data pada logistic regresion dengan menghasilkan nilai akurasi dan AUC lebih tinggi dibandingkan dengan metode logistic regression yang tidak menggunakan greedy forward selection dan bagging.

Kami juga membandingkan pengklasifikasi yang lainnya yaitu Naive Bayes (NB), Linear Discriminant Analys (LDA), K-Nearest Neighboard (KNN), C4.5, dan Random Forest (RF). Hasil rekap nilai AUC ditampilkan pada Tabel 10 dan Gambar 4.

Tabel 10 Perbandingan AUC Model Prediksi Cacat Perangkat Lunak

Model	Dataset								
	CM1	KC1	KC3	MC2	MW1	PC1	PC2	PC3	PC4
LR+GFS+BG	**0,884**	**0,862**	**0,845**	**0,756**	**0,917**	**0,931**	**0,990**	**0,879**	**0,911**
NB	0,708	0,786	0,677	0,712	0,752	0,775	0,885	0,756	0,840
LDA	0,500	0,500	0,500	0,500	0,500	0,500	0,500	0,500	0,500
KNN	0,500	0,500	0,500	0,500	0,500	0,500	0,500	0,500	0,500
C4.5	0,493	0,506	0,506	0,499	0,512	0,579	0,398	0,508	0,602
RF	0,583	0,610	0,578	0,686	0,717	0,616	0,521	0,581	0,305

Pada Tabel 10 dan Gambar 4 terlihat, metode yang diusulkan (*proposed method*) LR+GFS+BG mendapatkan performa *excellent* pada PC1, PC2, dan PC4. Nilai AUC *good* pada MC1, KC1, KC3, dan PC3. Dan nilai AUC cukup pada MC2. Jika dibandingkan dengan pengklasifikasi yang lain metode LR+GFS+BG mengungguli secara signifikan.



Gambar 4 Perbandingan Nilai AUC Model Prediksi Cacat Perangkat Lunak

V. PENUTUP

Penelitan dengan menerapkan metode greedy forward selection dan teknik bagging dapat menangani masalah ketidakseimbangan kelas dan *redudant* data pada dataset NASA MDP untuk prediksi cacat perangkat lunak dengan algoritma logistic regression. Hasil percobaan pada penelitian ini mendapatkan nilai akurasi tertinggi sebesar 0,990 pada dataset PC2, naik 0,19% dibandingkan dengan metode logistic regression tanpa GFS dan bagging. Sedangkan nilai AUC tertinggi sebesar 0,995 pada PC2, naik 7,94% dibandingkan dengan metode logistic regression tanpa GFS dan bagging.

Jika dibandingan dengan pengklasifikasi yang lain seperti Naive Bayes (NB), Linear Discriminant Analys (LDA), K-Nearest Neighboard (KNN), C4.5, dan Random Forest (RF), metode yang diusulkan (LR+GFS+BG) naik 13,59% dari rata-rata hasil tertinggi pada 9 dataset.

Meskipun diketahui bahwa metode LR+GFS+BG memiliki akurasi dan nilai AUC yang tinggi, namun untuk penelitian selanjutnya hal-hal berikut dapat disarankan, antara lain, 1) menerapkan langsung pada industri pembuatan perangkat lunak, 2) menggunakan feature selection yang lain seperti *Genetic algorithm*, *Particle Swarm Optimation*, atau *ant colony*, 3) menggunakan algoritma lain seperti *Adabost* atau *Boosting* untuk mengetahui algoritma yang paling tepat untuk prediksi cacat perangkat lunak.

DAFTAR PUSTAKA

Attenberg, J. (2013). Active Learning For Imbalanced Problems. *Class Imbalance and Active Learning*, (iii), 101 – 151.

Canu, S. (2006). Kernel methods and the exponential family. *Neurocomputing*,

69(October 2005), 1–14. doi:10.1016/j.neucom.2005.12.009

Chang, R., Mu, X., & Zhang, L. (2011). Software defect prediction using Non-Negative Matrix Factorization. *Journal of Software*, *6*(11 SPEC. ISSUE), 2114–2120. doi:10.4304/jsw.6.11.2114-2120

Chiş, M. (2008). Evolutionary Decision Trees and Software Metrics for Module Defects Identification. *Program*, *2*(2), 25–29.

Dawson, C. W. (2009). *Projects in Computing and Information Systems*. *Sentimentaltoday.Net*.

Fakhrahmad, S. M., & Sami, a. (2009). Effective Estimation of Modules' Metrics in Software Defect Prediction. *World Congress on Engineering*, *I*, 206–211.

Gorunescu, F. (2012). *Data Mining : Concepts, Models and Techniques*. *Springer* (Vol. XXXIII). Sport Management Association of Australia and New Zealand. doi:10.1007/s13398-014-0173-7.2

Gray, D., Bowes, D., Davey, N., Christianson, B., Sun, Y., & Christianson, B. (2011). The misuse of the NASA Metrics Data Program data sets for automated software defect prediction. *IET Seminar Digest*, *2011*(1), 96–103. doi:10.1049/ic.2011.0012

Hall, T., Beecham, S., Bowes, D., Gray, D., & Counsell, S. (2011). A Systematic Review of Fault Prediction Performance in Software Engineering. *IEEE Transactions on Software Engineering*, *38*(6), 1276–1304. doi:10.1109/TSE.2011.103

Han, J., Kamber, M., & Pei, J. (2012). *Data Mining Concepts and Technique*. *Data Mining* (third.). USA: Morgan Kaufmann. doi:10.1007/s13398-014-0173-7.2

Harrington, P. (2012). *Machine Learning in Action*. doi:10.1007/978-0-387-77242-4

Hosmer, D. W. (2000). Applied Logistic Regression. New York: A Wiley-Interscience.

Karsmakers, P., Pelckmans, K., & Suykens, J. a. K. (2007). Multi-class kernel logistic regression: A fixed-size implementation. *IEEE International Conference on Neural Networks - Conference Proceedings*, *1*(3), 1756–1761. doi:10.1109/IJCNN.2007.4371223

Khoshgoftaar, T. M., Gao, K., & Seliya, N. (2010). Attribute Selection and Imbalanced Data: Problems in Software Defect Prediction. *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*, 137–144. doi:10.1109/ICTAI.2010.27

Komarek, P., & Moore, A. (2005). Making Logistic Regression A Core Data Mining Tool A Practical Investigation of Accuracy , Speed , and Simplicity. *Compute*, (1), 1–4.

Laradji, I. H. I., Alshayeb, M., & Ghouti, L. (2015). Software defect prediction using ensemble learning on selected features. *Information and Software Technology*, *58*, 388–402. doi:10.1016/j.infsof.2014.07.005

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*, *34*(4), 485–496. doi:10.1109/TSE.2008.35

Lin, C.-J., Weng, R. C., & Keerthi, S. S. (2007). Trust region Newton method for large-scale logistic regression, 561–568.

Maalouf, M., & Trafalis, T. B. (2011). Robust weighted kernel logistic regression in imbalanced and rare events data. *Computational Statistics and Data Analysis*, *55*(1), 168–183. doi:10.1016/j.csda.2010.06.014

McDonald, M., Musson, R., & Smith, R. (2008). *The Practival Guide to Defect Prevention*. (W. Press, Ed.)*Control*. Washington: Microsoft.

Menzies, T., Greenwald, J., & Frank, A. (2007). Data Mining Static Code Attribute to Learn Defect Predictors. *IEEE Transactions on*

Software Engineering, *33*(1), 2–14. doi:10.1109/TSE.2007.10

Naik, K., & Tripathy, P. (2008). *AND QUALITY Theory and Practice*. *Theory and Practice*.

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

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J., & Folleco, A. (2014). An empirical study of the classification performance of learners on imbalanced and noisy software quality data. *Information Sciences*, *259*, 571–595. doi:10.1016/j.ins.2010.12.016

Song, Q., Jia, Z., Shepperd, M., Ying, S., & Liu, J. (2011). A general software defect-proneness prediction framework. *IEEE Transactions on Software Engineering*, *37*(3), 356–370. doi:10.1109/TSE.2010.90

Vercellis, C. (2009). *Business Intelligence : Data Mining and Optimization for Decision Making*. *John Wiley & Sons* (Vol. 1). Milan: John Wiley & Sons, Inc. doi:10.1017/CBO9781107415324.004

Wahono, R. S. (2015). *Software Defect Prediction Framework Based On Hybrid Metaheuristic Optimization Methods*. Universiti Teknikal Malaysia Melaka.

Wahono, R. S., & Herman, N. S. (2014). Genetic feature selection for software defect prediction. *Advanced Science Letters*, *20*(1), 239–244. doi:10.1166/asl.2014.5283

Wahono, R. S., & Suryana, N. (2013). Combining particle swarm optimization based feature selection and bagging technique for software defect prediction. *International Journal of Software Engineering and Its Applications*, *7*(5), 153–166. doi:10.14257/ijseia.2013.7.5.16

Witten, I. H., Frank, E., & Hall, M. a. (2011). *Data Mining*. *Data Mining* (Vol. 277). doi:10.1002/1521-3773(20010316)40:6<9823::AID-ANIE9823>3.3.CO;2-C

Zhang, H., & Wang, Z. (2011). A normal distribution-based over-sampling approach to imbalanced data classification. *Advanced Data Mining and Applications*, 83–96. doi:10.1007/978-3-642-25853-4_7