Resampling Logistic Regression untuk Penanganan Ketidakseimbangan *Class* pada Prediksi Cacat *Software*

Harsih Rianto
Sekolah Tinggi Manajemen Informatika dan Komputer Nusa Mandiri
hrsanto@gmail.com

Romi Satria Wahono
Fakultas Ilmu Komputer, Uni versitas Dian Nuswantoro
romi@romisatriawahono.net

Abstract: *Software* yang berkualitas tinggi adalah *software* yang dapat membantu proses bisnis perusahaan dengan efektif, efesien dan tidak ditemukan cacat selama proses pengujian, pemeriksaan, dan implementasi. Perbaikan *software* setelah pengirimana dan implementasi, membutuhkan biaya jauh lebih mahal dari pada saat pengembangan. Biaya yang dibutuhkan untuk pengujian *software* menghabisakan lebih dari 50% dari biaya pengembangan. Dibutuhkan model pengujian cacat *software* untuk mengurangi biaya yang dikeluarkan. Saat ini belum ada model prediksi cacat *software* yang berlaku umum pada saat digunakan digunakan. Model Logistic Regression merupakan model paling efektif dan efesien dalam prediksi cacat *software*. Kelemahan dari Logistic Regression adalah rentan terhadap *underfitting* pada dataset yang kelasnya tidak seimbang, sehingga akan menghasilkan akurasi yang rendah. Dataset NASA MDP adalah dataset umum yang digunakan dalam prediksi cacat *software*. Salah satu karakter dari dataset prediksi cacat software, termasuk didalamnya dataset NASA MDP adalah memiliki ketidakseimbangan pada kelas. Untuk menangani masalah ketidakseimbangan kelas pada dataset cacat software pada penelitian ini diusulkan metode *resampling*. Eksperimen dilakukan untuk membandingkan hasil kinerja Logistic Regression sebelum dan setelah diterapkan metode *resampling*. Demikian juga dilakukan eksperimen untuk membandingkan metode yang diusulkan hasil pengklasifikasi lain seperti Naïve Bayes, Linear Descriminant Analysis, C4.5, Random Forest, Neural Network, k-Nearest Network. Hasil eksperimen menunjukkan bahwa tingkat akurasi Logistic Regression dengan *resampling* lebih tinggi dibandingkan dengan metode Logistric Regression yang tidak menggunakan *resampling*, demikian juga bila dibandingkan dengan pengkalisifkasi yang lain. Dari hasil eksperimen di atas dapat disimpulkan bahwa metode *resampling* terbukti efektif dalam menyelesaikan ketidakseimbangan kelas pada prediksi cacat *software* dengan algoritma Logistic Regression.

Keywords: Ketidakseimbangan Kelas, Logistic Regression, *Resampling*.

1 PENDAHULUAN

Software yang dikembangkan dan dibuat oleh para pengembang sebagian besar digunakan oleh perusahaan atau instansi pemerintahan. Pembuatan *software* tersebut bertujuan agar proses bisnis pada perusahaan atau instansi pemerintahan berjalan dengan efesien, efektif, cepat dan akurat. Pengembangan sebuah *software* yang berkualitas tinggi membutuhkan biaya yang sangat mahal (Chang, Mu, & Zhang, 2011; Czibula, Marian, & Czibula, 2014; Ma, Luo, Zeng, & Chen, 2012; Wahono, Suryana, & Ahmad, 2014). Untuk meningkatkan efesiensi dan jaminan kualitas yang tinggi dari sebuah s*oftware*, dalam pengujiannya diperlukan program yang mampu memprediksi cacat *software* (Chang et al., 2011). Prediksi cacat *Software* digunakan untuk mengidentifikasi modul yang rawan terhadap cacat pada pengembangan modul yang akan dirilis dan membantu memprediksi kesalahan pada modul tersebut.

Penelitian dalam bidang *Software Engineering* khususnya tentang prediksi cacat *software* telah menjadi topik penelitian yang sangat penting (Hall, Beecham, Bowes, Gray, & Counsell, 2012). Saat ini penelitian prediksi cacat *software* fokus pada 1) estimasi jumlah cacat pada *software*, 2) asosiasi cacat pada *software*, 3) klasifikasi pada cacat *software* terutama pada penentuan cacat dan non-cacat (Song, Jia, Shepperd, Ying, & Liu, 2011). Klasifikasi merupakan pendekatan yang poluler untuk memprediksi cacat *software* (Lessmann, Member, Baesens, Mues, & Pietsch, 2008). Para pengembang dapat menghindari hal-hal yang merugikan pengguna dan tim pengembang dari cacat *software* sedini mungkin dengan menggunakan prediksi cacat *software*.

Algoritma klasifikasi seperti C4.5, Decision Tree, Linear Regression, Logistic Regression (LR), Naïve Bayes (NB), Neural Network (NN), Random Fores (RF) dan Support Vector Machine (SVM) menjadi focus topik penelitian yang banyak dilakukan (Hall et al., 2012). Hasil komparasi algoritma klasifikasi diperoleh dua metode algoritma terbaik yaitu Naïve Bayes dan Logistic Regression (Hall et al., 2012). Naïve Bayes adalah model klasifikasi probabilitas sederhana. Penggunaan algoritma Naïve Bayes sangat mudah dan nyaman karena tidak memerlukan estimasi parameter yang rumit. Sehingga Naïve Bayes bisa digunakan pada dataset yang sangat besar. Selain pada dataset yang besar Naïve Bayes juga menyajikan hasil klasifikasi kepada pengguna dengan sangat mudah tanpa harus memiliki pengetahuan teknologi klasifikasi terlebih dahulu (X. Wu & Kumar, 2010). Namun Naïve Bayes berasumsi pada semua atribut dataset adalah sama penting dan tidak terkait satu sama lain, sedangkan pada kenyataannya sulit dipahami keterkaitan antar atribut (J. Wu & Cai, 2011). Logistic Regression adalah metode klasifikasi statistik probabilitas. Keuntungan Logistic Regression adalah algoritma ini telah dipelajar secara ekstensif (Hosmer et al., 2013; Hosmer & Lemeshow, 2000) disamping pengembangan terbaru tentang penerapan *truncate newton* (Lin et al., 2008). Kelemahan dari Logistic Regression adalah rentan terhadap *underfitting* dan memiliki akurasi yang rendah (Harrington, 2012).

Logistic Regression merupakan klasifikasi linier yang telah terbukti menghasilkan klasifikasi yang powerful dengan statistik probabilitas dan menangani masalah klasifikasi multi kelas (Canu & Smola, 2006; Karsmakers, Pelckmans, &

Copyright @ 2015 IlmuKomputer.Com

Suykens, 2007). Masalah besar yang dialami oleh algoritma Logistic Regression adalah ketidakseimbangan kelas (*class imbalance*) pada dataset *berdimensi tinggi* (Lin et al., 2008). Jika dilihat dari dataset yang digunakan untuk prediksi cacat *software*, secara umum menggunakan dataset NASA masih mengalami ketidakseimbangan (*imbalance*) kelas (Khoshgoftaar, Gao, Napolitano, & Wald, 2013). Jumlah data yang rawan cacat (*fault-prone*) lebih sedikit dari pada jumlah data yang tidak rawan cacat (*nonfault-prone*). Membangun model klasifikasi prediksi cacat *software* tanpa melakukan pengolahan data awal, tidak akan menghasilkan prediksi yang efektif, karena jika kelas data awal tidak seimbang (*imbalance*) maka hasil prediksi cenderung menghasilkan kelas mayoritas (Khoshgoftaar et al., 2013). Karena rawan cacat merupakan kelas minoritas dari prediksi cacat *software.* Kinerja model prediksi cacat *software* berkurang secara signifikan, dikarenakan dataset yang digunakan mengandung ketidakseimbangan kelas (*class imbalance*). Secara umum seleksi fitur (*feature selection*) digunakan dalam *machine learning* ketika melibatkan dataset berdimensi tinggi dan atribut yang masih mengandung *noise* (Wahono et al., 2014).

Untuk menangani dataset berdimensi tinggi selain seleksi fitur (*feature selection*), dapat juga menggunakan teknik *resampling*. Dengan meningkatkan kelas minoritas dapat meningkatkan kemampuan algoritma *mechine learning* menjadi lebih baik, karena bisa mengenali sampel kelas minoritas dari sampel mayoritas (Thanathamathee & Lursinsap, 2013). *Resampling* merupakan cara yang paling populer untuk mengatasi masalah ini. Terdapat tiga pendekatan dasar untuk mengatasi masalah ketidakseimbangan kelas, yaitu *oversampling* kelas minoritas, *undersampling* kelas mayoritas atau menggunakan metode *hybrid* yang menggunakan dasar dari kedua metode ini. *Resampling* juga sebagai sarana mengubah distribusai kelas minoritas sehingga tidak kurang terwakili ketika training data pada algorita *mechine learning*. Metode *resampling* sudah terkenal diterapkan untuk memecahakn masalah ketidak seimbangan kelas (*class imbalace*) (Thanathamathee & Lursinsap, 2013).

Oversampling adalah metode yang paling sederhana untuk menangani kelas minoritas dengan melakukan *random* kelas selama proses pengambilan sampel. Proses pengambilan sampel dengan teknik *oversampling* ini adalah dengan menduplikasi kelas positif dan dilakukan penyeimbangkan kelas secara acak (Ganganwar, 2012). Namun, karena metode ini menduplikasi kelas positif yang ada dikelas minoritas, kemungkinan terjadi *overfitting* pasti akan terjadi. *Undersampling* hampir sama dengan teknik *oversampling* dengan menghitung selisih kelas mayoritas dan kelas minoritas. Selanjutnya dilakuakn perulangan sebanyak selisih kelas mayoritas dengan kelas minoritas. Selama proses perulangan dilakukan penghapusan terhadap kelas mayoritas sehingga didapatkan jumlah yang sama dengan kelas minoritas.

Pada penelitian ini yang akan dilakukan adalah penerapan *resampling* untuk penyelesaian ketidakseimbagan kelas (*class imbalance*) pada Logistic Regression untuk prediksi cacat *software,* sehingga dapat menghasilkan kinerja yang baik pada dataset yang seimbang.

2 PENELITIAN TERKAIT

Penelitian tentang penanganan ketidakseimbangan kelas pada Logistic Regression telah banyak dilakukan dan telah dipublikasikan. Untuk melakukan penelitian ini perlu ada kajian terhadap penelitian yang terkait sebelumnya agar dapat mengetahui metode apa saja yang digunakan, data seperti apa yang diproses, dan model seperti yang dihasilkan.

Penelitian yang dilakukan oleh Komarek dan Moore (Komarek & Moore, 2005), melakukan penelitan untuk meningkatkan akurasi prediksi model Logistic Regression dengan mengimplementasikan 3 metode yaitu: 1) *Iteratively re-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 performance tinggi. Dalam beberapa implementasi data mining, Logistic Regression dapat mengunguli algoritma lain seperti Naïve Bayes, Support Vector Mechine (SVM), dan K-Nearest Neighbor (KNN). 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, 0,98 imdb, 0.72 ds2, 0.94 ds1, 0,91 ds1.100 dan 0,84 ds1.10.

Penelitian yang dilakukan oleh Lin, Weng dan Keerthi (Lin et al., 2008), menunjukan hasil konvergensi yang lebih cepat dari pada metode *quasi Newton*. Metode *Truncated Newton* merupakan metode yang telah diakui mampu menangani dataset berdimensi tinggi seperti penelitian (Komarek & Moore, 2005) namun metode ini tidak sepenuhnya digunakan. Penelitian ini merupakan turunan pengunanan 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, rcv1 dengan 677399 instance dan yahoo-korea dengan 460554 instance. Penerapan metode *Trust Region Newton* (TRON) memperlihatkan hasil komputasi 50% lebih cepat dibadingkan metode *limited memory quasi Newton* (LBFGS) penelitian Liu dan Nocedal 1989 dalam (Lin et al., 2008).

Penelitian yang dilakukan oleh Maalouf dan Trafalis (Maalouf & Trafalis, 2011), mampu menangani dataset yang seimbang dan peristiwa langka. Dalam penelitian ini diterapkan model *rare event re-weighted kernel* logistic regression (RE-WKLR). Dataset yang digunakan dalam penelitian ini adalah UCI Machine Learning dan kejadian nyata angin tornado. Performance 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 menujukan bawah algoritma RE-WKLR sangat mudah diimplementasikan dan kuat dalam menangani data seimbang dan peristiwa langka. Metode RE-WKLR sesuai untuk dataset yang skala kecil dan menengah.

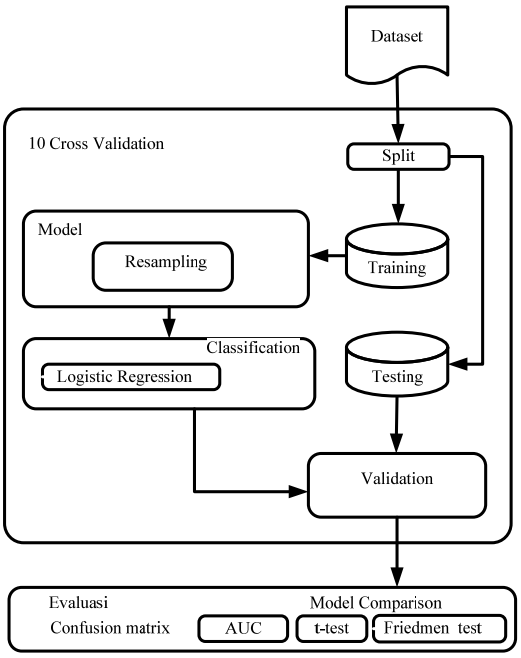
Penelitian yang dilakukan oleh Wahono, Suryana, dan Ahmad (Wahono et al., 2014) menerapan optimasi metaheuristik untuk menemukan solusi optimal dalam seleksi fitur (*feature selection*), secara signifikan mampu mencari solusi berkualitas tinggi dengan jangka waktu yang wajar. Metode yang diusulkan dalam penelitian ini adalah optimasi metaheuristik (algoritma genetika dan *particle swarm optimization* (PSO)) dan teknik Bagging untuk meningkatkan kinerja prediksi cacat *Software*. Bagging baik digunakan untuk model klasifikasi dan regressi. Bagging merupakan algoritma pembelajaran yang stabil pada dataset berdimensi tinggi dan attribut yang masih mengandung *noise* (Alpaydin, 2010). Dalam penelitian ini menggunakan 9 dataset NASA MDP dan 10 algoritma pengklasifikasi dan dikelompokan dalam 5 tipe, yaitu klasifikasi statistic tradisional (Logistic Regression (LR), Linear Discriminant Analysis (LDA), dan Naïve Bayes (NB)), *Nearest Neighbors* (k-Nearest Neighbor (k-NN) dan K*),

Copyright @ 2015 IlmuKomputer.Com

Neural Network (Back Propagation (BP), Support Vector Machine (SVM)), dan Decision Tree (C4.5, Classification and Regression Tree (CART), dan Random Forest (RF)). Hasilnya menunjukan bahwa metode yang diusulkan menunjukan peningkatan kenerja model prediksi cacat *Software*. Dari hasil Perbandingan model yang dilakukan, disimpulkan bahwa tidak ada perbedaan signifikan dalam penggunaan optimasi PSO dan algorima genetika saat digunakan pada seleksi fitur, untuk algoritma klasifikasi dalam prediksi cacat *Software*.

3 METODE YANG DIUSULKAN

Metode yang diusulkan pada penelitian ini yaitu untuk meningkatkan kinerja algoritma Logistic Regression dengan metode *resampling* untuk menangani ketidakseimbangan kelas (*class imbalance*) pada prediksi cacat *software*. Selanjutnya untuk validasi menggunakan 10-*fold cross validation.* Hasil pengukuran kinerja algoritma dengan menggunakan uji *t (t-test)* untuk mengetahui perbedaan kinerja model setelah dan sebelum diterapkan model resampling. Selanjutnya juga dilakukan pengujian kinerja model algoritam Logistic Regression dengan algoritma pengklasifikasi lain menggunakan uji Freidmen (*Freidmen test*). Model kerangka pemikiran metode yang diusulkan ditunjukan pada Gambar 1.



Gambar 1. Kerangka Pemikiran Model yang Diusulkan

Dalam penelitian ini dikumpulkan data sekunder yaitu NASA (*National Aeronautics and Space Administration)* MDP (*Metrics Data Program) repository* sebagai *software matrics* yang merupakan dataset yang sudah umum digunakan para peneliti dalam penelitian *Software Engineering* (Hall et al., 2012). Data NASA MDP dikhususkan untuk topik penelitian cacat *software* dan kegagalan *software*. Data NASA tidak hanya terdapat di *repository* MDP namun juga terdapat pada PROMISE. Kebanyakan peneliti menggunakan data NASA dari MDP karena sudah diperbaiki oleh Martin Shepperd (Liebchen & Shepperd, 2008) dengan menghilangkan data yang null atau data yang kosong. Dataset Nasa MDP *Repository* ditunjukan pada Tabel 1.

Tabel 1. Dataset NASA MDP Repository

Nama Atribut	NASA Dataset Repository									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LOC BLANK	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LOC CODE AND COMMENT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LOC COMMENTS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LOC EXECUTABLE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
LOC TOTAL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NUMBER OF LINES	✓			✓	✓	✓	✓	✓	✓	✓
HALSTEAD CONTENT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD DIFFICULTY	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD EFFORT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD ERROR EST	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD LENGTH	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD LEVEL	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD PROG TIME	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
HALSTEAD VOLUME	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NUM OPERANDS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NUM OPERATORS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NUM UNIQUE OPERANDS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
NUM UNIQUE OPERATORS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CYCLOMATIC COMPLEXITY	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CYCLOMATIC DENSITY	✓			✓	✓	✓	✓	✓	✓	✓
DESIGN COMPLEXITY	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ESSENTIAL COMPLEXITY	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
BRANCH COUNT	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
CALL PAIRS	✓			✓	✓	✓	✓	✓	✓	✓
CONDITION COUNT	✓			✓	✓	✓	✓	✓	✓	✓
DECISION COUNT	✓			✓	✓	✓	✓	✓	✓	✓
DECISION DENSITY	✓			✓		✓	✓	✓	✓	
DESIGN DENSITY	✓			✓	✓	✓	✓	✓	✓	✓
EDGE COUNT	✓			✓	✓	✓	✓	✓	✓	✓
ESSENTIAL DENSITY	✓			✓	✓	✓	✓	✓	✓	✓
GLOBAL DATA COMPLEXITY				✓	✓	✓				
GLOBAL DATA DENSITY				✓	✓	✓				
MAINTENANCE SEVERITY	✓			✓	✓	✓	✓	✓	✓	✓
MODIFIED CONDITION COUNT	✓			✓	✓	✓	✓	✓	✓	✓
MULTIPLE CONDITION COUNT	✓			✓	✓	✓	✓	✓	✓	✓
NODE COUNT	✓			✓	✓	✓	✓	✓	✓	✓
NORMALIZED CYLOMATIC COMPLEXITY	✓			✓	✓	✓	✓	✓	✓	✓
PARAMETER COUNT	✓			✓	✓	✓	✓	✓	✓	✓
PERCENT COMMENTS	✓			✓	✓	✓	✓	✓	✓	✓
PATHOLOGICAL COMPLEXITY										
DEFECTIVE	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Jumlah attribute	37	21	21	39	38	39	37	37	37	38
Jumlah modul	344	9593	2096	200	9277	127	759	1125	1399	17001
Jumlah modul cacat	42	1759	325	36	68	44	61	140	178	503
Presentase modul cacat	12%	18%	16%	18%	1%	35%	8%	12%	13%	3%
Jumlah modul tidak cacat	302	7834	1771	164	9209	83	698	985	1221	16498

Proses pengujian metode dimulai dari pembagian dataset dengan metode 10-*fold cross validation* yaitu membagi dataset menjadi dua segmen, segmen pertama digunakan sebagai data training dan segemen kedua digunakan sebagai data testing untuk mevalidasi model (Witten, Frank, & Hall, 2011). Selanjutnya diterapkan tahapan eveluasi menggunakan *Area Under Curve (AUC)* untuk mengukur hasil akurasi indikator dari performa model prediksi. Hasil akurasi dapat dilihat secara manual 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 positives* sebagai garis horisontal (Vercellis, 2011). Kurva ROC adalah grafik antara sensitivitas (*true positive rate*), pada sumbu Y dengan 1-spesifitas pada sumbu X (*false positive rate*), curva ROC ini menggambarkan seakan-akan ada terik-menarik antara sumbu Y dengan sumbu X (Dubey, Zhou, Wang, Thompson, & Ye, 2014)

Pengukuran akurasi dengan *confusion matrix* dapat dilihat pada Tabel 2.

Tabel 2. *Confusion Matrix*

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

Formulasi perhitungan yang dilakukan (Gorunescu, 2011) adalah sebagai berikut:

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

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

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

$$FP_{rate} = \frac{FP}{FP + TN}$$

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

$$F\ Measure = \frac{2RP}{R + P}$$

$$G - Mean = \sqrt{sensitivity * specificity}$$

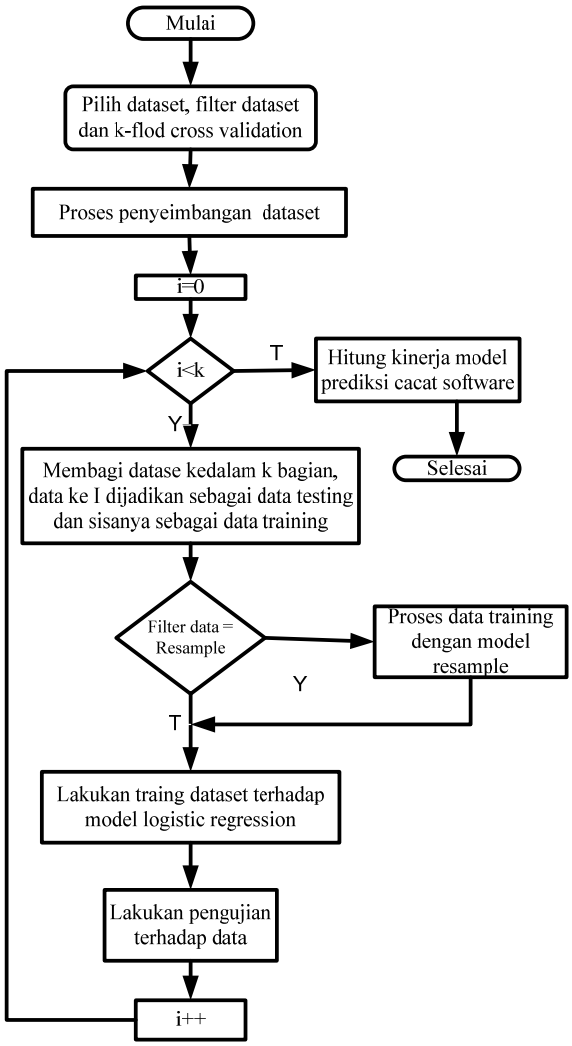
Dalam pengklasifikasi keakuratan dari tes diagnotik menggunakan *Area Under Curve(AUC)* (Gorunescu, 2011) dapat dijabarkan melalui Tabel 3.

Tabel 3. Nilai AUC, Keterangan dan Simbol

Nilai AUC	Klasifikasi	Simbol
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*	

Evaluasi dalam penelitian ini adalah menggunakan uji *t (t-test)*. Uji *t* adalah membandingkan hubungan antara dua variabel yaitu variabel *respon* dan variabel *predictor* (Larose, 2005). Uji *t* sample berpasangan (*paired-sample t-test*) dipergunakan untuk menguji perbandingan selisih dua rata-rata dari dua sample yang berpasangan dengan asumsi bahwa data terdistribusi dengan normal. Selanjutnya untuk mengevaluasi metode Logistic Regression dengan pengklasifikasi lain menggunakan uji Friedman (*Friedman test*). Uji Friedman di usulkan oleh Demsar untuk membandingkan model klasifikasi (Demšar, 2006). Uji Friedman (*Friedman test*) merupakan uji statistik non parametrik, yang juga disebut dengan Anova dua arah berdasarkan peringkat (*two-way anova by ranks*). Uji Friedman (*Friedman test*) berdasarkan peringkat kinerja dari perkiraan kinerja aktual, sehingga lebih tahan terhadap outlier. Semua model klasifikasi akan diperingkat berdasarkan performen terhdap dataset dan peringkat rata-rata dari model klasifikasi yang dibandingkan.

Flowchart metode yang diusulkan dapat dilihat pada Gambar 2, dimulai dengan memilih dataset yang akan diuji, filter dataset dan jumlah *cross validation* yang diinginkan. Selanjutnya dataset diproses kedalam model sebanyak jumlah *cross validation* sampai mendapakan hasil kinerja prediksi cacat *software*. Kemudian dilakukan evaluasi terhadap akurasi, *sensitivity*, *spesificity, FP_{rate}, Precision, F-Measure,* dan *G-Mean* dari hasil *confusion matrix.* Setelah mendapatkan hasil akurasi dan AUC kemudian dilakukan uji *t* (*t-test*) untuk mengetahui kinerja model setelah dan sebelum diterapakan *resampling.* Untuk mengetahui hasil kinerja model dengan pengklasifikasi lain dilakukan uji Friedman (*Friedman test).*



Gambar 2. Flowchart Metode yang Diusulkan

4 HASIL EKSPERIMEN

Eksperimen yang dilakukan dalam penelitian ini menggunakan sebuah platform komputer berbasis Intel Core i3-3217U @1.80GHz (4 CPUs), RAM 2GB, dan sistem operasi Microsoft Windows 7 Ultimate 32-bit. Sedangkan lingkungan pengembangan aplikasi menggunakan bahasa pemrograman Java Netbeans IDE 8.0.1 dan library Weka 3.6, untuk analisa hasil eksperimen menggunakan aplikasi Microsoft Excel 2007 dengan plugin XLSTAT.

Dalam eksperimen yang dilakukan dengan menggunakan 10 dataset NASA MDP (CM1, JM1, KC1, KC3, MC1, MC2, PC1, PC3, PC4, dan PC5). Metode yang diuji adalah metode pengklasifikasi Logistic Regression (LR), hasil eksperimen disajikan pada Tabel 4, informasi yang disajikan adalah akurasi, *sensitivty* (*recall*), *spesificity, Positive Predictive Value* (PPV) atau *Precision*, *Negative predictive Value* (NPV) atau FP_{rate}, *F-Measure*, *G-Mean* dan AUC. Hasil eksperimen pada Tabel 4, menunjukan rata-rata akurasi pada 10 dataset adalah 88.35% dan rata-rata AUC sebesar 0.818.

Copyright @ 2015 IlmuKomputer.Com

Tabel 4. Hasil Eksperimen Logistic Regression

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	9	19	33	283	84.88%	21.43%	93.71%	32.14%	89.56%	0.256	0.448	0.728
JM1	180	137	1579	7697	82.11%	10.23%	98.25%	56.78%	82.98%	0.173	0.317	0.705
KC1	70	44	255	1727	85.74%	21.54%	97.52%	61.40%	87.13%	0.319	0.458	0.800
KC3	12	17	24	147	79.50%	33.33%	89.63%	41.38%	85.97%	0.369	0.547	0.708
MC1	19	14	49	9195	99.32%	27.94%	99.85%	57.58%	99.47%	0.376	0.528	0.875
MC2	36	10	9	72	85.04%	80.00%	87.71%	78.26%	88.89%	0.791	0.038	0.863
PC1	11	13	50	658	91.70%	18.03%	98.14%	45.83%	93.20%	0.259	0.421	0.828
PC3	28	34	112	951	87.02%	20.00%	96.55%	45.16%	89.46%	0.277	0.439	0.819
PC4	86	34	92	1187	90.99%	48.32%	97.22%	71.67%	92.81%	0.577	0.685	0.907
PC5	147	108	356	16390	97.27%	29.23%	99.35%	57.65%	97.87%	0.388	0.539	0.955

Sedangan pada Tabel 5 ditunjukan hasil eksperimen metode Logistic Regression dengan *Resampling* untuk 10 dataset NASA MDP. Hasil eksperimen disajikan adalah akurasi, *sensitivty* (*recall*), *spesificity, Positive Predictive Value* (PPV) atau *Precision*, *Negative predictive Value* (NPV) atau FP_{rate}, *F-Measure*, *G-Mean* dan AUC. Hasil eksperimen pada Tabel 5, menunjukan rata-rata akurasi pada 10 dataset adalah 90.83% dan rata-rata AUC sebesar 0.856.

Tabel 5. Hasil Pengukuran LR dan Resampling

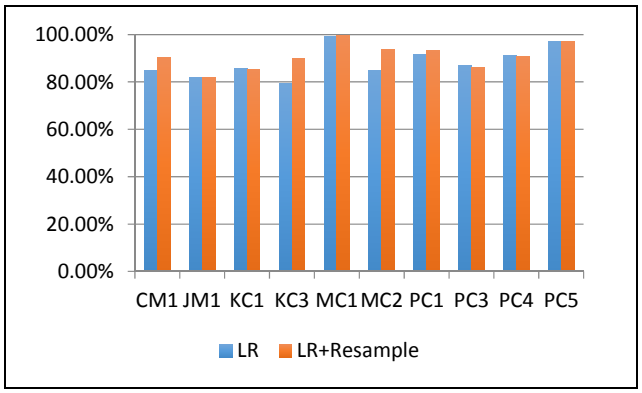
Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	18	10	23	293	90.40%	43.90%	96.70%	64.29%	92.72%	0.522	0.652	0.816
JM1	166	136	1600	7691	81.90%	9.40%	98.26%	54.97%	82.78%	0.161	0.304	0.708
KC1	82	58	253	1703	85.16%	24.48%	96.71%	58.57%	87.07%	0.345	0.487	0.791
KC3	32	15	5	148	90.00%	86.49%	90.80%	68.09%	96.73%	0.762	0.886	0.875
MC1	20	5	43	9209	99.48%	31.75%	99.95%	80.00%	99.54%	0.455	0.563	0.895
MC2	31	12	36	680	93.68%	46.27%	98.27%	72.09%	94.97%	0.564	0.674	0.900
PC1	17	12	38	692	93.41%	30.91%	98.30%	58.62%	94.80%	0.405	0.551	0.854
PC3	29	38	116	942	86.31%	20.00%	96.12%	43.28%	89.04%	0.274	0.438	0.846
PC4	112	43	88	1156	90.64%	56.00%	96.41%	72.26%	92.93%	0.631	0.735	0.926
PC5	143	103	355	16400	97.31%	28.72%	99.38%	58.13%	97.88%	0.384	0.534	0.951

Pada Tabel 6 disajikan rekap pengukuran akurasi model Logistic Regression (LR) dan Logistic Regression dengan penerapan *Resample* (LR+*Resample*).

Tabel 6. Rekap Pengukuran Akurasi LR dan LR+*Resample* pada Prediksi Cacat *Software*

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	84.88%	82.11%	85.74%	79.50%	99.32%	85.04%	91.70%	87.02%	90.99%	97.27%
LR+Resample	90.40%	81.90%	85.16%	90.00%	99.48%	93.68%	93.41%	86.31%	90.64%	97.31%

Dapat dilihat bawah terdapat peningkatan hasil kinerja model dengan melakukan penanganan ketidakseimbangan kelas (*class imbalance*) pada dataset NASA MDP. Perbedaan kinerja yang dihasilkan memang tidak cukup signifikan, yang dapat dilihat pada Gambar 3. Peningkatan kinerja hanya terjadi pada dataset CM1, KC3, dan MC2.



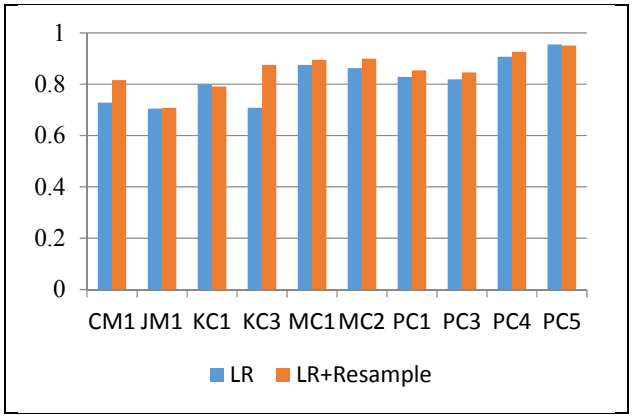
Gambar 3. Grafik Rekap Pengukuran Akurasi Pada Prediksi Cacat Software

Hasil perbandingan *Area Under Curve (*AUC) Logistic Regression (LR) dan Logistic Regression dengan penerapan *Resample* (LR+Resample) disajikan dalam Tabel 7. Dapat dilihat dari Gambar 4 grafik pengukuran AUC mengalami peningkatan kenerja setelah diterapkan metode resample pada dataset yang mengalami ketidakseimbangan kelas dengan peningkatan kinerja pada dataset CM1 dan KC3.

Tabel 7. Rekap Pengukuran AUC Model Prediksi Cacat *Software*

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	0.728	0.705	0.8	0.708	0.875	0.863	0.828	0.819	0.907	0.955
LR+Resample	0.816	0.708	0.791	0.875	0.895	0.9	0.854	0.846	0.926	0.951

Pada penelitian ini dilakukan pengujian hipotesis degan uji *paired sample t-test* untuk Logistic Legression (LR) dengan Logistic Regression dan Resample (LR+*Resample*) dan uji Freidmen (*Friedman test*) untuk membandingkan dengan pengklasifikasi lain. Uji *t (t-test)* adalah hubungan antara variabel respon dengan variable prediktor (Larose, 2005). Hipotesis nol (H_0) menyatakan bahwa tidak ada perbedaan hasil eksperimen antara motode LR dan LR+*Resample*, sedangkan hipotesis satu (H_1) menyatakan bahwa ada perbedaan hasil eksperiment antara LR dan LR+*Resample*.



Gambar 4 Grafik Rekap Pengukuran AUC LR dan LR+Resample pada Prediksi Cacat *Software*

Pada uji *t* sampel berpasangan (*paired-sample t-test)* untuk variabel akurasi LR dan variabel LR+*Resample* dapat dilihat pada Tabel 8.

Tabel 8. *Paired sample t-test* Akurasi LR dan LR+*Resample*

	LR	*LR+Resample*
Mean	88.357	90.829
Variance	40.77077889	29.42509889
Observations	10	10
Pearson Correlation	0.759543832	
Hypothesized Mean Difference	95	
df	9	
t Stat	-73.5139356	
P(T<=t) one-tail	4.03234E-14	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	8.06469E-14	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 8 dapat diambil kesimpulan hipotesis berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 73.513956, dan nilai *t* tabel yang diwakili oleh *t* Critical two-tail sebesar 2.262157158 maka dapat dipastikan nilai *t* hitung > *t* tabel yang artinya H_0 gagal diterima dan H_1 diterima, artinya ada perbedaan antara hasil akurasi LR dan

Copyright @ 2015 IlmuKomputer.Com

LR+*Resample*, sedangkan diketahui nilai probabilitas sebesar 8.06469E-14, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya H_0 gagal diterima dan H_1 diterima, artinya terdapat perbedaan yang signifikan dari rata-rata akurasi LR dan LR+*Resample,* hasil akurasi menunjukan LR+*Resample* lebih tinggi dibandingkan dengan LR.

Selanjutnya uji *t* sampel berpasangan (*paired-sample t-test)* untuk hasil AUC dari LR dan variabel LR+*Resample* dapat dilihat pada Tabel 9. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 5669.85953, dan nilai *t* tabel yang diwakili oleh *t* Critical two-tail sebesar 2.262157158 maka dapat dipastikan nilai *t* hitung > *t* tabel yang artinya H_0 gagal diterima dan H_1 diterima, artinya ada perbedaan antara hasil AUC LR dan LR+*Resample*, sedangkan diketahui nilai probabilitas sebesar 8.40652E-31, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya H_0 gagal diterima dan H_1 diterima, artinya terdapat perbedaan yang signifikan dari rata-rata AUC LR dan LR+*Resample,* hasil AUC menunjukan LR+*Resample* lebih tinggi dibandingkan dengan LR.

Tabel 9. Paired sample t-test Akurasi LR dan LR+Resample

	LR	*LR+Resample*
Mean	0.8188	0.8562
Variance	0.007261289	0.005063956
Observations	10	10
Pearson Correlation	0.784614645	
Hypothesized Mean Difference	95	
df	9	
t Stat	-5669.85953	
P(T<=t) one-tail	4.20426E-31	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	8.40852E-31	
t Critical two-tail	2.262157158	

Selanjutnya untuk mengetahui metode yang diusulkan dapat menangani ketidakseimbangan kelas (*class imbalance*) pada dataset NASA maka dibanginkan dengan algoritma pengklasifikasi lain yaitu: Naïve Bayes (NB), Linear Discriminant Analysis (LDA), k-nearest neighbor (k-NN), Decision Tree (C.45), Random Forest (RF), Support Vector Machine (SVM) dan neares neighbor (k*). Eksperimen untuk pengklasifikasi lain menggunakan *software* Rapid Miner versi 5.3. Rekap hasil akurasi dari eksperimen yang dilakukan pada LR, LR+Resample dan pengklasidikasi lain dapat dilihat pada Tabel 10.

Tabel 10. Rekap Akurasi dengan Pengklasifikasi Lain

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	84.88%	82.11%	85.74%	79.50%	99.32%	85.04%	91.70%	87.02%	90.99%	97.27%
LR+R	90.40%	81.90%	85.16%	90.00%	99.48%	93.68%	93.41%	86.31%	90.64%	97.31%
NB	82.56%	81.31%	82.25%	79.00%	93.54%	72.44%	90.93%	33.51%	87.42%	96.63%
LDA	43.02%	81.90%	85.73%	57.00%	97.75%	43.31%	81.03%	65.16%	85.70%	96.58%
KNN	81.69%	70.22%	80.44%	71.50%	99.16%	62.99%	87.35%	80.98%	81.63%	96.82%
C.45	79.07%	81.66%	84.49%	81.50%	99.40%	61.42%	91.83%	87.29%	87.28%	97.11%
SVM	87.50%	80.91%	85.35%	82.00%	99.27%	70.08%	91.83%	87.56%	89.42%	97.30%
RF	88.05%	81.67%	84.49%	82.00%	99.29%	66.14%	91.87%	87.56%	87.28%	97.10%
K*	87.79%	81.66%	84.49%	82.00%	99.27%	65.35%	91.96%	87.56%	12.72%	97.04%

Dari Tabel 10 rekap akurasi dipergunakan untuk uji Friedman (*Friedmant test*) yang menguji hipotesis nol (H_0) menyatakan bahwa tidak ada perbedaan hasil eksperimen antara motode Logitic Regression (LR), Logistic Regression menggunakan *Resample* (LR+R*),* Naïve Bayes (NB), Linear Discriminant Analysis (LDA), k-nearest neighbor (k-NN), Decision Tree (C.45), Random Forest (RF), Support Vector Machine (SVM) dan neares neighbor (k*). Sedangkan hipotesis satu (H_1) menyatakan bahwa ada perbedaan hasil eksperiment antara LR, LR+R, NB, LDA, KNN, C.45, RF, SVM dan k*. Uji *Friedman* dilakukan menggunakan aplikasi XLSTAT untuk hasil akurasi dari semua pengklasifikasi. Tabel 11 merupakan hasil dari uji *Friedman* untuk *pairwise differences*

Dari hasil uji Friedman model klasifikasi terbaik pada semua dataset dicatak tebal. Dalam uji Friedman didapatkan hasil signifikansi statistik pengujian P-value seperti telihat pada Tabel 11. Berdasarkan *p-value* dapat menjawab hipotesa H_0 diterima jika *p-value* signifikan dan menolak H_1 ketika p-value kurang signifikan. Dalam penelitian ini akan diatur seberapa signifikan statistik dengan symbol α dengan nilai 0.05. Sehingga dapat dirumuskan jika p-value = α menerima H_0 dan menolak H_1.

Tabel 11. Hasil Akurasi Uji Friedman untuk Pairwise Differences

	LR	LR+R	NB	LDA	KNN	C.45	SVM	RF	K*
LR	0	-1.300	4.250	5.200	5.500	2.450	0.700	1.050	1.900
LR+R	1.300	0	5.550	6.500	6.800	3.750	2.000	2.350	3.200
NB	-4.250	-5.550	0	0.950	1.250	-1.800	-3.550	-3.200	-2.350
LDA	-5.200	-6.500	-0.950	0	0.300	-2.750	-4.500	-4.150	-3.300
KNN	-5.500	-6.800	-1.250	-0.300	0	-3.050	-4.800	-4.450	-3.600
C.45	-2.450	-3.750	1.800	2.750	3.050	0	-1.750	-1.400	-0.550
SVM	-0.700	-2.000	3.550	4.500	4.800	1.750	0	0.350	1.200
RF	-1.050	-2.350	3.200	4.150	4.450	1.400	-0.350	0	0.850
K*	-1.900	-3.200	2.350	3.300	3.600	0.550	-1.200	-0.850	0

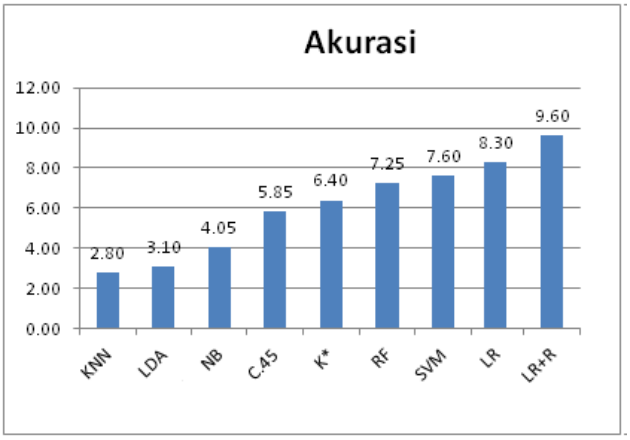
Untuk uji Friedman ini, kita mengatur tingkat signifikansi statistik (α) menjadi 0,05. Ini berarti bahwa ada perbedaan yang signifikan secara statistik jika P-value <0,05. Dari hasil percobaan, P-value adalah 0,0001, ini lebih rendah dari tingkat signifikansi α = 0,05, dengan demikian kita harus menolak hipotesis nol, dan itu berarti bahwa ada perbedaan yang signifikan secara statistik. Selanjutnya dilakukan uji Friedman dengan *Nemenyi post hoc tes* untuk mendeteksi pengklasifikasi tertentu berbeda secara signifikan. Tabel 12 memperlihatakan hasil akurasi uji Friedman untuk *p-value.*

Tabel 12. Hasil Akurasi Uji Friedman untuk P-values

	LR	LR+R	NB	LDA	KNN	C.45	SVM	RF	K*
LR	1	0.999	0.134	**0.020**	**0.010**	0.860	1.000	1.000	0.972
LR+R	0.999	1	**0.008**	**0.001**	**0.000**	0.288	0.960	0.889	0.536
NB	0.134	**0.008**	1	1.000	0.999	0.981	0.371	0.536	0.889
LDA	**0.020**	**0.001**	1.000	1	1.000	0.747	0.086	0.159	0.487
KNN	**0.010**	**0.000**	0.999	1.000	1	0.609	**0.047**	0.094	0.349
C.45	0.860	0.288	0.981	0.747	0.609	1	0.985	0.997	1.000
SVM	1.000	0.960	0.371	0.086	**0.047**	0.985	1	1.000	0.999
RF	1.000	0.889	0.536	0.159	0.094	0.997	1.000	1	1.000
K*	0.972	0.536	0.889	0.487	0.349	1.000	0.999	1.000	1

Dapat dilihat pada Gambar 5, Logistic Regression dengan *Resampling* (LR+R*)* menghasilkan akurasi tinggi berdasarkan uji *Freidmen.* Berdasarkan Tabel 12. dapat ditunjukan bahwa pengklasifiasi Naïve Bayes (NN) danLogistic Regression (LR) menghasilkan akurasi yang tinggi seperti penelitian oleh (Hall et al., 2012; Wahono et al., 2011). Seperti yang dilakukan oleh (Saifudin, 2014) yang menggunakan dataset NASA, hasil akurasi NB dan LR menunjukan hasil yang signifikan.

Copyright @ 2015 IlmuKomputer.Com



Gambar 5. Grafik Mean Akurasi Uji Friedman

Pada uji Friedman selanjutnya adalah menguji hasil AUC dari eksperimen yang sudah dilakukan. Tabel 13 menujukan rekap hasil eksperimen AUC. Dari Tabel 13 rekap AUC dipergunakan untuk uji Friedman (*Friedman test*) yang menguji hipotesis nol (H_0) menyatakan bahwa tidak ada perbedaan hasil eksperimen antara motode LR, LR+R, NB, LDA, k-NN, C.45, RF, SVM dan k*. Sedangkan hipotesis satu (H_1) menyatakan bahwa ada perbedaan hasil eksperiment antara LR, LR+R, NB, LDA, KNN, C.45, RF, SVM dan k*. Uji *Friedman* dilakukan menggunakan aplikasi XLSTAT untuk hasil akurasi dari semua pengklasifikasi. Tabel 11 merupakan hasil dari uji *Friedman* untuk *pairwise differences*

Tabel 13. Perbandingan AUC Model Prediksi Cacat Software

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	0.728	0.705	0.800	0.708	0.875	0.863	0.828	0.819	0.907	**0.955**
LR+R	0.816	0.708	0.791	0.875	0.895	**0.900**	0.854	0.846	**0.926**	0.951
NB	0.780	0.683	0.786	0.677	**0.916**	0.712	0.775	0.756	0.840	0.940
LDA	0.500	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	0.500
C.45	0.500	0.500	0.542	0.541	0.836	0.489	0.609	0.696	0.723	0.500
SVM	0.736	0.614	0.731	0.595	0.509	0.716	0.810	0.732	0.905	0.784
RF	0.518	0.000	0.549	0.599	0.639	0.627	0.523	0.509	0.150	0.676
K*	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500

Dari hasil AUC uji Friedman model klasifikasi terbaik pada semua dataset dicetak tebal. Dalam uji Friedman didapatkan hasil signifikansi statistik pengujian P-value seperti telihat pada Tabel 14. Berdasarkan *p-value* dapat menjawab hipotesa H_0 diterima jika *p-value* signifikan dan menolak H_1 ketika p-value kurang signifikan. Dalam penelitian ini akan diatur seberapa signifikan statistik dengan symbol α dengan nilai 0.05. Sehingga dapat dirumuskan jika p-value = α menerima H_0 dan menolak H_1.

Tabel 14. Hasil AUC Uji Friedman untuk *Pairwise Differences*

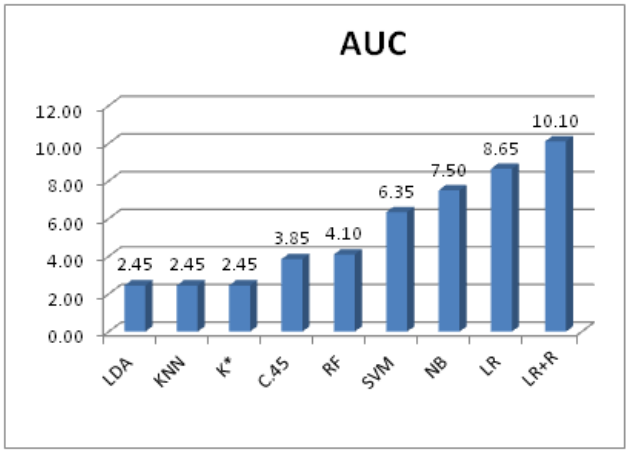
	LR	LR+R	NB	LDA	KNN	C.45	SVM	RF	K*
LR	0	-1.450	1.150	6.200	6.200	4.800	2.300	4.550	6.200
LR+R	1.450	0	2.600	7.650	7.650	6.250	3.750	6.000	7.650
NB	-1.150	-2.600	0	5.050	5.050	3.650	1.150	3.400	5.050
LDA	-6.200	-7.650	-5.050	0	0.000	-1.400	-3.900	-1.650	0.000
KNN	-6.200	-7.650	-5.050	0.000	0	-1.400	-3.900	-1.650	0.000
C.45	-4.800	-6.250	-3.650	1.400	1.400	0	-2.500	-0.250	1.400
SVM	-2.300	-3.750	-1.150	3.900	3.900	2.500	0	2.250	3.900
RF	-4.550	-6.000	-3.400	1.650	1.650	0.250	-2.250	0	1.650
K*	-6.200	-7.650	-5.050	0.000	0.000	-1.400	-3.900	-1.650	0

Untuk uji Friedmant ini, kita mengatur tingkat signifikansi statistik (α) menjadi 0,05. Ini berarti bahwa ada perbedaan yang signifikan secara statistik jika P-value <0,05. Dari hasil percobaan, P-value adalah 0,0001, ini lebih rendah dari tingkat signifikansi α = 0,05, dengan demikian kita harus menolak hipotesis nol, dan itu berarti bahwa ada perbedaan yang signifikan secara statistik. Selanjutnya dilakukan uji Friedman dengan *Nemenyi post hoc tes* untuk mendeteksi pengklasifikasi tertentu berbeda secara signifikan. Tabel 15 memperlihatakan hasil akurasi uji Friedman untuk *p-value.*

Tabel 15. Hasil AUC Uji Friedman untuk P-values

	LR	LR+R	NB	LDA	KNN	C.45	SVM	RF	K*
LR	1	0.997	1.000	**0.001**	**0.001**	**0.047**	0.902	0.078	**0.001**
LR+R	0.997	1	0.808	**0.0001**	**0.0001**	**0.001**	0.288	**0.003**	**0.0001**
NB	1.000	0.808	1	**0.028**	**0.028**	0.328	1.000	0.439	**0.028**
LDA	**0.001**	**0.0001**	**0.028**	1	1.000	0.997	0.233	0.990	1.000
KNN	**0.001**	**0.0001**	**0.028**	1.000	1	0.997	0.233	0.990	1.000
C.45	**0.047**	**0.001**	0.328	0.997	0.997	1	0.843	1.000	0.997
SVM	0.902	0.288	1.000	0.233	0.233	0.843	1	0.915	0.233
RF	0.078	**0.003**	0.439	0.990	0.990	1.000	0.915	1	0.990
K*	**0.001**	**0.0001**	**0.028**	1.000	1.000	0.997	0.233	0.990	1

Dapat dilihat pada Gambar 6, Logistic Regression dengan *Resampling* (LR+R*)* menghasilkan AUC tinggi berdasarkan uji *Freidmen.* Berdasarkan Tabel 15, dapat ditunjukan bahwa pengklasifiasi Naïve Bayes (NN) dan Logistic Regression (LR) menghasilkan AUC yang tinggi seperti penelitian oleh (Hall et al., 2012; Wahono et al., 2011). Seperti yang dilakukan oleh (Saifudin, 2014) yang menggunakan dataset NASA, hasil akurasi NB dan LR menunjukan hasil yang signifikan.



Gambar 6. Grafik Mean AUC uji Friedman

5 KESIMPULAN

Penelitian dengan menerapkan metode *resampling* untuk penyelesaian ketidakseimbangan kelas (*class Imbalance*) pada dataset NASA MDP untuk prediksi cacat *Software* dengan algoritma Logistic Regression. Hasil eksperimen pada penelitian ini mendapatakn nilai akurasi sebesar 99,48% pada dataset MC1 dengan model LR+*Resample*, mengalami peningkatan sebesar 0.16% dari LR tanpa *Resample.* Dan hasil AUC sebesar 0.951 pada dataset PC5 untuk model LR+*Resample.*

Hasil perbandingan dari eksperimen pada penelitian untuk semua dataset dengan pengklasifikasi lain (Naive Bayes (NB), Linear Discriminant Analysis (LDA), k-Nearest Neghbor (k-NN), C4.5, Support Vector Machine (SVM), Random Forest (RF), dan K*) tingkat akurasi Logistic Regression menunjukan hasil yang paling baik, baik dalam parameter AUC maupun akurasi.

Copyright @ 2015 IlmuKomputer.Com

Dari hasil pengujian di atas maka dapat disimpulkan bahwa pengunaan metode LR+*Resample* mampu menangani ketidakseimbangan dataset pada logistic regression dengan menghasilkan nilai akurasi dan AUC lebih tinggi dibandingkan dengan metode LR yang tidak menggunakan Resample. Dari hasil pengujian diatas dapat disimpulkan bahwa metode *resampling* terbukti efektif dalam menyelesaikan ketidakseimbangan kelas pada prefiksi cacat *Software* dengan algoritma Logistic Regression.

REFERENSI

Alpaydin, E. (2010). *Introduction to Machine Learning*. London: The MIT Press.

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

Chang, R., Mu, X., & Zhang, L. (2011). Software Defect Prediction Using Non-Negative Matrix Factorization. *Journal of Software*.

Czibula, G., Marian, Z., & Czibula, I. G. (2014). Software defect prediction using relational association rule mining. *Information Sciences*.

Dubey, R., Zhou, J., Wang, Y., Thompson, P. M., & Ye, J. (2014). Analysis of sampling techniques for imbalanced data: An n=648 ADNI study. *NeuroImage*.

Ganganwar, V. (2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*.

Gorunescu, F. (2011). Data mining: Concepts, models and techniques. *Intelligent Systems Reference Library*, *12*.

Hall, T., Beecham, S., Bowes, D., Gray, D., & Counsell, S. (2012). A Systematic Literature Review on Fault Prediction Performance in Software Engineering. *IEEE Transactions on Software Engineering*.

Harrington, P. (2012). *Machine Learning in Action*. Manning Publications Co.

Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression Second Edition*. New York, NY: John Wiley & Sons, Inc.

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression Third Edition*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Karsmakers, P., Pelckmans, K., & Suykens, J. a. K. (2007). Multi-class kernel logistic regression: a fixed-size implementation. *2007 International Joint Conference on Neural Networks*.

Khoshgoftaar, T. M., Gao, K., Napolitano, A., & Wald, R. (2013). A comparative study of iterative and non-iterative feature selection techniques for software defect prediction. *Information Systems Frontiers*.

Komarek, P., & Moore, A. W. (2005). Making Logistic Regression A Core Data Mining Tool. *School of Computer Science*.

Larose, D. T. (2005). *Discovering Knowladge In Data: An Introduction to Data Mining*. *Discovering Knowledge in Data: An Introduction to Data Mining*.

Lessmann, S., Member, S., Baesens, B., Mues, C., & Pietsch, S. (2008). Benchmarking Classification Models for Software Defect Prediction : A Proposed Framework and Novel Findings.

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*.

Lin, C., Weng, R. C., & Keerthi, S. S. (2008). Trust Region Newton Method for Large-Scale Logistic Regression. *Journal of Machine Learning Research*.

Ma, Y., Luo, G., Zeng, X., & Chen, A. (2012). Transfer learning for cross-company software defect prediction. *Information and Software Technology*.

Maalouf, M., & Trafalis, T. B. (2011). Robust weighted kernel logistic regression in imbalanced and rare events data. *Computational Statistics & Data Analysis*.

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

Thanathamathee, P., & Lursinsap, C. (2013). Handling imbalanced data sets with synthetic boundary data generation using bootstrap re-sampling and AdaBoost techniques. *Pattern Recognition Letters*.

Vercellis, C. (2011). *Business Intelligence: Data Mining and Optimization for Decision Making*. *Methods*. John Wiley & Sons.

Wahono, R. S., Suryana, N., & Ahmad, S. (2014). Metaheuristic Optimization based Feature Selection for Software Defect Prediction. *Journal of Software*.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining Third Edition*. Elsevier Inc.

Wu, J., & Cai, Z. (2011). Attribute Weighting via Differential Evolution Algorithm for Attribute Weighted Naive Bayes (WNB).

Wu, X., & Kumar, V. (2010). *The Top Ten Algorithms in Data Mining*. Taylor & Francis Group.

BIOGRAFI PENULIS

Harsih Rianto. Memperoleh gelar S.Kom dari Sekolah Tinggi Ilmu Komputer Nusa Mandiri Jakarta (STMIK Nusa Mandiri) dan M.Kom dari program pasca sarjana program studi Magister Ilmu Komputer STMIK Nusa Mandiri, Jakarta. Saat ini bekerja sebagai dosen di AMIK BSI Bekasi. Minat penelitiannya saat ini meliputi mechine learning dan software engineering (rekayasa perangkat lunak).

Romi Satria Wahono. Memperoleh gelar B.Eng dan M.Eng pada bidang ilmu komputer di Saitama University Japan, dan Ph.D pada bidang software engineering di Universiti Teknikal Malaysia Melaka. Pengajar dan peneliti di Fakultas Ilmu Komputer, Universitas Dian Nuswantoro. Pendiri dan CEO PT Brainmatics, perusahaan yang bergerak di bidang pengembangan software. Minat penelitian pada bidang software engineering dan machine learning. Profesional member dari asosiai ilmiah ACM, PMI dan IEEE Computer Society.

Copyright @ 2015 IlmuKomputer.Com