RESAMPLING LOGISTIC REGRESSION UNTUK PENANGANAN KETIDAKSEIMBANGAN *CLASS* PADA PREDIKSI CACAT *SOFTWARE*

TESIS

HARSIH RIANTO
14000856

PROGRAM PASCASARJANA MAGISTER ILMU KOMPUTER
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER
NUSA MANDIRI
JAKARTA
2015

RESAMPLING LOGISTIC REGRESSION UNTUK PENANGANAN KETIDAKSEIMBANGAN *CLASS* PADA PREDIKSI CACAT *SOFTWARE*

TESIS

Diajukan sebagai salah satu syarat untuk memperoleh gelar
Magister Ilmu Komputer (M.Kom)

HARSIH RIANTO

14000856

PROGRAM PASCASARJANA MAGISTER ILMU KOMPUTER

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER

NUSA MANDIRI

JAKARTA

2015

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini:

Nama	:	Harsih Rianto
NIM	:	14000856
Program Studi	:	Magsiter Ilmu Komputer
Jenjang	:	Strata Dua (S2)
Konsentrasi	:	*Management Information System*

Dengan ini menyatakan bahwa tesis yang telah saya buat dengan judul: "*resampling logistic regression* untuk penanganan ketidakseimbangan *class* pada prediksi cacat *software*" adalah hasil karya sendiri, dan semua sumber baik yang kutip maupun yang dirujuk telah saya nyatakan dengan benar dan tesis belum pernah diterbitkan atau dipublikasikan dimanapun dan dalam bentuk apapun.

Demikianlah surat pernyataan ini saya buat dengan sebenar-benarnya. Apabila dikemudian hari ternyata saya memberikan keterangan palsu dan atau ada pihak lain yang mengklaim bahwa tesis yang telah saya buat adalah hasil karya milik seseorang atau badan tertentu, saya bersedia diproses baik secara pidana maupun perdata dan kelulusan saya dari Program Pascasarjana Magister Ilmu Komputer Sekolah Tinggi Manajemen Informatika dan Komputer Nusa Mandiri dicabut/dibatalkan.

Jakarta, 2015
Yang menyatakan,

Harsih Rianto

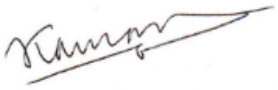
HALAMAN PENGESAHAN

Tesis ini diajukan oleh:

Nama	:	Harsih Rianto
NIM	:	14000856
Program Studi	:	Magsiter Ilmu Komputer
Jenjang	:	Strata Dua (S2)
Konsentrasi	:	*Management Information System*
Judul Tesis	:	"*Resampling Logistic Regression* untuk penanganan ketidakseimbangan *class* pada prediksi cacat *software*"

Telah berhasil dipertahankan dihadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Magister Ilmu Komputer (M.Kom) pada Program Pascasarjana Magister Ilmu Komputer Sekolah Tinggi Manajemen Informatika dan Komputer Nusa Mandiri (STMIK Nusa Mandiri).

Jakarta, 2015
Pascasarjana Magister Ilmu Komputer
STMIK Nusa Mandiri
Direktur



Prof. Dr. Ir. Kaman Nainggolan, MS

D E W A N P E N G U J I

Penguji I : Dr. Sularso Budilaksono .....

Penguji II : Dr. Windu Gata, M.Kom .....

Penguji III/Pembimbing : Romi Satia Wahono, M.Eng, Ph.D .....

	Lembar Konsultasi Bimbingan Tesis
	Pascasarja Magister Ilmu Komputer STMIK Nusa Mandiri

NIM : 14000856
Nama Lengkap : Harsih Rianto
Dosen Pembimbing : Romi Satria Wahono, Ph.D
Judul Tesis : *Resampling* Logistic Regression untuk Penanganan Ketidakseimbangan *Class* Pada Prediksi Cacat *Software*

No	Tanggal Bimbingan	Materi Bimbingan	Paraf Dosen Pembimbing
1	12 November 2014	Pengajuan Judul	
2	20 November 2014	Revisi Judul dan Pengajuan Bab I	
3	5 Desember 2014	Revisi Bab I	
4	15 Desember 2014	Pengajuan Bab II	
5	23 Desember 2014	Revisi Bab II	
6	8 Januari 2015	Pengajuan Bab III	
7	26 Januari 2015	Revisi Bab III dan Pengajuan Bab IV	
8	3 Februari 2015	Revisi Bab IV dan Pengajuan Bab V	
9	14 Februari 2015	Revisi Bab V	
10	28 Februari 2015	Acc Keseluruhan	

Bimbingan dimulai pada tanggal : 12 November 2015
Bimbingan diakhiri pada tanggal : 28 Februari 2015 Jumlah
Pertemuan : 10 Kali Peremuan

Jakarta, 28 Februari 2015

Dosen Pembimbing

Romi Satria Wahono, M.Eng, Ph.D

KATA PENGANTAR

Dengan memanjatkan puji syukur kehadirat Allah S.W.T yang telah melimpahkan segala rahmad dan hidayahnya kepada penulis, sehingga tersusunlah tesis yang berjudul ”*Resampling* Logistic Regression untuk Penanganan Ketidakseimbangan Data Skala Besar pada Prediksi Cacat *Software*”. Tujuan penulisan tesis ini dibuat sebagai salah satu syarat untuk mendapatkan gelar Magister Ilmu Komputer (M.Kom) pada Program Pascasarjana Magister Ilmu Komputer Sekolah Tinggi Manajemen Informatika dan Komputer Nusa Mandiri (STMIK Nusa Mandiri). Tesis ini dibuat berdasarkan hasil penelitian mengenai prediksi cacat software dengan mengusulkan metode resampling dan penanganan ketidakseimbangan kelas pada dataset skala besar. Penulis juga mencari dan menganalisa berbagai sumber referensi, berbentuk jurnal ilmiah, buku-buku literature, *internet* dan lain-lain yang berkitan dengan topik pada pembahasan tesis ini. Penulis menyadari bahwa tanpa bimbingan dan dukungan semua pihak dalam pembuatan tesis ini, maka penulis tidak dapat menyelesaikan tesis ini tepat pada waktunya. Untuk itu ijinkanah penulis dalam kesempatan ini untuk mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Bapak Romi Satria Wahono, Ph.D sebagai pembimbing yang selalu sabar membimbing, menyokong dan memotivasi penulisa serta memberikan sumbangsih ilmu dalam dunia penelitian.
2. Orang tua dan keluarga yang telah memberikan dukungan moral kepada penulis.
3. Seluruh Staff pengajar (Dosen) yang telah memberikan peluang kepada penulis untuk melanjutkan kuliah S2.
4. Yunita Darmastuti, S.Kom atas dukungan, kasih sayang, semangat, do’a dan pengertiannya dan perhatiannya untuk penulis dalam mengerjakan tesis.
5. Teman-teman penulis di Astel Group (Bapak Gentu Basukiyono, Bapak Edi Bastian, Mbak Mira, Mbak Frida) yang selalu memberikan semangat dan motivasi selama proses pengerjaan tesis.

6. Teman-teman seperjuangan (Hani Harafani, Indah, Anjar, Pak Dony, Ispandi, Lila Dini Utami) yang saling menyemangati dalam proses bimbingan serta pembuatan tesis.
7. Teman-teman di group Inteligent System yang selalu menyokong, mengomentari, memberikan semangat dan sumbangsih ilmu yang bermanfaat bagi penulis dalam penulisan tesis ini yaitu temen-temen seperjuangan dari Nusa Mandiri, Eresha dan Udinus antara lain (Adi Wijaya, Rizky Tri Asmono, Vinita Chandani, Tyas Setiyorini, Rizky D dan kawan-kawan yang tidak saya sebutkan satu-persatu).

Serta semua pihak yang tidak dapat penulis sebutkan satu-persatu. Penulis sangat menyadari bahwa penelitian ini sangat jauh dari kesempurnaan. Untuk itu penulis membutuhkan saran dan kritik yang sangat membangun demi kesempuranaan karya ilmiah dimasa mendatang. Akhir kata penulis ucapkan semoga tesis ini dapat bermanfaat terutama bagi para pembaca pada umumnya.

Jakarta, 3 Maret 2015

Harsih Rianto

Penulis

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini, saya:

Nama	:	Harsih Rianto
NIM	:	14000856
Program Studi	:	Magsiter Ilmu Komputer
Jenjang	:	Strata Dua (S2)
Konsentrasi	:	*Management Information System*
Jenis Karya	:	Tesis

Demi pengembangan ilmu pengetahuan, dengan ini menyetujui untuk memberikan ijin kepada pihak Program Pascasarjana Magister Ilmu Komputer Sekolah Tinggi Manajemen Inbentukika dan Komputer Nusa Mandiri (STMIK Nusa Mandiri) **Hak Bebas Royalti Non-Eksklusif** ***(Non-exclusive Royalti-Free Right)*** atas karya ilmiah kami yang berjudul: "*resampling logistic regression* untuk penanganan ketidakseimbangan *class* pada prediksi cacat *software*" beserta perangkat yang diperlukan (apabila ada).

Dengan **Hak Bebas Royalti Non-Eksklusif** ini pihak STMIK Nusa Mandiri berhak menyimpan, mengalih-media atau *bentuk*-kan, mengelolaannya dalam pangkalan data *(database)*, mendistribusikannya dan menampilkan atau mempublikasikannya di *internet* atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari kami selama tetap mencantumkan nama kami sebagai penulis/pencipta karya ilmiah tersebut.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak STMIK Nusa Mandiri, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini saya buat dengan sebenarnya.

Jakarta, 2015

Yang menyatakan,

Harsih Rianto

ABSTRAK

Nama : Harsih Rianto
NIM : 14000856
Program Studi : Magsiter Ilmu Komputer
Jenjang : Strata Dua (S2)
Konsentrasi : *Management Information System*
Judul Tesis : "*Resampling logistic regression* untuk penanganan ketidakseimbangan *class* pada prediksi cacat *software*"

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* menghabiskan 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. 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.

Kata Kunci: Ketidakseimbangan Kelas, Logistic Regression, *Resampling*.

ABSTRACT

Name	:	Harsih Rianto
NIM	:	14000856
Study of Program	:	Magsiter Ilmu Komputer
Levels	:	Strata Dua (S2)
Consentratrion	:	*Management Information System*
Title	:	"*Resampling Logistic Regression for Handling of Class Imbalance on Software Defect Prediction*"

High-quality software is software that can help the company's business processes with effective, efficient and not found to be defective during the testing process, inspection, and implementation. Improvement and implementation of software after delivery started, will cost much more expensive than at the time of development. Costs required for software testing spend more than 50% of the development costs. Software defects testing model is needed to reduce costs. Currently there is no predictive model of software defects generally accepted at the time of use. Logistic Regression Model is the most effective and efficient models in the prediction of software defects. The downside of Logistic Regression is susceptible to underfitting the dataset whose class is not balanced, so it will produce a low accuracy. NASA dataset MDP is a common dataset used in the prediction of software defects. One of the characters of the dataset prediction software defects, including NASA dataset MDP is to have an imbalance in the classroom. To address the issue of imbalances in the dataset class software defects in this study proposed resampling method. Experiments conducted to compare performance results before and after the Logistic Regression applied resampling methods. Likewise conducted experiments to compare the proposed method results such as Naive Bayes classifier, Descriminant Linear Analysis, C4.5, Random Forest, Neural Network, k-Nearest Network. The experimental results showed that the level of accuracy Logistic Regression with resampling higher than Logistric Regression methods that do not use resampling, as well as when compared to other classifier. From the above experimental results it can be concluded that the resampling method proved effective in resolving imbalances in the class prediction algorithm software defects with Logistic Regression.

Keywords: Logistic Regression, Class Imbalances, Resampling.

DAFTAR ISI

Halaman

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1 Latar Belakang

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 (R. Chang, Mu, & Zhang, 2011; Czibula, Marian, & Czibula, 2014; Ma, Luo, Zeng, & Chen, 2012; Wahono, Suryana, & Ahmad, 2014). Menurut Ruihua Chang, untuk meningkatkan efesiensi dan jaminan kualitas yang tinggi dari sebuah *software*, dalam pengujiannya diperlukan program yang mampu memprediksi cacat *software* (R. 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* berasumi pada semua atribut dataset adalah sama penting dan tidak terkait satu sama lain, sedangkan pada kenyataannya asumsi ini tidaklah benar (J. Wu & Cai, 2011). *Logistic Regression* adalah metode klasifikasi statistik probabilitas. Keuntungan *Logistic Regression* adalah metode komputasi lebih murah, mudah dalam penerapannya, dan mudah dalam mamahami hasil proses klasifikasi (Canu & Smola, 2006). Kelemahan dari *Logistic Regression* adalah rentan terhadap *underfitting* dan memiliki akurasi yang rendah (Harrington, 2012). *Underfiiting* adalah keadaan dimana prediktor dari *Logistic Regression* tidak fleksibel mengamati tren yang terjadi pada dataset.

Masalah besar yang dialami oleh algoritma *Logistic Regression* adalah ketidakseimbangan kelas (*class imbalance*) pada dataset skala besar (Lin, Weng, & Keerthi, 2008). Jika dilihat dari dataset yang digunakan untuk prediksi cacat *software*, secara umum menggunakan dataset NASA masih mengalami ketidakseimbangan (*imbalance*) data (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 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*.

Penanganan ketidakseimbangan kelas (*class imbalance*) yang terjadi pada dataset dapat diatasi dengan teknik *resampling* (Cateni, Colla, & Vannucci, 2014). Teknik *resampling* akan memanipulasi kelas distribusi dengan memper-baiki data latih sehingga didapatkan kelas yang seimbang. Terdapat beberapa teknik *resampling* antara lain: *random over sampling, random under sampling,*

directed over sampling, directed under sampling, kombinasi dari *over sampling* dan *under sampling,* dan *Synthetic Minority Oversampling Technique* (SMOTE) (Cateni et al., 2014; Chawla, Bowyer, Hall, & Kegelmeyer, 2002). Metode *cluster-based under-sampling* yang diterapkan oleh Show Jane Yen juga merupakan metode *resampling* yang menerapakn teknik *under sampling* menggunakan neural network untuk mengklasifikasikan dataset (Yen & Lee, 2009). Metode *Similarity-based UnderSampling and Normal Distribution-based Oversampling* (SUNDO) adalah metode *resampling* yang mengkombinasikan teknik *under sampling* dan *over sampling* sehingga menghasilkan kinerja dataset yang lebih baik (Cateni et al., 2014). Dengan menggunakan metode *resampling* pada pengolahan data, maka akan didapatkan keseimbangan kelas yang merata.

Logistic Regression merupakan klasifikasi linier yang telah terbukti menghasilkan klasifikasi yang powerful dengan statisik probabilitas dan menangani masalah klasifikasi multi *class* (Canu & Smola, 2006; Karsmakers, Pelckmans, & Suykens, 2007). Selain itu juga dilakukan penelitian secara ekstensif oleh (Hosmer, Lemeshow, & Sturdivant, 2013) pada *Logistic Regression*. Dan baru-baru ini telah dilakukan perbaikan algoritma LR melalui model *truncated newton's* oleh (Komarek & Moore, 2005; Maalouf & Trafalis, 2011). Dengan algoritma yang tepat pada LR maka waktu perhitungan dapat jauh lebih cepat dari pada model algoritma lain seperti *Support Vector Machine* (SVM). Penelitian pertama yang menerapkan model *Truncated Regularized Iteratively Re-weighted Least Squares* (TR-IRLS) di LR untuk mengklasifikasikan kumpulan dataset skala besar menujukan hasil klasifikasi yang mengunguli algoritma SVM (Komarek & Moore, 2004). Kemudian penelitian yang menerapkan model *Truncated Newton Interior-point* di LR untuk menanganai masalah dataset skala besar (Koh, Kim, & Boyd, 2007). Penerapan model *Trust Region Newton* yang merupakan jenis dari *Truncated Newton* juga diterapakan untuk menangani masalah dataset skala besar (Lin et al., 2008). Model *Trust Region Newton* ini mampu menangani dataset skala besar dan mempercepat proses komputasi yang dijalankan.

Berdasarkan penelitian sebelumnya, *Logistic Regression* merupakan algoritma klasifikasi yang banyak digunakan oleh para peneliti. Untuk mencari

solusi terbaik terhadap masalah ketidakseimbangan kelas (*class imbalance*) pada dataset skala besar, maka penelitian ini akan dilakukan pengukuran kinerja pada algoritma *Logistic Regression* dengan pengolahan data awal menggunakan *resampling*. Diharapkan mendapatkan peningkatan hasil kenerja algoritma *Logistic Regression* untuk menyelesaikan masalah ketidakseimbangan kelas data skala besar.

1.2 Identifikasi Masalah

Berdasarkan latar belakang permasalahan di atas maka rumusan masalah pada penelitan ini adalah *Logistic Regression* mampu menangani metode klasifikasi terutama pada dataset skala besar, tetapi *Logistic Regression* memiliki kelemahan pada prediktor yang tidak fleksibel untuk menangani tren yang terdapat dilatih yang disebut *underfitting* dan memiliki akurasi rendah terhadap ketidakseimbangan kelas (*class imbalance*) data skala besar.

1.3 Rumusan Masalah

Berdasarkan identifikasi masalah diatas, maka rumusan masalah pada penelitian ini adalah seberapa akurat algoritma klasifikasi *Logistic Regression*, jika diterapkan model *resampling logistic regression* untuk menangani ketidakseimbangan kelas *(class imbalance) dataset* skala besar dengan menerapkan metode *resampling* pada prediksi cacat *software?*.

1.4 Tujuan Penelitian

Tujuan penelitian ini adalah menerapkan metode *resampling logistic regression* untuk menangani ketidakseimbangan kelas (*class imbalance*) dataset skala besar pada *Logistic Regression* agar diperoleh akurasi tinggi pada prediksi cacat *software*.

1.5 Manfaat Penelitian

Berdasarkan tujuan penelitian, maka manfaat dari penelitian ini adalah:

1. Dapat dijadikan bahan pertimbangan peneliti lain dalam pemanfaatan algoritma *Logistic Regression*.
2. Dapat digunakan sebagai bahan pertimbangan dalam pengambilan keputusan untuk prediksi cacat *software* yang menggunakan dataset skala besar.

1.6 Ruang Lingkup Penelitian

Ruang lingkup pembahasan dalam penelitian ini dibatasi pada penerapan metode *resampling Logistic Regression* untuk menguji ketidakseimbangan kelas (*class imbalance*) dataset skala besar untuk meningkatkan akurasi algoritma *Logistic Regression* pada prediksi cacat *software*.

1.7 Sistematika Penulisan

Dalam penulisan tesis ini terdapat beberapa bab yang dibahas, diantaranya adalah:

Bab I Pendahuluan

Bab ini membahas tentang latar belakang penulisan, identifikasi masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, ruang lingkup penelitian dan sitematika penulisan.

Bab II Landasan Teori

Bab ini membahas tentang teori yang melandasi penelitian yaitu algoritma *Logistic Regression,* ketidakseimbangan kelas *(class imbalance)* dataset. Studi kasus disajikan untuk memberikan gambaran yang lebih jelas langkah-langkah algoritma *Logistic Regression.*

Bab III Metode Penelitian

Bab ini membahas tentang metode pengumpulan data dan eksperimen. Eksperimen merupakan inti dari pembahasan ini, yaitu menguji algoritma *Logistic Regression* tradisional dan *resampling Logistic Regression* untuk mendapatkan akurasi yang tinggi.

Bab IV Hasil dan Pembahasan

Bab ini menjelaskan tentang pengujian model yang dihasilkan pada bab sebelumnya, baik setelah menggunakan model dan sebelum diterapkan model *resampling.*

Bab V Penutup

Bab ini membahas tentang kesimpulan dari pembahasan pada bab-bab sebelumnya dan saran-saran untuk penelitian selanjutnya.

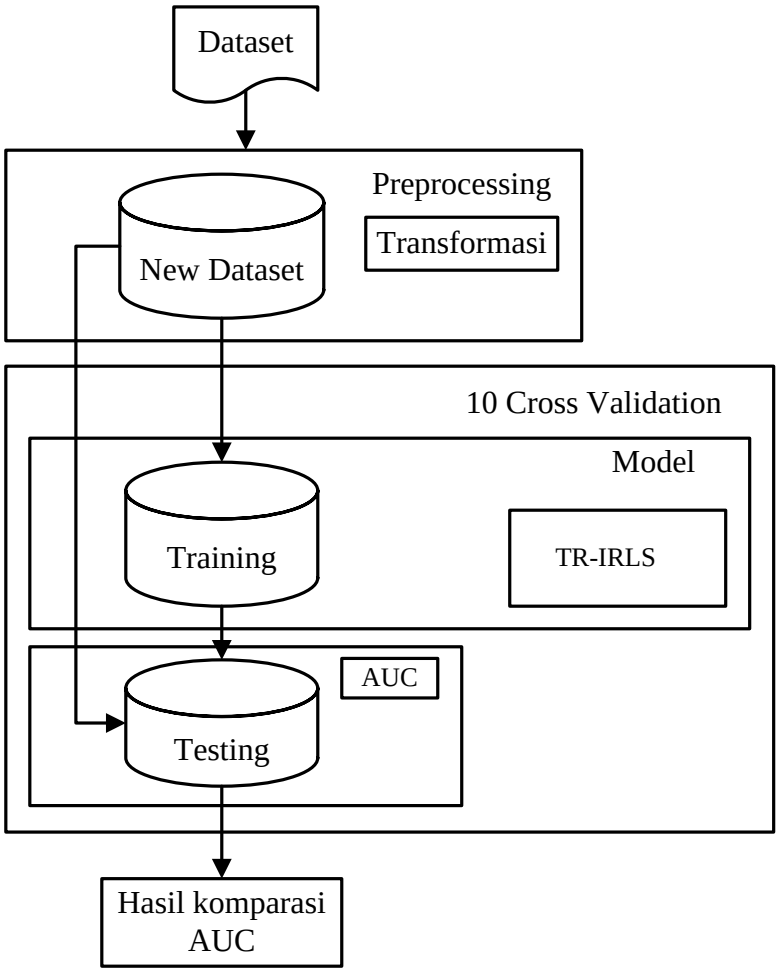
BAB II
LANDASAN TEORI

2.1 Tinjauan Studi

2.1.1 Model Komarek dan Moore (2005)

Pada penelitian yang dilakukan oleh Komarek dan Moore (Komarek & Moore, 2005) untuk meningkatkan akurasi *Logistic Regression*, menerapkan 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 mengungguli algoritma lain seperti *Naïve Bayes, Support Vector Machine* (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.

Model yang dilakukan (Komarek & Moore, 2005) dapat diilustrasikan pada Gambar 2.1. Dataset yang digunakan terlebih dahulu melalui tahapan *preprocessing* dan proses transformasi data. Tahapan selanjutnya adalah 10-*fold cross validation* dataset yang sudah ditransformasi dalam 2 bagian untuk data training dan data testing. Kemudian model TR-IRLS akan diterapkan pada data training dan hasil pengujian model akan diuji pada data testing dan didapatkan hasil AUC. Setelah proses berulang sebanyak 10 kali dan mendapatkan hasil AUC, tahapan selanjutnya adalah menguji hasil model dengan menghitung perbandingan nilai AUC yang dihasilkan.



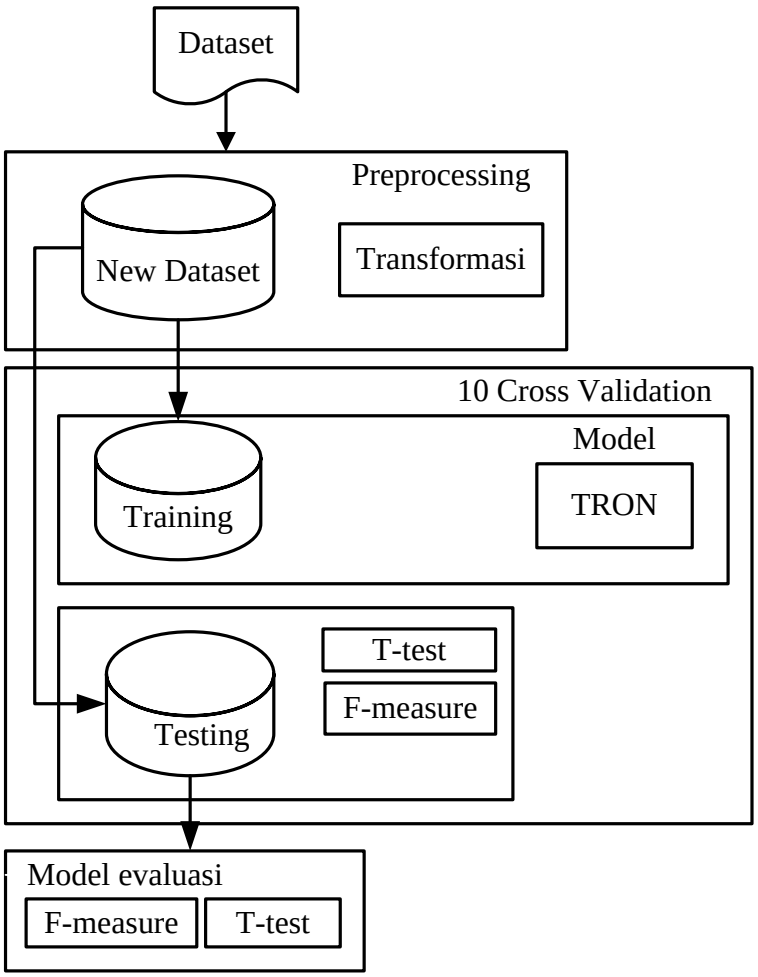
Gambar 2.1 Model Komarek & Moore (2005)

2.1.2 Model Lin, Weng dan Keerthi (2008)

Penelitian yang dilakukan oleh Lin, Weng dan Keerthi menunjukan hasil konvegensi yang lebih cepat dari pada metode *quasi Newton* (Lin et al., 2008). Metode *Truncated Newton* merupakan metode yang telah diakui mampu menangani dataset skala besar seperti penelitian (Komarek & Moore, 2005) namun metode ini tidak sepenuhnya digunakan. Penelitian ini merupakan turunan pengunanan model *Newton*, yang menggunakan dataset skala besar 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 yang lebih cepat dibadingkan metode

limited memory quasi Newton (LBFGS) penelitian Liu dan Nocedal 1989 dalam (Lin et al., 2008).

Model yang dilakukan (Lin et al., 2008) dapat dilihat pada Gambar 2.2. Dataset yang digunakan terlebih dahulu melalui tahapan *preprocessing* dan proses transformasi data. Tahapan selanjutnya adalah 10-*fold cross validation* dataset yang sudah ditransformasi dalam 2 bagian untuk data training dan data testing. Kemudian model TRON akan diterapkan pada data training dan hasil pengujian model akan diuji pada data testing. Pada tahapan testing dilakukan pengujian model dengan uji *t-test* dan uji *F-measure*. Setelah proses berulang sebanyak 10 kali dan mendapatkan hasil uji *t-test* dan uji *F-measure*, tahapan selanjutnya adalah mengevaluasi hasil model dengan menggunakan uji *t-test* dan *F-measure*.

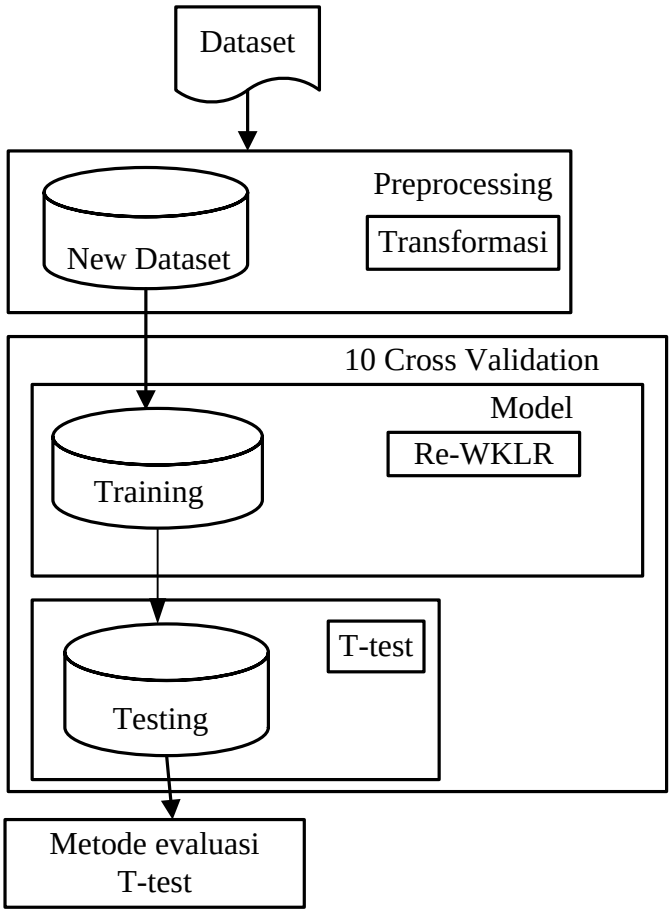


Gambar 2.2 Model Lin, Wen, & Keerthi (2008)

2.1.3 Model Maalouf dan Trafalis (2011)

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 klasfikasi yang lebih tinggi dari pada SVM. Secara keseluruhan hasil akurasi dari penelitian ini dengan menggunakan pengukuran komparasi paired t-test. Kesimpulan dari penelitian ini menunjukan 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.

Model yang dilakukan Maalouf dan Trafalis dapat dilihat pada Gambar 2.3. Dataset yang digunakan terlebih dahulu melalui tahapan *preprocessing* dan proses transformasi data. Tahapan selanjutnya adalah 10-*fold cross validation* dataset yang sudah ditransformasi dalam 2 bagian untuk data training dan data testing. Kemudian model Re-WLKR akan diterapkan pada data training dan hasil pengujian model akan diuji pada data testing. Pada tahapan testing dilakukan pengujian model dengan uji *t-test*. Setelah proses berulang sebanyak 10 kali dan mendapatkan hasil uji *t-test*, tahapan selanjutnya adalah mengevaluasi hasil model dengan menggunakan uji *t-test*.



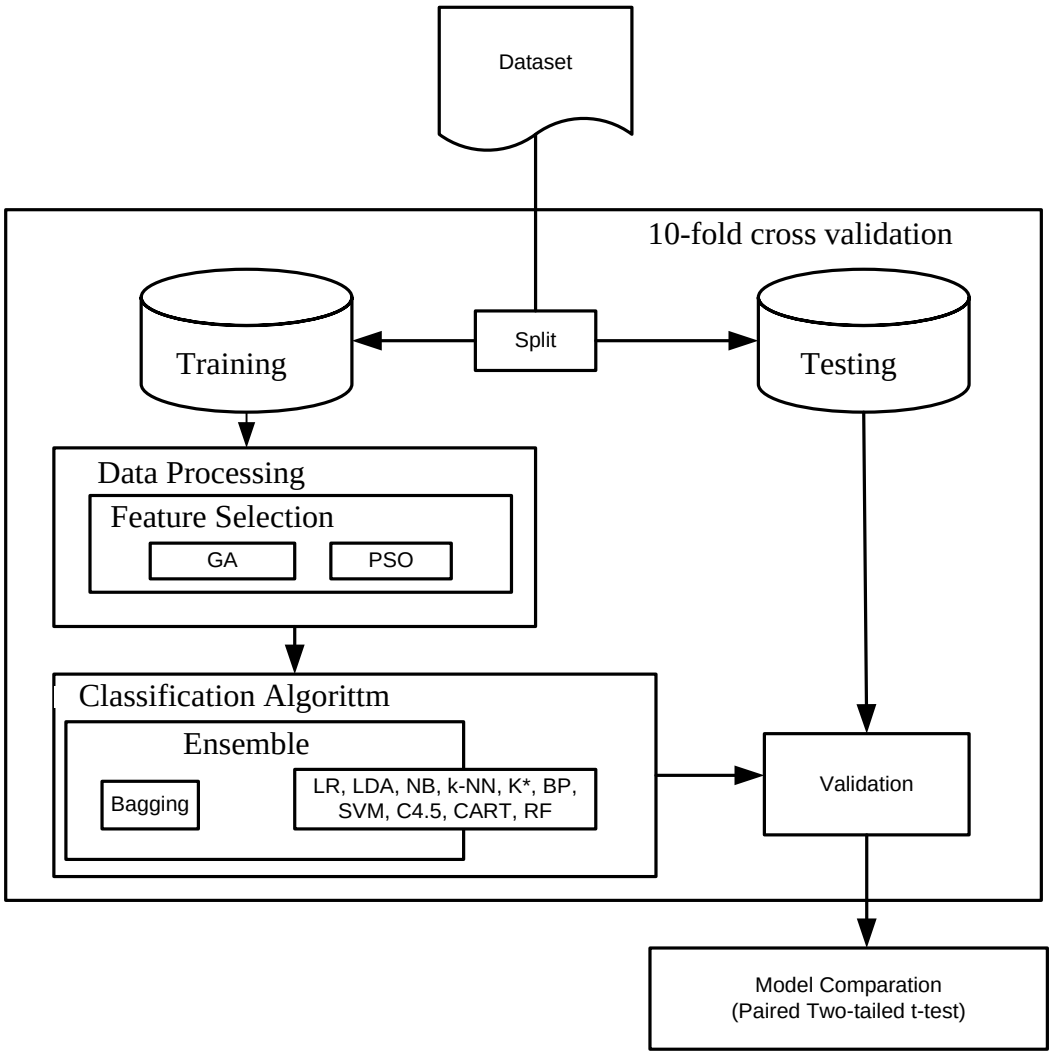
Gambar 2.3 Model Maalouf dan Trafalis (2011)

2.1.4 Model Wahono, Suryana, dan Ahmad (2014)

Penelitian yang dilakukan oleh Wahono, Suryana, dan Ahmad menyatakan bahwa penelitian pada prediksi cacat *software* telah menjadi topik yang penting dalam bidang *Software Engineering* (Wahono et al., 2014). 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.* Penelitian ini 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*. 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*), 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*.

Model Wahono, Suryana, dan Ahmad dapat dilihat pada Gambar 2.4. Dataset yang digunakan dimasukan kedalam model 10-*fold cross validation*. Kemudian dalam proses 10-*fold cross validation* dataset akan dibagi 2 menjadi data training dan data testing. Data training akan melalui proses data processing, dengan model *feature selection* menggunakan algoritma genetika (GA) dan *particle swarm optimization* (PSO). Proses selanjutnya adalah melakukan pengujian model algoritma klasifikasi dengan menggabungankan teknik *ensemble* bagging. Setelah proses klasifikasi dan ensemble selesai kemudian dilakukan validasi model pada data testing. Hasil validasi model klasifikasi akan dipergunakan untuk melakukan komparasi model menggunakan metode uji *paired two-tailed t-test.*



Gambar 2.4 Model Wahono, Suryana, dan Ahmad (2014)

2.1.5 Rangkuman Penelitian Tekait

Keempat penelitian terkait memiliki model prediksi yang berbeda pada penerapan algoritma *Logistic Regression*. Model Komarek dan Moore, menerapkan metode IRLS, TR-IRLS, dan *Generic Likelihood Maximization* dapat meningkatkan hasil prediksi di Logistic Regression (Komarek & Moore, 2005). Model TRON yang diterapkan oleh Lin, Weng, dan Keerthi mampu meningkatkan waktu komputasi yang lebih cepat dengan dataset skala besar (Lin et al., 2008). Model Maalouf dan Trafalis menerapkan metode RE-WKLR pada dataset yang skala menengah atau sedang (Maalouf & Trafalis, 2011). Model Wahono, Suryana, dan Ahmad melakukan optimasi metaheuristik dan teknik bagging pada algoritma pengklasifikasi salah satunya adalah Logistic Regression

(Wahono et al., 2014). Tabel 2.1 menjelaskan secara singkat mengenai perbandingan penelitian terkait.

Tabel 2.1 Perbandingan Penelitian Terkait

Peneliti	Tahun	Dataset	Metode		Evaluasi
			Data	Klasifikasi	
Komarek dan Moore	2005	Citeseer, Imdb, ds2, ds1, ds1.100, ds1.10, modapte.sub	*Iteratively re-resampling least squares (IRLS),Truncated Regularized IRLS (TR-IRLS), Generic Likelihood Maximization*	*Logitic Regression*	AUC 0.94 tertinggi dan 0.72 terendah.
Lin, Weng, dan Keerthi	2008	a9a, real-sim, news20, yahoo-japan, rcv1, yahoo-korea	*Trust Region Newton (TRON)*	*Logitic Regression*	t-test dan f-measure
Maalouf dan Trafalis	2011	UCI *Machine Learnin*, dan tornado	*rare event resampling kernel logistic regression* (RE-WKLR)	*Logitic Regression*	t-test 0.017
Wahono, Suryana, dan Ahmad	2015	NASA MDP	Optimasi metaheuristik (GA dan PSO)	ensamble Bagging dengan LR, LDA, NB, k-NN, K*, BP, SVM, C4.5, CART dan RF	AUC rata-rata meningkat 25,99% untuk GA dan 20,41% untuk PSO
Rianto	2015	NASA MDP	*resampling*	*Logitic Regression*	?

Pada penelitian ini menggunakan dataset NASA MDP yang diambil dari (http://nasa-softwaredefectdatasets.wikispaces.com/ dengan menggunakan algoritma *Logistic Regression* untuk prediksi cacat *software*, yang menerapkan metode *Resampling Logistic Regression* untuk menangani ketidakseimbangan kelas (*class imbalance*) dan skala besar data.

2.2 Tinjauan Pustaka

2.2.1 Prediksi Cacat *Software*

Cacat *software* atau perangkat lunak dapat didefinisikan sebagai cacat pada perangkat lunak seperti cacat pada dokumentasi, pada kode program, pada desain dan hal–hal lain yang menyebabkan kegagalan perangkat lunak. Cacat *software* dapat muncul pada berbagai tahap proses pengembangan (Pressman, 2010). Cacat *software* merupakan faktor penting yang mempengaruhi kualitas *software* tersebut. Teknik pencegahan cacat perangkat lunak pertama kali diusulkan oleh IBM corporation dan dapat digunakan untuk mencegah munculnya

cacat *software* di tahap lanjut pada proses pengembangan perangkat lunak (C. Chang & Chu, 2007). Kualitas *software* dapat ditingkatkan dengan mencegah munculnya cacat *software* melalui perbaikan modul yang mungkin menghasilkan cacat *software* pada proses pengembangan.

Cacat *software* dapat menjadikan penyebab kegagalan *software* akibat aksi dari user yang menggunakan software pada saat tertentu. Bagi pengguna, cacat *software* adalah segala hal yang menyebabkan hasil dari *software*, yang diinginkan pengguna tidak terpenuhi. Contohnya pengguna ingin melakukan perhitungan yang tepat, namun hasil perhitungan yang dikeluarkan salah. Secara umum, kesalahan penulisan sintaks bukan merupakan cacat *software*, karena kesalahan sintaks dapat ditemukan menggunakan pengujian dasar.

Pada Tabel 2.2 menampilkan beberapa contoh umum cacat pada *software*. Harapan pengguna *software* tidak bisa dijalankan atau dieksekusi oleh *software*, maka masuk dalam kategori cacat *software*. Jika harapan pengguna dapat dieksekusi oleh *software*, maka cacat pada *software* bisa diartikan tidak ada.

Tabel 2.2 Contoh Umum Cacat Software

Harapan pengguna	Cacat software
Software memanggil metode API dan menjalankan sesuai dokumentasi dari software	Motode API gagal dijalankan karena perubahan API tidak teregister dipaket software
Dapat melakukan pengkopian file	File mengalami kerusakan selama proses copy
Software dapat membantu pengguna menyelesaikan tugas	Fungsi dari perangkat lunak hilang dan tidak mampu membantu menyelesaikan tugas pengguna

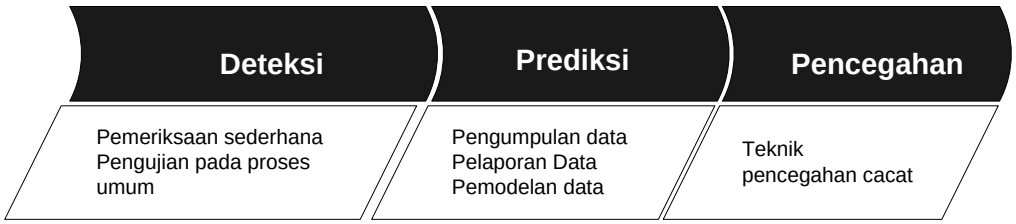
Sedangkan pada Tabel 2.3 menampilkan beberapa contoh spesifik dari cacat *software*. Pada contoh spesifik cacat *software* dapat dilihat bahwa, cacat *software* lebih spesifik kedalam sistem komputer.

Tabel 2.3 Contoh Nyata Cacat Software

Harapan Pengguna	Cacat Sofware
Software mampu merespon aksi pengguna dengan cepat.	*Software* tidak mampu memberikan respon dengan cepat.
Software aman dari serangan hacker.	Hacker mampu mengeksploitasi
Jika terjadi kesalahan yang fatal, *software* dapat menggunakan kode awal saat penggiriman, untuk mengurangi dampak dari kesalahan yang terjadi.	*Software* menggunakan kode awal, namun kode awal tidak bisa digunakan kembali.

Hasil cacat *software* yang terjadi karena beberapa jenis kesalahan antara lain (MacDonald, Musson, & Smits, 2008): 1) kesalahan manusia, ini biasa terjadi saat programmer mengetikan kode program, karena beban kerja programmer yang terlalu berat sehingga terjadi kesalahan pengetikan kode program. 2) Kesalahan sistemik pada proses pengembangan, ini bisa terjadi karena ada kesalahan proses inputan atau keluaran yang digunakan pada proses lain. 3) Kesalahan dari penggunaan perangkat keras saat dilakukan pengembangan software. 4) Akibat salah pengertian kebutuhan pengguna dengan pengembang. Namun tidak semua kesalahan yang terjadi dapat menyebabkan cacat *software*, karena sebab-sebab terjadinya kesalahan dapat ditelusuri kesalahan yang menyebabkannya.

Dalam meningkatkan kualitas perangkat lunak dan tingkat cacat *software* dapat diatas dengan tiga tingkatan (MacDonald et al., 2008) yaitu: 1) Deteksi, adalah proses paling umum dilakukan pengembang untuk melakukan pengujian *software* sampai ditemukannya cacat yang biasa terjadi. 2) Analisa, adalah proses mencari penyebab dan kemungkinan terjadinya cacat *software* karena cacat yang terjadi tidak terdeteksi sebelumnya. 3) Pencegahan, adalah proses yang dilakukan secara proaktif mencari dan menghilangkan cacat *software*. Model pencegahan cacat *software* dapat dilihat pada Gambar 2.4.



Gambar 2.5 Model Pencegahan Cacat Software

Pada Gambar 2.4 tahapan pertama adalah dilakukan pendeteksian terhadap cacat *software* dengan pemeriksaan sederhana dan pengujian pada tiap proses dengan lengkap untuk mengetahui cacat yang terjadi. Selanjutnya adalah tahapan prediksi, pada tahapan ini dilakukan pengumpulan data berdasarkan pada pengukuran yang tepat, kemudian dibuat pelaporan data agar bisa digunakan bagi yang membutuhkan, kemudian dilakukan pemodelan data untuk dijadikan model prediksi cacat pada pengembangan selanjutnya. Tahapan yang ketiga adalah

pencegahan, pada tahapan ini dilakukan dengan mengimplementasikan teknik pencegahan cacat berdasarkan hasil tahapan prediksi.

Pada tahapan prediksi seperti Gambar 2.4 dibutuhkan pengumpulan data, pelaporan data dan pemodelan data yang menggunakan dataset. Masalah utama dari prediksi cacat *software* adalah penggunaan dataset (Wahono et al., 2014). Beberapa perusahaan melakukan pengembangan prediksi cacat *software* menggunakan dataset perusahaan sendiri dan disajikan pada laporan konfrensi. Namun penggunaan dataset pribadi perusahaan sangat susah untuk dilakuan perbandingan model prediksi orang lain. Dibutuhkan dataset umum yang bisa digunakan secara berulang, tidak dapat disangkal dan dapat diverifikasi (Catal & Diri, 2009). Dataset prediksi cacat *software* yang umum digunakan baru-baru ini menurut Gray dalam (Wahono et al., 2014) adalah dataset NASA Matrik Data (MDP). Penjelasan mengenai dataset NASA MDP pada sub bab 2.2.2 Dataset NASA.

2.2.2 Dataset NASA

Dataset NASA sudah tersedia untuk umum dan merupakan dataset matrik perangkat lunak yang populer dalam mengembangkan model prediksi cacat *software*, karena sudah ada 62 peneliti yang menggunakan dataset NASA dari 208 jurnal penelitian (Hall et al., 2012). Dataset NASA untuk umum telah digunakan sebagai bagian dari penelitian cacat *software* (Song et al., 2011). Dataset NASA yang tersedia mencakup dalam 21 tingkat metode matrik yang diusulkan oleh Halsted dan McCabe (Catal & Diri, 2009). Dataset NASA sangat berguna karena memungkinkan peneliti menggunakan model berbeda dan variable independen untuk dibandingkan dengan peneliti lain dengan dataset yang sama.

Dataset NASA saat ini terdiri dari 13 dataset secara eksplisit ditujukan untuk para peneliti matrik perangkat lunak (Hall et al., 2012). Setiap set kumpulan data merupakan system/subsistem *software* NASA dan berisi matrik kode statis dan data kesalahan dari setiap modul. Matrik kode statis yang dicatat meliputi:

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

Pada dataset NASA yang masih asli, masih terdapat kegaduhan *(noise)* yaitu data yang tidak masuk akal atau tidak konsisten. Kegaduhan *(noise)* dapat disebut sebagai data yang salah (Liebchen & Shepperd, 2008). Untuk itu perlu dilakukan proses pembersihan data awal agar kegaduhan *(noise)* tidak ada. Prosedur pembersihan data dari kegaduhan *(noise)* adalah prosedur yang memakan waktu sangat padat dan merupakan suatu prosedur mutlak dalam proses pengambilan dataset (Witten, Frank, & Hall, 2011). Diskripsi dari dataset NASA ditunjukan pada Tabel 2.4. Spesifikasi dataset yang masih asli dapat dilihat pada Tabel 2.5. Spesifikasi dataset yang sudah dilakukan pembersihan kegaduhan *(noise)* pada Tabel 2.6 dan Tabel 2.7.

Dataset NASA MDP terdiri dari sejumlah modul *software*, masing-masing mewakili sejumlah sistem dan bahasa program yang dipergunakan. Bahasa pemrograman pada dataset NASA menggunakan C, C++ dan Java. Sistem yang ditangani oleh dataset NASA antara lain: CM1 merupakan sistem yang menangani instrumen pesawat ruang, JM1 menangani sistem prediksi pendarat-an *realtime*, KC1 dan KC3 menangani sistem manajemen penyimpanan data lapangan, MC2 menangani sistem panduan video, PC1, PC3 dan PC4 menangani sistem penerbangan satelit yang mengorbit di bumi, PC2 menangani simulator dinamis untuk sitem kontrol perilaku, dan PC5 menangani sistem peningkatan keamanan sistem upgrade kokpit. Diskripsi lengkap dataset NASA MDP dijabarkan pada Tabel 2.4.

Tabel 2.4 Diskripsi Dataset NASA MDP

Dataset	Sistem	Bahasa	Total LOC
CM1	Instrumen pesawat ruang angkasa	C	20K
JM1	Sistem prediksi pendaratan realtime	C	315K
KC1	Manajemen penyimpanan data lapangan	C++	18K
KC3	Manajemen penyimpanan data lapangan	Java	18K
MC2	Sistem panduan video	C,C++	6K
PC1	Software penerbangan satelit yang mengorbit bumi	C	40K
PC2	Simulator dinamis untuk sistem kontrol perilaku	C	26K
PC3	Software penerbangan satelit yang mengorbit bumi	C	40K
PC4	Software penerbangan satelit yang mengorbit bumi	C	36K
PC5	Peningkatan keamanan sistem upgrade kokpit	C++	164K

Spesifikasi dataset NASA MDP yang masih asli dapat dilihat pada Tabel 2.5. Jumlah atttribut pada dataset CM1, KC3, MC2, PC1, PC2, PC3 dan PC4 sebanyak 41 attribut, dataset JM1 dan KC1 sebanyak 22 attribut, dan dataset PC5 sebanyak 40 attribut. Jumlah modul, jumlah cacat dan persentase cacat pada dataset Nasa MDP yang masih asli dijabarkan pada Table 2.5.

Tabel 2.5 Spesifikasi Dataset NASA MDP

Dataset	Attribut	Modul	Cacat	Cacat (%)
CM1	41	523	48	9.18%
JM1	22	10878	2102	19.32%
KC1	22	2107	325	15.42%
KC3	41	458	43	9.39%
MC2	41	161	52	32.30%
PC1	41	1107	76	6.80%
PC2	41	5589	23	0.41%
PC3	41	1563	160	10.24%
PC4	41	1458	178	12.21%
PC5	40	17186	516	3.00%

Spesifikasi dataset NASA MDP yang masih asli dapat dilihat pada Tabel 2.6. Jumlah atttribut pada dataset CM1, PC1, PC3 dan PC4 sebanyak 38 attribut, dataset JM1 dan KC1 sebanyak 22 attribut, KC3 dan MC2 sebanyak 40, PC1 sebanyak 37 dan dataset PC5 sebanyak 39 attribut. Jumlah modul, jumlah cacat dan persentase cacat pada dataset Nasa MDP yang masih asli dijabarkan pada Table 2.6.

Tabel 2.6 Spesifikasi Dataset Pembersian Noise Kesatu

Dataset	Attribut	Modul	Cacat	Cacat (%)
CM1	38	344	42	12.21%
JM1	22	9591	1759	18.34%
KC1	22	2095	325	15.51%
KC3	40	200	36	18.00%
MC2	40	125	44	35.20%
PC1	38	735	61	8.30%
PC2	37	1493	16	10.70%
PC3	38	1099	138	12.56%
PC4	38	1379	178	12.91%
PC5	39	16962	502	2.96%

Spesifikasi dataset NASA MDP yang masih asli dapat dilihat pada Tabel 2.7. Jumlah atttribut pada dataset CM1, PC1, PC3 dan PC4 sebanyak 38 attribut,

dataset JM1 dan KC1 sebanyak 22 attribut, KC3 dan MC2 sebanyak 40, PC2 sebanyak 37 dan dataset PC5 sebanyak 39 attribut. Jumlah modul, jumlah cacat dan persentase cacat pada dataset Nasa MDP yang masih asli dijabarkan pada Table 2.7.

Tabel 2.1 Spesifikasi Dataset Pembersian Noise Kedua

Dataset	Attribut	Modul	Cacat	Cacat (%)
CM1	38	327	42	12,84%
JM1	22	7720	1612	20,88%
KC1	22	1162	294	25,30%
KC3	40	194	36	18,56%
MC2	40	124	44	35,48%
PC1	38	679	55	8,10%
PC2	37	722	16	2,22%
PC3	38	1053	130	12,35%
PC4	38	1270	176	13,86%
PC5	39	1694	458	27,04%

2.2.3 Logistic Regression

Menurut (Karsmakers et al., 2007), Logistic Regression dipresentasikan untuk prediksi dengan menggunakan lebih dari satu Linear Regression. Logistic Regression menampilkan persamaan linear yang saling berhubungan antara beberapa variabel acak, dimana variabel yang tergantung (*dependent*) adalah variabel yang berkelanjutan. Metode ini merupakan perluasan 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 disebut juga *Multiple Linear Logistic Regression*. Sedangkan menurut (Witten et al., 2011) model *Logistic Regression* merupakan probabilitas dari beberapa peristiwa, metode ini menggunakan fungsi linear untuk perhitungan prediksi pada beberapa variabel.

Menurut (Komarek & Moore, 2004) linier regression sangat berguna untuk dataset linier dan aplikasi yang memadai. Namun untuk dataset yang kontinue (0,1) atau dengan dataset bernilai biner (0,1) tidak mungkin tepat menggunakan liner 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, menang atau kalah, dan yang lainnya.

Notasi dalam Logistic Regression dimana variabel $\boldsymbol{X} \in Re^{Nxd}$ adalah data matrik di mana 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 $X_i \in R^d$ di mana X adalah vektor dari tiap baris data sampel, di mana $i = 1 \ldots . N$ dengan hasil dari $y_i = 1$ atau $y_i = 0$. Dari sampel data ini akan menghasilkan $y_i = 1$ masuk dalam *class* positif dan sampel yang menghasilkan $y_i = 0$ adalah masuk dalam *class* negatif. Tujuan dari klasifikasi seluruh data sampel X_i adalah nilai positif atau negatif. Data sampel dapat menggunakan metode percobaan Bernoulli dengan harapan hasil nilai $E(y_i)$ atau propabilitas p_i. Formula yang umum digunakan Logistic Regression untuk model masing-masing data sampel X_i menganut pada fungsi dari (Hosmer et al., 2013) sebagai berikut:

$$\boldsymbol{E[y_i|X_i|\beta] = P_i = \frac{e^{X_i\beta}}{1+e^{X_i\beta}}} \quad \ldots\ldots\ldots\ldots (2.1)$$

Dimana $\boldsymbol{\beta}$ adalah parameter vektor dengan asumsi bahwa $\boldsymbol{X_{i0} = 1}$ sehingga $\boldsymbol{\beta_0}$ adalah variable konstan dengan nilai 1.

Transformasi dari logistic regression adalah logit yang mana menghasilkan formulasi untuk mendapatkan nilai odds dengan variabel respon adalah positif dapat dilihat pada Formula 2.2 sebagai berikut:

$$n_i = ln\left(\frac{p_i}{1-p_i}\right) = x_i\beta \quad \ldots\ldots\ldots\ldots (2.2)$$

Dalam bentuk matrik, formula logit dapat di rumuskan menjadi:

$$n = X\beta \quad \ldots\ldots\ldots\ldots (2.3)$$

Sekarang dengan asumisi bahwa pengamatan terhadap data sampel menggunakan variabel independen, maka fungsi *likelihood* dapat dituliskan sebagai berikut:

$$L(\beta) = \prod_{i=1}^{l}(p_i)^{y_i}(1-p_i)^{1-y_i} = \prod_{i=1}^{l}\left(\frac{e^{x_i\beta}}{1+e^{x_i\beta}}\right)^{y_i}\left(\frac{1}{1+e^{x_i\beta}}\right)^{1-y_i} \quad \text{..... (2.4)}$$

Selanjutnya menggunakan formula dari (Hosmer et al., 2013) untuk mendapatkan *log likelihood* dituliskan sebagai berikut:

$$\log L(\beta) = \sum_{i=1}^{l}\left(y_i log\left(\frac{e^{x_i\beta}}{1+e^{x_i\beta}}\right) + (1-y_i)\log\left(\frac{1}{1+e^{x_i\beta}}\right)\right) - \frac{\lambda}{2}||\beta||^2 \quad \text{..... (2.5)}$$

$$= -\sum_{i=1}^{l} log\left(e^{-y_i x_i\beta}(1+e^{x_i\beta})\right) - \frac{\lambda}{2}||\beta||^2 \quad \text{..... (2.6)}$$

Dimana terdapat ketentuan $\frac{\lambda}{2}||\beta||^2$ yang ditambahkan untuk mendapatkan generalisasi yang lebih baik. Sejak diberlakukan pengetatan formula dalam menentukan *log likelihood,* formula ini secara objective digunakan untuk mencari *Maximum Likelihood Estimation* (MLE), β, yang memaksimalkan *log likelihood.* Untuk hasil biner, fungsi yang hilang atau penyimpangan dari *log likelihood* digunakan fungsi dari (Hosmer et al., 2013) yaitu:

$$DEV\ (\beta) = -2lnL(\beta) \quad \text{..... (2.7)}$$

Meminimalkan penyimpangan $DEV\ (\beta)$ seperti yang dilakukan oleh Busser at al.,1999 dalam (Maalouf & Trafalis, 2011) sama dengan memaksimalkan *log-likelihood* oleh (King & Zeng, 2001). Penyimpangan fungsi terhadap data nonliner dalam β. Minimalisasi dibutuhkan pada data numerik untuk mencari hasil *Maximum Likelihood Estimate* (MLE) dari β.

Bila terdapat populasi dari variabel **y** yang menglami kelangkaan *(rare event),* maka dilakukan proses seleksi secara acak pada variabel **y** untuk mendapatkan variabel yang signifikan (King & Zeng, 2001). Beberapa keuntungan terkait dengan pemilihan variabel respon secara acak. Pertama, tidak

perlu melakukan survey ulang, penghematan biaya dan menghemat waktu dengan hanya melakukan sampling terhadap data yang sudah ada secara acak. Kedua, penghematan secara efektif dan penggunaan komputasi yang efesien dapat dicapai karena tidak perlu menganalisa kembali dataset yang besar. Terakhir, model Logistic Regression dapat diperkaya dengan membuat proporsi data peristiwa dan non peristiwa yang seimbang.

2.2.4 Teknik *Resampling*

Teknik resampling didasarkan pada perulangan sample dalam kumpulan data yang sama (Yu, 2010). *Resampling* adalah teknik yang digunakan dalam konsep statistik inferensial. Teknik ini digunakan untuk mengambil sejumlah data sample dari data asli untuk mencapai perkiraan distribusi teoritis yang mendasari.

Resampling sangat penting bagi penelitian dalam bidang *software engineering*, sejak dataset *software engineering* dalam keadaan langka dan datanya terbatas. Kesulitan dalam mendapatkan dataset yang besar karena datanya rahasia atau datanya belum lengkap, maka teknik *resampling* adalah solusinya (Afzal & Torkar, 2008). Sehingga dataset *software enggineering* yang langka dapat dipergunakan dalam berbagai model penelitian.

Dalam *software engineering* masalah yang dihadapi adalah ketidakseimbangan kelas (*class imbalance*) dari dataset yang dimiliki. Dengan meningkatkan kelas minoritas dapat meningkatkan kemampuan algoritma *machine 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 distribusi kelas minoritas sehingga terwakili ketika dilakukan proses training data pada algorita *machine learning*. Metode *resampling* sudah terkenal diterapkan untuk memecahkan masalah ketidakseimbangan kelas (*class imbalace*) (Thanathamathee & Lursinsap, 2013).

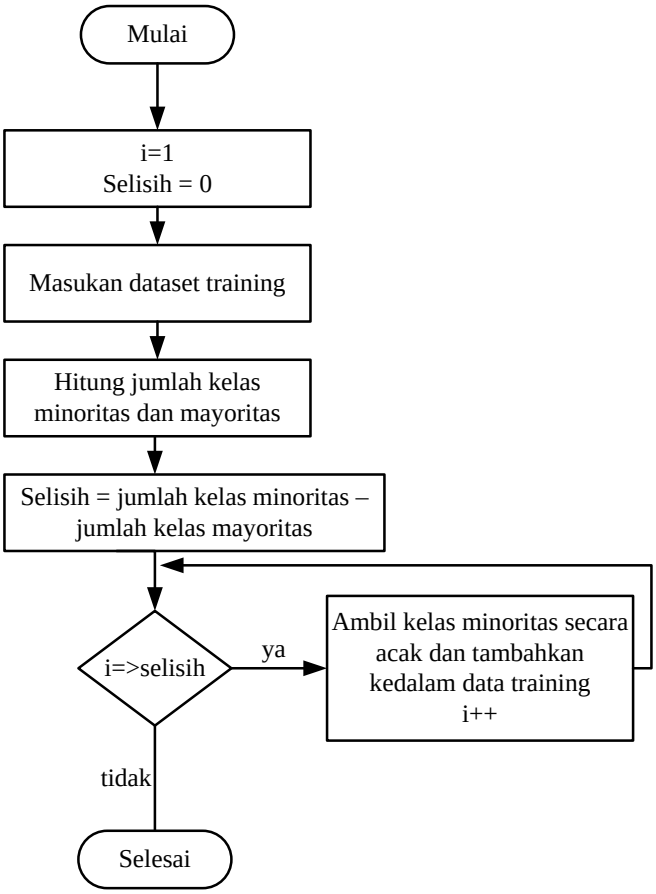
Dengan menerapkan teknik *over sampling, under sampling* dan kombinasi keduanya akan didapatkan kelas data yang seimbang.

2.2.4.1 Teknik *Oversampling*

Teknik *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 penyeimbangan kelas secara acak (Ganganwar, 2012). Namun, karena metode ini menduplikasi kelas positif yang ada di kelas minoritas, kemungkinan terjadi *overfitting* pasti akan terjadi.

Teknik *oversampling* dapat dicontohkan menggunakan data sampel, misalanya terdapat 2.200 data sampel pada data latih, dengan distribusi kelas minoritas sebanyak 200 sampel dan kelas mayoritas terdiri dari 2.000 sampel. Dengan menggunakan teknik *oversampling* maka kelas minoritas diambil secara acak sampai ukuran sampel minoritas mencapai 2.000 sampel, sehingga jumlah data menjadi 4.000 sample dengan 2.000 kelas positif dan 2.000 kelas negatif. Pada Gambar 2.6 menunjukan *flowchart* teknik *oversampling* untuk mendapatkan kelas data yang seimbang.

Proses yang dijalankan pada teknik *oversampling* seperti di tunjukan pada *flowchart* Gambar 2.6 dimulai dengan menginisiasi variabel i =1 dan selisih=0. Kemudian dataset training akan dibaca dan dilakukan perhitungan jumlah kelas minoritas dan mayoritas. Nilai variabel selisih akan diubah menjadi hasil perhitungan jumlah kelas minoritas dikurangi kelas mayoritas. Selanjutnya akan dilakukan pengecekan jika nilai i=> selisih maka dilakukan proses pengambilan nilai minoritas secara acak dan ditambahkan ke dalam dataset training, dan nilai i akan ditambah 1. Proses akan berulang sampai nilai i lebih besar dari selisih. Sehingga akan didapatkan data training yang seimbang antara kelas minoritas dan mayoritas.



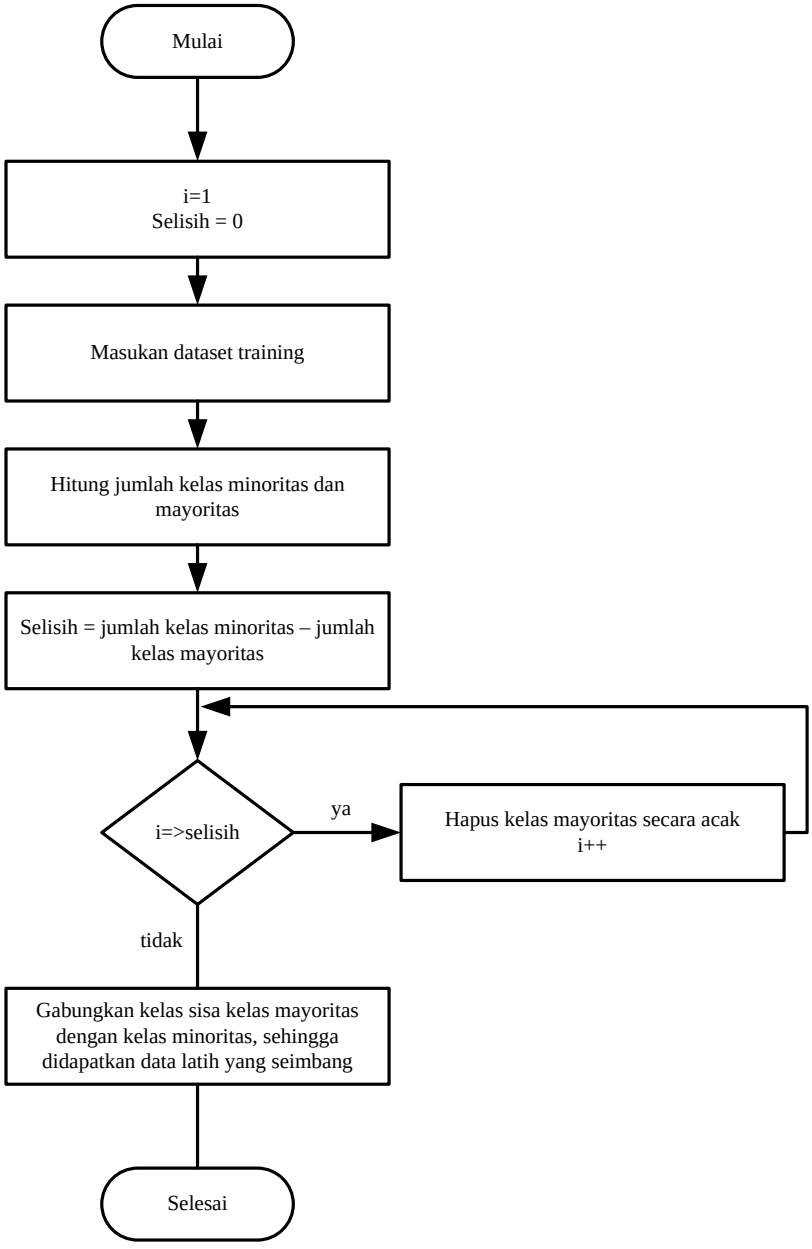
Gambar 2.6 Flowchart Tiknik *Oversamping*

2.2.4.2 Teknik *Undersampling*

Teknik *undersampling* hampir sama dengan teknik *oversampling* dengan menghitung selisih kelas mayoritas dan kelas minoritas. Selanjutnya dilakukan perulangan sebanyak selisih kelas mayoritas dengan kelas minoritas. Selama proses perulangan dilakukan penghapusan terhadap kelas mayoritas sehingga didapatkan jumlah yang sama dengan kelas minoritas. *Flowchart* dari teknik *undersampling* dapat dilihat pada Gambar 2.7.

Proses yang dijalankan pada teknik *undersampling* seperti di tunjukan pada *flowchart* Gambar 2.7 dimulai dengan menginisiasi variabel i=1 dan selisih=0. Kemudian dataset training akan dibaca dan dilakukan perhitungan jumlah kelas minoritas dan mayoritas. Nilai variabel selisih akan diubah menjadi hasil perhitungan jumlah kelas minoritas dikurangi kelas mayoritas. Selanjutnya akan dilakukan pengecekan jika nilai i=> selisih maka dilakukan proses penghapusan kelas mayoritas secara acak, dan nilai i akan ditambah 1. Proses

akan berulang sampai nilai i lebih besar dari selisih. Selanjutnya dilakukan penggabungan sisa kelas mayoritas dengan kelas minoritas. Sehingga akan didapatkan data training yang seimbang antara kelas minoritas dan mayoritas.

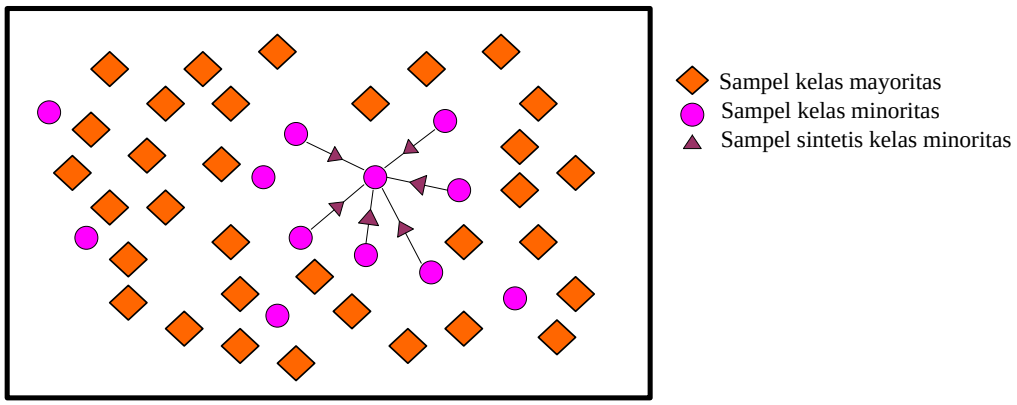


Gambar 2.7 Flowchart Teknik Undersampling

2.2.4.3 SMOTE (*Synthetic Minority Over-sampling Technique*)

Teknik menangani ketidakseimbangan kelas secara umum pada dataset menggunakan pendekatan *resampling*. Teknik *Synthetic Minority Over-sampling Technique (SMOTE)* merupakan pendekatan *oversampling* pada kelas minoritas

yaitu dengan melakukan *oversampling* untuk menciptakan sample sintetik bukan dengan *oversampling* dengan penggantian (Chawla et al., 2002). SMOTE bekerja dengan menciptakan *sample* kelas minoritas yang baru pada titik acak pada garis yang menghubungkan contoh kelas minoritas dengan kelas yang berdekatan dalam ditribusi data (Pelayo & Dick, 2007). Metode ini mensintetik sample kelas minoritas baru antara beberapa contoh minoritas yang terletak berdekatan, bukan hanya menduplikasi mereka secara *oversampling*. Keuntungan dari SMOTE adalah teknik SMOTE membuat daerah keputusan yang lebih besar dan kurang spesifik. Gambar 2.8 menunjukan hasil sampel sintetis kelas minoritas.



Gambar 2.8 Ilustrasi Sintetis Sample SMOTE

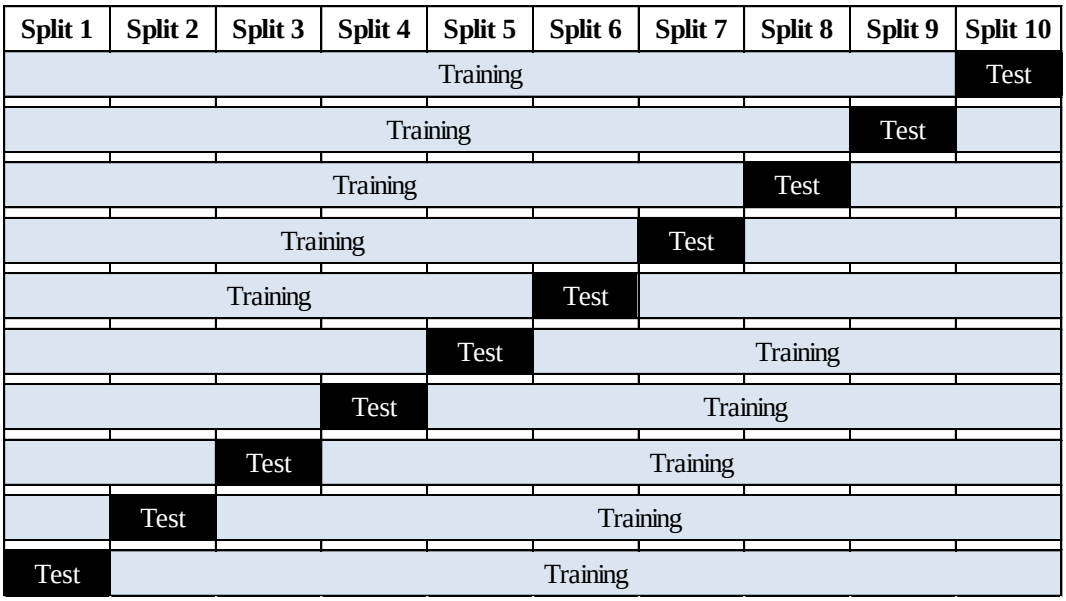
2.2.5 Teknik Validasi dan Evaluasi

Dalam pengujian dataset keseluruhan akurasi umumnya digunakan untuk mengevaluasi kinerja algoritma klasifikasi (Zhang & Wang, 2011). Namun untuk data yang tidak seimbang, kelas minoritas mendominasi akurasi yang mendalam, sehingga diperlukan alternatif matrik sebagai evaluasinya. Matrik evaluasi yang tepat termasuk *Area Under the ROC (Receiver Operating Characteristic) Curve (*AUC*), F-Measure, Geometric Mean (G-Mean),* semua akurasi dan rata-rata akurasi untuk kelas minoritas. Validasi terhadap model yang diusulkan diperlukan beberapa pengujian menggunakan *confusion matrix.*

2.2.5.1 K-*Fold Cross Validation*

Pengujian yang dilakukan dalam penelitian ini adalah menggunakan metode cross validation. *Cross validation* adalah metode statistik yang digunakan

untuk mengevaluasi dan membandingkan algoritma dengan membagi data menjadi dua segmen, segmen pertama digunakan sebagai data training dan segmen kedua sebagai data testing untuk memvalidasi model (Witten et al., 2011). Dalam cross validation segmen training dan segmen testing harus *crossover* sehingga setiap data memiliki kesempatan tervalidasi. Model *cross validation* yang terbagi dalam dua segmen dapat dilihat pada Gambar 2.9.



Gambar 2.9 10-Cross Fold Validation

2.2.5.2 *Area Under the ROC (Receiver Operating Characteristic) Curve*

Dalam penelitian ini diterapkan metode evaluasi 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 tarik-menarik antara sumbu Y dengan sumbu X

AUC dihitung berdasarkan rata-rata perkiraan bidang berbentuk trapesium untuk kurva yang dibuat oleh TP_{rate} dan FP_{rate} (Dubey, Zhou, Wang, Thompson, & Ye, 2014) dapat dirumuskan sebagai berikut:

$$AUC = \frac{1+TP_{rate}- FP_{rate}}{2} \quad \text{..... (2.19)}$$

2.2.5.3 Uji Siginfikan

Uji signifikansi dapat digunakan untuk membandingkan dua atau lebih model pengklasifikasi pada beberapa dataset. Evaluasi statistik hasil eksperimen dianggap sebagai bagian untuk validasi metode *machine lerning* (Demšar, 2006). Tingkat signifikansi dapat diperoleh dengan menggunakan metode parametrik dan non-parametrik. Sebagian besar prosedur analisis statistik didasarkan pada asumsi bahwa beberapa bentuk model linier menggambarkan perilaku respon interval atau rasio variabel (Freund, J, & L, 2003). Ini berarti perilaku variabel respon diperkirakan oleh model linear dan kesimpulan yang dibuat mengacu pada parameter model. Karena fokus utama adalah pada parameter, metode statistik berdasarkan model linear sering disebut metode parametrik.

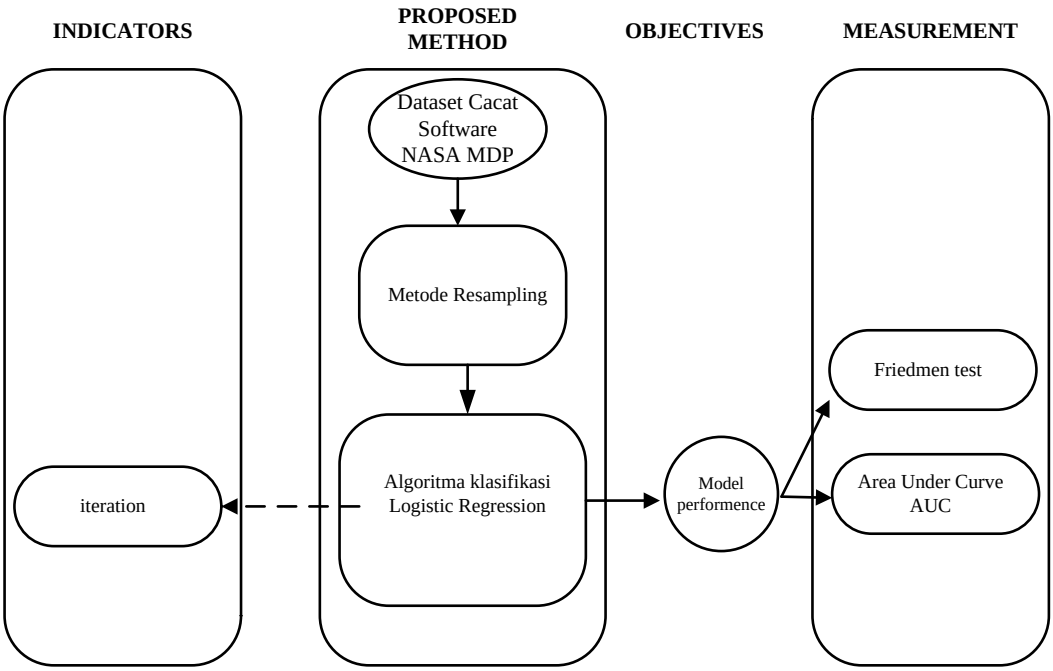
Uji *t* merupakan uji statistik untuk rata-rata dua populasi dari hasil eksperimen (Larose, 2005). Dua populasi *sample* uji *t* digunakan untuk memeriksa apakah dua sampel yang berbeda dan sering digunakan ketika varians dari dua distribusi normal tidak diketahui dan ketika percobaan menggunakan ukuran sampel yang kecil. Rumus dari uji *t* dapat dilihat pada persamaan 2.20.

$$t = \frac{x_1 - x_2}{\sqrt{\frac{var_1}{n_1} + \frac{var_2}{n_2}}} \quad \text{.....(2.20)}$$

2.3 Kerangka Pemikiran Penelitian

Penelitian ini adalah mengenai prediksi cacat *software* menggunakan *Logistic Regression* dengan dataset NASA MDP. Dataset akan diproses (*training*) dengan metode *resampling* untuk menangani ketidakseimbangan kelas (*class imbalance*). Dataset yang sudah seimbang akan diproses oleh model *Logistic Regression* dalam ***proposes method***. Dengan parameter (***indicator***) *iterations* yang diubah pada *Logistic Regression*. Tujuan (***objectives***) penelitian ini adalah untuk

mendapatkan klasifikasi cacat atau tidak cacat pada prediksi cacat *software*. Untuk mengevaluasi hasil klasifikasi, digunakan pengukuran (***measurement***) dengan Area Under the ROC (*Receiver Operating Characteristic*) Curve (AUC) dan *Friedmen test*. Kerangka pemikiran penelitian ini dapat dilihat pada Gambar 2.10.



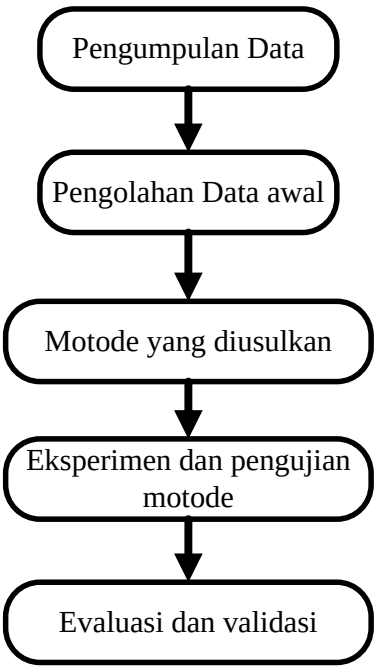
Gambar 2.10 Kerangka Pemikiran Penelitian

BAB III
METODE PENELITIAN

3.1 Perancangan Penelitian

Penelitian merupakan kegiatan yang bertujuan untuk membuat kontribusi orisinil terhadap ilmu pengetahuan (Dawson, 2009). Menurut (Dawson, 2009) terdapat empat metode penelitian yang umum digunakan, yaitu: penelitian langsung, eksperimen, study kasus dan survey. Penelitian ini menggunakan metode eksperimen, yaitu penelitian yang melibatkan penyelidikan pada perameter, variabel atau perbaikan metode tergantung dari penelitinya dan menggunakan data tes yang dikendalikan oleh peneliti itu sendiri.

Dalam penelitian ini menggunakan tahapan-tahapan penelitian yang dilakukan seperti pada Gambar 3.1.



Gambar 3.1 Metode Penelitian

Metode penelitian ini dibagi dalam lima tahapan sebagai berikut:

1. Pengumpulan data *(data gathering)*

   Pada tahapan ini dijelaskan tentang bagaimana serta darimana data untuk penelitian ini dikumpulkan, mengolah data yang sudah dikumpulkan agar dapat dipergunakan dalam ekseperimen penelitian ini.

2. Pengolahan data awal (*data pre-processing*)

   Pada tahapan ini dijelaskan bagaimana pengolahan data untuk mendapatkan data yang sudah ditransformasikan, agar sesuai bentuk yang diinginkan pada metode yang diusulkan.
3. Metode yang diusulkan (*Proposed Method/Model*)

   Data yang diteliti dan dianalisa kemudian dikelompokan ke dalam data latihan (training) dan data pengujian (testing) sesuai dengan model yang diperlukan.
4. Eksperimen dan pengujian model *(Method Test and Experiment)*

   Pengujian model yang diusulkan pada model yang akan diuji untuk melihat seberapa signifikan hasil yang diperoleh, untuk menentukan pengambilan keputusan dari peneliti.
5. Evaluasi dan Validasi Hasil *(Result Evalutation and Validation)*

   Pada sebuah penelitian dilakukan evaluasi terhadap metode yang diusulkan untuk mengetahui hasil terbaik dari metode yang digunakan.

3.2 Pengumpulan Data

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.

Dalam penelitian ini menggunakan dataset yang sudah diperbaiki yaitu: CM1, JM1, KC1, KC3, MC1, MC2, PC1, PC2, PC3, PC4 dan PC5. Spesifikasi dan attribut data NASA MDP *repository* ditunjukan pada Tabel 3.1. Spesifikasi dari attribut dataset NASA dibagi dalam 4 bagian yaitu: LOC count, attribut halstead, attribut McCabe, dan attribut miscellaneous. LOC count adalah jumlah baris kode dan komentar dari program. Attribut halsted adalah perhitungan operan

dan operator dalam program. Attribut McCabe adalah kompleksitas *cyclomatic* dalam program.

Tabel 3.1 Spesifikasi dan Atribut 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

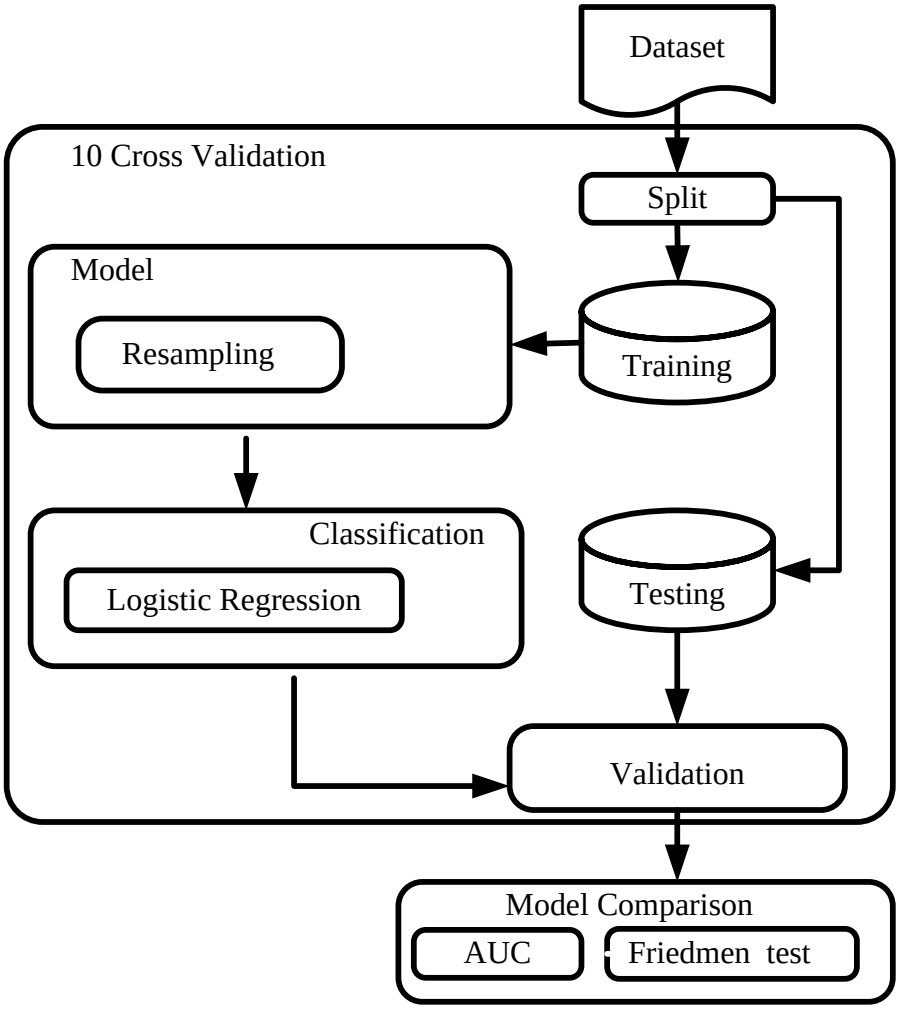
3.3 Pengolahan Data Awal

Dalam penelitian ini menggunakan 10 dateset dari NASA MDP yang sudah dilakukan pembersihan data yang tidak digunakan oleh Martin Shepperd (Liebchen & Shepperd, 2008). Dataset sudah bersih dari kegaduhan data (*noise*), sehingga dapat langsung digunakan dalam proses prediksi cacat *software*.

Dalam penelitian ini akan menghasilkan angka yang dihasilkan dari proses klasifikasi. Hasil angka tersebut akan diukur dan dibandingkan untuk mendapatkan model yang tepat. Hasil performance dengan level tertinggi merupakan hasil dari prediksi cacat *software*.

3.4 Metode Yang Diusulkan

Metode yang diusulkan yaitu penerapan *resampling logistic regression* untuk prediksi cacat *software*. Dimulai dari pembagian dataset dengan metode 10 *cross validation* yaitu data training dan data testing, kemudian data trainig diproses dengan metode *resampling* logistic regression, sehingga menghasilkan model evaluasi yang diukur dengan Friedmen test dan *Area Under Curve* (AUC), dapat dilihat pada Gambar 3.2.

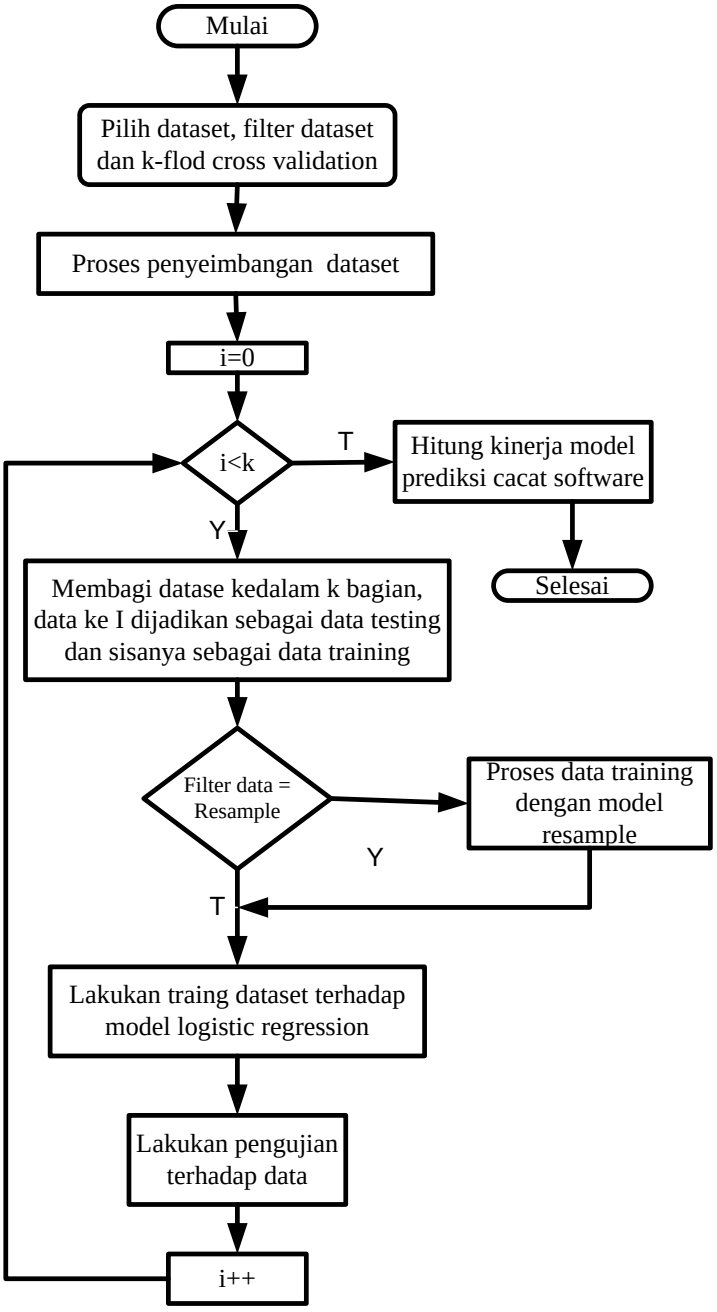


Gambar 3.2 Metode yang Diusulkan

Dapat dilihat pada Gambar 3.2 metode yang diusulkan pada penelitian ini. Dataset dibagi menjadi X sesuai dengan nilai validasi (*X-Fold Cross Validation*), dalam 1 bagian 1/X digunakan sebagai data uji (*testing*), dan sisanya digunakan sebagai data latih (*training*). Selanjutnya data dilatih (*training*) diproses dalam model dan menerapkan metode resampling. Hasil dari proses pelatihan selanjutnya digunakan sebagai evaluasi model. Kemudian sebagai komparasi dari model yang dilakukan, dataset uji diproses ke dalam model untuk mendapatkan hasil kinerja model algoritma. Hasil validasi dari proses digunakan untuk mengukur kinerja algoritma terhadap model yang diusulkan.

Flowchart metode yang diusulkan dapat dilihat pada Gambar 3.3, 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 prdiksi cacat *software*.

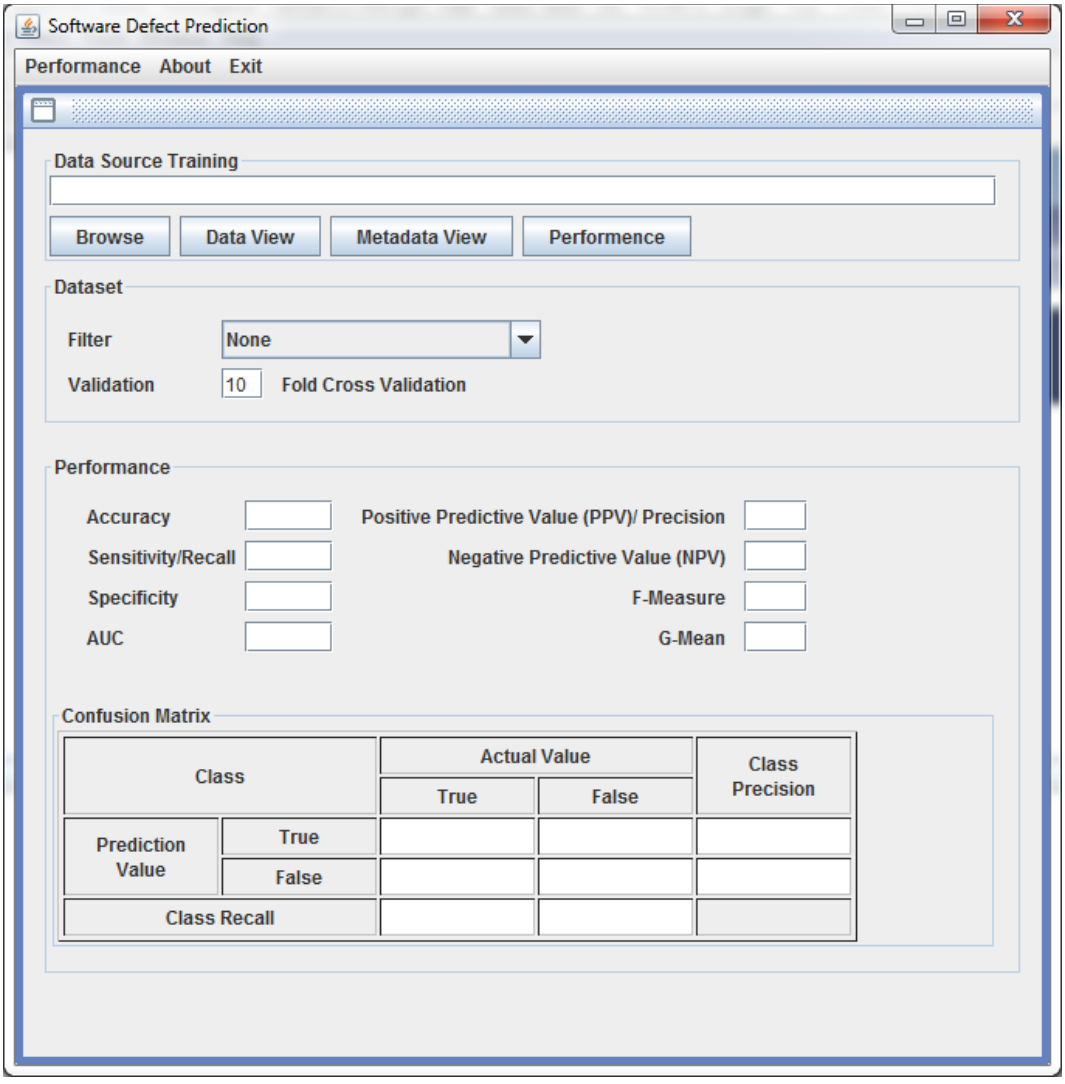


Gambar 3.3 Flowchart Metode yang Diusulkan

3.5 Eksperimen dan Pengujian Metode

Proses eksperimen dan pengujian metode dalam penelitian ini menggunakan antarmuka pengguna (*user interface*) dari aplikasi yang telah dikembangkan untuk mengukur kinerja model yang diusulkan, ditunjukkan pada

Gambar 3.4. Di bagian atas ada beberapa tombol yaitu: 1) *Browse* untuk memilih file pelatihan (dataset), 2) *Data View* untuk menampilkan isi dari dataser, 3) *Metadata View* digunakan untuk menampilkan metadata dari dataset, 4) *Performance* digunakan untuk menampilkan kinerja dari model yang diusulkan. Dibawahnya ada pilihan pengolahan data dan inputan *Cross Validations.* Pada bagian bawah menampilkan hasil perhitungan akurasi, *recall*, AUC, F-Measure dan G-Mean serta pengukuran kinerja dari model yang diusulkan dengan tabel *Confusion Matrix*.



Gambar 3.4 Program Aplikasi Prediksi Cacat Software

Tahapan eksperimen pada penelitian ini adalah:

1. Menyiapkan dataset untuk eksperiment yang sudah di download.

2. Memilih parameter filter dataset yang akan diuji.
3. Melakukan training dan testing pada dataset terhadap model Logistic Regression dan mencatat hasil perhitungan akurasi, AUC, F-Meansure, G-Mean, PPV dan NPV.
4. Melakukan training dan testing pada dataset terhadap model resampling Logistic Regression dan mencatat hasil perhitungan akurasi, AUC, F-Meansure, G-Mean, PPV dan NPV.
5. Melakukan komparasi hasil AUC pada model dan menguji beda dengan t-Test.

Dalam penelitian yang dilakukan ini adalah menggunakan komputer untuk melakukan proses perhitungan terhadap model yang diusulkan. Penggunaan komputer dengan spesifikasi komputer pada Tabel 3.2.

Tabel 3.2 Spesifikasi Komputer

Processor	Intel Core i3, 1.8 GHz
Memory	2 GB
Hardisk	500 GB
Sistem Operasi	Windows 7
Aplikasi	Java

3.6 Evaluasi dan Validasi

Tabel *confusion matrix* digunakan untuk melakukan pengukuran kinerja model. Kinerja yang diukur termasuk akurasi secara umum, akurasi dalam memprediksi kelas monoritas, dan *Area Under the ROC Curve* (AUC). *Confusion matrix* diperoleh dari proses validasi menggunakan 10-*fold cross validation*, sehingga model yang terbentuk dapat langsung diuji dengan melakukan 10 kali pengujian.

Kinerja model yang diperoleh digunakan untuk membandingkan antara model dasar algoritma *Logistic Regression* dengan model yang dibentuk menggunakan *Resampling Logistic Regression*. Untuk melihat kualitas model yang didapatkan, nilai AUC dapat dijadikan ukuran untuk melihat model yang

terbentuk. Kurva ROC dapat digunakan untuk mendapatkan nilai AUC, dimana nilai AUC digunakan untuk menentukan klasifikasi pengujian diagnostik.

Pedoman umum untuk mengklasifikasikan keakuratan pengujian diagnostik menggunakan AUC dapat dilihat pada sistem tradisional yang dijabarkan oleh Gorunescu (Wahono, Herman, & Ahmad, 2011), disajikan pada Tabel 3.3. Jika diperoleh nilai AUC kurang dari 0.60 diklasifikasikan gagal (*failure*) dan ditunjukan dengan simbol anak panak ke bawah. Jika nilai AUC diantara 0.60 sampai 0.70 maka klasifikasi jelek (p*oor classification*) dan ditunjukan dengan simbol anak panah ke bawah kanan. Jika nilai AUC diantara 0.70 sampai 0.80 maka termasuk dalam klasifikasi sedang dan ditunjukan dengan simbol anak panah mendatar. Jika nilai AUC diantara 0.80 sampain 0.90 maka termasuk dalam klasifikasi bagus dan ditunjukan dengan simbol anak panak keatas kanan. Dan terakhir jika nilai AUC antara 0.90 sampai 1.00 maka termasuk dalam klasifikasi yang sangat bagus dan ditunjukan dengan simbol anak panah keatas.

Tabel 3.3 Nilai AUC, Klasifikasi dan Simbol Pengujian Diagnostik

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*	↓

3.7 Teknik Analisa

Teknik analisa dilakukan dengan menggunakan uji statisik. Uji statistik digunakan untuk membandingkan kinerja model *Logistic Regression* dengan penangganan ketidakseimbangan kelas menggunakan *resampling*. Analisa ini bertujuan untuk mengetahui kinerja terbaik algoritma *Logistic Regression* dengan penanganan ketidakseimbangan kelas menggunakan *resampling*, sehingga keakuratan prediksi kelas minoritas dapat ditingkatkan. Pengukuran analisa kinerja adalah akurasi, sensitivitas, F-Measure, G-Mean dan AUC. Untuk mempermudah dalam melakukan analisa dipergunakan software Microsoft Excel

yang menerapkan plugin *xlstat.* Uji statistik yang dipergunakan adalah uji *t* sample berpasangan (*paired-sample t-test*) satu sisi untuk sisi bawah dan uji Friedman (*Friedman test*).

3.7.1 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. Terdapat tiga bentuk uji *t* sample berpasangan, yang mana dalam penggunaannya bergantung pada persoalan yang akan diuji. Ketiga bentuk uji *t* sampel tersebut adalah:

1. Uji *t* sampel berpasangan (*paired sample t test*) satu sisi (*one-tailed*) untuk sisi bawah menggunakan hipotesis:

   $H_0 = \mu_1 \geq \mu_2$ atau $H_0 = \mu_D \geq 0$

   $H_1 = \mu_1 < \mu_2$ atau $H_1 = \mu_D < 0$

   Di mana $\mu_D = \mu_1 - \mu_2$

2. Uji *t* sampel berpasangan (*paired-sample t-test*) satu sisi (*one-tailed*) untuk sisi atas menggunakan hipotesis:

   $H_0 = \mu_1 \leq \mu_2$ atau $H_0 = \mu_D \leq 0$

   $H_1 = \mu_1 > \mu_2$ atau $H_1 = \mu_D > 0$

   Di mana $\mu_D = \mu_1 - \mu_2$

3. Uji t sampel berpasangan (*paired-saple t-test*) dua sisi (*two-tailed*) menggunakan hipotesis:

   $H_0 : \mu_1 = \mu_2$ atau $H_0 : \mu_D = 0$

   $H_1 : \mu_1 \neq \mu_2$ atau $H_1 : \mu_D \neq 0$

   Di mana $\mu_D = \mu_1 - \mu_2$

Penentuan nilai $\mu_1$dan μ_2 tidak ada aturan, sehingga bebas untuk menetapkannya. Sedangan hipotesis mana yang diterima, ditentukan berdasarkan *P-value* pada aplikasi Microsoft Excel 2007 plugin xlstat digunakan istilah P(T<=t) *two-tail*. Ketentuan untuk menolak atau menerima hipotesis adalah:

a. Jika *P-value* < α, maka H_0 ditolak dan H_1 diterima

b. Jika *P-value* $\geq \alpha$, maka H_0 diterima dan H_1 ditolak

H_0 adalah hipotesis untuk mengguji sebuah pernyataan diterima atau ditolak dari sebuah pengujian. Sedangkan H_1 atau H alternatif adalah hipotesis alternatif dari pernyataan dari hipotesis nol.

3.7.2 Uji Friedman (*Friedman Test*)

Uji Friedman di usulkan oleh Demsar untuk membandingkan model klasifikasi (Demšar, 2006). Uji Friedman (*Friedman test*) merupakan uji statistik nonparametrik, 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 performance terhadap dataset dan peringkat rata-rata dari model klasifikasi yang dibandingkan.

Hipotesa pada uji Friedmen menggunakan model sebagai berikut:

$H_0: \eta_1 = \eta_2 = \eta_3 = \cdots = \eta_n$

$H_1: tidak\ semua\ median\ (\eta_i, i = 1 \cdots n) bernilai\ sama\ besar$

η dibaca ’eta’ merupakan simbol median.

Ketentuan untuk menerima atau menolak hipotesis berdasarkan pada nilai *P-value* pada aplikasi Microsoft Excel 2007 plugin xlstat digunakan istilah P(T<=t) *two-tail*. Jika *P-value* < α, maka H_0 akan ditolak dan H_1 akan diterima. Sehingga jika *P-value* $\geq \alpha$, maka H_0 akan diterima adan H_1 akan ditolak.

BAB IV
HASIL PENELITIAN DAN PEMBAHASAN

4.1 Hasil

Pada sub bab hasil akan dijabarkan hasil dari pengembangan aplikasi dan hasil dari pengukuran kinerja model. Pada pengembangan aplikasi akan dibahas dalam pengujian aplikasi untuk menunjukan bahwa hasil dari aplikasi yang dibuat sudah sesuai dengan yang diharapkan. Sedangkan pada pengukuran kinerja model akan dijabarkan hasil pengukuran kinerja model untuk prediksi cacat *software* mengunakan dataset NASA MDP pada aplikasi yang dikembangkan.

4.1.1 Hasil Pengukuran Manual Logistic Regression

Pada penelitian ini akan dilakukan perhitungan manual dengan menggunakan sampel dataset yang diambil dari dataset CM1, dapat dilihat pada Tabel 4.1.

Tabel 4.1 Dataset Uji Aplikasi

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
6	9	2	1	0	16	5	0.2	N
15	7	3	1	19	12	4	0.13	Y
27	9	1	4	22	16	5	0.15	Y
7	3	2	0	0	4	2	0.17	N
51	25	13	0	14	48	13	0.12	N
3	5	2	0	0	8	3	0.2	N
13	9	5	12	16	16	5	0.14	N
22	29	0	8	35	44	15	0.28	N
16	9	2	11	28	12	5	0.11	N
4	3	0	2	4	4	2	0.17	Y
23	13	0	10	17	24	7	0.21	N
23	13	0	10	17	24	7	0.21	N
40	9	4	2	15	16	5	0.15	N
23	9	6	3	20	16	5	0.19	N
8	3	1	6	5	4	2	0.11	N
19	3	1	2	0	4	2	0.06	Y
6	7	0	6	0	10	4	0.15	N
134	60	9	63	31	38	41	0.17	N
9	8	2	4	5	8	5	0.28	N
164	110	26	37	191	128	70	0.18	N

Teknik validasi digunakan pada perhitungan ini adalah 10-*fold cross validation*, sehingga dataset dibagi terlebih dahulu menjadi 10 bagian. Kemudian diambil satu bagian sebagai data uji dan sisanya digunakan sebagai data latih. Pengambilan data uji dan data latih dilakukan secara berurutan sampai perulangan selesai. Tabel 4.2 menunjukan data uji yang dilakukan pembagian sesusai dengan teknik validasi.

Tabel 4.2 Pembagian Dataset Berdasarkan *Fold Coss Validation*

SPLIT	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
1	6	9	2	1	0	16	5	0.2	N
	15	7	3	1	19	12	4	0.13	Y
2	27	9	1	4	22	16	5	0.15	Y
	7	3	2	0	0	4	2	0.17	N
3	51	25	13	0	14	48	13	0.12	N
	3	5	2	0	0	8	3	0.2	N
4	13	9	5	12	16	16	5	0.14	N
	22	29	0	8	35	44	15	0.28	N
5	16	9	2	11	28	12	5	0.11	N
	4	3	0	2	4	4	2	0.17	Y
6	23	13	0	10	17	24	7	0.21	N
	23	13	0	10	17	24	7	0.21	N
7	40	9	4	2	15	16	5	0.15	N
	23	9	6	3	20	16	5	0.19	N
8	8	3	1	6	5	4	2	0.11	N
	19	3	1	2	0	4	2	0.06	Y
9	6	7	0	6	0	10	4	0.15	N
	134	60	9	63	31	38	41	0.17	N
10	9	8	2	4	5	8	5	0.28	N
	164	110	26	37	191	128	70	0.18	N

Rumus prediksi (logistic regression) cacat *software* seperti pada persamaan 4.1.

$$\boldsymbol{E[y_i|X_i|\beta]} = \boldsymbol{P_i} = \frac{\boldsymbol{e^{X_i\beta}}}{\boldsymbol{1+e^{X_i\beta}}} \quad \text{.....(4.1)}$$

Dimana β adalah nilai *Maximum Loglikehood* dari persamaan 4.2.

$$L(\beta) = \prod_{i=1}^{l}(p_i)^{y_i}(1-p_i)^{1-y_i} = \prod_{i=1}^{l}\left(\frac{e^{x_i\beta}}{1+e^{x_i\beta}}\right)^{y_i}\left(\frac{1}{1+e^{x_i\beta}}\right)^{1-y_i} \text{.....(4.2)}$$

Penentuan nilai β didapatkan dari hasil komputasi menggunakan kode program java, kode program dapat dilihat pada lampiran yang disertakan.

Setiap bagian pada *fold cross validation* harus dilakukan pelatihan dan pengujian sesuai pembagian dataset. Di bawah ini merupakan tahapan proses perhitungan pada setiap tahapan.

1. Pada bagian ke-1 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      β_1= 0.475421895, β_2= -0.118603303, β_3= -4.388949785,
      β_4= -0.893487066, β_5= 0.287683428, β_6= -0.808008494,
      β_7= 1.367513407, β_8=0.782747922
   B. Berdasarkan pembagian *fold cross validation*, maka data bagian pertama dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.3.

Tabel 4.3 Rekap Perhitungan β Bagian ke-1

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
2.852531	-1.06743	-8.7779	-0.89349	0	-12.9281	6.837567	0.1565496
7.131328	-0.83022	-13.1668	-0.89349	5.465985	-9.6961	5.470054	0.1017572

Berdasarkan pada hasil perhitungan Tabel 4.3, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(2.852531+-1.06743+-8.7779+-0.89349+0+-
12.9281+6.837567+0.1565496)))
= 9.9521676860836E-07

Hasil probabilitas pada baris pertama sebesar (9.9521676860836E-07) atau sebesar 0.0001%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.

Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P= 1/(1+exp(-(7.131328+-0.83022
+-13.1668+-0.89349+5.465985+-9.6961+5.470054+0.1017572)))
= 0.00163001121232182

Hasil probabilitas pada baris kedua sebesar (0.00163001121232182) atau sebesar 0.163%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.

2. Pada bagian ke-2 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      β_1= 0.205921286, β_2= -1.073441093, β_3= -1.950427372,
      β_4= -0.783667283, β_5= 0.362856177, β_6= -0.273327328,
      β_7= 1.792202904, β_8=1.072879691
   B. Berdasarkan pembagian *fold cross validation*, maka data bagian kedua dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.4.

Tabel 4.4 Rekap Perhitungan β Bagian ke-2

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
5.559875	-9.66097	-1.95043	-3.13467	7.982836	-4.37324	8.961015	0.160932
1.441449	-3.22032	-3.90085	0	0	-1.09331	3.584406	0.1823895

Berdasarkan pada hasil perhitungan Tabel 4.4, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(5.559875+-9.66097+-1.95043+-3.13467+7.982836+-4.37324+8.961015+0.160932)))

= 0.971951029751943

Hasil probabilitas pada baris pertama sebesar (0.971951029751943) atau sebesar 97.1951%, karena nilai persentase probabilitas lebih dari nol maka diklasifikasikan sebagai rawan cacat.

Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P= 1/(1+exp(-(1.441449002+-3.22032328 -3.900854743+0+0+-1.093309312+3.584405809+0.182389547)))

= 0.0471446319045691

Hasil probabilitas pada baris kedua sebesar (0.0471446319045691) atau sebesar 4.71446%, karena nilai persentase probabilitas lebih dari nol dan kurang dari 5% maka diklasifikasikan sebagai tidak rawan cacat.

3. Pada bagian ke-3 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.

   A. Nilai β yang di hasilkan.

      β_1= 0.260405386, β_2= -1.129958332, β_3= -2.497400134,
      β_4= -0.908342359, β_5= 0.454888781, β_6= -0.343603683,
      β_7= 1.899506578, β_8=1.032564705

   B. Berdasarkan pembagian *fold cross validation*, maka data bagian ketiga dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.5.

Tabel 4.5 Rekap Perhitungan *β* Bagian ke-3

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
13.28067	-28.249	-32.4662	0	6.368443	-16.493	24.69359	0.1239078
0.781216	-5.64979	-4.9948	0	0	-2.74883	5.69852	0.2065129

Berdasarkan pada hasil perhitungan Tabel 4.5, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(13.2806747+-28.24895831+-
32.46620175+0+6.36844294+-
16.49297678+24.69358552+0.123907765)))
= 6.0330366572347E-15

Hasil probabilitas pada baris pertama sebesar (6.0330366572347E-15) atau sebesar 0.00000%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.

Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P= 1/(1+exp(-(0.781216159-5.649791661+
-4.994800269+0+0+-2.748829464+5.698519734+ 0.206512941)))
= 0.001220623

Hasil probabilitas pada baris kedua sebesar (0.001220623) atau sebesar 0.1221%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.

4. Pada bagian ke-4 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      $\beta_1 = 0.261992794$, $\beta_2 = -0.939432531$, $\beta_3 = -2.657126932$,
      $\beta_4 = -0.955540427$, $\beta_5 = 0.453050077$, $\beta_6 = -0.334683914$,
      $\beta_7 = 1.67034747$, $\beta_8 = 1.012973586$
   B. Berdasarkan pembagian *fold cross validation*, maka data bagian keempat dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.6.

Tabel 4.6 Rekap Perhitungan *β* Bagian ke-4

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
3.405906	-8.45489	-13.2856	-11.4665	7.248801	-5.35494	8.351737	0.1418163
5.763841	-27.2435	0	-7.64432	15.85675	-14.7261	25.05521	0.2836326

Berdasarkan pada hasil perhitungan Tabel 4.6, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(3.405906317+-8.454892781+-13.28563466+-
11.46648513+7.248801226+-
5.354942626+8.351737351+0.141816302)))
= 3.70458719075371E-09

Hasil probabilitas pada baris pertama sebesar (3.70458719075371E-09) atau sebesar 0.00000%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.
Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(5.763841459+-27.2435434+0+-7.644323419+15.85675268+-
14.72609222+25.05521205+0.283632604)))
= 0.065710952500683

Hasil probabilitas pada baris kedua sebesar (0.065710952500683) atau sebesar 6.5710%, karena nilai persentase probabilitas lebih dari nol dan lebih dari 5% maka diklasifikasikan sebagai rawan cacat.

5. Pada bagian ke-5 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      β_1= 0.694324313, β_2= -0.477042223, β_3= -1.706444628,
      β_4= -0.58066434, β_5=1.453810389, β_6= -2.661278555,

β_7= 0.893174513, β_8=0.1039067

B. Berdasarkan pembagian *fold cross validation*, maka data bagian kelima dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.7.

Tabel 4.7 Rekap Perhitungan *β* Bagian ke-5

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
11.10919	-4.29338	-3.41289	-6.38731	40.70669	-31.9353	4.465873	0.0114297
2.777297	-1.43113	0	-1.16133	5.815242	-10.6451	1.786349	0.0176641

Berdasarkan pada hasil perhitungan Tabel 4.7, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(11.109189+-4.293380003+-3.412889256+-
6.387307743+40.7066909+-
31.93534266+4.465872566+0.011429737)))
= 0.999965144422656

Hasil probabilitas pada baris pertama sebesar (0.999965144422656) atau sebesar 99,9965%, karena nilai persentase probabilitas lebih dari nol maka diklasifikasikan sebagai rawan cacat.

Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(2.777297251+-1.431126668+0+-1.161328681+5.815241557+-10.64511422+1.786349026+0.017664139)))
= 0.0551474906516793

Hasil probabilitas pada baris kedua sebesar (0.0551474906516793) atau sebesar 5.51474%, karena nilai persentase probabilitas lebih dari nol dan lebih dari 5% maka diklasifikasikan sebagai rawan cacat.

6. Pada bagian ke-6 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      β_1= 0.251795448, β_2= -1.215487652, β_3= -2.486150591,
      β_4= -0.866704036, β_5= 0.439143109, β_6= -0.28563102,
      β_7= 1.963821462, β_8= 0.982571014
   B. Berdasarkan pembagian *fold cross validation*, maka data bagian keenam dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.8.

Tabel 4.8 Rekap Perhitungan β Bagian ke-6

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
5.791295	-15.8013	0	-8.66704	7.465433	-6.85514	13.74675	0.2063399
5.791295	-15.8013	0	-8.66704	7.465433	-6.85514	13.74675	0.2063399

Berdasarkan pada hasil perhitungan Tabel 4.8, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(5.791295303+-15.80133948+0+-8.667040364+7.46543286+6.855144487+13.74675023+0.206339913)))
= 0.0160841505839197

Hasil probabilitas pada baris pertama sebesar (0.0160841505839197) atau sebesar 1.6084%, karena nilai persentase probabilitas lebih dari nol dan kurang dari 5% maka diklasifikasikan sebagai tidak rawan cacat.
Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(5.791295303+-15.80133948+0+-8.667040364+7.46543286+-6.855144487+13.74675023+0.206339913)))
= 0.0160841505839197

Hasil probabilitas pada baris kedua sebesar (0.0160841505839197) atau sebesar 1.608415%, karena nilai persentase probabilitas lebih dari nol dan kuranng dari 5% maka diklasifikasikan sebagai tidak rawan cacat.

7. Pada bagian ke-7 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      β_1= 0.632331922, β_2= -0.756612713, β_3= -2.953537429,
      β_4= -1.358140517, β_5= 0.510527277, β_6= -0.356532368,
      β_7= 0.713441646, β_8= 0.947315875
   B. Berdasarkan pembagian *fold cross validation*, maka data bagian ketujuh dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.9.

Tabel 4.9 Rekap Perhitungan β Bagian ke-7

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
25.29328	-6.80951	-11.8141	-2.71628	7.657909	-5.70452	3.567208	0.1420974
14.54363	-6.80951	-17.7212	-4.07442	10.21055	-5.70452	3.567208	0.17999

Berdasarkan pada hasil perhitungan Tabel 4.9, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(25.29327689+-6.809514414+-11.81414971+-
2.716281033+7.657909158+-
5.704517888+3.567208229+0.142097381)))
= 0.999933352649125

Hasil probabilitas pada baris pertama sebesar (0.999933352649125) atau sebesar 99,99333%, karena nilai persentase probabilitas lebih dari nol maka diklasifikasikan sebagai rawan cacat.
Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(14.54363421+-6.809514414+-17.72122457+-
4.07442155+10.21054554+-5.704517888+3.567208229+0.179990016)))
= 0.00299354057644845

Hasil probabilitas pada baris kedua sebesar (0.00299354057644845) atau sebesar 2.99354%, karena nilai persentase probabilitas lebih dari

nol dan kuranng dari 5% maka diklasifikasikan sebagai tidak rawan cacat.

8. Pada bagian ke-8 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.

   A. Nilai β yang di hasilkan.

      β_1= 0.101153471, β_2= -1.200892214, β_3= -2.305514052,

      β_4= -1.22696011, β_5= 0.671765672, β_6= -0.201330909,

      β_7= 1.61505419, β_8= 1.10000027

   B. Berdasarkan pembagian *fold cross validation*, maka data bagian kedelapan dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.10.

Tabel 4.10 Rekap Perhitungan *β* Bagian ke-8

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
0.809228	-3.60268	-2.30551	-7.36176	3.358828	-0.80532	3.230108	0.121
1.921916	-3.60268	-2.30551	-2.45392	0	-0.80532	3.230108	0.066

Berdasarkan pada hasil perhitungan Tabel 4.10, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(0.809227766+-3.602676641+-2.305514052+-
7.361760658+3.358828362+-
0.805323635+3.23010838+0.121000029)))
= 0.00141938607219321

Hasil probabilitas pada baris pertama sebesar (0.00141938607219321) atau sebesar 0.1419%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.

Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(1.921915945+-3.602676641+-2.305514052+-2.453920219+0+-0.805323635+3.23010838+0.066000016)))
= 0.0189018963066569

Hasil probabilitas pada baris kedua sebesar (0.0189018963066569) atau sebesar 1.89019%, karena nilai persentase probabilitas lebih dari nol dan kuranng dari 5% maka diklasifikasikan sebagai tidak rawan cacat.

9. Pada bagian ke-9 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
   A. Nilai β yang di hasilkan.
      β_1= 0.262416515, β_2= -1.124986022, β_3= -2.515381199,
      β_4= -0.91241027, β_5= 0.457908998, β_6= -0.347924178,
      β_7= 1.895476749, β_8= 1.028785581
   B. Berdasarkan pembagian *fold cross validation*, maka data bagian kesembilan dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.11.

Tabel 4.11 Rekap Perhitungan *β* Bagian ke-9

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
1.574499	-7.8749	0	-5.47446	0	-3.47924	7.581907	0.1543178
35.16381	-67.4992	-22.6384	-57.4818	14.19518	-13.2211	77.71455	0.1748935

Berdasarkan pada hasil perhitungan Tabel 4.11, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(1.574499088+-7.874902155+0+-5.474461622+0+-3.47924178+7.581906995+0.154317837)))
= 0.000542987221775189

Hasil probabilitas pada baris pertama sebesar (0.000542987221775189) atau sebesar 0.0543%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.
Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp*(-(LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(35.16381296+-67.49916132+-22.63843079+-57.48184703+14.19517893+-13.22111876+77.7145467+0.174893549)))
= 2.57706360937752E-15

Hasil probabilitas pada baris kedua sebesar (2.57706360937752E-15) atau sebesar 0.0000%, karena nilai persentase probabilitas kurang dari nol maka diklasifikasikan sebagai tidak rawan cacat.

10. Pada bagian ke-10 dijadikan sebagai data uji, dan data selebihnya dijadikan sebagai data pelatihan. Pertama dilakukan pelatihan data menggunakan kode program java untuk mendapatkan nilai prediksi β.
    A. Nilai β yang di hasilkan.

       β_1= 0.333893663, β_2= -0.048692604, β_3= -2.986092044,
       β_4= -1.652764809, β_5= 0.665139153, β_6= -1.450805817,
       β_7= 2.908857015, β_8= 0.750637264

    B. Berdasarkan pembagian *fold cross validation*, maka data bagian kesepuluh dijadikan data uji, kemudian dikalikan dengan β sehingga mendapatkan hasil perhitungan seperti terlihat pada Tabel 4.12.

Tabel 4.12 Rekap Perhitungan β Bagian ke-10

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
3.005043	-0.38954	-5.97218	-6.61106	3.325696	-11.6064	14.54429	0.2101784
54.75856	-5.35619	-77.6384	-61.1523	127.0416	-185.703	203.62	0.1351147

Berdasarkan pada hasil perhitungan Tabel 4.12, maka dapat dihitung nilai probabilitasnya dari masing-masing baris data uji sebagai berikut:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(3.005042968+-0.389540835+-5.972184088+-6.611059235+3.325695766+-11.60644654+14.54428508+0.210178434)))
= 0.0294826177468072

Hasil probabilitas pada baris pertama sebesar (0.0294826177468072) atau sebesar 2.9483%, karena nilai persentase probabilitas lebih dari nol dan kurang dari 5%, maka diklasifikasikan sebagai tidak rawan cacat.

Dengan cara yang sama maka dilakukan perhitungan untuk baris kedua, yaitu:

P=1/(1+exp(-(*LOC_BLANK+BRANCH_COUNT+CALL_PAIRS+LOC_CODE_AND_COMMENT+ LOC_COMMENTS+CONDITION_COUNT+CYCLOMATIC_COMPLEXITY+CYCLOMATIC_DENSITY))*

P=1/(1+exp(-(54.75856076+-5.356186478+-77.63839315+-61.15229792+127.0415783+-185.7031446+203.6199911+0.135114707)))
= 1.00E+00

Hasil probabilitas pada baris kedua sebesar (1.00E+00) atau sebesar 100%, karena nilai persentase probabilitas lebih dari nol dan lebih dari 5%, maka diklasifikasikan sebagai rawan cacat.

Berdasarkan perhitungan probabilitas secara manual model Logistic Regression dan 10-*fold cross validation* diatas, maka dapat dibuat rekapitulasinya seperti pada Tabel 4.13.

Tabel 4.13 Rekap Perhitungan Manual Model *Logistic Regression*

Split	Actual	Prediction
1	N	N
	Y	N
2	Y	Y
	N	N
3	N	N
	N	N
4	N	N
	N	Y
5	N	Y
	Y	Y
6	N	N
	N	N
7	N	Y
	N	N
8	N	N
	Y	N
9	N	N
	N	N
10	N	N
	N	Y

Dari hasil rekap perhitungan manual, kemudian dibuat tabel *confusion matrix* seperti Tabel 4.14, dan dilakukan perhitungan kinerja model.

Tabel 4.14 Confusion Matrix Perhitungan Manual

Class		Actual		Class Precision
		TRUE	FALSE	
Prediction	TRUE	2	4	33.33%
	FALSE	2	12	85.71%
Class recall		50.00%	75.00%	

Berdasarkan pada Tabel 4.14 maka dapat dihitung akurasi, *sensitivitas/recall, Spesificity, Precision, F-measure, G-Mean* dan AUC sebagai berikut:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} = \frac{2 + 12}{2 + 12 + 4 + 2} = \frac{14}{20} = 0.70 = 70\%$$

$$Precision = PPV = \frac{TP}{TP + FP} = \frac{2}{2 + 4} = \frac{2}{6} = 0.33333 = 33.33\%$$

$$NPV = \frac{TN}{TN + FN} = \frac{12}{12 + 2} = \frac{12}{14} = 0.857142 = 85.71\%$$

$$Sensitivitas = recall = TP_{rate} = \frac{TP}{TP + FN} = \frac{2}{2 + 2} = \frac{2}{4} = 0{,}50 = 50\%$$

$$Specificity = TN_{rate} = \frac{TN}{TN + FP} = \frac{12}{12 + 4} = \frac{12}{16} = 0.75 = 75\%$$

$$FP_{rate} = \frac{FP}{FP + TN} = \frac{4}{4 + 12} = \frac{4}{16} = 0.25 = 25\%$$

$$FN_{rate} = \frac{FN}{FN + TP} = \frac{2}{2 + 2} = \frac{2}{4} = 0.50 = 50\%$$

$$f - Measure = \frac{2xrecallxprecision}{(recall + precision)} = \frac{2x0.50x0.3333}{(0.50 + 0.3333)} = \frac{0.3333}{0.8333} = 0.3999$$

$$G - Mean = \sqrt{sensivitas\ x\ specificity} = \sqrt{0.50\ x\ 0.75} = \sqrt{0.375} = 0.612$$

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} = \frac{1 + 0.5 - 0.25}{2} = \frac{1.25}{2} = 0.625$$

4.1.2 Hasil Eksperimen *Logistic Regression*

Hasil eksperimen *Logisic Regression* yang dilakukan dengan menggunakan 10 dataset NASA MDP (CM1, JM1, KC1, KC3, MC1, MC2, PC1, PC3, PC4, dan PC5). Model yang diuji adalah model pengklasifikasi *Logistic Regression* (LR), hasil eksperimen disajikan pada Tabel 4.15.

Tabel 4.15 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

4.1.3 Hasil Pengukuran Manual *Logistic Regression* dengan *Resampling*

Pada penelitian ini akan dilakukan perhitungan manual dengan menggunakan sampel dataset yang diambil dari dataset CM1, dapat dilihat pada Tabel 4.1 dan Tabel 4.2 dataset yang displit 10 bagian sesuai 10-*fold cross validation.* Rumus prediksi (*Logistic Regression*) cacat *software* seperti pada persamaan 4.1, dimana β adalah nilai *Maximum Loglikehood* dari persamaan 4.2.

Tahapan pada *fold cross validation* sama dengan tahapan pada *Logistic Regression* pada subbab 4.1.1, sehingga pada tiap bagian tahapan hanya akan dimunculkan tabel hasil dan tabel hasil *resampling* saja.

1. Pada tahapan split ke-1 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.16.

Tabel 4.16 Dataset Training Split ke-1 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
27	9	1	4	22	16	5	0.2	Y
4	3	0	2	4	4	2	0.2	Y
19	3	1	2	0	4	2	0.1	Y
23	9	6	3	20	16	5	0.2	N
23	13	0	10	17	24	7	0.2	N
16	9	2	11	28	12	5	0.1	N

Selanjutan dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.17.

Tabel 4.17 Rekap Perhitungan *β* Bagian ke-1 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
3.570136	11.51278	-3.40528	-0.85092	0	-24.0799	9.354079	0.10242
8.925339	8.954384	-5.10792	-0.85092	-2.84232	-18.0599	7.483263	0.066573

Berdasarkan pada hasil perhitungan Tabel 4.17, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.18.

Tabel 4.18 Perhitungan Probabilitas Bagian ke-1

Nilai Probabilitas	Persentase	Prediksi
0.021952502	2.195%	Tidak Cacat
0.192861409	19.286%	Cacat

2. Pada bagian ke-2 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.19.

Tabel 4.19 Dataset Training Split ke-2 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
4	3	0	2	4	4	2	0.2	Y
19	3	1	2	0	4	2	0.1	Y
3	5	2	0	0	8	3	0.2	N
6	7	0	6	0	10	4	0.2	N
164	110	26	37	191	128	70	0.2	N

Selanjutan dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.20.

Tabel 4.20 Rekap Perhitungan *β* Bagian ke-2 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
16.28167	-16.6522	-1.81744	4.240418	14.9547	4.184513	-4.00408	0.097777
4.221173	-5.55074	-3.63487	0	0	1.046128	-1.60163	0.110814

Berdasarkan pada hasil perhitungan Tabel 4.20, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.21.

Tabel 4.21 Perhitungan Probabilitas Bagian ke-2

Nilai Probabilitas	Persentase	Prediksi
0.999999969875392	99.999%	Cacat
0.00454317351970367	0.454%	Tidak Cacat

3. Pada bagian ke-3 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.22.

Tabel 4.22 Dataset Training Split ke-3 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
4	3	0	2	4	4	2	0.2	Y
19	3	1	2	0	4	2	0.1	Y
16	9	2	11	28	12	5	0.1	N
8	3	1	6	5	4	2	0.1	N
22	29	0	8	35	44	15	0.3	N
40	9	4	2	15	16	5	0.2	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.23.

Tabel 4.23 Rekap Perhitungan *β* Bagian ke-3 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
13.9833	16.66585	-28.4154	0	7.875165	-100.162	48.48374	0.087319
0.822547	3.33317	-4.3716	0	0	-16.6937	11.18856	0.145532

Berdasarkan pada hasil perhitungan Tabel 4.23, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.24.

Tabel 4.24 Perhitungan Probabilitas Bagian ke-2

Nilai Probabilitas	Persentase	Prediksi
9.64857E-19	0.000%	Tidak Cacat
0.003775241	0.378%	Tidak Cacat

4. Pada bagian ke-4 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.25.

Tabel 4.25 Dataset Training Split ke-4 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
4	3	0	2	4	4	2	0.2	Y
19	3	1	2	0	4	2	0.1	Y
134	60	9	63	31	38	41	0.2	N
23	13	0	10	17	24	7	0.2	N
164	110	26	37	191	128	70	0.2	N
16	9	2	11	28	12	5	0.1	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.26.

Tabel 4.26 Rekap Perhitungan *β* Bagian ke-4 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
5.657297	-7.20629	-5.20167	-10.4435	2.693818	-0.80276	3.255418	0.172319
9.573887	-23.2203	0	-6.96232	5.892726	-2.20758	9.766254	0.344638

Berdasarkan pada hasil perhitungan Tabel 4.26, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.27.

Tabel 4.27 Perhitungan Probabilitas Bagian ke-4

Nilai Probabilitas	Persentase	Prediksi
6.95984E-06	0.001%	Tidak Cacat
0.001098545	0.110%	Tidak Cacat

5. Pada bagian ke-5 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.28.

Tabel 4.28 Dataset Training Split ke-5 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
19	3	1	2	0	4	2	0.1	Y
164	110	26	37	191	128	70	0.2	N
8	3	1	6	5	4	2	0.1	N
3	5	2	0	0	8	3	0.2	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.29.

Tabel 4.29 Rekap Perhitungan β Bagian ke-5 LR dan Resampling

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
11.88012	-6.0087	-0.31163	-15.8686	12.80426	-4.98292	-2.52707	-0.0023
2.970029	-2.0029	0	-2.8852	1.82918	-1.66097	-1.01083	-0.00356

Berdasarkan pada hasil perhitungan Tabel 4.29, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.30.

Tabel 4.30 Perhitungan Probabilitas Bagian ke-5

Nilai Probabilitas	Persentase	Prediksi
0.006581768	0.658%	Tidak Cacat
0.059286654	5.928%	Cacat

6. Pada bagian ke-6 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.31.

Tabel 4.31 Dataset Training Split ke-6 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
19	3	1	2	0	4	2	0.1	Y
4	3	0	2	4	4	2	0.2	Y
134	60	9	63	31	38	41	0.2	N
13	9	5	12	16	16	5	0.1	N
51	25	13	0	14	48	13	0.1	N
8	3	1	6	5	4	2	0.1	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.32.

Tabel 4.32 Rekap Perhitungan β Bagian ke-6 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
13.68339	3.512505	0	-18.5727	21.71342	-19.6575	4.322257	0.065705
13.68339	3.512505	0	-18.5727	21.71342	-19.6575	4.322257	0.065705

Berdasarkan pada hasil perhitungan Tabel 4.32, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.33.

Tabel 4.33 Perhitungan Probabilitas Bagian ke-6

Nilai Probabilitas	Persentase	Prediksi
0.993738716	99.373%	Cacat
0.993738716	99.373%	Cacat

7. Pada bagian ke-7 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.34.

Tabel 4.34 Dataset Training Split ke-7 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
19	3	1	2	0	4	2	0.1	Y
4	3	0	2	4	4	2	0.2	Y
51	25	13	0	14	48	13	0.1	N
6	9	2	1	0	16	5	0.2	N
23	13	0	10	17	24	7	0.2	N
164	110	26	37	191	128	70	0.2	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.35

Tabel 4.35 Rekap Perhitungan β bagian ke-7 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
22.39193	-11.9337	-4.41183	0.998107	11.01634	-11.47	0.792669	0.173374
12.87536	-11.9337	-6.61774	1.49716	14.68845	-11.47	0.792669	0.219608

Berdasarkan pada hasil perhitungan Tabel 4.35, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.36.

Tabel 4.36 Perhitungan Probabilitas Bagian ke-7

Nilai Probabilitas	Persentase	Prediksi
0.999477763	99,9477%	Cacat
0.512942537	51.294%	Cacat

8. Pada bagian ke-8 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.37.

Tabel 4.37 Dataset Training Split ke-8 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
4	3	0	2	4	4	2	0.2	Y
22	29	0	8	35	44	15	0.3	N
16	9	2	11	28	12	5	0.1	N
9	8	2	4	5	8	5	0.3	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.38.

Tabel 4.38 Rekap Perhitungan β Bagian ke-8 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
12.37136	-4.5376	-2.23804	-4.47731	-1.48168	1.721808	0.29852	0.051705
29.38197	-4.5376	-2.23804	-1.49244	0	1.721808	0.29852	0.028202

Berdasarkan pada hasil perhitungan Tabel 4.38, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.39.

Tabel 4.39 Perhitungan Probabilitas Bagian ke-8

Nilai Probabilitas	Persentase	Prediksi
0.846675752	84.66%	Cacat
1	100%	Cacat

9. Pada bagian ke-9 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.40.

Tabel 4.40 Dataset Training Split ke-9 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
4	3	0	2	4	4	2	0.2	Y
19	3	1	2	0	4	2	0.1	Y
16	9	2	11	28	12	5	0.1	N
22	29	0	8	35	44	15	0.3	N
51	25	13	0	14	48	13	0.1	N
13	9	5	12	16	16	5	0.1	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.41.

Tabel 4.41 Rekap Perhitungan β Bagian ke-9 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
6.249256	2.559231	0	-7.87914	0	-13.1557	7.202715	0.098672
139.5667	21.93626	-19.6484	-82.731	2.520271	-49.9916	73.82783	0.111828

Berdasarkan pada hasil perhitungan Tabel 4.41, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.42.

Tabel 4.42 Perhitungan Probabilitas Bagian ke-9

Nilai Probabilitas	Persentase	Prediksi
0.007210788	0.721%	Tidak Cacat
1	100%	Cacat

10. Pada bagian ke-10 dataset yang digunakan sebagai data latih dari hasil proses *resampling* dapat dilihat pada Tabel 4.43.

Tabel 4.43 Dataset Training Split ke-10 Hasil *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	Defective
15	7	3	1	19	12	4	0.1	Y
27	9	1	4	22	16	5	0.2	Y
4	3	0	2	4	4	2	0.2	Y
19	3	1	2	0	4	2	0.1	Y
16	9	2	11	28	12	5	0.1	N
23	13	0	10	17	24	7	0.2	N
22	29	0	8	35	44	15	0.3	N
6	9	2	1	0	16	5	0.2	N

Selanjutnya dilakukan perhitungan nilai β, sehingga mendapatkan hasil perhitungan seperti pada Tabel 4.34

Tabel 4.44 Rekap Perhitungan β Bagian ke-10 LR dan *Resampling*

LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY
9.697536	7.92338	-1.59844	-5.32678	-0.62581	-13.1831	9.451183	0.162289
176.7107	108.9465	-20.7797	-49.2727	-23.9058	-210.929	132.3166	0.104329

Berdasarkan pada hasil perhitungan Tabel 4.44, maka didapat nilai probabilitas dan prediksi cacat *software* seperti pada Tabel 4.45.

Tabel 4.45 Perhitungan Probabilitas Bagian ke-10

Nilai Probabilitas	Persentase	Prediksi
0.998499233	99,84%	Cacat
1.00E+00	100%	Cacat

Berdasarkan perhitungan probabilitas secara manual model *Logistic Regression* dan 10-*fold cross validation* diatas, maka dapat dibuat rekapitulasinya seperti pada Tabel 4.46.

Tabel 4.46 Rekap Perhitungan Manual Model *Logistic Regression* dan *Resampling*

Split	Actual	Prediction
1	N	N
	Y	Y
2	Y	Y
	N	N
3	N	N
	N	N
4	N	N
	N	N
5	N	N
	Y	Y
6	N	N
	N	Y
7	N	Y
	N	Y
8	N	Y
	Y	Y
9	N	N
	N	Y
10	N	Y
	N	Y

Dari hasil rekap perhitungan manual, kemudian dibuat tabel *confusion matrix* seperti Tabel 4.47, dan dilakukan perhitungan kinerja model.

Tabel 4.47 Confusion Matrix Perhitungan Manual LR dan *Resampling*

Class		Actual		Class Precision
		TRUE	FALSE	
Prediction	TRUE	4	7	36.36%
	FALSE	0	9	100.00%
Class Recall		100.00%	56.25%	

Berdasarkan pada Tabel 4.47 maka dapat dihitung akurasi, *sensitivitas/recall, Spesificity, Precision, F-measure, G-Mean* dan AUC sebagai berikut:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} = \frac{4 + 9}{4 + 9 + 7 + 0} = \frac{13}{20} = 0.65 = 65\%$$

$$Precision = PPV = \frac{TP}{TP + FP} = \frac{4}{4 + 7} = \frac{4}{11} = 0.363636 = 36.36\%$$

$$NPV = \frac{TN}{TN + FN} = \frac{9}{9 + 0} = \frac{9}{9} = 1 = 100\%$$

$$Sensitivitas = recall = TP_{rate} = \frac{TP}{TP + FN} = \frac{4}{4 + 0} = \frac{4}{4} = 1 = 100\%$$

$$Specificity = TN_{rate} = \frac{TN}{TN + FP} = \frac{9}{9 + 7} = \frac{9}{16} = 0.5625 = 56.25\%$$

$$FP_{rate} = \frac{FP}{FP + TN} = \frac{7}{7 + 9} = \frac{7}{16} = 0.4375 = 43.75\%$$

$$FN_{rate} = \frac{FN}{FN + TP} = \frac{0}{0 + 4} = \frac{0}{4} = 0 = 0\%$$

$$f - Measure = \frac{2xrecallxprecision}{(recall + precision)} = \frac{2x1x0.363636}{(1 + 0.3636)} = \frac{0.727272}{1.363636}$$

$$= 0.53333$$

$$G - Mean = \sqrt{sensivitas\ x\ specificity} = \sqrt{1\ x\ 0.5625} = \sqrt{0.5625} = 0.75$$

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} = \frac{1 + 1 - 0.4375}{2} = \frac{1.5625}{2} = 0.78125$$

4.1.4 Hasil Eksperimen *Logistic Regression* dengan dengan *Resampling*

Hasil eksperimen *Logisic Regression* dan *Resampling* yang dilakukan dengan menggunakan 10 dataset NASA MDP (CM1, JM1, KC1, KC3, MC1, MC2, PC1, PC3, PC4, dan PC5). Model yang diuji adalah model *Resampling Logistic Regression* (LR). Hasil eksperimen pada Tabel 4.48 menunjukan model *Resampling* dan *Logistic Regression*, Tabel 4.49 menunjukan model *Resampling* SMOTE dan Logistic Regression, dan Table 4.50 menunjukan model *Resampling Spread Sub Sample* dan *Logistic Regression*.

Tabel 4.48 Hasil Pengukuran Model 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

Tabel 4.49 Hasil Pengukuran Model *Logistic Regression* dan SMOTE

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	33	276	26	51	80.052%	39.286%	91.391%	55.932%	84.40%	0.462	0.599	0.727
JM1	852	7385	449	2666	72.560%	24.218%	94.269%	65.488%	73.48%	0.354	0.478	0.728
KC1	244	1645	126	406	78.026%	37.538%	92.885%	65.946%	80.20%	0.478	0.590	0.801
KC3	34	146	18	38	76.271%	47.222%	89.024%	65.385%	79.35%	0.548	0.648	0.669
MC1	46	9195	14	90	98.887%	33.824%	99.848%	76.667%	99.03%	0.469	0.581	0.896
MC2	54	65	18	34	69.591%	61.364%	78.313%	75.000%	65.66%	0.675	0.693	0.731
PC1	44	679	19	78	88.171%	36.066%	97.278%	69.841%	89.70%	0.476	0.592	0.835
PC3	99	915	70	181	80.158%	35.357%	92.893%	58.580%	83.49%	0.441	0.573	0.816
PC4	225	1156	65	131	87.571%	63.202%	94.676%	77.586%	89.82%	0.697	0.774	0.923
PC5	475	16237	261	531	95.475%	47.217%	98.418%	64.538%	96.83%	0.545	0.682	0.956

Tabel 4.50 Hasil Pengukuran Model *Logistic Regression* dan *Spread Sub Sample*

Dataset	TP	TN	FP	FN	Accuracy	Recall	Specificity	PPV	NPV	F-Measure	G-Mean	AUC
CM1	10	282	20	32	84.884%	23.810%	93.377%	33.333%	89.81%	0.278	0.472	0.761
JM1	181	7700	134	1578	82.154%	10.290%	98.290%	57.460%	82.99%	0.175	0.318	0.706
KC1	66	1726	45	259	85.496%	20.308%	97.459%	59.459%	86.95%	0.303	0.445	0.800
KC3	13	144	20	23	78.500%	36.111%	87.805%	39.394%	86.23%	0.377	0.563	0.679
MC1	20	9197	12	48	99.353%	29.412%	99.870%	62.500%	99.48%	0.400	0.542	0.893
MC2	25	67	16	19	72.441%	56.818%	80.723%	60.976%	77.91%	0.588	0.677	0.713
PC1	15	680	18	46	91.568%	24.590%	97.421%	45.455%	93.66%	0.319	0.489	0.830
PC3	29	945	31	111	87.276%	20.714%	96.824%	48.333%	89.49%	0.290	0.448	0.823
PC4	86	1191	30	92	91.279%	48.315%	97.543%	74.138%	92.83%	0.585	0.686	0.905
PC5	148	16387	111	355	97.259%	29.423%	99.327%	57.143%	97.88%	0.388	0.541	0.956

4.1.5 Hasil Pengukuran Kinerja *Logistic Regression* dan *Resampling*

Pengukuran model dilakukan dengan menguji menggunakan 10 dataset NASA MDP (CM1, JM1, KC1, KC3, MC1, MC2, PC1, PC3, PC4, dan PC5). Model yang diuji adalah model pengklasifikasi *Logistic Regression* (LR), *Logistic*

Regression dan *Resample* (LR+Resample), *Logistic Regression* dan SMOTE (LR+SMOTE), dan *Logistic Regression* dengan *Spread Sub Sample* (LR+SS).

Hasil rekap pengukuran akurasi disajikan pada Tabel 4.51 untuk semua dataset. Rekap menyeluruh pada eksperimen *Logistic Regression* tanpa *Resampling* dan *Logistic Regression* dengan *Resampling.*

Tabel 4.51 Rekap Pengukuran Akurasi Model 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%
LR+SMOTE	80.052%	72.560%	78.026%	76.271%	98.887%	69.591%	88.171%	80.158%	87.571%	95.475%
LR+SpreadSubSample	84.88%	82.15%	85.50%	78.50%	99.35%	72.44%	91.57%	87.28%	91.28%	97.26%

Hasil rekap pengukuran *Sensitivitas* disajikan pada Tabel 4.52 untuk semua dataset. Rekap menyeluruh pada eksperimen *Logistic Regression* tanpa *Resampling* dan *Logistic Regression* dengan *Resampling.*

Tabel 4.52 Rekap Pengukuran Sensitivitas Model Prediksi Cacat *Software*

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	21.43%	10.23%	21.54%	33.33%	27.94%	80.00%	18.03%	20.00%	48.32%	29.23%
LR+Resample	43.90%	9.40%	24.48%	86.49%	31.75%	46.27%	30.91%	20.00%	56.00%	28.72%
LR+SMOTE	39.29%	24.22%	37.54%	47.22%	33.82%	61.36%	36.07%	35.36%	63.20%	47.22%
LR+SpreadSubSample	23.81%	10.29%	20.31%	36.11%	29.41%	56.82%	24.59%	20.71%	48.31%	29.42%

Hasil rekap pengukuran *F-Measure* disajikan pada Tabel 4.53 untuk semua dataset. Rekap menyeluruh pada eksperimen *Logistic Regression* tanpa *Resampling* dan *Logistic Regression* dengan *Resampling.*

Tabel 4.53 Rekap Pengukuran F-Measure Model Prediksi Cacat *Software*

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	0.256	0.173	0.319	0.369	0.376	0.791	0.259	0.277	0.577	0.388
LR+Resample	0.522	0.161	0.345	0.762	0.455	0.564	0.405	0.274	0.631	0.384
LR+SMOTE	0.462	0.354	0.478	0.548	0.469	0.675	0.476	0.441	0.697	0.545
LR+SpreadSubSample	0.278	0.175	0.303	0.377	0.400	0.588	0.319	0.290	0.585	0.388

Hasil rekap pengukuran *G-Mean* disajikan pada Tabel 4.54 untuk semua dataset. Rekap menyeluruh pada eksperimen *Logistic Regression* tanpa *Resampling* dan *Logistic Regression* dengan *Resampling.*

Tabel 4.54 Rekap Pengukuran G-Mean Model Prediksi Cacat *Software*

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	0.448	0.317	0.458	0.547	0.528	0.038	0.421	0.439	0.685	0.539
LR+Resample	0.652	0.304	0.487	0.886	0.563	0.674	0.551	0.438	0.735	0.534
LR+SMOTE	0.599	0.478	0.590	0.648	0.581	0.693	0.592	0.573	0.774	0.682
LR+SpreadSubSample	0.472	0.318	0.445	0.563	0.542	0.677	0.489	0.448	0.686	0.541

Hasil rekap pengukuran AUC disajikan pada Tabel 4.55 untuk semua dataset. Rekap menyeluruh pada eksperimen *Logistic Regression* tanpa *Resampling* dan *Logistic Regression* dengan *Resampling.*

Tabel 4.55 Rekap Pengukuran AUC Model Prediksi Cacat *Software*

Model	Dataset									
	CM1	JM1	KC1	KC3	MC1	MC2	PC1	PC3	PC4	PC5
LR	0.728	0.705	0.800	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	0.900	0.854	0.846	0.926	0.951
LR+SMOTE	0.727	0.728	0.801	0.669	0.896	0.731	0.835	0.816	0.923	0.956
LR+SpreadSubSample	0.761	0.706	0.800	0.679	0.893	0.713	0.830	0.823	0.905	0.956

4.2 Pembahasan

4.2.1 Perbandingan Kinerja *Logistic Regression* dan *Logistic Regression* dengan *Resampling*

Pada bagian sub bab ini akan membahas hasil perbandingan *Logistic Regression* (LR), *Logistic Regression* dengan *Resampling* (LR+*Resample*), *Logistic Regression* dengan SMOTE (LR+SMOTE), dan *Logistic Regression* dengan *Spread Sub Sample* (LR+*SpreadSubSample*).

4.2.1.1 Perbandingan Kinerja LR dan LR+*Resample*

Pada tahapan ini akan dilakukan uji *t* dua sampel berpasangan dengan menggunakan nilai rata-rata dari hasil akurasi, *sensitivitas, F-Measure*, *G-Mean* dan AUC dari eksperimen yang sudah dijalanan.

A. Akurasi LR dan LR+Resample

Uji beda dilakukan dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata akurasi antara LR dengan LR+*Resample*

H_1: Ada perbedaan nilai rata-rata akurasi antara LR dengan LR+*Resample*

Pada Tabel 4.56 menunjukan nilai akurasi LR dan LR+*Resample*, yang selanjutnya dilakukan perhitungan uji *t* .

Tabel 4.56 Perbandingan Akurasi LR dan LR+*Resample*

NO	Dataset	LR	LR+Resample
1	CM1	84.88	90.4
2	JM1	82.11	81.9
3	KC1	85.74	85.16
4	KC3	79.5	90
5	MC1	99.32	99.48
6	MC2	85.04	93.68
7	PC1	91.7	93.41
8	PC3	87.02	86.31
9	PC4	90.99	90.64
10	PC5	97.27	97.31

Berdasarakan hasil akurasi yang tertera pada Tabel 4.56 dilakukan analisa statistik uji *t* (*t-test*) sampel berpasangan (*paired-sample t-test)* menggunakan aplikasi Microsoft Excel 2007, dan diperoleh hasil seperti terlihat pada Tabel 4.57.

Tabel 4.57 Tabel Uji Beda Akurasi LR dan LR+*Resample*

t-Test: Paired Two Sample for Means

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

B. Sensitivitas LR dan LR+*Resample*

Uji beda juga dilakukan terhadap sensitivitas dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata sensitivitas antara LR dengan LR+*Resample*

H_1: Ada perbedaan nilai rata-rata sensitivitas antara LR dengan LR+*Resample*

Pada Tabel 4.58 menunjukan nilai sensitivitas LR dan LR+*Resample*.

Tabel 4.58 Perbandingan Sensitivitas LR dan LR+*Resample*

NO	Dataset	LR	LR+Resample
1	CM1	21.43	43.9
2	JM1	10.23	9.4
3	KC1	21.54	24.48
4	KC3	33.33	86.49
5	MC1	27.94	31.75
6	MC2	80	46.27
7	PC1	18.03	30.91
8	PC3	20	20
9	PC4	48.32	56
10	PC5	29.23	28.72

Berdasarkan pada Tabel 4.58 perbandingan senitivitas LR dan LR+*Resample* maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.59.

Tabel 4.59 Tabel Uji Beda *Sensitivitas* LR dan LR+*Resample*

t-Test: Paired Two Sample for Means

	LR	*LR+Resample*
Mean	31.005	37.792
Variance	402.28865	475.3610844
Observations	10	10
Pearson Correlation	0.462673238	
Hypothesized Mean Difference	95	
df	9	
t Stat	-14.8000858	
P(T<=t) one-tail	6.33728E-08	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	1.26746E-07	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.59 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 14.8000858, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 1.267467E-07, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata sensitivitas LR dan LR+*Resample*, hasil sensitivitas menunjukan LR+*Resample* lebih tinggi dibandingkan dengan LR.

C. *F-Measure* LR dan LR+*Resample*

Uji beda juga dilakukan terhadap *F-Measure* dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata *F-Measure* antara LR dengan LR+*Resample*

H_1: Ada perbedaan nilai rata-rata F-Measure antara LR dengan LR+*Resample*

Pada Tabel 4.60 menunjukan nilai *F-Measure* LR dan LR+*Resample*.

Tabel 4.60 Perbandingan *F-Measure* LR dan LR+*Resample*

NO	Dataset	LR	LR+Resample
1	CM1	0.256	0.522
2	JM1	0.173	0.161
3	KC1	0.319	0.345
4	KC3	0.369	0.762
5	MC1	0.376	0.455
6	MC2	0.791	0.564
7	PC1	0.259	0.405
8	PC3	0.277	0.274
9	PC4	0.577	0.631
10	PC5	0.388	0.384

Berdasarkan pada Tabel 4.60 perbandingan *F-Measure* LR dan LR+*Resample* maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.61.

Tabel 4.61 Tabel Uji Beda *F-Measure* LR dan LR+*Resample*

t-Test: Paired Two Sample for Means

	LR	*LR+Resample*
Mean	0.3785	0.4503
Variance	0.032789389	0.031134678
Observations	10	10
Pearson Correlation	0.554344977	
Hypothesized Mean Difference	95	
df	9	
t Stat	-1780.85937	
P(T<=t) one-tail	1.41309E-26	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	2.82618E-26	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.61 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan t tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 1780.85937, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 2.82618E-26, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata *F-Measure* LR dan LR+*Resample,* hasil *F-Measure* menunjukan LR+*Resample* lebih tinggi dibandingkan dengan LR.

D. *G-Mean* LR dan LR+*Resample*

Uji beda juga dilakukan terhadap *G-Mean* dengan menggunakan metode statistic untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata *G-Mean* antara LR dengan LR+*Resample*

H_1: Ada perbedaan nilai rata-rata *G-Mean* antara LR dengan LR+*Resample*

Pada Tabel 4.62 menunjukan nilai *G-Mean* LR dan LR+*Resample*.

Tabel 4.62 Perbandingan *G-Mean* LR dan LR+*Resample*

NO	Dataset	LR	LR+Resample
1	CM1	0.448	0.652
2	JM1	0.317	0.304
3	KC1	0.458	0.487
4	KC3	0.547	0.886
5	MC1	0.528	0.563
6	MC2	0.038	0.674
7	PC1	0.421	0.551
8	PC3	0.439	0.438
9	PC4	0.685	0.735
10	PC5	0.539	0.534

Berdasarkan pada Tabel 4.62 perbandingan *G-Mean* LR dan LR+*Resample* maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.63.

Tabel 4.63 Tabel Uji Beda *G-Mean* LR dan LR+*Resample*

t-Test: Paired Two Sample for Means

	LR	*LR+Resample*
Mean	0.442	0.5824
Variance	0.029606889	0.026650933
Observations	10	10
Pearson Correlation	0.238811283	
Hypothesized Mean Difference	95	
df	9	
t Stat	-1453.56099	
P(T<=t) one-tail	8.78889E-26	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	1.75778E-25	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.63 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 1453.56099, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 1.75778E-25, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata *G-Mean* LR dan LR+*Resample*, hasil *G-Mean* menunjukan LR+*Resample* lebih tinggi dibandingkan dengan LR.

E. AUC LR dan LR+*Resample*

Uji beda juga dilakukan terhadap AUC dengan menggunakan metode statistic untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata AUC antara LR dengan LR+*Resample*

H_1: Ada perbedaan nilai rata-rata AUC antara LR dengan LR+*Resample*

Pada Tabel 4.64 menunjukan nilai AUC LR dan LR+*Resample*.

Tabel 4.64 Perbandingan AUC LR dan LR+Resample

NO	Dataset	LR	LR+Resample
1	CM1	0.728	0.816
2	JM1	0.705	0.708
3	KC1	0.8	0.791
4	KC3	0.708	0.875
5	MC1	0.875	0.895
6	MC2	0.863	0.9
7	PC1	0.828	0.854
8	PC3	0.819	0.846
9	PC4	0.907	0.926
10	PC5	0.955	0.951

Berdasarkan pada Tabel 4.64 perbandingan AUC LR dan LR+*Resample* maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.65.

Tabel 4.65 Tabel Uji Beda AUC LR dan LR+*Resample*

t-Test: Paired Two Sample for Means

	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	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.65 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 8.40852E-31, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata AUC LR dan LR+*Resample,* hasil AUC menunjukan LR+*Resample* lebih tinggi dibandingkan dengan LR.

4.2.1.2 Perbandingan Kinerja LR dan LR+SMOTE

Pada tahapan ini akan dilakukan uji *t* dua sampel berpasangan dengan menggunakan nilai rata-rata dari hasil akurasi, sensitivitas, *F-Measure, G-Mean* dan AUC dari eksperimen yang sudah dijalanan.

A. Akurasi LR dan LR+SMOTE

Uji beda dilakukan dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata akurasi antara LR dengan LR+ SMOTE.

H_1: Ada perbedaan nilai rata-rata akurasi antara LR dengan LR+ SMOTE.

Pada Tabel 4.66 menunjukan nilai akurasi LR dan LR+ SMOTE.

Tabel 4.66 Perbandingan Akurasi LR dan LR+ SMOTE

NO	Dataset	LR	LR+SMOTE
1	CM1	84.88	80.05
2	JM1	82.11	72.56
3	KC1	85.74	78.03
4	KC3	79.5	76.27
5	MC1	99.32	98.89
6	MC2	85.04	69.59
7	PC1	91.7	88.17
8	PC3	87.02	80.16
9	PC4	90.99	87.57
10	PC5	97.27	95.48

Berdasarakan hasil akurasi yang tertera pada Tabel 4.66 dilakukan analisa statistik uji *t* (*t-test*) sampel berpasangan (*paired-sample t-test)* menggunakan aplikasi Microsoft Excel 2007, dan diperoleh hasil seperti terlihat pada Tabel 4.67.

Tabel 4.67 Tabel Uji Beda Akurasi LR dan LR+ SMOTE

t-Test: Paired Two Sample for Means

	LR	*LR+SMOTE*
Mean	88.357	82.677
Variance	40.77077889	92.26629
Observations	10	10
Pearson Correlation	0.925943913	
Hypothesized Mean Difference	95	
df	9	
t Stat	-64.0377147	
P(T<=t) one-tail	1.39323E-13	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	2.78646E-13	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.67 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 64.0377147, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 2.78646E-13, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata akurasi LR dan LR+SMOTE, hasil akurasi menunjukan LR+SMOTE lebih tinggi dibandingkan dengan LR.

B. Sensitivitas LR dan LR+SMOTE

Uji beda juga dilakukan terhadap sensitivitas dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata sensitivitas antara LR dengan LR+SMOTE.

H_1: Ada perbedaan nilai rata-rata sensitivitas antara LR dengan LR+SMOTE.

Pada Tabel 4.68 menunjukan nilai sensitivitas LR dan LR+SMOTE.

Tabel 4.68 Perbandingan Sensitivitas LR dan LR+SMOTE

NO	Dataset	LR	LR+SMOTE
1	CM1	21.43	39.29
2	JM1	10.23	24.22
3	KC1	21.54	37.54
4	KC3	33.33	47.22
5	MC1	27.94	33.82
6	MC2	80	61.36
7	PC1	18.03	36.07
8	PC3	20	35.36
9	PC4	48.32	63.2
10	PC5	29.23	47.22

Berdasarkan pada Tabel 4.68 perbandingan senitivitas LR dan LR+SMOTE maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.69.

Tabel 4.69 Tabel Uji Beda Sensitivitas LR dan LR+SMOTE

t-Test: Paired Two Sample for Means

	LR	*LR+SMOTE*
Mean	31.005	42.53
Variance	402.28865	151.7187111
Observations	10	10
Pearson Correlation	0.868517158	
Hypothesized Mean Difference	95	
df	9	
t Stat	-30.1455634	
P(T<=t) one-tail	1.18938E-10	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	2.37876E-10	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.69 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai t hitung yang diwakili oleh *t* Stat sebesar 30.1455634, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 2.37876E-10, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata sensitivitas LR dan LR+SMOTE, hasil sensitivitas menunjukan LR+SMOTE lebih tinggi dibandingkan dengan LR.

C. *F-Measure* LR dan LR+SMOTE

Uji beda juga dilakukan terhadap *F-Measure* dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata *F-Measure* antara LR dengan LR+SMOTE.

H_1: Ada perbedaan nilai rata-rata *F-Measure* antara LR dengan LR+SMOTE. Pada Tabel 4.70 menunjukan nilai *F-Measure* LR dan LR+SMOTE.

Tabel 4.70 Perbandingan *F-Measure* LR dan LR+SMOTE

NO	Dataset	LR	LR+SMOTE
1	CM1	0.256	0.462
2	JM1	0.173	0.354
3	KC1	0.319	0.478
4	KC3	0.369	0.548
5	MC1	0.376	0.469
6	MC2	0.791	0.675
7	PC1	0.259	0.476
8	PC3	0.277	0.441
9	PC4	0.577	0.697
10	PC5	0.388	0.545

Berdasarkan pada Tabel 4.70 perbandingan *F-Measure* LR dan LR+SMOTE maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.71.

Tabel 4.71 Tabel Uji Beda F-Measure LR dan LR+SMOTE

t-Test: Paired Two Sample for Means

	LR	*LR+SMOTE*
Mean	0.3785	0.5145
Variance	0.032789389	0.0111025
Observations	10	10
Pearson Correlation	0.90941634	
Hypothesized Mean Difference	95	
df	9	
t Stat	-3138.47475	
P(T<=t) one-tail	8.6173E-29	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	1.72346E-28	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.71 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t*

tabel, juga berdasarkan nilai probabilitas. Nilai t hitung yang diwakili oleh *t* Stat sebesar 3138.47475, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 1.72346E-28, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata *F-Measure* LR dan LR+SMOTE, hasil *F-Measure* menunjukan LR+SMOTE lebih tinggi dibandingkan dengan LR.

D. *G-Mean* LR dan LR+SMOTE

Uji beda juga dilakukan terhadap *G-Mean* dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata *G-Mean* antara LR dengan LR+SMOTE.

H_1: Ada perbedaan nilai rata-rata *G-Mean* antara LR dengan LR+SMOTE.

Pada Tabel 4.72 menunjukan nilai *G-Mean* LR dan LR+SMOTE.

Tabel 4.72 Perbandingan *G-Mean* LR dan LR+SMOTE

NO	Dataset	LR	LR+SMOTE
1	CM1	0.448	0.599
2	JM1	0.317	0.478
3	KC1	0.458	0.59
4	KC3	0.547	0.648
5	MC1	0.528	0.581
6	MC2	0.038	0.693
7	PC1	0.421	0.592
8	PC3	0.439	0.573
9	PC4	0.685	0.774
10	PC5	0.539	0.682

Berdasarkan pada Tabel 4.72 perbandingan *G-Mean* LR dan LR+SMOTE maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.73.

Tabel 4.73 Tabel Uji Beda *G-Mean* LR dan LR+SMOTE

t-Test: Paired Two Sample for Means

	LR	*LR+SMOTE*
Mean	0.442	0.621
Variance	0.029606889	0.006631333
Observations	10	10
Pearson Correlation	0.249018615	
Hypothesized Mean Difference	95	
df	9	
t Stat	-1759.56626	
P(T<=t) one-tail	1.57465E-26	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	3.14931E-26	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.73 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 1759.56626, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 3.14931E-26, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata *G-Mean* LR dan LR+SMOTE, hasil *G-Mean* menunjukan LR+SMOTE lebih tinggi dibandingkan dengan LR.

E. AUC LR dan LR+SMOTE

Uji beda juga dilakukan terhadap AUC dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata AUC antara LR dengan LR+SMOTE.

H_1: Ada perbedaan nilai rata-rata AUC antara LR dengan LR+SMOTE.

Pada Tabel 4.74 menunjukan nilai AUC LR dan LR+SMOTE.

Tabel 4.74 Perbandingan AUC LR dan LR+SMOTE

NO	Dataset	LR	LR+SMOTE
1	CM1	0.728	0.727
2	JM1	0.705	0.728
3	KC1	0.8	0.801
4	KC3	0.708	0.669
5	MC1	0.875	0.896
6	MC2	0.863	0.731
7	PC1	0.828	0.835
8	PC3	0.819	0.816
9	PC4	0.907	0.923
10	PC5	0.955	0.956

Berdasarkan pada Tabel 4.74 perbandingan AUC LR dan LR+SMOTE maka dapat dilakukan uji statisik dengan hasil disajikan pada Tabel 4.75.

Tabel 4.75 Tabel Uji Beda AUC LR dan LR+SMOTE

t-Test: Paired Two Sample for Means

	LR	*LR+SMOTE*
Mean	0.8188	0.8082
Variance	0.007261289	0.009102844
Observations	10	10
Pearson Correlation	0.875819065	
Hypothesized Mean Difference	95	
df	9	
t Stat	-6519.04888	
P(T<=t) one-tail	1.19724E-31	
t Critical one-tail	1.833112923	
P(T<=t) two-tail	2.39449E-31	
t Critical two-tail	2.262157158	

Dari hasil uji *t* sampel berpasangan (*paired-sample t-test*) pada Tabel 4.75 dapat diambil kesimpulan hipotesa berdasarkan perbandingan *t* hitung dan *t* tabel, juga berdasarkan nilai probabilitas. Nilai *t* hitung yang diwakili oleh *t* Stat sebesar 6519.04888, 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 ditolak, sedangkan diketahui nilai probabilitas sebesar 2.39449E-31, maka dapat dipatikan bahwa nilai probabilitas < 0,05 yang artinya terdapat perbedaan yang signifikan dari rata-rata AUC LR dan LR+SMOTE, hasil AUC menunjukan LR+SMOTE lebih tinggi dibandingkan dengan LR.

4.2.2 Perbandingan Logistic Regression dengan Algoritma Pengklasifikasi Lain.

Pada penelitian ini dilakukan komparasi untuk mengetahui hasil eksperimen Logistic Regression dengan *Resample* terhadap algoritma pengklasifikasi lain yaitu: *Naive Bayes* (NB), *Linear Discriminant Analysis* (LDA), *k-Nearest Neghbor* (k-NN), C4.5, *Support Vector Machine* (SVM), *Random Forest* (RF), dan K*). Eksperimen yang dilakukan adalah menggunakan *software* Rapid Miner versi 5.3 dengan dataset NASA. Untuk mengetahui hasil perbandingan kinerja dijelaskan pada subbab 4.2.2.1 dan 4.2.2.2.

4.2.2.1 Perbandingan Akurasi untuk Uji *Friedman*

Berdasarkan hasil eksperimen pada penelitian ini, untuk mendapatkan hasil yang terbaik perlu dibandingkan dengan model pengklasifikasi lain yang sudah dilakukan oleh para peneliti lain. Eksperimen yang dilakukan untuk model pengklasifikasi lain dalam penelitian ini menggunakan *software* Rapid Miner versi 5.3. Hasil akurasi dari eksperimen yang dilakukan dapat dilihat pada Tabel 4.76 untuk dataset NASA pada prediksi cacat *software*.

Tabel 4.76 Perbandingan Akurasi Model 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%
LR+SMOTE	80.05%	72.56%	78.03%	76.27%	98.89%	69.59%	88.17%	80.16%	87.57%	95.48%
LR+SpreadSubSample	84.88%	82.15%	85.50%	78.50%	99.35%	72.44%	91.57%	87.28%	91.28%	97.26%
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%

Uji *Friedmen* dilakukan dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata akurasi antara LR dengan pengklasifikasi lain

H_1: Ada perbedaan nilai rata-rata akurasi antara LR dengan pengklasifikasi lain

Uji *Friedmen* dilakukan menggunakan aplikasi XLSTAT untuk hasil akurasi dari semua pengklasifikasi. Tabel 4.77 merupakan hasil dari uji *Friedmen* untuk *pairwise differences*

Tabel 4.77 Hasil Akurasi Uji Friedmen untuk *Pairwise Differences*

	LR	LR+R	LR+SMOTE	LR+SSS	NB	LDA	KNN	C.45	SVM	RF	K*
LR	0	-1.300	5.100	0.450	4.250	5.200	5.500	2.450	0.700	1.050	1.900
LR+R	1.300	0	6.400	1.750	5.550	6.500	6.800	3.750	2.000	2.350	3.200
LR+SMOTE	-5.100	-6.400	0	-4.650	-0.850	0.100	0.400	-2.650	-4.400	-4.050	-3.200
LR+SSS	-0.450	-1.750	4.650	0	3.800	4.750	5.050	2.000	0.250	0.600	1.450
NB	-4.250	-5.550	0.850	-3.800	0	0.950	1.250	-1.800	-3.550	-3.200	-2.350
LDA	-5.200	-6.500	-0.100	-4.750	-0.950	0	0.300	-2.750	-4.500	-4.150	-3.300
KNN	-5.500	-6.800	-0.400	-5.050	-1.250	-0.300	0	-3.050	-4.800	-4.450	-3.600
C.45	-2.450	-3.750	2.650	-2.000	1.800	2.750	3.050	0	-1.750	-1.400	-0.550
SVM	-0.700	-2.000	4.400	-0.250	3.550	4.500	4.800	1.750	0	0.350	1.200
RF	-1.050	-2.350	4.050	-0.600	3.200	4.150	4.450	1.400	-0.350	0	0.850
K*	-1.900	-3.200	3.200	-1.450	2.350	3.300	3.600	0.550	-1.200	-0.850	0

Dari hasil uji Friedmen model klasifikasi terbaik pada semua dataset dicatak tebal. Dalam uji Friedmen didapatkan hasil signifikansi statistik pengujian *P-value* seperti telihat pada Tabel 4.78. Berdasarkan *p-value* dapat menjawab hipotesa H_0

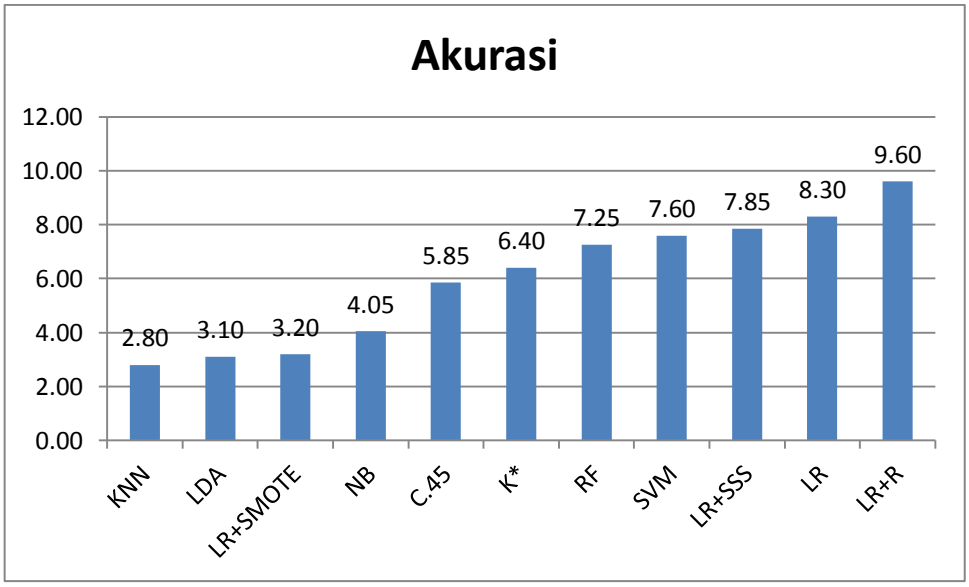
diterima jika *p-value* signifikan dan menolak H_1 ketika p-value kurang signifikan. Dalam penelitian ini akan diatur seberapa signifikan statistik dengan symbol α dengan nilai 0.05. Sehingga dapat dirumuskan jika p-value = α menerima H_0 dan menolak H_1

Dalam hal 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 signifikan α = 0,05, dengan demikian kita harus menolak hipotesis nol, dan itu berarti bahwa ada perbedaan yang signifikan secara statistik. Selanjutnya dilakukan uji Friedmen dengan *Nemenyi post hoc tes* untuk mendeteksi pengklasifikasi tertentu berbeda secara signifikan. Tabel 4.78 memperlihatakan hasil akurasi uji Friedmen untuk *p-value.*

Tabel 4.78 Hasil Akurasi Uji Friedmen untuk *P-values*

	LR	LR+R	LR+SMOTE	LR+SSS	NB	LDA	KNN	C.45	SVM	RF	K*
LR	1	0.999	**0.025**	1.000	0.134	**0.020**	**0.010**	0.860	1.000	1.000	0.972
LR+R	0.999	1	**0.001**	0.985	**0.008**	**0.001**	**0.000**	0.288	0.960	0.889	0.536
LR+SMOTE	**0.025**	**0.001**	1	0.064	1.000	1.000	1.000	0.788	0.103	0.186	0.536
LR+SSS	1.000	0.985	0.064	1	0.269	0.053	**0.028**	0.960	1.000	1.000	0.997
NB	0.134	**0.008**	1.000	0.269	1	1.000	0.999	0.981	0.371	0.536	0.889
LDA	**0.020**	**0.001**	1.000	0.053	1.000	1	1.000	0.747	0.086	0.159	0.487
KNN	**0.010**	**0.000**	1.000	**0.028**	0.999	1.000	1	0.609	**0.047**	0.094	0.349
C.45	0.860	0.288	0.788	0.960	0.981	0.747	0.609	1	0.985	0.997	1.000
SVM	1.000	0.960	0.103	1.000	0.371	0.086	**0.047**	0.985	1	1.000	0.999
RF	1.000	0.889	0.186	1.000	0.536	0.159	0.094	0.997	1.000	1	1.000
K*	0.972	0.536	0.536	0.997	0.889	0.487	0.349	1.000	0.999	1.000	1

Dapat dilihat pada Gambar 4.1, Logistic Regression dengan *Resampling* menghasilkan akurasi tinggi berdasarkan uji *Freidmen.* Berdasarkan Tabel 4.78 dapat ditunjukan bahwa pengklasifiasi Naïve Bayes (NN) dan *Logistic 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.



Gambar 4.1 Grafik Mean Akurasi Uji Friedmen

4.2.2.2 Perbandingan AUC untuk Uji *Friedman*

Berdasarkan hasil ekeperimen pada penelitian ini, untuk mendapatkan hasil yang terbaik perlu dibandingkan dengan model pengklasifikasi lain yang sudah dilakukan oleh para peneliti lain. Eksperimen yang dilakukan untuk model pengklasifikasi lain dalam penelitian ini menggunakan *software* Rapid Miner versi 5.3. Hasil AUC dari eksperimen yang dilakukan dapat dilihat pada Tabel 4.79 untuk dataset NASA pada prediksi cacat *software*.

Tabel 4.79 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+Resample	0.816	0.708	0.791	0.875	0.895	**0.900**	0.854	0.846	**0.926**	0.951
LR+SMOTE	0.727	0.728	0.801	0.669	0.896	0.731	0.835	0.816	0.923	**0.956**
LR+SpreadSubSample	0.761	0.706	0.800	0.679	0.893	0.713	0.830	0.823	0.905	**0.956**
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

Uji *Friedmen* dilakukan dengan menggunakan metode statistik untuk menguji hipotesa:

H_0: Tidak ada perbedaan nilai rata-rata akurasi antara LR dengan pengklasifikasi lain

H_1: Ada perbedaan nilai rata-rata akurasi antara LR dengan pengklasifikasi lain

Uji *Friedmen* dilakukan menggunakan aplikasi XLSTAT untuk hasil AUC dari semua pengklasifikasi. Tabel 4.80 merupakan hasil dari uji *Friedmen* untuk *pairwise differences*

Tabel 4.80 Hasil AUC Uji Friedmen untuk *Pairwise Differences*

	LR	LR+R	LR+SMOTE	LR+SSS	NB	LDA	KNN	C.45	SVM	RF	K*
LR	0	-1.450	-0.600	-0.200	1.150	6.200	6.200	4.800	2.300	4.550	6.200
LR+R	1.450	0	0.850	1.250	2.600	7.650	7.650	6.250	3.750	6.000	7.650
LR+SMOTE	0.600	-0.850	0	0.400	1.750	6.800	6.800	5.400	2.900	5.150	6.800
LR+SSS	0.200	-1.250	-0.400	0	1.350	6.400	6.400	5.000	2.500	4.750	6.400
NB	-1.150	-2.600	-1.750	-1.350	0	5.050	5.050	3.650	1.150	3.400	5.050
LDA	-6.200	-7.650	-6.800	-6.400	-5.050	0	0.000	-1.400	-3.900	-1.650	0.000
KNN	-6.200	-7.650	-6.800	-6.400	-5.050	0.000	0	-1.400	-3.900	-1.650	0.000
C.45	-4.800	-6.250	-5.400	-5.000	-3.650	1.400	1.400	0	-2.500	-0.250	1.400
SVM	-2.300	-3.750	-2.900	-2.500	-1.150	3.900	3.900	2.500	0	2.250	3.900
RF	-4.550	-6.000	-5.150	-4.750	-3.400	1.650	1.650	0.250	-2.250	0	1.650
K*	-6.200	-7.650	-6.800	-6.400	-5.050	0.000	0.000	-1.400	-3.900	-1.650	0

Dari hasil uji Friedmen model klasifikasi terbaik pada semua dataset dicatak tebal. Dalam uji Friedmen didapatkan hasil signifikansi statistik pengujian *P-value* seperti telihat pada Tabel 4.78. 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

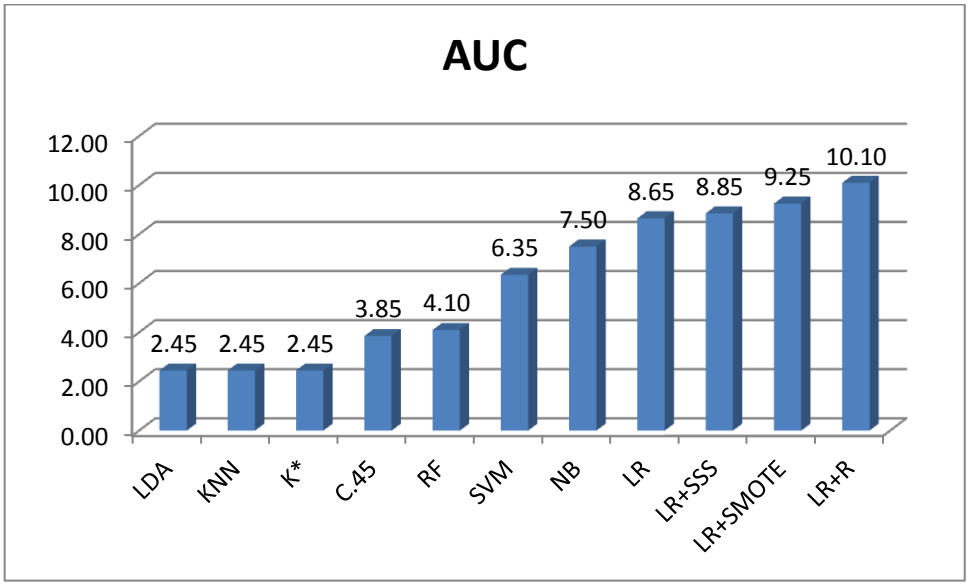
Dalam hal ini, kita mengatur tingkat signifikansi statistik (α) menjadi 0,05. Ini berarti bahwa ada perbedaan yang signifikan secara statistik jika *P-value* <0,05. Dari hasil percobaan, *P-value* adalah 0,0001, ini lebih rendah dari tingkat signifikansi α = 0,05, dengan demikian kita harus menolak hipotesis nol, dan itu berarti bahwa ada perbedaan yang signifikan secara statistik. Selanjutnya

dilakukan uji Friedmen dengan *Nemenyi post hoc tes* untuk mendeteksi pengklasifikasi tertentu berbeda secara signifikan. Tabel 4.81 memperlihatakan hasil AUC uji Friedmen untuk *p-value.*

Tabel 4.81 Hasil AUC Uji Friedmen untuk P-values

	LR	LR+R	LR+SMOTE	LR+SSS	NB	LDA	KNN	C.45	SVM	RF	K*
LR	1	0.997	1.000	1.000	1.000	**0.001**	**0.001**	**0.047**	0.902	0.078	**0.001**
LR+R	0.997	1	1.000	0.999	0.808	**0.0001**	**0.0001**	**0.001**	0.288	**0.003**	**0.0001**
LR+SMOTE	1.000	1.000	1	1.000	0.985	**0.000**	**0.000**	**0.012**	0.680	**0.022**	**0.000**
LR+SSS	1.000	0.999	1.000	1	0.998	**0.001**	**0.001**	**0.031**	0.843	0.053	**0.001**
NB	1.000	0.808	0.985	0.998	1	**0.028**	**0.028**	0.328	1.000	0.439	**0.028**
LDA	**0.001**	**0.0001**	**0.000**	**0.001**	**0.028**	1	1.000	0.997	0.233	0.990	1.000
KNN	**0.001**	**0.0001**	**0.000**	**0.001**	**0.028**	1.000	1	0.997	0.233	0.990	1.000
C.45	**0.047**	**0.001**	**0.012**	**0.031**	0.328	0.997	0.997	1	0.843	1.000	0.997
SVM	0.902	0.288	0.680	0.843	1.000	0.233	0.233	0.843	1	0.915	0.233
RF	0.078	**0.003**	**0.022**	0.053	0.439	0.990	0.990	1.000	0.915	1	0.990
K*	**0.001**	**0.0001**	**0.000**	**0.001**	**0.028**	1.000	1.000	0.997	0.233	0.990	1

Dapat dilihat pada Gambar 4.2, Logistic Regression dengan *Resampling* menghasilkan akurasi tinggi berdasarkan uji *Freidmen.* Dari hasil Perbandingan AUC Tabel 4.80 menunjukan bahwa model LR adalah algoritma yang paling bagus dengan hasil paling tinggi (Laradji, Alshayeb, & Ghouti, 2015; Wahono et al., 2011). Disamping algoritma NB yang juga menunjukan hasil AUC cukup signifikan.

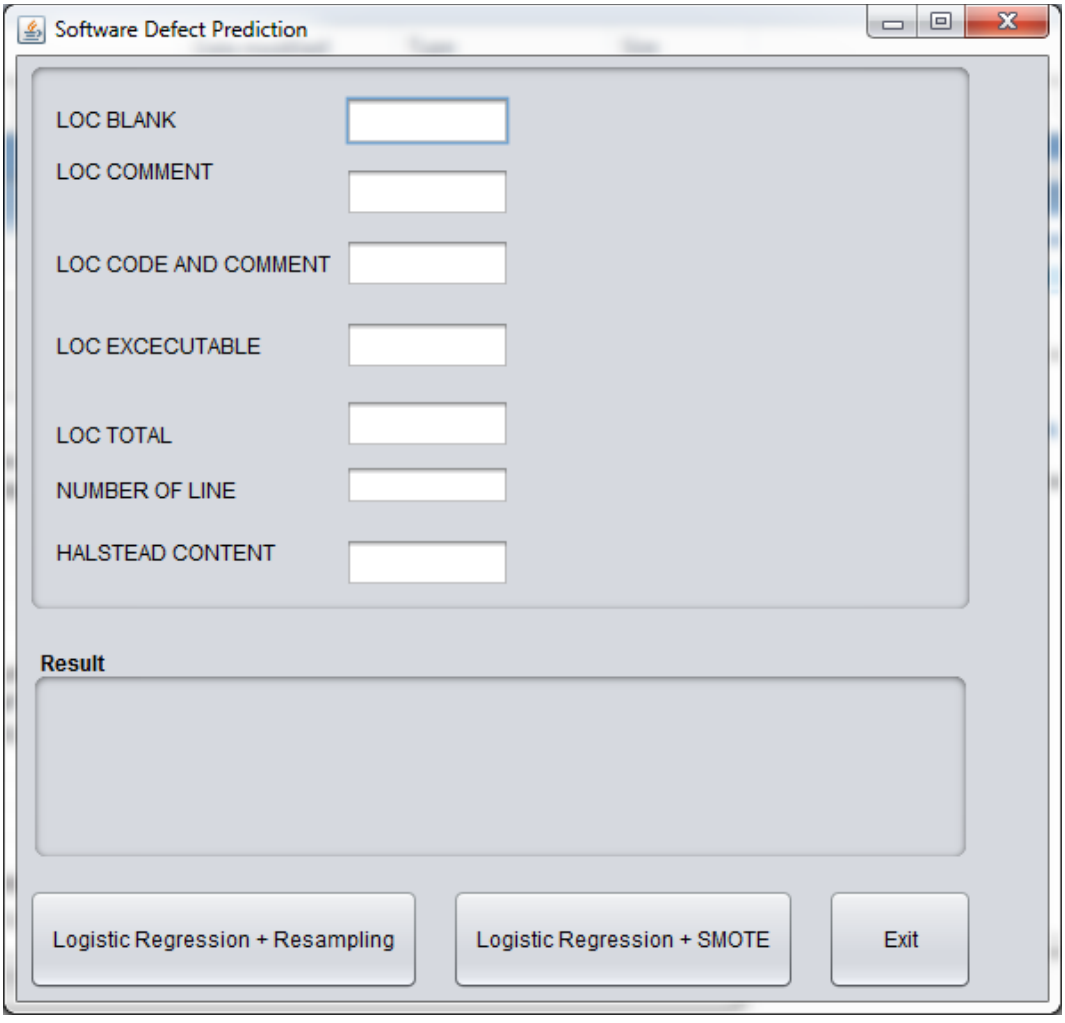


Gambar 4.2 Grafik Mean AUC uji Friedmen

.

4.3 Pengembangan Aplikasi

Selanjutnya dari hasil ekeperimen yang sudah dilakukan, maka dibuatkan model aplikasi prediksi cacat *software* yang dapat dilihat pada Gambar 4.3. Pada form aplikasi yang diinputkan berupa *loc blank, loc comment, loc code and exceutable, loc excutable, loc total, number of line, dan helstead content.* Aplikasi akan menerima inputan nilai, yang kemudian akan diolah saat tombol Logistic Regression + *Resampling* atau Logistic Regression + SMOTE. Maka akan dihasilkan prediksi cacat software yang ditamiplkan pada kolom *result.*



Gambar 4.3 Form Prediksi Cacat *Software*

BAB V
PENUTUP

5.1 Kesimpulan

Dalam pengembangan sebuah *software,* tujuan utamanya adalah menghasilkan *software* yang berkualitas. Kualitas *software* ditentukan dari jumlah cacat yang ditemukan selama pemeriksaan dan pengujian. Sebagian besar biaya pengembangan *software* dihabiskan untuk pemeriksaan dan pengujian (Wahono & Herman, 2014), karena merupakan kegiatan utama untuk menjamin kualitas *software* yang dihasilkan. Telah banyak metode yang digunakan untuk melakukan pemeriksaan dan pengujian. Dalam penelitian ini algoritam *Logistic Regression* digunakan untuk melakukan pengujian berdasarkan probabilitas cacat dan tidak cacat sebuah *software*.

Hasil eksperimen pada penelitian ini mendapatkan 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* sedangkan model LR+SMOTE dan LR+ *SpreadSubSample* menghasilkan nilai sama yaitu 0.956.

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*) nilai AUC *Logistic Regression* menjunkan hasil yang paling bagus, di samping *Naive Bayes* juga menghasilkan nilai AUC yang signifikan.

Dari hasil pengujian di atas maka dapat disimpulkan bahwa penggunaan metode LR+*Resample* mampu menangani ketidakseimbangan kelas pada *Logistic Regression* dengan menghasilkan nilai akurasi dan AUC lebih tinggi dibandingkan dengan metode LR yang tidak menggunakan *Resample*.

5.2 Saran

Penelitian ini telah memberikan beberapa cara untuk menangani ketidakseimbangan kelas pada model prediksi cacat *software*. Beberapa hasil

penelitian mungkin menjadi pendorong adanya penelitian lain. Beberapa saran terkait hasil penelitian ini dan penelitan lebih lanjut untuk menangani ketidakseimbangan kelas pada model prediksi cacat *software* antara lain:

1. *Logistic Regression* dan *Naive Bayes* memilik kinerja yang relatif baik ketika diterapkan pada dataset NASA MDP (Wahono & Herman, 2014). Maka dapat dilakukan penelitian lebih lanjut dengan penambahan selesi fitur (GA dan PSO) pada *Logistic Regression* menangani ketidakseimbangan kelas untuk mendapatkan model prediksi cacat *software* yang lebih baik.
2. Pada *Logistic Regression* dapat diterapkan Model *Robust Kernal* untuk menangani ketidakseimbangan kelas dan rare event data (Maalouf & Trafalis, 2011). Maka dapat dilakukan penelitian lebih lanjut dengan menerapkan model *Robust Kernal dan Resampling* untuk menangani ketidakseimbangan kelas pada prediksi cacat *software.*
3. *Logistic Regression* dapat menagani dataset skala besar namun membutuhkan waktu proses yang lama, oleh karena itu dapat diterapkan model *weighted Logistic Regression* untuk menangani ketidakseimbangan kelas skala besar dengan proses waktu yang relatif lebih cepat (Maalouf & Siddiqi, 2014). Maka dapat dilakukan penelitian lebih lanjut dengan menerapkan *weighted Logistic Regression* pada prediksi cacat *software*.

DAFTAR REFERENSI

Afzal, W., & Torkar, R. (2008). Lessons from applying experimentation in software engineering prediction systems.

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

Catal, C., & Diri, B. (2009). A systematic review of software fault prediction studies. *Expert Systems with Applications*.

Cateni, S., Colla, V., & Vannucci, M. (2014). A method for resampling imbalanced datasets in binary classification tasks for real-world problems. *Neurocomputing*.

Chang, C., & Chu, C. (2007). Defect prevention in software processes : An action-based approach.

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

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*.

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

Dawson, C. W. (2009). *Projects in Computing and Information Systems A Student's Guide Second Edition*. *Information Systems Journal* (Vol. 2). Harlow, England: Addison-Wesley.

Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *The Journal of Machine Learning Research*.

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*.

Freund, R. J., J, W. W., & L, M. D. (2003). *Statistical Methods* (Vol. 2). Academic Press.

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

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., & 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*.

King, G., & Zeng, L. (2001). Logistic Regression in Rare. *Political Analysis*.

Koh, K., Kim, S.-J., & Boyd, S. (2007). An Interior-Point Method for Large-Scale Logistic Regression. *Journal of Machine Learning Research*.

Komarek, P., & Moore, A. (2004). Logistic Regression for Data Mining and High-Dimensional Classification. *School of Computer Science*.

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

Laradji, I. H., Alshayeb, M., & Ghouti, L. (2015). Software defect prediction using ensemble learning on selected features. *Information and Software Technology*.

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., & Siddiqi, M. (2014). Weighted logistic regression for large-scale imbalanced and rare events data. *Knowledge-Based Systems*.

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

MacDonald, M., Musson, R., & Smits, R. (2008). *The Practical Guide to Defect Prevention*. Redmond, Washington: H.B. Fenn and Company Ltd.

Pelayo, L., & Dick, S. (2007). Applying novel resampling strategies to software defect prediction. *Annual Conference of the North American Fuzzy Information Processing Society - NAFIPS*.

Pressman, R. S. (2010). *Software Engineering A Practitioner's Approach Sevent Edition*. New York, NY: McGraw-Hill Companies, Inc.

Saifudin, A. (2014). *Penerapan Pendekatan Level Data dan Algoritma Untuk Penanganan Prediksi Cacat Software Berbasis Naive Bayes*.

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., & Herman, N. S. (2014). Genetic Feature Selection for Software Defect Prediction. *Advanced Science Letters*.

Wahono, R. S., Herman, N. S., & Ahmad, S. (2011). A Comparison Framework of Classification Models for Software Defect Prediction. *Advanced Science Letters*.

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 Practical Mechine Learning Tools and Techniques Third Edition*.

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.

Yen, S. J., & Lee, Y. S. (2009). Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, *36*.

Yu, C. H. (2010). Resampling methods : Concepts , Applications , and Justification What is resampling ? Types of resampling.

Zhang, H., & Wang, Z. (2011). A normal distribution-based over-sampling approach to imbalanced data classification. In *Artificial Intelligence and Lecture Notes in Bioinformatics*.

Lampiran 1 Dataset CM1

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DECISION_DENSITY	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_CO	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	6.0	9.0	2.0	1.0	.0	16.0	5.0	.2	8.0	2.0	3.0	.6	17.0	1.0	.0	24.0	3.0	32.5	9.5	2936.8	.1	63.0	.1	163.2	309.1	.2	4.0	8.0	14.0	.2	19.0	44.0	15.0	15.0	32.0	4.0	25.0	N
2	15.0	7.0	3.0	1.0	19.0	12.0	4.0	.1	6.0	2.0	2.0	.5	17.0	1.0	.0	31.0	1.0	38.6	21.5	17846.2	.3	141.0	.1	991.5	829.5	.3	3.0	6.0	15.0	.1	51.0	90.0	32.0	27.0	67.0	39.2	32.0	Y
3	27.0	9.0	1.0	4.0	22.0	16.0	5.0	.2	8.0	2.0	3.0	.6	18.0	1.0	.0	29.0	.0	52.0	12.3	7914.7	.2	111.0	.1	439.7	641.7	.2	4.0	8.0	15.0	.1	37.0	74.0	33.0	22.0	83.0	47.3	33.0	Y
4	7.0	3.0	2.0	.0	.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	6.0	1.0	.0	12.0	2.0	14.6	6.4	604.4	.0	23.0	.2	33.6	94.0	.5	1.0	2.0	6.0	.1	9.0	14.0	7.0	10.0	20.0	.0	12.0	N
5	51.0	25.0	13.0	.0	14.0	48.0	13.0	.1	24.0	2.0	11.0	.9	96.0	1.0	.0	106.0	2.0	101.3	27.0	74089.9	.9	421.0	.0	4116.1	2739.8	.1	12.0	24.0	85.0	.1	192.0	229.0	71.0	20.0	172.0	11.7	106.0	N
6	3.0	5.0	2.0	.0	.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	12.0	1.0	.0	15.0	1.0	8.2	18.0	2648.7	.1	36.0	.1	147.2	147.2	.3	2.0	4.0	11.0	.2	15.0	21.0	5.0	12.0	19.0	.0	15.0	N
7	13.0	9.0	5.0	12.0	16.0	16.0	5.0	.1	8.0	2.0	5.0	1.0	21.0	5.0	1.0	25.0	4.0	38.5	14.3	7820.9	.2	107.0	.1	434.5	548.8	1.0	4.0	8.0	18.0	.1	38.0	69.0	20.0	15.0	67.0	52.8	37.0	N
8	22.0	29.0	.0	8.0	35.0	44.0	15.0	.3	16.0	2.8	1.0	.1	45.0	3.0	.1	46.0	.0	61.1	22.3	30378.0	.5	239.0	.0	1687.7	1362.4	.2	14.0	22.0	32.0	.1	110.0	129.0	37.0	15.0	112.0	48.3	54.0	N
9	16.0	9.0	2.0	11.0	28.0	12.0	5.0	.1	4.0	3.0	1.0	.2	21.0	5.0	1.0	34.0	2.0	41.2	20.8	17773.1	.3	155.0	.1	987.4	856.2	1.0	4.0	6.0	18.0	.1	59.0	96.0	27.0	19.0	90.0	53.4	45.0	N
10	4.0	3.0	.0	2.0	4.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	10.0	1.0	15.9	9.0	1287.6	.1	35.0	.1	71.5	143.1	.5	1.0	2.0	6.0	.1	16.0	19.0	8.0	9.0	21.0	37.5	12.0	N
11	23.0	13.0	.0	10.0	17.0	24.0	7.0	.2	12.0	2.0	1.0	.1	27.0	5.0	.7	23.0	3.0	27.4	28.1	21659.6	.3	157.0	.0	1203.3	770.4	.7	6.0	12.0	22.0	.1	43.0	114.0	13.0	17.0	74.0	54.0	33.0	N
12	23.0	13.0	.0	10.0	17.0	24.0	7.0	.2	12.0	2.0	1.0	.1	27.0	5.0	.7	23.0	3.0	17.5	27.2	12929.9	.2	105.0	.0	718.3	475.0	.7	6.0	12.0	22.0	.1	35.0	70.0	9.0	14.0	74.0	54.0	33.0	N
13	40.0	9.0	4.0	2.0	15.0	16.0	5.0	.2	8.0	2.0	2.0	.4	21.0	1.0	.0	31.0	.0	47.4	27.5	35852.6	.4	231.0	.0	1991.8	1303.7	.2	4.0	8.0	18.0	.1	70.0	161.0	28.0	22.0	89.0	35.4	33.0	N
14	23.0	9.0	6.0	3.0	20.0	16.0	5.0	.2	8.0	2.0	2.0	.4	23.0	1.0	.0	24.0	.0	43.1	15.2	9958.0	.2	120.0	.1	553.2	655.1	.2	4.0	8.0	20.0	.1	40.0	80.0	25.0	19.0	71.0	48.9	27.0	N
15	8.0	3.0	1.0	6.0	5.0	4.0	2.0	.1	2.0	2.0	1.0	.5	7.0	1.0	.0	13.0	1.0	29.6	9.2	2481.0	.1	57.0	.1	137.8	271.0	.5	1.0	2.0	7.0	.1	17.0	40.0	13.0	14.0	33.0	45.8	19.0	N
16	19.0	3.0	1.0	2.0	.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	23.0	1.0	.0	33.0	.0	64.0	11.7	8685.0	.3	135.0	.1	482.5	745.7	.5	1.0	2.0	23.0	.0	66.0	69.0	34.0	12.0	55.0	5.7	35.0	N
17	6.0	7.0	.0	6.0	.0	10.0	4.0	.2	4.0	2.5	1.0	.3	13.0	4.0	1.0	21.0	1.0	21.3	24.4	12749.0	.2	98.0	.0	708.3	521.5	1.0	3.0	5.0	11.0	.1	40.0	58.0	18.0	22.0	34.0	22.2	27.0	N
18	134.0	60.0	9.0	63.0	31.0	38.0	41.0	.2	16.0	2.4	23.0	.6	118.0	6.0	.1	177.0	5.0	101.8	54.8	305928.6	1.9	767.0	.0	16996.0	5580.8	.2	11.0	19.0	79.0	.1	268.0	499.0	110.0	45.0	406.0	34.7	240.0	N
19	9.0	8.0	2.0	4.0	5.0	8.0	5.0	.3	4.0	2.0	3.0	.6	14.0	1.0	.0	14.0	.0	12.4	15.0	2782.0	.1	41.0	.1	154.6	185.5	.2	2.0	4.0	11.0	.2	16.0	25.0	8.0	15.0	33.0	39.1	18.0	N
20	164.0	110.0	26.0	37.0	191.0	128.0	70.0	.2	54.0	2.4	46.0	.7	241.0	27.0	.4	361.0	1.0	157.0	97.7	1499684.3	5.1	1844.0	.0	83315.8	15345.6	.4	37.0	66.0	173.0	.1	711.0	1133.0	251.0	69.0	764.0	38.7	398.0	N
21	27.0	24.0	11.0	3.0	48.0	34.0	13.0	.2	14.0	2.4	10.0	.8	53.0	7.0	.5	76.0	1.0	60.5	32.2	62767.7	.7	298.0	.0	3487.1	1948.7	.5	10.0	19.0	42.0	.1	102.0	196.0	57.0	36.0	155.0	40.2	79.0	N
22	5.0	5.0	1.0	.0	.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	10.0	1.0	.0	20.0	1.0	34.2	10.3	3620.9	.1	71.0	.1	201.2	351.8	.3	2.0	4.0	9.0	.1	25.0	46.0	17.0	14.0	26.0	.0	20.0	N
23	7.0	9.0	3.0	4.0	5.0	14.0	5.0	.2	6.0	2.3	3.0	.6	20.0	3.0	.5	27.0	.0	24.7	21.9	11802.9	.2	108.0	.1	655.7	540.0	.6	4.0	7.0	17.0	.1	34.0	74.0	14.0	18.0	44.0	25.0	31.0	N
24	16.0	5.0	7.0	2.0	24.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	15.0	1.0	.0	15.0	1.0	36.2	7.2	1867.9	.1	52.0	.1	103.8	260.0	.3	2.0	4.0	14.0	.1	21.0	31.0	19.0	13.0	58.0	63.4	17.0	N
25	15.0	7.0	3.0	3.0	8.0	12.0	4.0	.3	6.0	2.0	4.0	1.0	15.0	1.0	.0	13.0	.0	26.8	11.0	3253.7	.1	59.0	.1	180.8	295.0	.3	3.0	6.0	13.0	.1	25.0	34.0	17.0	15.0	40.0	45.8	16.0	N
26	19.0	5.0	4.0	2.0	16.0	8.0	3.0	.1	4.0	2.0	3.0	1.0	14.0	1.0	.0	25.0	1.0	48.5	14.1	9649.3	.2	120.0	.1	536.1	684.1	.3	2.0	4.0	13.0	.0	49.0	71.0	33.0	19.0	70.0	41.9	27.0	N
27	19.0	7.0	5.0	5.0	10.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	15.0	4.0	1.0	22.0	1.0	28.4	13.4	5120.9	.1	75.0	.1	284.5	381.6	1.0	3.0	6.0	13.0	.1	34.0	41.0	19.0	15.0	57.0	40.5	27.0	N

28	15.0	11.0	4.0	5.0	19.0	20.0	6.0	.2	10.0	2.0	4.0	.7	23.0	1.0	.0	28.0	2.0	30.6	23.1	16332.1	.2	128.0	.0	907.3	707.0	.2	5.0	10.0	19.0	.1	55.0	73.0	25.0	21.0	75.0	46.2	33.0	N
29	13.0	3.0	1.0	3.0	18.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	6.0	1.0	.0	18.0	2.0	25.6	24.4	15282.0	.2	122.0	.0	849.0	625.8	.5	1.0	2.0	6.0	.0	58.0	64.0	19.0	16.0	53.0	53.9	21.0	N
30	14.0	9.0	3.0	2.0	20.0	16.0	5.0	.2	8.0	2.0	3.0	.6	21.0	1.0	.0	24.0	1.0	27.1	39.7	42716.4	.4	197.0	.0	2373.1	1075.5	.2	4.0	8.0	18.0	.1	87.0	110.0	23.0	21.0	68.0	47.8	26.0	N
31	12.0	9.0	4.0	2.0	12.0	16.0	5.0	.1	8.0	2.0	5.0	1.0	22.0	1.0	.0	38.0	1.0	54.4	19.3	20172.8	.4	176.0	.1	1120.7	1047.9	.2	4.0	8.0	19.0	.1	70.0	106.0	40.0	22.0	73.0	26.9	40.0	N
32	23.0	9.0	5.0	10.0	9.0	16.0	5.0	.1	8.0	2.0	5.0	1.0	21.0	5.0	1.0	31.0	1.0	54.6	14.8	12009.6	.3	140.0	.1	667.2	809.4	1.0	4.0	8.0	18.0	.1	61.0	79.0	37.0	18.0	74.0	38.0	41.0	N
33	155.0	71.0	5.0	14.0	170.0	122.0	37.0	.2	58.0	2.1	26.0	.7	148.0	21.0	.6	196.0	1.0	127.7	71.5	653537.1	3.1	1177.0	.0	36307.6	9135.4	.6	32.0	61.0	113.0	.1	454.0	723.0	165.0	52.0	536.0	48.4	210.0	Y
34	42.0	11.0	10.0	5.0	44.0	20.0	6.0	.2	10.0	2.0	6.0	1.0	30.0	1.0	.0	32.0	.0	53.3	12.9	8852.8	.2	123.0	.1	491.8	687.0	.2	5.0	10.0	26.0	.1	47.0	76.0	31.0	17.0	124.0	60.5	37.0	N
35	30.0	19.0	5.0	6.0	45.0	36.0	10.0	.1	18.0	2.0	9.0	.9	48.0	8.0	.8	71.0	1.0	46.2	27.1	33930.4	.4	211.0	.0	1885.0	1251.4	.8	9.0	18.0	40.0	.1	73.0	138.0	35.0	26.0	153.0	41.8	77.0	Y
36	9.0	3.0	2.0	.0	7.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	6.0	1.0	.0	15.0	.0	27.1	14.3	5493.4	.1	74.0	.1	305.2	385.5	.5	1.0	2.0	6.0	.1	27.0	47.0	18.0	19.0	32.0	31.8	15.0	Y
37	26.0	17.0	4.0	5.0	32.0	28.0	9.0	.1	12.0	2.3	6.0	.7	39.0	4.0	.4	62.0	2.0	97.5	20.9	42622.0	.7	314.0	.1	2367.9	2038.4	.4	8.0	14.0	32.0	.1	115.0	199.0	66.0	24.0	126.0	37.4	67.0	N
38	40.0	11.0	12.0	3.0	47.0	20.0	6.0	.1	10.0	2.0	5.0	.8	40.0	1.0	.0	41.0	.0	61.6	17.0	17733.6	.4	177.0	.1	985.2	1045.5	.2	5.0	10.0	36.0	.1	63.0	114.0	39.0	21.0	132.0	55.0	44.0	N
39	7.0	7.0	2.0	.0	8.0	12.0	4.0	.2	6.0	2.0	1.0	.3	16.0	1.0	.0	25.0	.0	31.7	15.0	7125.0	.2	95.0	.1	395.8	475.0	.3	3.0	6.0	14.0	.1	30.0	65.0	16.0	16.0	41.0	24.2	25.0	N
40	93.0	25.0	10.0	11.0	57.0	48.0	13.0	.1	24.0	2.0	12.0	.9	69.0	1.0	.0	143.0	1.0	175.2	28.5	142535.3	1.7	695.0	.0	7918.6	4996.9	.1	12.0	24.0	58.0	.0	290.0	405.0	122.0	24.0	305.0	32.2	154.0	Y
	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
324	.0	7.0	3.0	.0	.0	10.0	4.0	.2	4.0	2.5	3.0	.8	15.0	4.0	1.0	22.0	1.0	41.7	12.8	6780.1	.2	98.0	.1	376.7	531.8	1.0	3.0	5.0	13.0	.2	39.0	59.0	26.0	17.0	23.0	.0	22.0	N
325	2.0	15.0	2.0	1.0	9.0	24.0	8.0	.2	10.0	2.4	6.0	.8	24.0	6.0	.7	41.0	4.0	75.2	14.8	16481.4	.4	187.0	.1	915.6	1113.4	.8	7.0	12.0	18.0	.2	67.0	120.0	43.0	19.0	54.0	19.6	42.0	N
326	2.0	5.0	2.0	2.0	13.0	8.0	3.0	.1	4.0	2.0	3.0	1.0	14.0	1.0	.0	47.0	1.0	55.9	28.0	43785.9	.5	256.0	.0	2432.6	1563.8	.3	2.0	4.0	13.0	.1	112.0	144.0	46.0	23.0	65.0	24.2	49.0	Y
327	5.0	7.0	1.0	.0	.0	10.0	4.0	.2	4.0	2.5	2.0	.5	12.0	3.0	.7	22.0	.0	21.1	14.8	4591.9	.1	64.0	.1	255.1	310.9	.8	3.0	5.0	10.0	.1	24.0	40.0	13.0	16.0	28.0	.0	22.0	N
328	4.0	7.0	2.0	1.0	7.0	10.0	4.0	.2	4.0	2.5	3.0	.8	14.0	3.0	.7	24.0	1.0	30.7	19.1	11241.6	.2	107.0	.1	624.5	587.6	.8	3.0	5.0	12.0	.1	40.0	67.0	23.0	22.0	37.0	25.0	25.0	Y
329	35.0	21.0	9.0	9.0	59.0	36.0	11.0	.1	16.0	2.3	10.0	.9	64.0	3.0	.2	82.0	3.0	70.6	44.7	141084.2	1.1	475.0	.0	7838.0	3155.8	.3	10.0	18.0	55.0	.1	190.0	285.0	68.0	32.0	186.0	45.3	91.0	Y
330	4.0	3.0	2.0	1.0	12.0	4.0	2.0	.2	2.0	2.0	1.0	.5	7.0	1.0	.0	10.0	1.0	23.1	6.5	977.7	.1	32.0	.2	54.3	150.4	.5	1.0	2.0	7.0	.1	13.0	19.0	13.0	13.0	28.0	56.5	11.0	Y
331	.0	11.0	3.0	2.0	7.0	18.0	6.0	.2	8.0	2.3	5.0	.8	20.0	5.0	.8	28.0	1.0	33.2	15.9	8407.1	.2	98.0	.1	467.1	528.5	.8	5.0	9.0	16.0	.2	35.0	63.0	22.0	20.0	38.0	24.3	30.0	Y
332	14.0	9.0	2.0	5.0	1.0	16.0	5.0	.2	8.0	2.0	5.0	1.0	21.0	4.0	.8	28.0	2.0	18.7	21.5	8684.6	.1	83.0	.1	482.5	403.2	.8	4.0	8.0	18.0	.1	35.0	48.0	13.0	16.0	49.0	17.7	33.0	N
333	5.0	3.0	1.0	.0	.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	7.0	1.0	.0	10.0	1.0	10.4	6.8	474.6	.0	19.0	.2	26.4	70.3	.5	1.0	2.0	7.0	.1	6.0	13.0	4.0	9.0	16.0	.0	10.0	N
334	12.0	19.0	2.0	.0	.0	36.0	10.0	.2	18.0	2.0	9.0	.9	47.0	10.0	1.0	50.0	2.0	65.4	11.2	8218.6	.2	136.0	.1	456.6	733.4	1.0	9.0	18.0	39.0	.2	50.0	86.0	29.0	13.0	63.0	.0	50.0	N
335	8.0	19.0	6.0	1.0	.0	32.0	10.0	.3	14.0	2.3	8.0	.8	40.0	10.0	1.0	29.0	2.0	31.3	20.4	13084.4	.2	118.0	.1	726.9	640.3	1.0	9.0	16.0	32.0	.3	47.0	71.0	23.0	20.0	39.0	3.3	30.0	N
336	12.0	19.0	2.0	.0	.0	36.0	10.0	.2	18.0	2.0	9.0	.9	47.0	10.0	1.0	50.0	2.0	74.6	10.2	7697.2	.3	138.0	.1	427.6	757.9	1.0	9.0	18.0	39.0	.2	50.0	88.0	32.0	13.0	63.0	.0	50.0	N
337	7.0	5.0	1.0	2.0	.0	8.0	3.0	.1	4.0	2.0	2.0	.7	10.0	1.0	.0	22.0	2.0	56.0	7.2	2931.9	.1	81.0	.1	162.9	405.0	.3	2.0	4.0	9.0	.1	37.0	44.0	23.0	9.0	32.0	8.3	24.0	N
338	5.0	11.0	3.0	2.0	3.0	18.0	6.0	.3	8.0	2.3	5.0	.8	21.0	4.0	.6	18.0	1.0	26.4	12.1	3845.7	.1	67.0	.1	213.7	318.6	.7	5.0	9.0	17.0	.2	26.0	41.0	14.0	13.0	29.0	21.7	20.0	N
339	3.0	5.0	1.0	.0	.0	8.0	3.0	.3	4.0	2.0	2.0	.7	10.0	1.0	.0	12.0	2.0	15.2	12.0	2191.9	.1	43.0	.1	121.8	182.7	.3	2.0	4.0	9.0	.2	14.0	29.0	7.0	12.0	16.0	.0	12.0	N
340	17.0	21.0	4.0	1.0	8.0	36.0	11.0	.3	16.0	2.3	5.0	.5	43.0	7.0	.6	41.0	5.0	42.2	35.9	54431.2	.5	241.0	.0	3024.0	1514.8	.6	10.0	18.0	34.0	.2	98.0	143.0	45.0	33.0	68.0	18.0	42.0	N
341	16.0	13.0	6.0	.0	5.0	24.0	7.0	.1	12.0	2.0	6.0	.9	31.0	1.0	.0	59.0	4.0	95.6	31.4	94278.1	1.0	449.0	.0	5237.7	3002.2	.1	6.0	12.0	26.0	.1	186.0	263.0	77.0	26.0	81.0	7.8	59.0	N
342	11.0	9.0	6.0	.0	4.0	16.0	5.0	.1	8.0	2.0	5.0	1.0	24.0	1.0	.0	42.0	1.0	66.6	18.5	22884.2	.4	209.0	.1	1271.3	1234.5	.2	4.0	8.0	21.0	.1	80.0	129.0	41.0	19.0	58.0	8.7	42.0	N
343	1.0	3.0	.0	.0	.0	4.0	2.0	.2	2.0	2.0	1.0	.5	5.0	1.0	.0	9.0	2.0	14.8	3.5	181.6	.0	15.0	.3	10.1	51.9	.5	1.0	2.0	5.0	.2	4.0	11.0	4.0	7.0	11.0	.0	9.0	N
344	10.0	13.0	6.0	1.0	16.0	24.0	7.0	.2	12.0	2.0	5.0	.7	29.0	1.0	.0	39.0	5.0	29.5	36.3	38771.2	.4	185.0	.0	2154.0	1069.6	.1	6.0	12.0	24.0	.1	65.0	120.0	26.0	29.0	67.0	30.4	40.0	N

Lampiran 2 Dataset JM1

NO	LOC_BLANK	BRANCH_COUNT	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CYCLOMATIC_COMPLEXITY	DESIGN_COMPLEXITY	ESSENTIAL_COMPLEXITY	LOC_EXECUTABLE	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	LOC_TOTAL	LABEL
1	447.0	826.0	12.0	157.0	470.0	385.0	113.0	2824.0	210.3	384.5	31079782.3	27.0	8441.0	.0	1726654.6	80843.1	3021.0	5420.0	609.0	155.0	3442.0	Y
2	37.0	29.0	8.0	42.0	19.0	19.0	6.0	133.0	108.1	46.3	232043.5	1.7	685.0	.0	12891.3	5009.3	295.0	390.0	121.0	38.0	222.0	Y
3	11.0	405.0	.0	17.0	404.0	2.0	1.0	814.0	101.2	206.0	4294926.5	7.0	2033.0	.0	238607.1	20848.5	813.0	1220.0	811.0	411.0	844.0	Y
4	106.0	240.0	7.0	344.0	127.0	105.0	33.0	952.0	218.2	215.2	10100866.9	15.7	5669.0	.0	561159.3	46943.7	2301.0	3368.0	262.0	49.0	1411.0	Y
5	101.0	464.0	11.0	75.0	263.0	256.0	140.0	1339.0	106.5	337.4	12120796.2	12.0	4308.0	.0	673377.6	35928.1	1556.0	2752.0	226.0	98.0	1532.0	Y
6	67.0	187.0	4.0	1.0	94.0	63.0	27.0	391.0	233.1	58.0	785182.0	4.5	1780.0	.0	43621.2	13527.8	718.0	1062.0	167.0	27.0	466.0	Y
7	105.0	344.0	9.0	40.0	207.0	171.0	58.0	1124.0	122.6	269.5	8901671.3	11.0	3848.0	.0	494537.3	33034.9	1432.0	2416.0	279.0	105.0	1280.0	Y
8	18.0	47.0	.0	10.0	24.0	13.0	1.0	75.0	87.7	30.1	79302.3	.9	438.0	.0	4405.7	2637.8	157.0	281.0	47.0	18.0	107.0	Y
9	39.0	163.0	1.0	6.0	94.0	67.0	3.0	656.0	114.8	174.2	3483952.8	6.7	2702.0	.0	193552.9	19997.2	784.0	1918.0	117.0	52.0	706.0	N
10	143.0	67.0	7.0	49.0	34.0	25.0	1.0	589.0	569.8	49.3	1385089.7	9.4	3281.0	.0	76949.4	28092.7	1522.0	1759.0	355.0	23.0	790.0	Y
11	202.0	503.0	3.0	78.0	286.0	197.0	82.0	1599.0	293.3	188.0	10367011.0	18.4	5392.0	.0	575945.0	55140.8	2243.0	3149.0	1026.0	172.0	1882.0	Y
12	56.0	175.0	4.0	32.0	104.0	86.0	46.0	559.0	106.4	133.5	1896161.7	4.7	1822.0	.0	105342.3	14201.4	645.0	1177.0	157.0	65.0	657.0	Y
13	23.0	39.0	2.0	9.0	20.0	8.0	6.0	91.0	72.8	27.6	55527.3	.7	327.0	.0	3084.9	2011.0	123.0	204.0	49.0	22.0	128.0	Y
14	53.0	338.0	17.0	30.0	173.0	65.0	90.0	622.0	201.6	112.1	2533263.1	7.5	2869.0	.0	140736.8	22597.7	1093.0	1776.0	195.0	40.0	725.0	Y
15	63.0	65.0	14.0	11.0	33.0	17.0	19.0	244.0	174.4	47.7	396298.6	2.8	1180.0	.0	22016.6	8312.4	504.0	676.0	111.0	21.0	334.0	Y
16	4.0	3.0	1.0	1.0	2.0	2.0	1.0	11.0	46.3	4.6	976.7	.1	47.0	.2	54.3	212.6	21.0	26.0	16.0	7.0	20.0	N
17	102.0	36.0	13.0	120.0	25.0	24.0	1.0	384.0	149.6	106.9	1710253.6	5.3	2049.0	.0	95014.1	15997.3	784.0	1265.0	176.0	48.0	621.0	Y
18	11.0	21.0	6.0	12.0	11.0	10.0	5.0	51.0	55.4	27.0	40271.6	.5	255.0	.0	2237.3	1493.8	95.0	160.0	37.0	21.0	82.0	Y
19	5.0	3.0	.0	.0	2.0	2.0	1.0	16.0	29.1	13.0	4911.8	.1	77.0	.1	272.9	377.8	26.0	51.0	15.0	15.0	23.0	Y
20	28.0	45.0	.0	31.0	23.0	15.0	10.0	184.0	233.1	27.4	175381.7	2.1	843.0	.0	9743.4	6394.1	384.0	459.0	168.0	24.0	243.0	Y

21	17.0	17.0	.0	9.0	9.0	6.0	4.0	78.0	79.4	28.1	62666.9	.7	349.0	.0	3481.5	2230.9	149.0	200.0	61.0	23.0	106.0	Y
22	88.0	53.0	.0	51.0	27.0	15.0	17.0	329.0	258.6	46.3	553655.0	4.0	1579.0	.0	30758.6	11964.8	676.0	903.0	168.0	23.0	470.0	Y
23	61.0	7.0	1.0	3.0	4.0	3.0	1.0	221.0	182.6	50.0	456476.8	3.0	1294.0	.0	25359.8	9129.5	565.0	729.0	113.0	20.0	288.0	Y
24	10.0	39.0	.0	4.0	20.0	15.0	16.0	76.0	88.3	25.7	58274.1	.8	362.0	.0	3237.5	2268.6	137.0	225.0	56.0	21.0	93.0	Y
25	32.0	25.0	1.0	20.0	13.0	11.0	11.0	105.0	111.3	38.4	164197.5	1.4	660.0	.0	9122.1	4274.0	234.0	426.0	67.0	22.0	161.0	Y
26	17.0	169.0	1.0	17.0	85.0	58.0	57.0	287.0	209.6	24.6	126435.8	1.7	733.0	.0	7024.2	5147.4	287.0	446.0	111.0	19.0	325.0	Y
27	35.0	97.0	.0	47.0	49.0	35.0	17.0	241.0	149.5	42.6	270569.6	2.1	900.0	.0	15031.7	6359.5	371.0	529.0	109.0	25.0	325.0	Y
28	18.0	17.0	.0	10.0	9.0	7.0	1.0	46.0	39.9	30.0	35902.2	.4	207.0	.0	1994.6	1196.7	72.0	135.0	30.0	25.0	77.0	Y
29	14.0	3.0	1.0	26.0	2.0	2.0	1.0	41.0	40.4	30.0	36384.1	.4	222.0	.0	2021.3	1212.0	79.0	143.0	25.0	19.0	85.0	Y
30	1.0	11.0	.0	35.0	10.0	10.0	1.0	96.0	49.8	69.0	237396.8	1.2	472.0	.0	13188.7	3438.7	153.0	319.0	82.0	74.0	135.0	Y
31	22.0	27.0	.0	17.0	14.0	12.0	6.0	90.0	97.9	29.9	87650.7	1.0	462.0	.0	4869.5	2929.0	171.0	291.0	60.0	21.0	132.0	Y
32	5.0	274.0	.0	13.0	273.0	2.0	1.0	552.0	113.4	140.5	2239582.4	5.3	1644.0	.0	124421.3	15939.0	551.0	1093.0	549.0	280.0	572.0	Y
33	32.0	57.0	.0	43.0	29.0	17.0	22.0	158.0	277.1	27.3	207114.0	2.5	1084.0	.0	11506.3	7575.7	524.0	560.0	115.0	12.0	235.0	Y
34	10.0	7.0	.0	8.0	4.0	3.0	1.0	44.0	67.0	21.2	29982.9	.5	242.0	.1	1665.7	1417.6	94.0	148.0	40.0	18.0	65.0	Y
35	9.0	21.0	2.0	15.0	11.0	7.0	3.0	35.0	28.3	23.3	15415.8	.2	125.0	.0	856.4	660.7	40.0	85.0	18.0	21.0	63.0	Y
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
9577	1.0	3.0	.0	.0	2.0	2.0	1.0	7.0	10.8	6.8	493.3	.0	22.0	.2	27.4	73.1	9.0	13.0	4.0	6.0	10.0	N
9578	1.0	7.0	1.0	.0	4.0	3.0	1.0	23.0	21.1	21.3	9526.6	.2	99.0	.1	529.3	447.8	39.0	60.0	11.0	12.0	27.0	N
9579	3.0	5.0	.0	.0	3.0	2.0	3.0	7.0	13.4	9.6	1247.1	.0	29.0	.1	69.3	129.3	9.0	20.0	7.0	15.0	12.0	N
9580	2.0	5.0	.0	8.0	3.0	1.0	1.0	13.0	16.1	7.7	956.6	.0	31.0	.1	53.1	124.0	12.0	19.0	7.0	9.0	25.0	N
9581	4.0	11.0	2.0	.0	6.0	4.0	1.0	29.0	22.1	23.8	12459.7	.2	104.0	.0	692.2	524.6	35.0	69.0	14.0	19.0	37.0	N
9582	2.0	5.0	.0	7.0	3.0	1.0	1.0	13.0	16.1	7.7	956.6	.0	31.0	.1	53.1	124.0	12.0	19.0	7.0	9.0	24.0	N
9583	4.0	11.0	2.0	.0	6.0	4.0	1.0	29.0	22.1	23.8	12459.7	.2	104.0	.0	692.2	524.6	35.0	69.0	14.0	19.0	37.0	N
9584	16.0	17.0	.0	5.0	9.0	1.0	1.0	65.0	38.1	37.3	53017.4	.5	236.0	.0	2945.4	1421.3	105.0	131.0	38.0	27.0	88.0	Y
9585	3.0	3.0	.0	.0	2.0	1.0	1.0	5.0	22.4	3.7	300.8	.0	21.0	.3	16.7	82.0	11.0	10.0	9.0	6.0	10.0	N
9586	1.0	1.0	.0	.0	1.0	1.0	1.0	2.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	3.0	1.0	3.0	5.0	N
9587	.0	1.0	.0	.0	1.0	1.0	1.0	2.0	7.7	1.5	17.4	.0	5.0	.7	1.0	11.6	2.0	3.0	2.0	3.0	4.0	N
9588	4.0	3.0	.0	.0	2.0	2.0	1.0	18.0	38.9	12.1	5694.9	.2	95.0	.1	316.4	470.7	44.0	51.0	20.0	11.0	24.0	N
9589	2.0	7.0	.0	.0	4.0	4.0	1.0	13.0	32.9	7.3	1770.9	.1	52.0	.1	98.4	241.5	22.0	30.0	15.0	10.0	18.0	N
9590	2.0	3.0	.0	.0	2.0	2.0	1.0	5.0	15.7	8.3	1069.7	.0	30.0	.1	59.4	129.7	11.0	19.0	8.0	12.0	9.0	N
9591	10.0	7.0	.0	1.0	4.0	2.0	1.0	29.0	19.7	26.4	13716.7	.2	103.0	.0	762.0	519.6	44.0	59.0	15.0	18.0	42.0	N
9592	2.0	1.0	.0	.0	1.0	1.0	1.0	6.0	17.4	8.4	1241.6	.1	36.0	.1	69.0	147.2	15.0	21.0	8.0	9.0	10.0	N
9593	2.0	5.0	1.0	.0	3.0	1.0	1.0	13.0	23.6	11.6	3154.7	.1	58.0	.1	175.3	272.6	27.0	31.0	14.0	12.0	19.0	N

Lampiran 3 Dataset KC1

NO	LOC_BLANK	BRANCH_COUNT	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CYCLOMATIC_COMPLEXITY	DESIGN_COMPLEXITY	ESSENTIAL_COMPLEXITY	LOC_EXECUTABLE	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	LOC_TOTAL	Defective
1	.0	1.0	.0	.0	1.0	1.0	1.0	3.0	11.6	2.7	82.4	.0	11.0	.4	4.6	30.9	4.0	7.0	3.0	4.0	5.0	N
2	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
3	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
4	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
5	2.0	1.0	.0	.0	1.0	1.0	1.0	8.0	18.0	3.5	220.9	.0	19.0	.3	12.3	63.1	7.0	12.0	5.0	5.0	12.0	N
6	.0	1.0	.0	.0	1.0	1.0	1.0	3.0	11.6	2.7	82.4	.0	11.0	.4	4.6	30.9	4.0	7.0	3.0	4.0	5.0	N
7	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
8	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
9	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
10	2.0	1.0	.0	.0	1.0	1.0	1.0	9.0	29.4	3.8	414.1	.0	29.0	.3	23.0	110.4	10.0	19.0	8.0	6.0	13.0	N
11	2.0	1.0	.0	.0	1.0	1.0	1.0	8.0	18.0	3.5	220.9	.0	19.0	.3	12.3	63.1	7.0	12.0	5.0	5.0	12.0	N
12	.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	3.0	N
13	2.0	1.0	.0	.0	1.0	1.0	1.0	8.0	18.0	3.5	220.9	.0	19.0	.3	12.3	63.1	7.0	12.0	5.0	5.0	12.0	N
14	1.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	1.0	4.0	N
15	2.0	1.0	.0	.0	1.0	1.0	1.0	8.0	14.0	7.3	753.9	.0	27.0	.1	41.9	102.8	11.0	16.0	6.0	8.0	13.0	N
16	2.0	5.0	.0	.0	3.0	2.0	3.0	12.0	18.9	10.0	1888.7	.1	43.0	.1	104.9	188.9	15.0	28.0	9.0	12.0	16.0	N
17	2.0	5.0	.0	.0	3.0	2.0	3.0	12.0	18.9	10.0	1888.7	.1	43.0	.1	104.9	188.9	15.0	28.0	9.0	12.0	16.0	N
18	2.0	5.0	.0	.0	3.0	2.0	3.0	12.0	18.9	10.0	1888.7	.1	43.0	.1	104.9	188.9	15.0	28.0	9.0	12.0	16.0	N
19	2.0	1.0	.0	.0	1.0	1.0	1.0	10.0	27.3	5.2	737.2	.1	34.0	.2	41.0	141.8	13.0	21.0	10.0	8.0	16.0	N
20	6.0	11.0	.0	1.0	6.0	6.0	5.0	26.0	36.3	11.4	4697.1	.1	85.0	.1	261.0	412.9	28.0	57.0	16.0	13.0	35.0	N

21	6.0	21.0	.0	10.0	11.0	11.0	1.0	65.0	40.3	23.0	21378.6	.3	171.0	.0	1187.7	927.9	64.0	107.0	25.0	18.0	83.0	Y
22	1.0	3.0	.0	.0	2.0	2.0	1.0	10.0	21.2	5.8	708.5	.0	30.0	.2	39.4	122.6	13.0	17.0	9.0	8.0	14.0	N
23	1.0	7.0	.0	.0	4.0	4.0	1.0	32.0	30.5	15.9	7677.2	.2	103.0	.1	426.5	484.2	37.0	66.0	14.0	12.0	40.0	N
24	3.0	7.0	.0	3.0	4.0	4.0	3.0	27.0	42.3	11.6	5663.4	.2	94.0	.1	314.6	489.7	38.0	56.0	23.0	14.0	38.0	N
25	5.0	15.0	.0	2.0	8.0	8.0	6.0	37.0	51.8	14.9	11436.7	.3	141.0	.1	635.4	769.8	52.0	89.0	28.0	16.0	46.0	Y
26	5.0	3.0	.0	.0	2.0	2.0	1.0	19.0	29.0	10.0	2898.9	.1	66.0	.1	161.1	289.9	22.0	44.0	11.0	10.0	26.0	N
27	1.0	3.0	.0	.0	2.0	2.0	1.0	12.0	16.6	9.2	1393.3	.1	38.0	.1	77.4	152.0	11.0	27.0	6.0	10.0	15.0	N
28	4.0	1.0	.0	.0	1.0	1.0	1.0	15.0	20.2	10.3	2139.4	.1	52.0	.1	118.9	208.0	16.0	36.0	7.0	9.0	21.0	N
29	2.0	1.0	.0	.0	1.0	1.0	1.0	7.0	13.9	4.8	319.7	.0	18.0	.2	17.8	66.6	6.0	12.0	5.0	8.0	11.0	N
30	3.0	3.0	.0	.0	2.0	2.0	1.0	12.0	16.2	6.7	720.0	.0	27.0	.2	40.0	108.0	8.0	19.0	6.0	10.0	17.0	N
31	2.0	5.0	.0	.0	3.0	3.0	1.0	21.0	27.3	9.4	2382.0	.1	58.0	.1	132.3	254.8	17.0	41.0	10.0	11.0	25.0	N
32	2.0	19.0	.0	4.0	10.0	10.0	9.0	48.0	46.7	17.7	14573.2	.3	155.0	.1	809.6	824.9	53.0	102.0	24.0	16.0	56.0	N
33	2.0	5.0	.0	.0	3.0	3.0	1.0	21.0	27.3	9.4	2382.0	.1	58.0	.1	132.3	254.8	17.0	41.0	10.0	11.0	25.0	Y
34	4.0	5.0	.0	2.0	3.0	3.0	1.0	35.0	50.5	11.3	6423.7	.2	115.0	.1	356.9	569.7	41.0	74.0	20.0	11.0	43.0	Y
35	4.0	9.0	.0	4.0	5.0	4.0	1.0	49.0	45.1	15.8	11291.1	.2	138.0	.1	627.3	713.5	56.0	82.0	23.0	13.0	59.0	N
36	5.0	23.0	.0	9.0	12.0	11.0	5.0	104.0	77.5	23.3	42021.1	.6	315.0	.0	2334.5	1804.3	118.0	197.0	38.0	15.0	120.0	N
37	2.0	11.0	.0	2.0	6.0	6.0	1.0	41.0	48.7	15.4	11596.3	.3	149.0	.1	644.2	751.6	54.0	95.0	21.0	12.0	48.0	Y
38	2.0	23.0	.0	3.0	12.0	12.0	1.0	62.0	44.5	27.3	33061.9	.4	231.0	.0	1836.8	1212.3	75.0	156.0	22.0	16.0	69.0	Y
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
2083	.0	3.0	.0	.0	2.0	2.0	1.0	12.0	29.5	4.8	680.5	.1	34.0	.2	37.8	141.8	12.0	22.0	10.0	8.0	14.0	N
2084	1.0	3.0	.0	.0	2.0	2.0	1.0	9.0	17.7	6.9	830.5	.0	31.0	.2	46.1	121.1	12.0	19.0	7.0	8.0	14.0	N
2085	.0	1.0	.0	.0	1.0	1.0	1.0	.0	11.2	2.0	44.9	.0	8.0	.5	2.5	22.5	3.0	5.0	3.0	4.0	2.0	N
2086	1.0	3.0	.0	.0	2.0	2.0	1.0	11.0	21.3	5.1	562.6	.0	28.0	.2	31.3	109.4	9.0	19.0	7.0	8.0	14.0	N
2087	.0	1.0	.0	.0	1.0	1.0	1.0	.0	7.7	1.5	17.4	.0	5.0	.7	1.0	11.6	2.0	3.0	2.0	3.0	4.0	N
2088	3.0	31.0	.0	2.0	16.0	16.0	13.0	75.0	67.3	16.5	18211.2	.4	204.0	.1	1011.7	1107.0	85.0	119.0	31.0	12.0	82.0	Y
2089	2.0	3.0	.0	1.0	2.0	2.0	1.0	12.0	25.8	6.8	1197.9	.1	40.0	.2	66.6	175.7	15.0	25.0	11.0	10.0	19.0	N
2090	.0	1.0	.0	.0	1.0	1.0	1.0	6.0	16.3	4.7	355.4	.0	20.0	.2	19.7	76.2	7.0	13.0	6.0	8.0	8.0	Y
2091	2.0	5.0	.0	1.0	3.0	3.0	3.0	18.0	28.8	9.7	2700.6	.1	60.0	.1	150.0	278.6	21.0	39.0	13.0	12.0	23.0	N
2092	.0	1.0	.0	.0	1.0	1.0	1.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	3.0	1.0	3.0	2.0	N
2093	5.0	1.0	.0	.0	1.0	1.0	1.0	6.0	15.2	4.0	243.8	.0	17.0	.3	13.5	60.9	8.0	9.0	6.0	6.0	13.0	N
2094	.0	9.0	.0	.0	5.0	5.0	1.0	18.0	39.5	6.0	1423.1	.1	54.0	.2	79.1	237.2	16.0	38.0	12.0	9.0	20.0	Y
2095	2.0	13.0	.0	.0	7.0	6.0	3.0	32.0	26.8	13.4	4843.4	.1	75.0	.1	269.1	360.6	31.0	44.0	15.0	13.0	36.0	Y
2096	.0	3.0	.0	.0	2.0	2.0	1.0	9.0	17.1	6.0	616.8	.0	27.0	.2	34.3	102.8	9.0	18.0	6.0	8.0	11.0	N

Lampiran 4 Dataset KC3

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DECISION_DENSITY	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	GLOBAL_DATA_COMPLEXITY	GLOBAL_DATA_DENSITY	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	22.0	43.0	20.0	3.0	11.0	38.0	26.0	.2	18.0	2.1	18.0	.7	132.0	15.0	.6	128.0	.0	19.0	.7	109.6	42.0	193686.4	1.5	666.0	.0	10760.4	4608.0	.6	10.0	19.0	108.0	.2	255.0	411.0	91.0	30.0	166.0	9.9	131.0	N
2	2.0	11.0	10.0	.0	1.0	12.0	7.0	.2	6.0	2.0	6.0	.9	38.0	1.0	.0	35.0	.0	5.0	.7	55.7	17.1	16353.0	.3	165.0	.1	908.5	953.9	.1	3.0	6.0	33.0	.2	60.0	105.0	35.0	20.0	39.0	2.8	35.0	N
3	.0	7.0	2.0	.0	.0	4.0	5.0	.2	2.0	2.0	4.0	.8	19.0	3.0	.5	23.0	.0	4.0	.8	29.4	14.4	6090.8	.1	89.0	.1	338.4	423.2	.6	1.0	2.0	16.0	.2	31.0	58.0	14.0	13.0	24.0	.0	23.0	N
4	5.0	18.0	15.0	.0	.0	26.0	10.0	.2	12.0	2.2	10.0	1.0	65.0	5.0	.4	52.0	.0	9.0	.9	80.8	27.1	59492.5	.7	353.0	.0	3305.1	2191.9	.5	7.0	13.0	57.0	.2	137.0	216.0	53.0	21.0	59.0	.0	52.0	N
5	1.0	13.0	10.0	.0	.0	24.0	7.0	.3	12.0	2.0	7.0	1.0	40.0	1.0	.0	22.0	3.0	7.0	1.0	54.5	15.6	13243.5	.3	153.0	.1	735.8	849.9	.1	6.0	12.0	35.0	.3	55.0	98.0	30.0	17.0	25.0	.0	22.0	N
6	5.0	11.0	9.0	.0	11.0	12.0	7.0	.2	6.0	2.0	5.0	.7	40.0	1.0	.0	44.0	3.0	6.0	.9	42.1	32.7	45051.1	.5	236.0	.0	2502.8	1376.6	.1	3.0	6.0	35.0	.1	90.0	146.0	33.0	24.0	62.0	20.0	44.0	N
7	5.0	11.0	11.0	.0	2.0	20.0	6.0	.3	10.0	2.0	5.0	.8	36.0	1.0	.0	22.0	2.0	5.0	.8	35.3	19.6	13490.6	.2	128.0	.1	749.5	690.2	.2	5.0	10.0	32.0	.2	43.0	85.0	22.0	20.0	31.0	8.3	22.0	N
8	3.0	7.0	9.0	.0	2.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	24.0	1.0	.0	17.0	2.0	3.0	.8	33.7	16.2	8841.7	.2	104.0	.1	491.2	545.8	.3	3.0	6.0	22.0	.2	36.0	68.0	20.0	18.0	25.0	10.5	17.0	N
9	1.0	7.0	9.0	.0	.0	12.0	4.0	.3	6.0	2.0	4.0	1.0	24.0	1.0	.0	12.0	3.0	4.0	1.0	44.3	7.3	2383.9	.1	70.0	.1	132.4	325.1	.3	3.0	6.0	22.0	.3	22.0	48.0	15.0	10.0	14.0	.0	12.0	N
10	5.0	11.0	11.0	.0	5.0	20.0	6.0	.2	10.0	2.0	6.0	1.0	38.0	1.0	.0	28.0	1.0	4.0	.7	44.4	18.4	15056.5	.3	157.0	.1	836.5	817.9	.2	5.0	10.0	34.0	.2	54.0	103.0	22.0	15.0	39.0	15.2	28.0	N
11	2.0	5.0	7.0	.0	.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	18.0	1.0	.0	15.0	1.0	3.0	1.0	41.0	14.0	8034.1	.2	111.0	.1	446.3	573.9	.3	2.0	4.0	17.0	.2	44.0	67.0	22.0	14.0	18.0	.0	15.0	N
12	3.0	5.0	9.0	.0	.0	8.0	3.0	.1	4.0	2.0	3.0	1.0	22.0	1.0	.0	21.0	2.0	3.0	1.0	45.8	15.0	10297.9	.2	129.0	.1	572.1	686.5	.3	2.0	4.0	21.0	.1	50.0	79.0	25.0	15.0	25.0	.0	21.0	N
13	3.0	11.0	7.0	.0	.0	18.0	6.0	.3	8.0	2.3	4.0	.7	28.0	4.0	.6	21.0	2.0	4.0	.7	37.7	18.8	13244.9	.2	131.0	.1	735.8	706.4	.7	5.0	9.0	24.0	.2	50.0	81.0	24.0	18.0	25.0	.0	21.0	N
14	11.0	43.0	16.0	.0	4.0	76.0	22.0	.2	34.0	2.2	17.0	.8	119.0	12.0	.5	103.0	3.0	18.0	.8	70.9	48.9	169799.1	1.2	557.0	.0	9433.3	3469.5	.6	21.0	38.0	99.0	.2	208.0	349.0	51.0	24.0	119.0	3.7	103.0	N
15	1.0	5.0	3.0	.0	.0	8.0	3.0	.3	4.0	2.0	3.0	1.0	13.0	1.0	.0	12.0	1.0	3.0	1.0	26.2	9.2	2229.1	.1	52.0	.1	123.8	241.5	.3	2.0	4.0	12.0	.2	20.0	32.0	13.0	12.0	14.0	.0	12.0	N
16	1.0	3.0	3.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	9.0	1.0	.0	8.0	1.0	1.0	.5	18.7	7.7	1110.9	.1	36.0	.1	61.7	144.0	.5	1.0	2.0	9.0	.2	12.0	24.0	7.0	9.0	10.0	.0	8.0	N
17	3.0	11.0	6.0	.0	.0	20.0	6.0	.2	10.0	2.0	6.0	1.0	35.0	1.0	.0	25.0	1.0	6.0	1.0	30.2	25.0	18875.0	.3	151.0	.0	1048.6	755.0	.2	5.0	10.0	31.0	.2	50.0	101.0	16.0	16.0	29.0	.0	25.0	N
18	.0	3.0	4.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	9.0	1.0	.0	8.0	2.0	2.0	1.0	20.8	9.0	1685.7	.1	42.0	.1	93.7	187.3	.5	1.0	2.0	9.0	.2	15.0	27.0	10.0	12.0	9.0	.0	8.0	N
19	.0	3.0	3.0	.0	.0	4.0	2.0	.5	2.0	2.0	2.0	1.0	9.0	1.0	.0	4.0	2.0	2.0	1.0	16.3	4.7	355.4	.0	20.0	.2	19.7	76.2	.5	1.0	2.0	9.0	.4	7.0	13.0	6.0	8.0	5.0	.0	4.0	N
20	5.0	11.0	7.0	.0	.0	20.0	6.0	.3	10.0	2.0	5.0	.8	32.0	1.0	.0	22.0	2.0	4.0	.7	39.5	17.0	11411.2	.2	127.0	.1	634.0	671.3	.2	5.0	10.0	28.0	.2	44.0	83.0	22.0	17.0	28.0	.0	22.0	N
21	1.0	5.0	4.0	.0	.0	8.0	3.0	.3	4.0	2.0	3.0	1.0	13.0	1.0	.0	11.0	1.0	3.0	1.0	23.3	11.4	3011.0	.1	57.0	.1	167.3	264.7	.3	2.0	4.0	12.0	.2	21.0	36.0	12.0	13.0	13.0	.0	11.0	N
22	15.0	33.0	13.0	.0	9.0	58.0	17.0	.2	26.0	2.2	13.0	.8	97.0	12.0	.7	76.0	3.0	10.0	.6	69.5	40.7	115062.6	.9	473.0	.0	6392.4	2827.3	.7	16.0	29.0	82.0	.2	175.0	298.0	43.0	20.0	102.0	10.6	76.0	N
23	15.0	19.0	14.0	1.0	8.0	34.0	10.0	.2	16.0	2.1	9.0	.9	70.0	4.0	.3	64.0	2.0	7.0	.7	59.8	42.4	107406.7	.8	412.0	.0	5967.0	2533.7	.4	9.0	17.0	62.0	.1	156.0	256.0	46.0	25.0	89.0	12.3	65.0	N
24	5.0	19.0	24.0	.0	.0	24.0	11.0	.2	12.0	2.0	11.0	1.0	81.0	7.0	.6	71.0	.0	10.0	.9	122.9	19.9	48468.2	.8	385.0	.1	2692.7	2440.8	.6	6.0	12.0	72.0	.1	139.0	246.0	63.0	18.0	78.0	.0	71.0	N
25	1.0	20.0	11.0	.0	2.0	26.0	11.0	.3	10.0	2.6	8.0	.7	52.0	1.0	.0	39.0	3.0	10.0	.9	60.9	22.0	29456.0	.5	224.0	.1	1636.5	1338.9	.1	8.0	13.0	43.0	.3	82.0	142.0	41.0	22.0	44.0	4.9	39.0	N
26	5.0	3.0	6.0	.0	1.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	17.0	1.0	.0	17.0	1.0	1.0	.5	50.1	14.0	9816.4	.2	139.0	.1	545.4	701.2	.5	1.0	2.0	17.0	.1	56.0	83.0	22.0	11.0	24.0	5.6	17.0	N
27	12.0	11.0	13.0	.0	.0	8.0	7.0	.1	4.0	2.0	5.0	.7	42.0	4.0	.5	54.0	2.0	4.0	.6	88.1	18.2	29080.6	.5	271.0	.1	1615.6	1600.8	.6	2.0	4.0	37.0	.1	109.0	162.0	45.0	15.0	67.0	.0	54.0	N
28	19.0	18.0	23.0	.0	1.0	16.0	11.0	.1	8.0	2.0	9.0	.8	79.0	7.0	.6	88.0	2.0	8.0	.7	151.9	21.7	71508.8	1.1	504.0	.1	3972.7	3295.7	.6	4.0	8.0	70.0	.1	194.0	310.0	76.0	17.0	109.0	1.1	88.0	N
29	9.0	12.0	10.0	.0	2.0	14.0	7.0	.2	6.0	2.3	7.0	1.0	40.0	6.0	.8	30.0	.0	6.0	.9	52.0	24.0	29970.6	.4	216.0	.0	1665.0	1248.8	.9	4.0	7.0	35.0	.2	84.0	132.0	35.0	20.0	42.0	6.3	30.0	N
30	2.0	3.0	5.0	.0	1.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	10.0	1.0	.0	8.0	1.0	2.0	1.0	26.8	8.2	1796.9	.1	50.0	.1	99.8	219.6	.5	1.0	2.0	10.0	.2	18.0	32.0	11.0	10.0	12.0	11.1	8.0	N

31	1.0	6.0	6.0	.0	.0	4.0	4.0	.2	2.0	2.0	3.0	.8	21.0	1.0	.0	20.0	.0	3.0	.8	36.9	14.6	7876.2	.2	106.0	.1	437.6	539.3	.3	1.0	2.0	19.0	.2	37.0	69.0	19.0	15.0	23.0	.0	20.0	N
32	1.0	6.0	6.0	.0	.0	4.0	4.0	.2	2.0	2.0	3.0	.8	18.0	3.0	.7	19.0	.0	3.0	.8	29.3	13.5	5332.5	.1	79.0	.1	296.3	395.0	.8	1.0	2.0	16.0	.2	27.0	52.0	16.0	16.0	22.0	.0	19.0	N
33	28.0	59.0	46.0	1.0	38.0	42.0	36.0	.2	20.0	2.1	34.0	.9	233.0	20.0	.5	241.0	.0	33.0	.9	238.1	40.8	395438.8	3.2	1287.0	.0	21968.8	9702.9	.6	11.0	21.0	199.0	.1	480.0	807.0	159.0	27.0	310.0	13.9	242.0	N
34	12.0	15.0	20.0	.0	16.0	16.0	9.0	.2	8.0	2.0	9.0	1.0	61.0	1.0	.0	61.0	.0	9.0	1.0	90.7	22.5	46024.9	.7	327.0	.0	2556.9	2043.1	.1	4.0	8.0	54.0	.1	118.0	209.0	55.0	21.0	91.0	20.8	61.0	N
35	.0	3.0	4.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	9.0	1.0	.0	8.0	.0	2.0	1.0	20.3	9.8	1943.9	.1	46.0	.1	108.0	198.8	.5	1.0	2.0	9.0	.2	16.0	30.0	9.0	11.0	9.0	.0	8.0	N
36	4.0	5.0	11.0	.0	.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	30.0	1.0	.0	19.0	2.0	3.0	1.0	58.1	12.4	8983.1	.2	142.0	.1	499.1	722.4	.3	2.0	4.0	29.0	.1	52.0	90.0	23.0	11.0	24.0	.0	19.0	N
37	.0	3.0	4.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	9.0	1.0	.0	8.0	.0	2.0	1.0	20.3	9.8	1943.9	.1	46.0	.1	108.0	198.8	.5	1.0	2.0	9.0	.2	16.0	30.0	9.0	11.0	9.0	.0	8.0	N
38	20.0	25.0	29.0	2.0	18.0	36.0	14.0	.1	18.0	2.0	9.0	.6	102.0	5.0	.3	102.0	.0	12.0	.9	163.3	25.1	103097.3	1.4	590.0	.0	5727.6	4103.0	.4	9.0	18.0	90.0	.1	233.0	357.0	102.0	22.0	144.0	16.4	104.0	N
39	2.0	13.0	7.0	.0	.0	12.0	8.0	.2	6.0	2.0	6.0	.8	41.0	3.0	.3	36.0	1.0	7.0	.9	46.1	20.7	19687.6	.3	179.0	.1	1093.8	952.6	.4	3.0	6.0	35.0	.2	62.0	117.0	24.0	16.0	40.0	.0	36.0	N
40	3.0	6.0	4.0	.0	.0	4.0	4.0	.3	2.0	2.0	3.0	.8	15.0	3.0	.7	13.0	1.0	3.0	.8	28.7	7.0	1404.7	.1	45.0	.1	78.0	200.7	.8	1.0	2.0	13.0	.2	14.0	31.0	11.0	11.0	18.0	.0	13.0	N
41	5.0	11.0	9.0	.0	14.0	14.0	6.0	.3	4.0	3.5	6.0	1.0	28.0	5.0	.8	18.0	1.0	5.0	.8	42.8	14.6	9144.9	.2	116.0	.1	508.1	625.5	.8	5.0	8.0	24.0	.2	43.0	73.0	25.0	17.0	39.0	43.8	18.0	N
42	2.0	6.0	4.0	.0	.0	4.0	4.0	.3	2.0	2.0	3.0	.8	15.0	3.0	.7	13.0	1.0	3.0	.8	28.7	7.0	1404.7	.1	45.0	.1	78.0	200.7	.8	1.0	2.0	13.0	.2	14.0	31.0	11.0	11.0	17.0	.0	13.0	N
43	.0	6.0	3.0	.0	.0	4.0	4.0	.3	2.0	2.0	3.0	.8	14.0	1.0	.0	13.0	.0	2.0	.5	21.1	11.3	2664.4	.1	51.0	.1	148.0	236.8	.3	1.0	2.0	12.0	.3	15.0	36.0	10.0	15.0	14.0	.0	13.0	N
44	.0	6.0	4.0	.0	.0	4.0	4.0	.3	2.0	2.0	3.0	.8	15.0	1.0	.0	16.0	.0	2.0	.5	24.1	11.6	3264.5	.1	59.0	.1	181.4	280.5	.3	1.0	2.0	13.0	.2	16.0	43.0	11.0	16.0	17.0	.0	16.0	N
45	3.0	10.0	7.0	.0	2.0	12.0	6.0	.3	6.0	2.0	6.0	1.0	31.0	1.0	.0	23.0	.0	6.0	1.0	39.0	11.0	4689.2	.1	88.0	.1	260.5	427.5	.2	3.0	6.0	27.0	.2	27.0	61.0	16.0	13.0	29.0	8.0	23.0	N
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
181	7.0	9.0	20.0	1.0	.0	16.0	5.0	.1	8.0	2.0	5.0	1.0	49.0	3.0	.5	47.0	.0	4.0	.8	90.1	19.4	33861.9	.6	285.0	.1	1881.2	1746.9	.6	4.0	8.0	46.0	.1	112.0	173.0	52.0	18.0	56.0	2.1	48.0	Y
182	7.0	17.0	21.0	.0	2.0	20.0	10.0	.1	10.0	2.0	10.0	1.0	75.0	7.0	.7	73.0	.0	9.0	.9	118.1	20.6	50170.6	.8	385.0	.1	2787.3	2433.9	.7	5.0	10.0	67.0	.1	142.0	243.0	62.0	18.0	84.0	2.7	73.0	N
183	19.0	27.0	27.0	.0	1.0	52.0	14.0	.1	26.0	2.0	10.0	.7	149.0	1.0	.0	106.0	.0	10.0	.7	125.8	44.8	252155.8	1.9	829.0	.0	14008.7	5632.6	.1	13.0	26.0	137.0	.1	308.0	521.0	86.0	25.0	127.0	.9	106.0	Y
184	1.0	5.0	5.0	.0	3.0	8.0	3.0	.3	4.0	2.0	3.0	1.0	14.0	1.0	.0	11.0	.0	2.0	.7	28.5	9.5	2569.9	.1	59.0	.1	142.8	270.5	.3	2.0	4.0	13.0	.2	19.0	40.0	12.0	12.0	17.0	21.4	11.0	Y
185	5.0	5.0	10.0	1.0	1.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	21.0	1.0	.0	12.0	1.0	3.0	1.0	46.3	10.2	4832.7	.2	93.0	.1	268.5	473.1	.3	2.0	4.0	20.0	.1	33.0	60.0	21.0	13.0	21.0	14.3	13.0	Y
186	2.0	5.0	6.0	.0	7.0	8.0	3.0	.3	4.0	2.0	3.0	1.0	15.0	1.0	.0	12.0	1.0	3.0	1.0	29.1	10.5	3208.1	.1	65.0	.1	178.2	305.5	.3	2.0	4.0	14.0	.1	21.0	44.0	13.0	13.0	23.0	36.8	12.0	Y
187	2.0	8.0	6.0	.0	.0	8.0	5.0	.2	4.0	2.0	4.0	.8	29.0	1.0	.0	25.0	1.0	4.0	.8	42.9	17.7	13404.5	.3	153.0	.1	744.7	758.0	.2	2.0	4.0	26.0	.2	56.0	97.0	19.0	12.0	28.0	.0	25.0	N
188	2.0	7.0	6.0	.0	5.0	12.0	4.0	.3	6.0	2.0	4.0	1.0	19.0	1.0	.0	12.0	1.0	3.0	.8	28.8	11.5	3814.6	.1	69.0	.1	211.9	331.7	.3	3.0	6.0	17.0	.2	23.0	46.0	14.0	14.0	21.0	29.4	12.0	Y
189	.0	3.0	2.0	.0	1.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	9.0	1.0	.0	9.0	2.0	2.0	1.0	18.6	14.9	4119.0	.1	57.0	.1	228.8	276.9	.5	1.0	2.0	9.0	.2	21.0	36.0	12.0	17.0	12.0	10.0	9.0	Y
190	.0	5.0	2.0	.0	1.0	6.0	3.0	.3	2.0	3.0	2.0	.7	12.0	3.0	1.0	10.0	3.0	3.0	1.0	19.1	17.9	6086.8	.1	67.0	.1	338.2	340.9	1.0	2.0	3.0	11.0	.2	25.0	42.0	14.0	20.0	13.0	9.1	10.0	Y
191	5.0	6.0	6.0	1.0	.0	4.0	4.0	.2	2.0	2.0	3.0	.8	19.0	1.0	.0	21.0	.0	3.0	.8	32.2	13.7	6017.7	.2	88.0	.1	334.3	440.0	.3	1.0	2.0	17.0	.1	31.0	57.0	17.0	15.0	28.0	4.6	22.0	Y
192	2.0	6.0	4.0	.0	.0	4.0	4.0	.3	2.0	2.0	3.0	.8	16.0	1.0	.0	16.0	.0	2.0	.5	25.4	14.0	4980.4	.1	74.0	.1	276.7	355.7	.3	1.0	2.0	14.0	.2	28.0	46.0	14.0	14.0	19.0	.0	16.0	Y
193	21.0	41.0	28.0	.0	11.0	64.0	22.0	.2	30.0	2.1	19.0	.9	151.0	8.0	.3	111.0	1.0	16.0	.7	141.9	34.6	169489.0	1.6	715.0	.0	9416.1	4903.5	.4	17.0	32.0	131.0	.2	265.0	450.0	92.0	24.0	144.0	9.0	111.0	Y
194	3.0	5.0	7.0	1.0	2.0	8.0	3.0	.2	4.0	2.0	1.0	.3	18.0	1.0	.0	16.0	.0	3.0	1.0	45.0	14.7	9703.7	.2	125.0	.1	539.1	660.7	.3	2.0	4.0	17.0	.1	47.0	78.0	24.0	15.0	25.0	15.8	17.0	N
195	5.0	8.0	8.0	.0	2.0	8.0	5.0	.2	4.0	2.0	5.0	1.0	28.0	1.0	.0	23.0	.0	5.0	1.0	55.5	8.5	3988.7	.2	91.0	.1	221.6	470.5	.2	2.0	4.0	25.0	.2	30.0	61.0	23.0	13.0	31.0	8.0	23.0	Y
196	19.0	24.0	17.0	2.0	12.0	16.0	15.0	.2	8.0	2.0	9.0	.6	83.0	1.0	.0	86.0	.0	11.0	.7	92.9	35.1	114410.4	1.1	495.0	.0	6356.1	3259.6	.1	4.0	8.0	70.0	.1	189.0	306.0	70.0	26.0	122.0	14.0	88.0	Y
197	1.0	8.0	5.0	.0	1.0	8.0	5.0	.2	4.0	2.0	4.0	.8	22.0	3.0	.5	21.0	1.0	4.0	.8	28.5	10.6	3218.6	.1	66.0	.1	178.8	302.6	.6	2.0	4.0	19.0	.2	18.0	48.0	11.0	13.0	24.0	4.6	21.0	N
198	4.0	9.0	5.0	.0	1.0	14.0	5.0	.3	6.0	2.3	4.0	.8	26.0	3.0	.5	16.0	.0	5.0	1.0	22.8	17.0	6590.0	.1	79.0	.1	366.1	387.6	.6	4.0	7.0	23.0	.2	26.0	53.0	13.0	17.0	23.0	5.9	16.0	N
199	2.0	3.0	2.0	.0	.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	7.0	1.0	.0	9.0	.0	2.0	1.0	15.9	8.6	1189.3	.1	33.0	.1	66.1	137.6	.5	1.0	2.0	7.0	.2	11.0	22.0	7.0	11.0	13.0	.0	9.0	N
200	1.0	7.0	1.0	.0	.0	12.0	4.0	.3	6.0	2.0	4.0	1.0	16.0	1.0	.0	13.0	.0	4.0	1.0	16.1	13.7	3029.4	.1	52.0	.1	168.3	220.9	.3	3.0	6.0	14.0	.3	16.0	36.0	7.0	12.0	15.0	.0	13.0	N

Lampiran 5 Dataset MC1

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	GLOBAL_DATA_COMPLEXITY	GLOBAL_DATA_DENSITY	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
2	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
3	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
4	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
5	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
6	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
7	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
8	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
9	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
10	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
11	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
12	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
13	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
14	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
15	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
16	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
17	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
18	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
19	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N
20	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	.0	.0	.0	1.0	.0	.0	N

21	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	1.0	1.0	1.0	7.4	8.0	470.5	.0	17.0	.1	26.1	58.8	1.0	.0	.0	2.0	.3	6.0	11.0	3.0	8.0	4.0	.0	2.0	N
22	.0	1.0	1.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	3.0	1.0	.0	2.0	1.0	1.0	1.0	7.8	9.0	632.8	.0	19.0	.1	35.2	70.3	1.0	.0	.0	4.0	.3	8.0	11.0	4.0	9.0	4.0	.0	2.0	N
23	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	Y
24	18.0	18.0	3.0	15.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	129.0	38.5	82.0	N
25	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	N
26	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	N
27	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	N
28	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	N
29	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	N
30	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	N
31	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	N
32	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	Y
33	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	Y
34	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	N
35	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	N
36	18.0	18.0	3.0	15.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	129.0	38.5	82.0	N
37	18.0	18.0	3.0	15.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	129.0	38.5	82.0	N
38	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	1.0	1.0	31.0	44.4	61064.3	.5	238.0	.0	3392.5	1376.0	1.0	4.0	8.0	31.0	.0	99.0	139.0	29.0	26.0	128.0	38.0	81.0	N
39	18.0	18.0	3.0	14.0	27.0	16.0	1.0	.0	8.0	1.0	1.0	42.0	1.0	.0	67.0	.0	.0	.0	28.6	47.7	64981.4	.5	238.0	.0	3610.1	1363.3	1.0	4.0	8.0	31.0	.0	99.0	139.0	27.0	26.0	128.0	38.0	81.0	N
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
9256	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9257	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9258	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9259	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9260	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9261	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9262	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9263	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9264	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9265	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9266	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9267	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9268	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9269	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9270	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9271	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9272	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9273	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9274	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9275	1.0	1.0	.0	.0	4.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	11.0	50.0	4.0	N
9276	.0	1.0	.0	2.0	1.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	7.0	60.0	4.0	N
9277	.0	1.0	.0	2.0	1.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	21.7	1.8	70.2	.0	13.0	.6	3.9	39.0	1.0	.0	.0	2.0	.1	6.0	7.0	5.0	3.0	7.0	60.0	4.0	N

Lampiran 6 Dataset MC2

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DECISION_DENSITY	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	GLOBAL_DATA_COMPLEXITY	GLOBAL_DATA_DENSITY	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	2.0	3.0	.0	2.0	7.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	9.0	.0	2.0	1.0	13.5	7.3	725.9	.0	26.0	.1	40.3	99.0	.5	1.0	2.0	6.0	.1	11.0	15.0	6.0	8.0	22.0	50.0	11.0	N
2	1.0	3.0	1.0	1.0	.0	4.0	2.0	.1	2.0	2.0	1.0	.5	7.0	1.0	.0	14.0	.0	2.0	1.0	17.2	7.5	969.5	.0	31.0	.1	53.9	129.3	.5	1.0	2.0	7.0	.1	12.0	19.0	8.0	10.0	18.0	6.7	15.0	N
3	6.0	5.0	1.0	4.0	5.0	8.0	3.0	.2	4.0	2.0	2.0	.7	11.0	1.0	.0	11.0	2.0	3.0	1.0	24.9	11.5	3265.5	.1	63.0	.1	181.4	285.0	.3	2.0	4.0	10.0	.1	25.0	38.0	12.0	11.0	29.0	45.0	15.0	N
4	6.0	5.0	1.0	4.0	3.0	8.0	3.0	.2	4.0	2.0	1.0	.3	11.0	1.0	.0	11.0	1.0	3.0	1.0	19.7	14.3	4019.9	.1	64.0	.1	223.3	281.1	.3	2.0	4.0	10.0	.1	26.0	38.0	10.0	11.0	26.0	38.9	15.0	N
5	2.0	3.0	.0	.0	1.0	4.0	2.0	.3	2.0	2.0	1.0	.5	6.0	1.0	.0	7.0	.0	2.0	1.0	12.1	6.3	479.7	.0	20.0	.2	26.7	76.2	.5	1.0	2.0	6.0	.2	7.0	13.0	5.0	9.0	12.0	12.5	7.0	N
6	18.0	46.0	2.0	6.0	24.0	64.0	26.0	.2	26.0	2.5	15.0	.6	79.0	23.0	.9	114.0	.0	24.0	.9	43.0	57.7	143350.9	.8	395.0	.0	7963.9	2482.7	.9	19.0	33.0	55.0	.2	166.0	229.0	46.0	32.0	164.0	20.8	120.0	Y
7	.0	3.0	.0	.0	.0	4.0	2.0	.4	2.0	2.0	1.0	.5	6.0	1.0	.0	5.0	.0	2.0	1.0	14.0	2.5	87.2	.0	11.0	.4	4.8	34.9	.5	1.0	2.0	6.0	.3	4.0	7.0	4.0	5.0	7.0	.0	5.0	N
8	.0	3.0	.0	.0	1.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	11.0	1.0	2.0	1.0	16.3	7.6	931.6	.0	29.0	.1	51.8	123.2	.5	1.0	2.0	6.0	.1	11.0	18.0	8.0	11.0	14.0	8.3	11.0	N
9	4.0	13.0	3.0	.0	3.0	24.0	7.0	.2	12.0	2.0	7.0	1.0	33.0	6.0	.8	35.0	2.0	7.0	1.0	34.5	12.8	5673.3	.2	91.0	.1	315.2	442.1	.9	6.0	12.0	28.0	.2	42.0	49.0	18.0	11.0	44.0	7.9	35.0	N
10	.0	3.0	.0	.0	1.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	11.0	1.0	2.0	1.0	16.9	8.1	1118.1	.1	33.0	.1	62.1	137.6	.5	1.0	2.0	6.0	.1	13.0	20.0	8.0	10.0	14.0	8.3	11.0	N
11	1.0	19.0	1.0	.0	.0	36.0	10.0	.2	18.0	2.0	3.0	.3	40.0	4.0	.3	43.0	2.0	9.0	.9	41.8	26.7	29766.9	.4	202.0	.0	1653.7	1115.8	.4	9.0	18.0	32.0	.2	83.0	119.0	28.0	18.0	46.0	.0	43.0	N
12	1.0	7.0	1.0	.0	.0	12.0	4.0	.4	6.0	2.0	2.0	.5	16.0	1.0	.0	9.0	1.0	2.0	.5	9.9	20.4	4149.7	.1	47.0	.1	230.5	203.1	.3	3.0	6.0	14.0	.3	22.0	25.0	7.0	13.0	12.0	.0	9.0	N
13	1.0	19.0	1.0	.0	.0	36.0	10.0	.4	18.0	2.0	4.0	.4	41.0	1.0	.0	28.0	2.0	4.0	.4	14.4	37.1	19752.7	.2	112.0	.0	1097.4	532.6	.1	9.0	18.0	33.0	.3	51.0	61.0	11.0	16.0	31.0	.0	28.0	N
14	2.0	9.0	.0	.0	.0	14.0	5.0	.3	6.0	2.3	1.0	.2	15.0	3.0	.5	16.0	2.0	5.0	1.0	32.0	18.8	11252.4	.2	117.0	.1	625.1	600.1	.6	4.0	7.0	12.0	.3	50.0	67.0	20.0	15.0	20.0	.0	16.0	N
15	1.0	21.0	5.0	.0	.0	36.0	11.0	.4	16.0	2.3	7.0	.6	40.0	9.0	.8	28.0	1.0	11.0	1.0	25.1	31.6	25154.2	.3	155.0	.0	1397.5	795.0	.8	10.0	18.0	31.0	.4	67.0	88.0	18.0	17.0	31.0	.0	28.0	N
16	1.0	7.0	1.0	.0	.0	10.0	4.0	.6	4.0	2.5	2.0	.5	12.0	3.0	.7	7.0	1.0	3.0	.8	11.0	13.0	1854.1	.1	33.0	.1	103.0	142.6	.8	3.0	5.0	10.0	.4	14.0	19.0	7.0	13.0	10.0	.0	7.0	N
17	1.0	13.0	.0	6.0	.0	24.0	7.0	.3	12.0	2.0	1.0	.1	28.0	1.0	.0	17.0	2.0	4.0	.6	14.3	38.6	21360.0	.2	115.0	.0	1186.7	552.9	.1	6.0	12.0	23.0	.3	50.0	65.0	11.0	17.0	26.0	26.1	23.0	N
18	.0	3.0	1.0	.0	.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	7.0	1.0	.0	13.0	3.0	2.0	1.0	12.6	12.9	2075.4	.1	38.0	.1	115.3	161.4	.5	1.0	2.0	7.0	.1	15.0	23.0	7.0	12.0	16.0	.0	13.0	N
19	1.0	21.0	4.0	.0	.0	36.0	11.0	.4	16.0	2.3	7.0	.6	40.0	9.0	.8	28.0	1.0	11.0	1.0	22.4	34.5	26651.0	.3	153.0	.0	1480.6	771.8	.8	10.0	18.0	31.0	.4	65.0	88.0	16.0	17.0	31.0	.0	28.0	N
20	13.0	11.0	.0	.0	2.0	20.0	6.0	.4	10.0	2.0	1.0	.2	22.0	1.0	.0	17.0	1.0	5.0	.8	10.6	29.0	8908.6	.1	67.0	.0	494.9	307.2	.2	5.0	10.0	18.0	.2	29.0	38.0	8.0	16.0	34.0	10.5	17.0	N

21	6.0	15.0	1.0	.0	2.0	26.0	8.0	.3	12.0	2.2	2.0	.3	32.0	3.0	.3	31.0	2.0	4.0	.5	33.6	28.1	26512.3	.3	174.0	.0	1472.9	944.2	.4	7.0	13.0	26.0	.2	78.0	96.0	25.0	18.0	41.0	6.1	31.0	N
22	5.0	5.0	.0	.0	7.0	8.0	3.0	.2	4.0	2.0	1.0	.3	9.0	1.0	.0	14.0	2.0	1.0	.3	16.0	6.7	710.7	.0	28.0	.2	39.5	106.6	.3	2.0	4.0	8.0	.1	10.0	18.0	6.0	8.0	28.0	33.3	14.0	Y
23	5.0	5.0	2.0	.0	4.0	8.0	3.0	.4	4.0	2.0	2.0	.7	13.0	1.0	.0	7.0	2.0	3.0	1.0	20.6	8.2	1372.0	.1	42.0	.1	76.2	168.0	.3	2.0	4.0	12.0	.2	21.0	21.0	9.0	7.0	17.0	36.4	7.0	Y
24	4.0	7.0	.0	.0	2.0	12.0	4.0	.3	6.0	2.0	1.0	.3	14.0	3.0	.7	12.0	1.0	2.0	.5	17.2	21.0	7600.6	.1	77.0	.1	422.3	361.9	.8	3.0	6.0	12.0	.2	36.0	41.0	12.0	14.0	20.0	14.3	12.0	Y
25	9.0	11.0	3.0	.0	33.0	20.0	6.0	.3	10.0	2.0	2.0	.3	28.0	1.0	.0	23.0	2.0	5.0	.8	19.5	40.2	31575.9	.3	153.0	.0	1754.2	784.8	.2	5.0	10.0	24.0	.1	76.0	77.0	17.0	18.0	68.0	58.9	23.0	Y
26	22.0	38.0	9.0	2.0	61.0	66.0	20.0	.2	32.0	2.1	10.0	.5	127.0	13.0	.6	124.0	1.0	11.0	.6	59.4	88.8	467827.7	1.8	820.0	.0	25990.4	5269.5	.7	17.0	33.0	109.0	.1	388.0	432.0	59.0	27.0	211.0	33.7	126.0	Y
27	17.0	45.0	2.0	5.0	44.0	86.0	23.0	.4	42.0	2.1	2.0	.1	97.0	5.0	.2	57.0	2.0	9.0	.4	26.4	78.1	161381.3	.7	374.0	.0	8965.6	2065.8	.2	22.0	43.0	76.0	.2	186.0	188.0	25.0	21.0	125.0	46.2	62.0	Y
28	.0	3.0	.0	.0	.0	4.0	2.0	.7	2.0	2.0	1.0	.5	5.0	1.0	.0	3.0	2.0	2.0	1.0	6.2	5.0	154.4	.0	11.0	.2	8.6	30.9	.5	1.0	2.0	5.0	.4	4.0	7.0	2.0	5.0	5.0	.0	3.0	N
29	.0	7.0	.0	.0	.0	10.0	4.0	1.0	4.0	2.5	1.0	.3	11.0	3.0	.7	4.0	3.0	4.0	1.0	7.8	9.0	627.7	.0	22.0	.1	34.9	69.7	.8	3.0	5.0	9.0	.7	9.0	13.0	3.0	6.0	6.0	.0	4.0	N
30	.0	3.0	.0	.0	.0	4.0	2.0	.7	2.0	2.0	1.0	.5	5.0	1.0	.0	3.0	2.0	2.0	1.0	6.2	5.0	154.4	.0	11.0	.2	8.6	30.9	.5	1.0	2.0	5.0	.4	4.0	7.0	2.0	5.0	5.0	.0	3.0	N
31	.0	7.0	.0	.0	.0	10.0	4.0	1.0	4.0	2.5	1.0	.3	11.0	3.0	.7	4.0	3.0	4.0	1.0	7.8	9.0	627.7	.0	22.0	.1	34.9	69.7	.8	3.0	5.0	9.0	.7	9.0	13.0	3.0	6.0	6.0	.0	4.0	N
32	.0	3.0	.0	.0	.0	4.0	2.0	.7	2.0	2.0	1.0	.5	5.0	1.0	.0	3.0	1.0	2.0	1.0	8.4	9.0	677.6	.0	21.0	.1	37.6	75.3	.5	1.0	2.0	5.0	.4	6.0	15.0	3.0	9.0	5.0	.0	3.0	Y
33	7.0	25.0	2.0	.0	18.0	44.0	13.0	.3	20.0	2.2	3.0	.2	55.0	6.0	.4	48.0	1.0	8.0	.6	39.2	58.5	133998.6	.8	400.0	.0	7444.4	2291.2	.5	12.0	22.0	44.0	.2	193.0	207.0	33.0	20.0	75.0	27.3	48.0	Y
34	8.0	27.0	1.0	.0	20.0	52.0	14.0	.4	26.0	2.0	4.0	.3	68.0	1.0	.0	39.0	5.0	11.0	.8	32.2	40.5	52735.3	.4	254.0	.0	2929.7	1302.8	.1	13.0	26.0	56.0	.2	137.0	117.0	22.0	13.0	69.0	33.9	39.0	Y
35	1.0	5.0	1.0	.0	.0	8.0	3.0	.3	4.0	2.0	1.0	.3	11.0	1.0	.0	12.0	3.0	3.0	1.0	15.5	36.3	20339.0	.2	118.0	.0	1130.0	561.1	.3	2.0	4.0	10.0	.2	58.0	60.0	12.0	15.0	15.0	.0	12.0	Y
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
100	5.0	5.0	.0	.0	28.0	8.0	3.0	.3	4.0	2.0	1.0	.3	9.0	1.0	.0	12.0	.0	.0	.0	37.3	9.0	3020.3	.1	79.0	.1	167.8	335.6	.3	2.0	4.0	8.0	.1	20.0	59.0	10.0	9.0	47.0	70.0	12.0	N
101	2.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	8.0	.0	2.0	1.0	17.3	7.3	916.7	.0	34.0	.1	50.9	125.8	.5	1.0	2.0	7.0	.2	17.0	17.0	7.0	6.0	12.0	.0	8.0	N
102	11.0	15.0	3.0	8.0	21.0	28.0	8.0	.1	14.0	2.0	5.0	.6	41.0	1.0	.0	49.0	1.0	2.0	.3	29.0	65.2	123506.5	.6	349.0	.0	6861.5	1893.8	.1	7.0	14.0	35.0	.1	150.0	199.0	23.0	20.0	91.0	37.2	57.0	Y
103	24.0	24.0	.0	3.0	17.0	12.0	20.0	.2	6.0	2.0	1.0	.1	49.0	1.0	.0	91.0	1.0	20.0	1.0	65.6	23.7	36775.6	.5	241.0	.0	2043.1	1552.8	.1	3.0	6.0	31.0	.2	90.0	151.0	57.0	30.0	137.0	18.0	94.0	N
104	1.0	9.0	.0	.0	.0	16.0	5.0	.2	8.0	2.0	1.0	.2	21.0	1.0	.0	23.0	2.0	5.0	1.0	37.0	20.7	15931.2	.3	151.0	.1	885.1	768.2	.2	4.0	8.0	18.0	.2	67.0	84.0	21.0	13.0	26.0	.0	23.0	N
105	9.0	13.0	2.0	.0	14.0	28.0	7.0	.2	14.0	2.0	3.0	.4	28.0	1.0	.0	30.0	.0	7.0	1.0	29.4	18.2	9702.0	.2	99.0	.1	539.0	533.8	.1	7.0	14.0	23.0	.1	44.0	55.0	23.0	19.0	55.0	31.8	30.0	N
106	4.0	7.0	2.0	3.0	3.0	12.0	4.0	.2	6.0	2.0	3.0	.8	16.0	1.0	.0	15.0	.0	1.0	.3	24.0	10.4	2583.7	.1	55.0	.1	143.5	248.8	.3	3.0	6.0	14.0	.2	27.0	28.0	13.0	10.0	27.0	28.6	18.0	N
107	12.0	7.0	.0	.0	16.0	12.0	4.0	.1	6.0	2.0	1.0	.3	16.0	1.0	.0	39.0	2.0	4.0	1.0	18.2	33.7	20674.5	.2	132.0	.0	1148.6	613.0	.3	3.0	6.0	14.0	.1	53.0	79.0	11.0	14.0	69.0	29.1	39.0	N
108	14.0	5.0	9.0	4.0	16.0	8.0	3.0	.1	4.0	2.0	1.0	.3	23.0	1.0	.0	31.0	.0	3.0	1.0	97.3	7.6	5668.7	.3	133.0	.1	314.9	742.8	.3	2.0	4.0	22.0	.0	58.0	75.0	38.0	10.0	67.0	39.2	35.0	N
109	8.0	23.0	3.0	.0	14.0	36.0	12.0	.4	14.0	2.6	5.0	.4	40.0	6.0	.5	32.0	.0	10.0	.8	39.6	24.1	22915.9	.3	168.0	.0	1273.1	953.0	.5	11.0	18.0	30.0	.2	81.0	87.0	32.0	19.0	56.0	30.4	32.0	Y
110	.0	3.0	1.0	.0	.0	4.0	2.0	.4	2.0	2.0	2.0	1.0	6.0	1.0	.0	5.0	1.0	1.0	.5	12.3	4.8	282.3	.0	17.0	.2	15.7	58.8	.5	1.0	2.0	6.0	.3	8.0	9.0	5.0	6.0	7.0	.0	5.0	N
111	6.0	20.0	1.0	1.0	4.0	18.0	13.0	.4	6.0	3.0	7.0	.5	31.0	7.0	.5	31.0	.0	8.0	.6	28.3	19.0	10212.8	.2	101.0	.1	567.4	537.5	.5	6.0	9.0	20.0	.3	38.0	63.0	20.0	20.0	44.0	13.9	32.0	Y
112	13.0	58.0	8.0	15.0	12.0	80.0	32.0	.2	30.0	2.7	21.0	.7	105.0	20.0	.6	120.0	.0	27.0	.8	49.5	51.8	132801.6	.9	408.0	.0	7377.9	2564.4	.6	25.0	40.0	75.0	.2	175.0	233.0	49.0	29.0	162.0	18.4	135.0	Y
113	8.0	7.0	1.0	.0	6.0	12.0	4.0	.2	6.0	2.0	2.0	.5	16.0	1.0	.0	19.0	.0	4.0	1.0	33.7	13.7	6324.9	.2	88.0	.1	351.4	461.8	.3	3.0	6.0	14.0	.1	42.0	46.0	23.0	15.0	35.0	24.0	19.0	N
114	5.0	3.0	.0	.0	3.0	4.0	2.0	.3	2.0	2.0	1.0	.5	6.0	1.0	.0	7.0	.0	2.0	1.0	25.8	4.5	522.0	.0	29.0	.2	29.0	116.0	.5	1.0	2.0	6.0	.1	15.0	14.0	10.0	6.0	17.0	30.0	7.0	N
115	5.0	3.0	.0	.0	3.0	4.0	2.0	.3	2.0	2.0	1.0	.5	6.0	1.0	.0	7.0	.0	2.0	1.0	26.7	4.5	540.0	.0	30.0	.2	30.0	120.0	.5	1.0	2.0	6.0	.1	15.0	15.0	10.0	6.0	17.0	30.0	7.0	N
116	5.0	3.0	.0	.0	3.0	4.0	2.0	.3	2.0	2.0	1.0	.5	6.0	1.0	.0	7.0	.0	2.0	1.0	26.7	4.2	470.4	.0	28.0	.2	26.1	112.0	.5	1.0	2.0	6.0	.1	14.0	14.0	10.0	6.0	17.0	30.0	7.0	N
117	30.0	69.0	17.0	4.0	52.0	118.0	35.0	.3	54.0	2.2	18.0	.5	152.0	17.0	.5	125.0	.0	25.0	.7	84.7	27.6	64595.1	.8	363.0	.0	3588.6	2338.8	.5	32.0	59.0	119.0	.2	145.0	218.0	63.0	24.0	213.0	30.9	129.0	N
118	6.0	9.0	2.0	6.0	4.0	4.0	7.0	.1	2.0	2.0	4.0	.6	22.0	1.0	.0	57.0	1.0	7.0	1.0	51.6	33.0	56139.1	.6	288.0	.0	3118.8	1701.2	.1	1.0	2.0	17.0	.1	132.0	156.0	40.0	20.0	75.0	14.9	63.0	N
119	8.0	11.0	2.0	.0	6.0	20.0	6.0	.3	10.0	2.0	2.0	.3	31.0	1.0	.0	19.0	1.0	6.0	1.0	34.7	15.0	7800.0	.2	104.0	.1	433.3	520.0	.2	5.0	10.0	27.0	.2	50.0	54.0	20.0	12.0	35.0	24.0	19.0	Y
120	16.0	13.0	3.0	11.0	18.0	24.0	7.0	.1	12.0	2.0	2.0	.3	35.0	1.0	.0	53.0	.0	5.0	.7	104.9	24.9	64889.5	.9	407.0	.0	3605.0	2608.6	.1	6.0	12.0	30.0	.1	199.0	208.0	68.0	17.0	100.0	35.4	64.0	N
121	24.0	51.0	3.0	11.0	21.0	82.0	26.0	.3	32.0	2.6	2.0	.1	91.0	13.0	.5	82.0	.0	8.0	.3	93.2	39.5	145307.9	1.2	555.0	.0	8072.7	3679.3	.5	25.0	41.0	67.0	.2	261.0	294.0	76.0	23.0	140.0	28.1	93.0	N
122	1.0	9.0	3.0	.0	.0	16.0	5.0	.2	8.0	2.0	4.0	.8	22.0	5.0	1.0	27.0	.0	4.0	.8	29.2	10.7	3341.5	.1	63.0	.1	185.6	312.1	1.0	4.0	8.0	19.0	.2	26.0	37.0	17.0	14.0	30.0	.0	27.0	Y
123	14.0	24.0	8.0	.0	14.0	30.0	15.0	.3	14.0	2.1	9.0	.6	49.0	3.0	.1	59.0	.0	9.0	.6	46.7	21.2	20950.2	.3	162.0	.1	1163.9	989.6	.2	8.0	15.0	36.0	.2	62.0	100.0	41.0	28.0	89.0	19.2	59.0	Y
124	26.0	31.0	7.0	9.0	21.0	60.0	16.0	.2	30.0	2.0	2.0	.1	83.0	1.0	.0	78.0	.0	7.0	.4	85.2	35.8	109011.3	1.0	478.0	.0	6056.2	3047.3	.1	15.0	30.0	69.0	.1	241.0	237.0	64.0	19.0	136.0	27.8	87.0	Y
125	19.0	13.0	3.0	11.0	19.0	24.0	7.0	.1	12.0	2.0	2.0	.3	35.0	1.0	.0	57.0	.0	5.0	.7	113.8	25.0	71345.7	1.0	439.0	.0	3963.7	2849.9	.1	6.0	12.0	30.0	.1	215.0	224.0	73.0	17.0	108.0	34.5	68.0	N
126	5.0	3.0	3.0	7.0	3.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	10.0	1.0	.0	18.0	.0	1.0	.5	69.2	12.8	11396.1	.3	159.0	.1	633.1	888.0	.5	1.0	2.0	10.0	.1	77.0	82.0	36.0	12.0	35.0	35.7	25.0	N
127	23.0	15.0	3.0	10.0	27.0	28.0	8.0	.1	14.0	2.0	2.0	.3	39.0	1.0	.0	75.0	.0	5.0	.6	136.4	27.4	102014.8	1.2	542.0	.0	5667.5	3730.4	.1	7.0	14.0	33.0	.1	268.0	274.0	98.0	20.0	137.0	33.0	85.0	N

Lampiran 7 Dataset PC1

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DECISION_DENSITY	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	.0	5.0	1.0	.0	.0	8.0	3.0	.4	4.0	2.0	3.0	1.0	10.0	1.0	.0	8.0	2.0	9.4	17.0	2723.1	.1	41.0	.1	151.3	160.2	.3	2.0	4.0	9.0	.3	17.0	24.0	5.0	10.0	9.0	.0	8.0	N
2	1.0	3.0	2.0	.0	.0	4.0	2.0	.4	2.0	2.0	2.0	1.0	8.0	1.0	.0	5.0	.0	13.7	5.7	446.5	.0	20.0	.2	24.8	78.1	.5	1.0	2.0	8.0	.3	10.0	10.0	7.0	8.0	7.0	.0	5.0	N
3	18.0	19.0	5.0	1.0	58.0	34.0	10.0	.2	16.0	2.1	2.0	.2	41.0	5.0	.4	60.0	1.0	61.6	40.2	99691.1	.8	392.0	.0	5538.4	2478.2	.5	9.0	17.0	33.0	.1	177.0	215.0	55.0	25.0	138.0	49.6	61.0	N
4	2.0	3.0	.0	.0	9.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	10.0	.0	45.9	6.8	2118.3	.1	68.0	.2	117.7	311.8	.5	1.0	2.0	6.0	.1	33.0	35.0	17.0	7.0	22.0	47.4	10.0	N
5	36.0	13.0	8.0	.0	42.0	18.0	7.0	.1	8.0	2.3	4.0	.6	38.0	3.0	.3	120.0	1.0	118.3	59.9	424928.0	2.4	1002.0	.0	23607.1	7091.0	.4	5.0	9.0	33.0	.0	458.0	544.0	107.0	28.0	218.0	25.9	120.0	Y
6	43.0	39.0	20.0	1.0	35.0	56.0	22.0	.2	24.0	2.3	17.0	.8	98.0	5.0	.2	107.0	.0	97.3	33.0	106264.8	1.1	464.0	.0	5903.6	3215.9	.2	16.0	28.0	78.0	.1	194.0	270.0	91.0	31.0	221.0	25.2	108.0	N
7	10.0	3.0	5.0	.0	2.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	10.0	1.0	.0	22.0	1.0	49.5	9.5	4487.3	.2	88.0	.1	249.3	471.5	.5	1.0	2.0	10.0	.1	41.0	47.0	28.0	13.0	35.0	8.3	22.0	N
8	34.0	23.0	6.0	.0	31.0	34.0	13.0	.2	16.0	2.1	7.0	.5	60.0	1.0	.0	83.0	2.0	73.3	64.5	305254.2	1.6	703.0	.0	16958.6	4729.7	.1	9.0	17.0	49.0	.1	327.0	376.0	76.0	30.0	174.0	27.2	83.0	N
9	17.0	15.0	5.0	.0	7.0	8.0	11.0	.2	4.0	2.0	7.0	.6	32.0	1.0	.0	54.0	1.0	46.5	27.6	35438.4	.4	214.0	.0	1968.8	1284.0	.1	2.0	4.0	23.0	.1	92.0	122.0	40.0	24.0	79.0	11.5	54.0	N
10	55.0	40.0	5.0	.0	55.0	44.0	25.0	.2	20.0	2.2	13.0	.5	91.0	6.0	.2	153.0	1.0	96.0	61.9	367939.9	2.0	841.0	.0	20441.1	5942.6	.2	12.0	22.0	68.0	.1	379.0	462.0	101.0	33.0	288.0	26.4	153.0	Y
11	32.0	57.0	3.0	.0	39.0	86.0	29.0	.3	30.0	2.9	5.0	.2	89.0	20.0	.7	106.0	2.0	92.2	39.9	146524.1	1.2	539.0	.0	8140.2	3676.1	.7	28.0	53.0	62.0	.2	242.0	297.0	85.0	28.0	178.0	26.9	106.0	Y
12	9.0	11.0	3.0	.0	10.0	20.0	6.0	.2	10.0	2.0	3.0	.5	24.0	3.0	.4	33.0	1.0	43.8	34.3	51695.7	.5	257.0	.0	2872.0	1505.5	.5	5.0	10.0	20.0	.1	121.0	136.0	37.0	21.0	53.0	23.3	33.0	N
13	17.0	24.0	4.0	1.0	16.0	28.0	14.0	.2	12.0	2.3	8.0	.6	47.0	9.0	.6	83.0	2.0	65.3	30.6	61049.3	.7	295.0	.0	3391.6	1996.6	.6	8.0	14.0	35.0	.1	119.0	176.0	72.0	37.0	118.0	17.0	84.0	N
14	225.0	236.0	15.0	2.0	40.0	384.0	136.0	.2	188.0	2.0	123.0	.9	581.0	123.0	.9	600.0	2.0	246.5	105.3	2730637.2	8.7	2785.0	.0	151702.1	25942.7	.9	98.0	192.0	447.0	.2	1144.0	1641.0	538.0	99.0	868.0	6.5	602.0	Y
15	19.0	9.0	1.0	.0	2.0	16.0	5.0	.1	8.0	2.0	1.0	.2	22.0	1.0	.0	54.0	.0	148.4	23.9	84587.1	1.2	507.0	.0	4699.3	3543.3	.2	4.0	8.0	19.0	.1	241.0	266.0	106.0	21.0	76.0	3.6	54.0	N
16	27.0	17.0	3.0	.0	20.0	28.0	9.0	.1	14.0	2.0	3.0	.3	47.0	1.0	.0	84.0	1.0	140.1	46.4	301439.4	2.2	876.0	.0	16746.6	6498.0	.1	7.0	14.0	40.0	.1	419.0	457.0	140.0	31.0	132.0	19.2	84.0	Y
17	39.0	40.0	7.0	4.0	32.0	40.0	25.0	.2	16.0	2.5	14.0	.6	81.0	7.0	.3	103.0	1.0	81.0	52.9	226588.8	1.4	619.0	.0	12588.3	4282.8	.3	12.0	22.0	58.0	.1	260.0	359.0	86.0	35.0	179.0	25.9	107.0	Y
18	31.0	24.0	12.0	.0	27.0	20.0	17.0	.2	10.0	2.0	12.0	.7	56.0	4.0	.2	93.0	1.0	70.2	38.2	102358.8	.9	409.0	.0	5686.6	2680.8	.2	5.0	10.0	41.0	.1	180.0	229.0	66.0	28.0	152.0	22.5	93.0	Y
19	56.0	36.0	19.0	.0	50.0	40.0	24.0	.2	20.0	2.0	18.0	.8	93.0	9.0	.4	141.0	1.0	87.7	51.9	236377.0	1.5	657.0	.0	13132.1	4553.5	.4	10.0	20.0	71.0	.1	292.0	365.0	90.0	32.0	258.0	26.2	141.0	Y
20	5.0	9.0	2.0	.0	7.0	16.0	5.0	.2	8.0	2.0	5.0	1.0	19.0	5.0	1.0	29.0	1.0	43.8	18.6	15166.5	.3	146.0	.1	842.6	815.4	1.0	4.0	8.0	16.0	.1	62.0	84.0	30.0	18.0	42.0	19.4	29.0	Y

21	7.0	9.0	2.0	.0	9.0	16.0	5.0	.2	8.0	2.0	5.0	1.0	19.0	5.0	1.0	29.0	1.0	44.7	20.6	18980.6	.3	164.0	.1	1054.5	920.8	1.0	4.0	8.0	16.0	.1	71.0	93.0	31.0	18.0	46.0	23.7	29.0	Y
22	4.0	3.0	1.0	.0	7.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	13.0	1.0	33.4	14.5	7052.7	.2	98.0	.1	391.8	485.5	.5	1.0	2.0	6.0	.1	46.0	52.0	19.0	12.0	25.0	35.0	13.0	N
23	5.0	3.0	2.0	.0	4.0	4.0	2.0	.1	2.0	2.0	1.0	.5	7.0	1.0	.0	14.0	1.0	41.6	15.9	10529.3	.2	129.0	.1	585.0	661.7	.5	1.0	2.0	7.0	.1	61.0	68.0	23.0	12.0	24.0	22.2	14.0	N
24	10.0	12.0	2.0	.0	18.0	16.0	7.0	.2	8.0	2.0	3.0	.4	25.0	1.0	.0	40.0	1.0	60.7	33.0	66017.1	.7	330.0	.0	3667.6	2001.8	.1	4.0	8.0	20.0	.1	155.0	175.0	47.0	20.0	69.0	31.0	40.0	N
25	7.0	3.0	4.0	.0	4.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	10.0	1.0	.0	14.0	.0	31.5	4.7	702.8	.1	35.0	.2	39.1	148.7	.5	1.0	2.0	10.0	.1	13.0	22.0	11.0	8.0	26.0	22.2	14.0	N
26	6.0	9.0	.0	1.0	4.0	14.0	5.0	.2	6.0	2.3	1.0	.2	16.0	3.0	.5	21.0	2.0	19.6	28.3	15698.6	.2	111.0	.0	872.1	555.0	.6	4.0	7.0	13.0	.2	44.0	67.0	14.0	18.0	33.0	19.2	22.0	N
27	39.0	51.0	6.0	1.0	49.0	100.0	26.0	.2	50.0	2.0	23.0	.9	122.0	1.0	.0	166.0	.0	218.7	22.4	109272.5	1.6	671.0	.0	6070.7	4888.5	.0	25.0	50.0	98.0	.1	304.0	367.0	136.0	20.0	263.0	23.2	167.0	Y
28	37.0	49.0	5.0	13.0	23.0	76.0	25.0	.2	28.0	2.7	9.0	.4	103.0	21.0	.8	97.0	.0	97.7	45.1	198677.1	1.5	663.0	.0	11037.6	4404.9	.8	24.0	38.0	80.0	.2	302.0	361.0	77.0	23.0	171.0	27.1	110.0	Y
29	20.0	12.0	3.0	.0	4.0	14.0	7.0	.3	6.0	2.3	4.0	.6	22.0	1.0	.0	27.0	5.0	23.6	21.5	10872.5	.2	94.0	.1	604.0	506.9	.1	4.0	7.0	17.0	.1	39.0	55.0	20.0	22.0	52.0	12.9	27.0	Y
30	8.0	11.0	2.0	2.0	13.0	18.0	6.0	.2	8.0	2.3	2.0	.3	22.0	4.0	.6	23.0	2.0	53.6	19.0	19256.0	.3	175.0	.1	1069.8	1016.3	.7	5.0	9.0	18.0	.1	80.0	95.0	38.0	18.0	47.0	39.5	25.0	N
31	5.0	11.0	.0	.0	20.0	12.0	7.0	.4	6.0	2.0	1.0	.1	24.0	1.0	.0	20.0	3.0	17.5	39.3	27034.7	.2	134.0	.0	1501.9	687.3	.1	3.0	6.0	19.0	.2	59.0	75.0	15.0	20.0	46.0	50.0	20.0	N
32	14.0	9.0	7.0	.0	7.0	4.0	7.0	.2	2.0	2.0	7.0	1.0	23.0	1.0	.0	42.0	.0	35.8	24.6	21677.1	.3	156.0	.0	1204.3	880.4	.1	1.0	2.0	18.0	.1	68.0	88.0	29.0	21.0	64.0	14.3	42.0	N
33	6.0	11.0	2.0	.0	1.0	20.0	6.0	.2	10.0	2.0	3.0	.5	26.0	1.0	.0	39.0	3.0	65.3	12.0	9407.9	.3	135.0	.1	522.7	784.0	.2	5.0	10.0	22.0	.1	60.0	75.0	40.0	16.0	47.0	2.5	39.0	N
34	1.0	3.0	2.0	.0	3.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	7.0	1.0	.0	7.0	.0	27.6	6.4	1120.0	.1	40.0	.2	62.2	175.7	.5	1.0	2.0	7.0	.2	17.0	23.0	12.0	9.0	12.0	30.0	7.0	N
35	4.0	27.0	.0	1.0	4.0	46.0	14.0	.6	20.0	2.3	1.0	.1	46.0	9.0	.6	21.0	2.0	15.7	32.0	15993.3	.2	102.0	.0	888.5	500.5	.6	13.0	23.0	34.0	.5	37.0	65.0	11.0	19.0	31.0	19.2	22.0	N
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
744	1.0	7.0	.0	.0	.0	12.0	4.0	.2	6.0	2.0	1.0	.3	14.0	1.0	.0	26.0	.0	47.1	21.6	21902.5	.3	185.0	.1	1216.8	1016.0	.3	3.0	6.0	12.0	.1	59.0	126.0	26.0	19.0	28.0	.0	26.0	N
745	1.0	9.0	1.0	.0	.0	16.0	5.0	.3	8.0	2.0	2.0	.4	22.0	1.0	.0	18.0	.0	15.6	23.2	8390.3	.1	77.0	.0	466.1	361.9	.2	4.0	8.0	19.0	.3	34.0	43.0	11.0	15.0	20.0	.0	18.0	N
746	1.0	11.0	1.0	.0	.0	18.0	6.0	.3	8.0	2.3	2.0	.3	25.0	5.0	.8	23.0	.0	18.2	25.4	11715.2	.2	96.0	.0	650.8	461.5	.8	5.0	9.0	21.0	.2	44.0	52.0	13.0	15.0	25.0	.0	23.0	N
747	4.0	24.0	3.0	10.0	1.0	28.0	16.0	.2	14.0	2.0	5.0	.3	49.0	1.0	.0	64.0	.0	36.8	82.8	252471.1	1.0	488.0	.0	14026.2	3049.0	.1	7.0	14.0	35.0	.2	194.0	294.0	41.0	35.0	80.0	14.7	74.0	N
748	1.0	3.0	.0	.0	.0	4.0	2.0	.2	2.0	2.0	1.0	.5	5.0	1.0	.0	9.0	.0	10.5	13.2	1834.5	.1	34.0	.1	101.9	139.0	.5	1.0	2.0	5.0	.2	11.0	23.0	5.0	12.0	11.0	.0	9.0	N
749	1.0	5.0	5.0	1.0	.0	8.0	3.0	.1	4.0	2.0	3.0	1.0	15.0	1.0	.0	21.0	.0	41.7	8.6	3111.4	.1	72.0	.1	172.9	360.0	.3	2.0	4.0	14.0	.1	33.0	39.0	21.0	11.0	24.0	4.6	22.0	N
750	4.0	5.0	.0	.0	6.0	8.0	3.0	.2	4.0	2.0	1.0	.3	19.0	1.0	.0	13.0	.0	16.6	15.8	4150.2	.1	58.0	.1	230.6	262.4	.3	2.0	4.0	18.0	.1	29.0	29.0	11.0	12.0	24.0	31.6	13.0	N
751	1.0	19.0	3.0	.0	.0	30.0	10.0	.8	12.0	2.5	5.0	.5	35.0	6.0	.6	13.0	.0	19.8	20.5	8269.9	.1	85.0	.1	459.4	404.2	.6	9.0	15.0	27.0	.7	38.0	47.0	13.0	14.0	15.0	.0	13.0	N
752	5.0	69.0	8.0	.0	.0	130.0	35.0	.5	62.0	2.1	11.0	.3	141.0	19.0	.5	65.0	.0	70.8	49.5	173359.9	1.2	526.0	.0	9631.1	3502.2	.5	34.0	65.0	108.0	.4	204.0	322.0	68.0	33.0	80.0	.0	65.0	N
753	62.0	71.0	14.0	42.0	24.0	140.0	36.0	.3	70.0	2.0	17.0	.5	183.0	1.0	.0	102.0	.0	83.1	87.5	636348.8	2.4	1070.0	.0	35352.7	7270.0	.0	35.0	70.0	149.0	.2	495.0	575.0	82.0	29.0	239.0	39.3	144.0	N
754	1.0	3.0	.0	3.0	3.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	7.0	1.0	14.5	36.4	19192.7	.2	115.0	.0	1066.3	527.3	.5	1.0	2.0	6.0	.1	52.0	63.0	10.0	14.0	15.0	46.2	10.0	N
755	56.0	71.0	13.0	42.0	24.0	140.0	36.0	.3	70.0	2.0	17.0	.5	181.0	1.0	.0	99.0	.0	79.5	89.8	640223.8	2.4	1056.0	.0	35568.0	7133.2	.0	35.0	70.0	147.0	.2	489.0	567.0	79.0	29.0	230.0	40.0	141.0	N
756	6.0	7.0	2.0	2.0	7.0	12.0	4.0	.1	6.0	2.0	2.0	.5	17.0	1.0	.0	26.0	.0	27.2	19.1	9929.6	.2	108.0	.1	551.6	519.2	.3	3.0	6.0	15.0	.1	51.0	57.0	16.0	12.0	42.0	25.7	28.0	N
757	4.0	20.0	8.0	2.0	2.0	24.0	12.0	.3	12.0	2.0	5.0	.4	48.0	1.0	.0	43.0	.0	47.5	28.6	38870.1	.5	221.0	.0	2159.5	1359.1	.1	6.0	12.0	38.0	.2	99.0	122.0	45.0	26.0	52.0	8.5	45.0	N
758	4.0	11.0	4.0	.0	3.0	18.0	6.0	.3	8.0	2.3	4.0	.7	22.0	4.0	.6	24.0	2.0	29.2	19.5	11102.3	.2	111.0	.1	616.8	569.4	.7	5.0	9.0	18.0	.2	52.0	59.0	20.0	15.0	32.0	11.1	24.0	N
759	2.0	3.0	1.0	.0	1.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	5.0	1.0	.0	7.0	.0	17.8	5.6	562.9	.0	24.0	.2	31.3	100.1	.5	1.0	2.0	5.0	.2	9.0	15.0	8.0	10.0	11.0	12.5	7.0	N

Lampiran 8 Dataset PC3

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DECISION_DENSITY	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	14.0	11.0	4.0	3.0	.0	20.0	6.0	.2	10.0	2.0	3.0	.5	26.0	1.0	.0	23.0	1.0	41.9	14.2	8403.0	.2	110.0	.1	466.8	593.2	.2	5.0	10.0	22.0	.2	51.0	59.0	27.0	15.0	41.0	11.5	26.0	N
2	6.0	3.0	1.0	1.0	3.0	4.0	2.0	.5	2.0	2.0	2.0	1.0	5.0	1.0	.0	3.0	1.0	12.2	3.0	109.6	.0	11.0	.3	6.1	36.5	.5	1.0	2.0	5.0	.1	4.0	7.0	4.0	6.0	14.0	57.1	4.0	N
3	14.0	19.0	6.0	5.0	12.0	24.0	11.0	.3	10.0	2.4	9.0	.8	37.0	3.0	.2	34.0	2.0	37.3	23.9	21248.5	.3	152.0	.0	1180.5	890.4	.3	7.0	12.0	28.0	.2	63.0	89.0	33.0	25.0	66.0	33.3	39.0	N
4	20.0	17.0	2.0	3.0	12.0	20.0	10.0	.3	8.0	2.5	3.0	.3	31.0	3.0	.2	36.0	5.0	31.4	19.3	11716.3	.2	107.0	.1	650.9	607.0	.3	6.0	10.0	23.0	.1	47.0	60.0	28.0	23.0	72.0	29.4	39.0	N
5	6.0	3.0	6.0	1.0	.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	10.0	1.0	.0	10.0	1.0	31.9	6.9	1518.5	.1	48.0	.1	84.4	220.1	.5	1.0	2.0	10.0	.1	23.0	25.0	15.0	9.0	18.0	9.1	11.0	Y
6	17.0	7.0	4.0	4.0	7.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	22.0	1.0	.0	16.0	1.0	46.8	14.5	9823.4	.2	125.0	.1	545.8	678.3	.3	3.0	6.0	20.0	.1	60.0	65.0	29.0	14.0	45.0	40.7	20.0	N
7	7.0	5.0	2.0	2.0	4.0	8.0	3.0	.3	4.0	2.0	3.0	1.0	11.0	1.0	.0	8.0	1.0	17.8	11.1	2188.1	.1	42.0	.1	121.6	197.4	.3	2.0	4.0	10.0	.1	19.0	23.0	12.0	14.0	22.0	42.9	10.0	N
8	3.0	5.0	6.0	1.0	.0	6.0	3.0	.3	2.0	3.0	2.0	.7	12.0	1.0	.0	10.0	.0	33.0	6.2	1260.5	.1	48.0	.2	70.0	203.9	.3	2.0	3.0	11.0	.2	17.0	31.0	11.0	8.0	15.0	9.1	11.0	N
9	2.0	11.0	.0	7.0	6.0	14.0	6.0	.2	4.0	3.5	1.0	.2	16.0	4.0	.6	23.0	7.0	36.8	15.6	8933.3	.2	107.0	.1	496.3	573.3	.7	5.0	8.0	12.0	.2	44.0	63.0	24.0	17.0	39.0	36.1	30.0	N
10	9.0	11.0	2.0	8.0	15.0	20.0	6.0	.1	10.0	2.0	3.0	.5	24.0	1.0	.0	40.0	.0	56.5	17.4	17102.1	.3	180.0	.1	950.1	982.7	.2	5.0	10.0	20.0	.1	83.0	97.0	31.0	13.0	73.0	36.5	48.0	N
11	.0	5.0	2.0	.0	.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	11.0	1.0	.0	13.0	6.0	23.4	7.3	1260.6	.1	38.0	.1	70.0	171.9	.3	2.0	4.0	10.0	.2	16.0	22.0	12.0	11.0	14.0	.0	13.0	N
12	8.0	8.0	3.0	.0	1.0	4.0	6.0	.2	2.0	2.0	6.0	1.0	17.0	1.0	.0	39.0	.0	55.8	18.8	19738.6	.4	186.0	.1	1096.6	1049.8	.2	1.0	2.0	13.0	.1	73.0	113.0	33.0	17.0	49.0	2.5	39.0	N
13	10.0	7.0	5.0	1.0	3.0	10.0	4.0	.1	4.0	2.5	4.0	1.0	16.0	4.0	1.0	28.0	1.0	45.1	11.1	5572.1	.2	93.0	.1	309.6	501.5	1.0	3.0	5.0	14.0	.1	40.0	53.0	27.0	15.0	51.0	12.5	29.0	N
14	11.0	3.0	.0	4.0	9.0	4.0	2.0	.0	2.0	2.0	1.0	.5	6.0	1.0	.0	41.0	1.0	135.5	10.0	13474.9	.5	249.0	.1	748.6	1351.1	.5	1.0	2.0	6.0	.0	123.0	126.0	37.0	6.0	66.0	24.1	45.0	N
15	2.0	3.0	2.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	7.0	1.0	.0	8.0	1.0	18.5	9.0	1501.2	.1	40.0	.1	83.4	166.8	.5	1.0	2.0	7.0	.2	18.0	22.0	9.0	9.0	11.0	.0	8.0	N
16	6.0	3.0	2.0	.0	5.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	7.0	1.0	.0	16.0	1.0	57.3	10.2	5954.4	.2	113.0	.1	330.8	584.2	.5	1.0	2.0	7.0	.1	53.0	60.0	26.0	10.0	35.0	23.8	16.0	N
17	2.0	3.0	2.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	7.0	1.0	.0	8.0	1.0	18.5	9.0	1501.2	.1	40.0	.1	83.4	166.8	.5	1.0	2.0	7.0	.2	18.0	22.0	9.0	9.0	11.0	.0	8.0	N
18	2.0	3.0	2.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	7.0	1.0	.0	8.0	1.0	18.5	9.0	1501.2	.1	40.0	.1	83.4	166.8	.5	1.0	2.0	7.0	.2	18.0	22.0	9.0	9.0	11.0	.0	8.0	N
19	2.0	3.0	2.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	7.0	1.0	.0	8.0	1.0	18.5	9.0	1501.2	.1	40.0	.1	83.4	166.8	.5	1.0	2.0	7.0	.2	18.0	22.0	9.0	9.0	11.0	.0	8.0	N
20	6.0	5.0	3.0	1.0	1.0	8.0	3.0	.2	4.0	2.0	2.0	.7	11.0	1.0	.0	12.0	2.0	25.2	11.2	3155.7	.1	58.0	.1	175.3	281.8	.3	2.0	4.0	10.0	.1	24.0	34.0	15.0	14.0	21.0	14.3	13.0	N

21	7.0	25.0	10.0	4.0	16.0	42.0	13.0	.2	18.0	2.3	10.0	.8	47.0	5.0	.3	71.0	1.0	70.9	18.6	24562.8	.4	210.0	.1	1364.6	1319.9	.4	12.0	23.0	36.0	.1	89.0	121.0	55.0	23.0	99.0	22.0	75.0	Y
22	8.0	5.0	2.0	1.0	9.0	8.0	3.0	.2	4.0	2.0	2.0	.7	12.0	1.0	.0	13.0	2.0	30.8	10.5	3393.0	.1	63.0	.1	188.5	323.1	.3	2.0	4.0	11.0	.1	28.0	35.0	20.0	15.0	32.0	43.5	14.0	N
23	13.0	7.0	.0	.0	2.0	12.0	4.0	.0	6.0	2.0	1.0	.3	19.0	1.0	.0	397.0	.0	1535.5	11.1	188118.2	5.7	1916.0	.1	10451.0	16995.6	.3	3.0	6.0	17.0	.0	1129.0	787.0	459.0	9.0	413.0	.5	397.0	N
24	22.0	55.0	13.0	.0	51.0	108.0	28.0	.0	54.0	2.0	27.0	1.0	125.0	27.0	1.0	817.0	2.0	1350.0	39.9	2153049.9	18.0	5580.0	.0	119613.9	53912.7	1.0	27.0	54.0	99.0	.0	2733.0	2847.0	787.0	23.0	891.0	5.9	817.0	N
25	.0	5.0	1.0	.0	.0	8.0	3.0	.4	4.0	2.0	3.0	1.0	10.0	1.0	.0	8.0	2.0	9.4	17.0	2723.1	.1	41.0	.1	151.3	160.2	.3	2.0	4.0	9.0	.3	17.0	24.0	5.0	10.0	9.0	.0	8.0	N
26	1.0	3.0	2.0	.0	.0	4.0	2.0	.4	2.0	2.0	2.0	1.0	8.0	1.0	.0	5.0	.0	13.7	5.7	446.5	.0	20.0	.2	24.8	78.1	.5	1.0	2.0	8.0	.3	10.0	10.0	7.0	8.0	7.0	.0	5.0	N
27	13.0	5.0	8.0	1.0	13.0	8.0	3.0	.1	4.0	2.0	2.0	.7	22.0	1.0	.0	20.0	.0	50.4	8.5	3644.6	.1	85.0	.1	202.5	428.8	.3	2.0	4.0	21.0	.1	34.0	51.0	22.0	11.0	48.0	41.2	21.0	Y
28	11.0	5.0	8.0	1.0	10.0	8.0	3.0	.2	4.0	2.0	3.0	1.0	18.0	1.0	.0	16.0	.0	54.4	6.1	2001.5	.1	66.0	.2	111.2	330.0	.3	2.0	4.0	17.0	.1	31.0	35.0	23.0	9.0	39.0	40.7	17.0	Y
29	2.0	25.0	.0	13.0	.0	48.0	13.0	.9	24.0	2.0	1.0	.1	49.0	1.0	.0	2.0	1.0	102.2	7.2	5262.9	.2	136.0	.1	292.4	733.4	.1	12.0	24.0	38.0	.7	61.0	75.0	34.0	8.0	18.0	86.7	15.0	Y
30	3.0	17.0	.0	17.0	.0	32.0	9.0	.5	16.0	2.0	1.0	.1	33.0	1.0	.0	2.0	1.0	76.9	15.9	19327.8	.4	222.0	.1	1073.8	1219.2	.1	8.0	16.0	26.0	.4	98.0	124.0	34.0	11.0	23.0	89.5	19.0	Y
31	2.0	5.0	.0	10.0	.0	8.0	3.0	.3	4.0	2.0	1.0	.3	9.0	1.0	.0	2.0	1.0	59.4	12.0	8577.8	.2	136.0	.1	476.6	713.7	.3	2.0	4.0	8.0	.2	59.0	77.0	27.0	11.0	15.0	83.3	12.0	N
32	2.0	59.0	.0	31.0	.0	116.0	30.0	.9	58.0	2.0	1.0	.0	117.0	1.0	.0	2.0	1.0	185.5	11.1	22714.2	.7	317.0	.1	1261.9	2052.8	.0	29.0	58.0	89.0	.8	142.0	175.0	77.0	12.0	36.0	93.9	33.0	Y
33	2.0	43.0	.0	25.0	.0	84.0	22.0	.8	42.0	2.0	1.0	.1	85.0	1.0	.0	2.0	1.0	112.1	18.1	36742.6	.7	345.0	.1	2041.3	2029.5	.1	21.0	42.0	65.0	.7	158.0	187.0	48.0	11.0	30.0	92.6	27.0	N
34	2.0	59.0	.0	31.0	.0	116.0	30.0	.9	58.0	2.0	1.0	.0	117.0	1.0	.0	2.0	1.0	130.8	15.9	33012.6	.7	345.0	.1	1834.0	2077.7	.0	29.0	58.0	89.0	.8	156.0	189.0	54.0	11.0	36.0	93.9	33.0	N
35	2.0	17.0	.0	14.0	.0	32.0	9.0	.6	16.0	2.0	1.0	.1	33.0	1.0	.0	2.0	1.0	85.3	11.0	10321.0	.3	168.0	.1	573.4	938.3	.1	8.0	16.0	26.0	.5	74.0	94.0	37.0	11.0	19.0	87.5	16.0	Y
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
1110	1.0	7.0	.0	.0	.0	12.0	4.0	.2	6.0	2.0	1.0	.3	14.0	1.0	.0	18.0	1.0	40.0	5.5	1210.4	.1	48.0	.2	67.3	220.1	.3	3.0	6.0	12.0	.2	22.0	26.0	16.0	8.0	20.0	.0	18.0	N
1111	13.0	9.0	3.0	1.0	.0	16.0	5.0	.2	8.0	2.0	2.0	.4	20.0	1.0	.0	24.0	2.0	26.4	18.4	8961.0	.2	92.0	.1	497.8	486.3	.2	4.0	8.0	17.0	.1	43.0	49.0	21.0	18.0	39.0	4.0	25.0	N
1112	6.0	3.0	3.0	.0	.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	9.0	1.0	.0	10.0	1.0	28.6	8.0	1827.2	.1	52.0	.1	101.5	228.4	.5	1.0	2.0	9.0	.1	26.0	26.0	13.0	8.0	17.0	.0	10.0	N
1113	.0	13.0	.0	1.0	2.0	24.0	7.0	.6	12.0	2.0	1.0	.1	31.0	1.0	.0	10.0	.0	13.1	19.8	5137.4	.1	70.0	.1	285.4	259.0	.1	6.0	12.0	26.0	.5	34.0	36.0	6.0	7.0	14.0	23.1	11.0	N
1114	13.0	19.0	2.0	3.0	13.0	34.0	10.0	.2	16.0	2.1	4.0	.4	38.0	1.0	.0	49.0	.0	61.4	21.3	27799.1	.4	217.0	.1	1544.4	1306.9	.1	9.0	17.0	30.0	.1	103.0	114.0	46.0	19.0	79.0	24.6	52.0	Y
1115	6.0	17.0	9.0	.0	8.0	30.0	9.0	.2	14.0	2.1	7.0	.8	35.0	5.0	.5	37.0	.0	58.7	10.1	5972.3	.2	106.0	.1	331.8	592.0	.6	8.0	15.0	28.0	.2	49.0	57.0	34.0	14.0	52.0	17.8	37.0	N
1116	5.0	5.0	1.0	3.0	1.0	8.0	3.0	.1	4.0	2.0	1.0	.3	12.0	1.0	.0	20.0	1.0	43.2	11.2	5397.8	.2	92.0	.1	299.9	482.8	.3	2.0	4.0	11.0	.1	43.0	49.0	25.0	13.0	30.0	16.7	23.0	N
1117	.0	3.0	.0	.0	.0	4.0	2.0	.7	2.0	2.0	1.0	.5	6.0	1.0	.0	3.0	2.0	5.0	6.0	180.0	.0	10.0	.2	10.0	30.0	.5	1.0	2.0	6.0	.5	4.0	6.0	2.0	6.0	4.0	.0	3.0	N
1118	1.0	3.0	.0	.0	.0	4.0	2.0	.2	2.0	2.0	1.0	.5	5.0	1.0	.0	9.0	1.0	35.8	7.0	1752.5	.1	57.0	.1	97.4	250.4	.5	1.0	2.0	5.0	.2	28.0	29.0	14.0	7.0	11.0	.0	9.0	N
1119	1.0	3.0	.0	.0	.0	4.0	2.0	.2	2.0	2.0	1.0	.5	5.0	1.0	.0	9.0	1.0	35.8	7.0	1752.5	.1	57.0	.1	97.4	250.4	.5	1.0	2.0	5.0	.2	28.0	29.0	14.0	7.0	11.0	.0	9.0	N
1120	4.0	7.0	5.0	2.0	7.0	12.0	4.0	.1	6.0	2.0	2.0	.5	22.0	1.0	.0	35.0	.0	58.2	22.0	28043.5	.4	221.0	.1	1558.0	1277.7	.3	3.0	6.0	20.0	.1	107.0	114.0	39.0	16.0	49.0	20.5	37.0	N
1121	1.0	3.0	.0	.0	.0	4.0	2.0	.1	2.0	2.0	1.0	.5	7.0	1.0	.0	20.0	3.0	54.2	22.0	26232.2	.4	243.0	.1	1457.4	1192.4	.5	1.0	2.0	7.0	.1	121.0	122.0	22.0	8.0	22.0	.0	20.0	N
1122	1.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	8.0	.0	16.4	11.5	2162.0	.1	47.0	.1	120.1	188.0	.5	1.0	2.0	7.0	.2	23.0	24.0	8.0	8.0	10.0	.0	8.0	N
1123	.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	8.0	3.0	16.4	11.5	2162.0	.1	47.0	.1	120.1	188.0	.5	1.0	2.0	7.0	.2	23.0	24.0	8.0	8.0	9.0	.0	8.0	N
1124	.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	7.0	3.0	11.8	11.3	1510.3	.0	35.0	.1	83.9	133.3	.5	1.0	2.0	7.0	.3	17.0	18.0	6.0	8.0	8.0	.0	7.0	N
1125	1.0	3.0	1.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	7.0	1.0	.0	6.0	3.0	11.2	9.2	936.7	.0	25.0	.1	52.0	102.2	.5	1.0	2.0	7.0	.3	10.0	15.0	6.0	11.0	8.0	.0	6.0	N

Lampiran 9 Dataset PC4

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DECISION_DENSITY	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	22.0	15.0	9.0	4.0	11.0	26.0	8.0	.2	12.0	2.2	5.0	.6	35.0	1.0	.0	32.0	.0	43.4	14.6	9265.5	.2	120.0	.1	514.8	634.3	.1	7.0	13.0	29.0	.1	42.0	78.0	23.0	16.0	71.0	31.9	36.0	Y
2	7.0	1.0	2.0	.0	2.0	.0	1.0	.1	.0	.0	1.0	1.0	3.0	1.0	.0	7.0	.0	14.5	10.0	1446.8	.1	38.0	.1	80.4	144.7	1.0	.0	.0	4.0	.1	8.0	30.0	4.0	10.0	18.0	22.2	7.0	N
3	10.0	3.0	2.0	.0	2.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	9.0	1.0	.0	10.0	.0	17.8	9.0	1439.2	.1	37.0	.1	80.0	159.9	.5	1.0	2.0	9.0	.1	12.0	25.0	8.0	12.0	24.0	16.7	10.0	N
4	9.0	3.0	2.0	.0	6.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	9.0	1.0	.0	10.0	.0	18.3	9.0	1478.1	.1	38.0	.1	82.1	164.2	.5	1.0	2.0	9.0	.1	12.0	26.0	8.0	12.0	27.0	37.5	10.0	N
5	3.0	1.0	.0	.0	.0	.0	1.0	.3	.0	.0	1.0	1.0	1.0	1.0	.0	4.0	.0	8.5	4.7	186.0	.0	12.0	.2	10.3	39.9	1.0	.0	.0	2.0	.1	4.0	8.0	3.0	7.0	10.0	.0	4.0	N
6	9.0	1.0	4.0	.0	3.0	.0	1.0	.1	.0	.0	1.0	1.0	5.0	1.0	.0	8.0	1.0	21.9	4.0	350.3	.0	23.0	.3	19.5	87.6	1.0	.0	.0	6.0	.1	8.0	15.0	7.0	7.0	22.0	27.3	8.0	N
7	8.0	3.0	4.0	9.0	5.0	4.0	2.0	.1	2.0	2.0	2.0	1.0	8.0	1.0	.0	10.0	1.0	33.3	10.0	3306.0	.1	69.0	.1	183.7	331.7	.5	1.0	2.0	8.0	.1	23.0	46.0	15.0	13.0	34.0	58.3	19.0	Y
8	5.0	1.0	1.0	.0	1.0	.0	1.0	.2	.0	.0	1.0	1.0	2.0	1.0	.0	6.0	1.0	16.3	4.1	272.0	.0	18.0	.2	15.1	66.6	1.0	.0	.0	3.0	.1	7.0	11.0	6.0	7.0	14.0	14.3	6.0	N
9	5.0	1.0	1.0	.0	1.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	7.0	1.0	18.5	6.8	843.9	.0	32.0	.2	46.9	125.0	1.0	.0	.0	3.0	.1	9.0	23.0	6.0	9.0	15.0	12.5	7.0	N
10	6.0	5.0	1.0	1.0	1.0	6.0	3.0	.2	2.0	3.0	1.0	.3	9.0	3.0	1.0	17.0	1.0	20.8	13.7	3916.4	.1	64.0	.1	217.6	285.4	1.0	2.0	3.0	8.0	.1	19.0	45.0	9.0	13.0	27.0	10.5	18.0	N
11	4.0	1.0	1.0	1.0	2.0	.0	1.0	.3	.0	.0	1.0	1.0	2.0	1.0	.0	2.0	1.0	11.6	3.0	104.6	.0	11.0	.3	5.8	34.9	1.0	.0	.0	3.0	.1	3.0	8.0	3.0	6.0	11.0	60.0	3.0	Y
12	6.0	3.0	1.0	1.0	1.0	4.0	2.0	.1	2.0	2.0	1.0	.5	7.0	1.0	.0	16.0	1.0	23.3	10.4	2514.4	.1	56.0	.1	139.7	242.0	.5	1.0	2.0	7.0	.1	17.0	39.0	9.0	11.0	26.0	11.1	17.0	Y
13	6.0	1.0	1.0	.0	1.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	10.0	1.0	21.1	8.4	1470.9	.1	44.0	.1	81.7	176.0	1.0	.0	.0	3.0	.1	13.0	31.0	7.0	9.0	19.0	9.1	10.0	N
14	5.0	1.0	2.0	.0	1.0	.0	1.0	.2	.0	.0	1.0	1.0	3.0	1.0	.0	6.0	1.0	19.2	6.2	736.0	.0	28.0	.2	40.9	118.9	1.0	.0	.0	4.0	.1	9.0	19.0	8.0	11.0	14.0	14.3	6.0	N
15	7.0	1.0	1.0	.0	1.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	10.0	1.0	21.1	8.4	1470.9	.1	44.0	.1	81.7	176.0	1.0	.0	.0	3.0	.1	13.0	31.0	7.0	9.0	20.0	9.1	10.0	N
16	5.0	1.0	1.0	.0	1.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	7.0	1.0	17.4	6.8	791.2	.0	30.0	.2	44.0	117.2	1.0	.0	.0	3.0	.1	9.0	21.0	6.0	9.0	15.0	12.5	7.0	N
17	6.0	1.0	2.0	.0	2.0	.0	1.0	.1	.0	.0	1.0	1.0	3.0	1.0	.0	8.0	1.0	25.3	7.2	1319.2	.1	43.0	.1	73.3	182.7	1.0	.0	.0	4.0	.1	13.0	30.0	9.0	10.0	18.0	20.0	8.0	N
18	6.0	3.0	1.0	1.0	1.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	9.0	1.0	22.7	10.3	2409.4	.1	55.0	.1	133.9	233.6	.5	1.0	2.0	6.0	.1	15.0	40.0	8.0	11.0	19.0	18.2	10.0	Y
19	6.0	3.0	1.0	.0	1.0	4.0	2.0	.2	2.0	2.0	1.0	.5	6.0	1.0	.0	10.0	1.0	19.5	11.8	2703.0	.1	55.0	.1	150.2	229.4	.5	1.0	2.0	6.0	.1	15.0	40.0	7.0	11.0	19.0	9.1	10.0	N
20	5.0	1.0	2.0	.0	1.0	.0	1.0	.1	.0	.0	1.0	1.0	3.0	1.0	.0	7.0	1.0	23.5	5.5	711.0	.0	31.0	.2	39.5	129.3	1.0	.0	.0	4.0	.1	11.0	20.0	9.0	9.0	15.0	12.5	7.0	N

21	4.0	5.0	2.0	.0	.0	8.0	3.0	.3	4.0	2.0	3.0	1.0	10.0	1.0	.0	12.0	.0	18.6	9.6	1712.4	.1	40.0	.1	95.1	178.4	.3	2.0	4.0	9.0	.2	16.0	24.0	10.0	12.0	18.0	.0	12.0	N
22	5.0	3.0	2.0	1.0	1.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	6.0	1.0	.0	7.0	4.0	20.2	4.4	399.7	.0	22.0	.2	22.2	89.9	.5	1.0	2.0	6.0	.1	10.0	12.0	9.0	8.0	19.0	22.2	8.0	N
23	4.0	3.0	2.0	.0	1.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	6.0	1.0	.0	8.0	4.0	20.2	4.4	399.7	.0	22.0	.2	22.2	89.9	.5	1.0	2.0	6.0	.1	10.0	12.0	9.0	8.0	18.0	11.1	8.0	N
24	8.0	5.0	3.0	1.0	1.0	6.0	3.0	.1	2.0	3.0	2.0	.7	12.0	3.0	1.0	24.0	3.0	18.2	30.6	17051.3	.2	116.0	.0	947.3	557.7	1.0	2.0	3.0	11.0	.1	53.0	63.0	13.0	15.0	38.0	7.7	25.0	Y
25	5.0	1.0	1.0	2.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	7.0	3.0	19.6	5.5	594.0	.0	27.0	.2	33.0	108.0	1.0	.0	.0	3.0	.1	11.0	16.0	8.0	8.0	19.0	22.2	9.0	N
26	5.0	3.0	2.0	.0	.0	4.0	2.0	.3	2.0	2.0	2.0	1.0	8.0	1.0	.0	8.0	3.0	16.2	9.4	1444.3	.1	36.0	.1	80.2	152.9	.5	1.0	2.0	8.0	.1	17.0	19.0	9.0	10.0	17.0	.0	8.0	N
27	8.0	3.0	4.0	.0	3.0	4.0	2.0	.2	2.0	2.0	2.0	1.0	8.0	1.0	.0	10.0	4.0	40.2	4.4	790.8	.1	40.0	.2	43.9	178.4	.5	1.0	2.0	8.0	.1	19.0	21.0	15.0	7.0	26.0	23.1	10.0	N
28	7.0	3.0	.0	1.0	.0	4.0	2.0	.1	2.0	2.0	1.0	.5	6.0	1.0	.0	14.0	6.0	20.5	14.8	4470.3	.1	66.0	.1	248.4	302.6	.5	1.0	2.0	6.0	.1	25.0	41.0	11.0	13.0	30.0	6.7	15.0	Y
29	3.0	7.0	2.0	20.0	5.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	15.0	1.0	.0	2.0	.0	21.6	13.6	3988.1	.1	64.0	.1	221.6	293.4	.3	3.0	6.0	13.0	.1	23.0	41.0	11.0	13.0	32.0	92.6	22.0	Y
30	3.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	6.0	1.0	.0	7.0	3.0	11.5	7.5	644.6	.0	22.0	.1	35.8	86.0	.5	1.0	2.0	6.0	.1	10.0	12.0	6.0	9.0	14.0	.0	7.0	N
31	9.0	7.0	3.0	1.0	5.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	16.0	1.0	.0	26.0	.0	23.4	21.9	11272.2	.2	101.0	.1	626.2	513.8	.3	3.0	6.0	14.0	.1	39.0	62.0	16.0	18.0	43.0	18.8	27.0	Y
32	9.0	7.0	3.0	1.0	5.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	16.0	1.0	.0	26.0	.0	23.4	21.9	11272.2	.2	101.0	.1	626.2	513.8	.3	3.0	6.0	14.0	.1	39.0	62.0	16.0	18.0	43.0	18.8	27.0	N
33	9.0	7.0	3.0	1.0	5.0	12.0	4.0	.2	6.0	2.0	4.0	1.0	16.0	1.0	.0	26.0	.0	23.4	21.9	11272.2	.2	101.0	.1	626.2	513.8	.3	3.0	6.0	14.0	.1	39.0	62.0	16.0	18.0	43.0	18.8	27.0	Y
34	1.0	1.0	.0	.0	.0	.0	1.0	.5	.0	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	5.8	2.0	23.2	.0	5.0	.5	1.3	11.6	1.0	.0	.0	2.0	.2	1.0	4.0	1.0	4.0	5.0	.0	2.0	N
35	3.0	1.0	.0	.0	.0	.0	1.0	.3	.0	.0	1.0	1.0	1.0	1.0	.0	3.0	4.0	12.9	3.5	157.4	.0	13.0	.3	8.7	45.0	1.0	.0	.0	2.0	.1	4.0	9.0	4.0	7.0	12.0	.0	3.0	N
36	4.0	1.0	.0	.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	1.0	1.0	.0	11.0	3.0	20.2	15.4	4795.8	.1	84.0	.1	266.4	310.8	1.0	.0	.0	2.0	.1	36.0	48.0	7.0	6.0	20.0	.0	11.0	N
37	11.0	9.0	3.0	6.0	11.0	16.0	5.0	.1	8.0	2.0	4.0	.8	23.0	1.0	.0	40.0	1.0	32.2	22.8	16658.8	.2	154.0	.0	925.5	732.3	.2	4.0	8.0	20.0	.1	49.0	105.0	14.0	13.0	70.0	29.8	46.0	Y
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
1385	1.0	9.0	2.0	.0	.0	.0	5.0	.5	.0	.0	4.0	.8	21.0	1.0	.0	10.0	.0	17.5	14.2	3513.3	.1	62.0	.1	195.2	248.0	.2	.0	.0	18.0	.3	17.0	45.0	6.0	10.0	17.0	.0	10.0	N
1386	1.0	9.0	1.0	.0	.0	.0	5.0	.5	.0	.0	4.0	.8	21.0	1.0	.0	10.0	.0	14.3	17.0	4117.9	.1	62.0	.1	228.8	242.2	.2	.0	.0	18.0	.3	17.0	45.0	5.0	10.0	17.0	.0	10.0	N
1387	2.0	5.0	2.0	.0	.0	.0	3.0	.4	.0	.0	2.0	.7	11.0	1.0	.0	7.0	.0	18.6	9.4	1651.3	.1	42.0	.1	91.7	175.1	.3	.0	.0	10.0	.2	12.0	30.0	7.0	11.0	15.0	.0	7.0	N
1388	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
1389	37.0	1.0	10.0	.0	7.0	.0	1.0	.0	.0	.0	1.0	1.0	107.0	1.0	.0	107.0	.0	1028.5	10.4	111346.3	3.6	1364.0	.1	6185.9	10701.3	1.0	.0	.0	108.0	.0	511.0	853.0	221.0	9.0	153.0	6.1	107.0	N
1390	.0	1.0	1.0	.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	7.0	2.0	9.4	17.3	2818.3	.1	47.0	.1	156.6	162.6	1.0	.0	.0	3.0	.1	13.0	34.0	3.0	8.0	9.0	.0	7.0	N
1391	1.0	3.0	.0	.0	.0	4.0	2.0	.1	2.0	2.0	1.0	.5	7.0	1.0	.0	19.0	3.0	14.7	45.5	30339.5	.2	186.0	.0	1685.5	666.8	.5	1.0	2.0	7.0	.1	65.0	121.0	5.0	7.0	22.0	.0	19.0	N
1392	.0	1.0	.0	.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	1.0	1.0	.0	12.0	2.0	7.9	22.0	3810.2	.1	67.0	.1	211.7	173.2	1.0	.0	.0	2.0	.1	22.0	45.0	2.0	4.0	14.0	.0	12.0	N
1393	.0	1.0	1.0	.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	14.0	2.0	9.5	36.0	12329.4	.1	99.0	.0	685.0	342.5	1.0	.0	.0	3.0	.1	27.0	72.0	3.0	8.0	16.0	.0	14.0	N
1394	1.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	7.0	.0	12.1	11.9	1706.4	.1	40.0	.1	94.8	143.4	.5	1.0	2.0	7.0	.2	17.0	23.0	5.0	7.0	10.0	.0	7.0	N
1395	.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	7.0	3.0	12.1	11.9	1706.4	.1	40.0	.1	94.8	143.4	.5	1.0	2.0	7.0	.2	17.0	23.0	5.0	7.0	9.0	.0	7.0	N
1396	.0	3.0	.0	.0	.0	4.0	2.0	.3	2.0	2.0	1.0	.5	7.0	1.0	.0	6.0	3.0	11.8	9.1	978.7	.0	30.0	.1	54.4	107.6	.5	1.0	2.0	7.0	.3	13.0	17.0	5.0	7.0	8.0	.0	6.0	N
1397	.0	1.0	1.0	.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	2.0	1.0	.0	15.0	2.0	9.4	38.7	14045.3	.1	105.0	.0	780.3	363.2	1.0	.0	.0	3.0	.1	29.0	76.0	3.0	8.0	17.0	.0	15.0	N
1398	1.0	3.0	1.0	.0	.0	4.0	2.0	.4	2.0	2.0	2.0	1.0	7.0	1.0	.0	5.0	3.0	11.5	8.3	800.0	.0	24.0	.1	44.4	96.0	.5	1.0	2.0	7.0	.3	10.0	14.0	6.0	10.0	8.0	.0	5.0	N
1399	1.0	1.0	.0	.0	.0	.0	1.0	.1	.0	.0	1.0	1.0	1.0	1.0	.0	19.0	2.0	10.4	45.0	21097.3	.2	167.0	.0	1172.1	468.8	1.0	.0	.0	2.0	.1	36.0	131.0	2.0	5.0	22.0	.0	19.0	N

Lampiran 10 Dataset PC5

NO	LOC_BLANK	BRANCH_COUNT	CALL_PAIRS	LOC_CODE_AND_COMMENT	LOC_COMMENTS	CONDITION_COUNT	CYCLOMATIC_COMPLEXITY	CYCLOMATIC_DENSITY	DECISION_COUNT	DESIGN_COMPLEXITY	DESIGN_DENSITY	EDGE_COUNT	ESSENTIAL_COMPLEXITY	ESSENTIAL_DENSITY	LOC_EXECUTABLE	PARAMETER_COUNT	GLOBAL_DATA_COMPLEXITY	GLOBAL_DATA_DENSITY	HALSTEAD_CONTENT	HALSTEAD_DIFFICULTY	HALSTEAD_EFFORT	HALSTEAD_ERROR_EST	HALSTEAD_LENGTH	HALSTEAD_LEVEL	HALSTEAD_PROG_TIME	HALSTEAD_VOLUME	MAINTENANCE_SEVERITY	MODIFIED_CONDITION_COUNT	MULTIPLE_CONDITION_COUNT	NODE_COUNT	NORMALIZED_CYLOMATIC_COMPLEXITY	NUM_OPERANDS	NUM_OPERATORS	NUM_UNIQUE_OPERANDS	NUM_UNIQUE_OPERATORS	NUMBER_OF_LINES	PERCENT_COMMENTS	LOC_TOTAL	Defective
1	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
2	8.0	5.0	9.0	.0	6.0	8.0	3.0	.2	4.0	1.0	.3	25.0	1.0	.0	16.0	.0	1.0	.3	26.5	16.5	7218.2	.2	91.0	.1	401.0	437.5	.3	2.0	4.0	24.0	.1	33.0	58.0	14.0	14.0	34.0	27.3	16.0	N
3	3.0	1.0	4.0	.0	3.0	.0	1.0	.2	.0	1.0	1.0	5.0	1.0	.0	6.0	.0	1.0	1.0	16.4	7.1	820.3	.0	29.0	.1	45.6	116.0	1.0	.0	.0	6.0	.1	11.0	18.0	7.0	9.0	15.0	33.3	6.0	N
4	.0	1.0	1.0	.0	.0	.0	1.0	.3	.0	1.0	1.0	2.0	1.0	.0	3.0	.0	1.0	1.0	11.7	4.0	186.4	.0	13.0	.3	10.4	46.6	1.0	.0	.0	3.0	.1	4.0	9.0	4.0	8.0	7.0	.0	3.0	N
5	3.0	25.0	.0	18.0	9.0	40.0	13.0	.2	16.0	1.0	.1	43.0	9.0	.7	50.0	.0	12.0	.9	22.1	57.0	71859.2	.4	242.0	.0	3992.2	1260.7	.7	12.0	20.0	32.0	.2	108.0	134.0	18.0	19.0	86.0	35.1	68.0	N
6	2.0	25.0	.0	16.0	8.0	40.0	13.0	.2	16.0	1.0	.1	43.0	9.0	.7	52.0	.0	12.0	.9	22.1	57.0	71859.2	.4	242.0	.0	3992.2	1260.7	.7	12.0	20.0	32.0	.2	108.0	134.0	18.0	19.0	84.0	31.6	68.0	N
7	2.0	21.0	.0	5.0	.0	32.0	11.0	.3	12.0	1.0	.1	33.0	7.0	.6	28.0	.0	11.0	1.0	9.1	37.4	12716.0	.1	85.0	.0	706.4	340.0	.6	10.0	16.0	24.0	.3	34.0	51.0	5.0	11.0	39.0	15.2	33.0	N
8	2.0	21.0	.0	5.0	.0	32.0	11.0	.3	12.0	1.0	.1	33.0	7.0	.6	28.0	.0	11.0	1.0	9.1	37.4	12716.0	.1	85.0	.0	706.4	340.0	.6	10.0	16.0	24.0	.3	34.0	51.0	5.0	11.0	39.0	15.2	33.0	N
9	2.0	21.0	.0	5.0	.0	32.0	11.0	.3	12.0	1.0	.1	33.0	7.0	.6	28.0	.0	11.0	1.0	9.1	37.4	12716.0	.1	85.0	.0	706.4	340.0	.6	10.0	16.0	24.0	.3	34.0	51.0	5.0	11.0	39.0	15.2	33.0	N
10	.0	9.0	1.0	3.0	.0	12.0	5.0	.7	2.0	5.0	1.0	15.0	4.0	.8	4.0	2.0	5.0	1.0	7.0	27.6	5307.1	.1	47.0	.0	294.8	192.1	.8	5.0	8.0	12.0	.5	17.0	30.0	4.0	13.0	10.0	42.9	7.0	N
11	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	1.0	1.0	13.2	5.3	363.2	.0	20.0	.2	20.2	69.2	1.0	.0	.0	2.0	.3	6.0	14.0	4.0	7.0	4.0	.0	2.0	N
12	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
13	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
14	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
15	.0	1.0	1.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	2.0	1.0	.0	2.0	.0	.0	.0	11.3	6.1	423.8	.0	20.0	.2	23.5	69.2	1.0	.0	.0	3.0	.3	7.0	13.0	4.0	7.0	4.0	.0	2.0	N
16	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	1.0	1.0	7.7	1.5	17.4	.0	5.0	.7	1.0	11.6	1.0	.0	.0	2.0	.3	2.0	3.0	2.0	3.0	4.0	.0	2.0	N
17	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	1.0	1.0	7.7	1.5	17.4	.0	5.0	.7	1.0	11.6	1.0	.0	.0	2.0	.3	2.0	3.0	2.0	3.0	4.0	.0	2.0	N
18	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	1.0	1.0	7.7	1.5	17.4	.0	5.0	.7	1.0	11.6	1.0	.0	.0	2.0	.3	2.0	3.0	2.0	3.0	4.0	.0	2.0	N
19	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N
20	.0	1.0	.0	.0	.0	.0	1.0	1.0	.0	1.0	1.0	1.0	1.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	1.0	.0	.0	.0	1.0	.0	.0	2.0	1.0	.0	1.0	.0	1.0	1.0	.0	.0	N

21	.0	9.0	1.0	3.0	.0	12.0	5.0	.7	2.0	5.0	1.0	15.0	4.0	.8	4.0	2.0	5.0	1.0	7.0	27.6	5307.1	.1	47.0	.0	294.8	192.1	.8	5.0	8.0	12.0	.5	17.0	30.0	4.0	13.0	10.0	42.9	7.0	N
22	.0	1.0	.0	.0	.0	.0	1.0	.2	.0	1.0	1.0	1.0	1.0	.0	6.0	.0	.0	.0	7.9	1.0	7.9	.0	5.0	1.0	.4	7.9	1.0	.0	.0	2.0	.1	4.0	1.0	2.0	1.0	7.0	.0	6.0	N
23	.0	1.0	.0	.0	.0	.0	1.0	.2	.0	1.0	1.0	1.0	1.0	.0	6.0	.0	1.0	1.0	3.6	4.0	57.1	.0	9.0	.3	3.2	14.3	1.0	.0	.0	2.0	.1	4.0	5.0	1.0	2.0	7.0	.0	6.0	N
24	.0	1.0	.0	.0	.0	.0	1.0	.2	.0	1.0	1.0	1.0	1.0	.0	6.0	.0	1.0	1.0	2.5	2.0	10.0	.0	5.0	.5	.6	5.0	1.0	.0	.0	2.0	.1	4.0	1.0	1.0	1.0	7.0	.0	6.0	N
25	.0	1.0	3.0	.0	.0	.0	1.0	.2	.0	1.0	1.0	4.0	1.0	.0	6.0	.0	1.0	1.0	8.5	3.0	76.6	.0	11.0	.3	4.3	25.5	1.0	.0	.0	5.0	.1	4.0	7.0	2.0	3.0	8.0	.0	6.0	N
26	.0	1.0	.0	.0	.0	.0	1.0	.2	.0	1.0	1.0	1.0	1.0	.0	6.0	.0	1.0	1.0	23.2	.5	5.8	.0	5.0	2.0	.3	11.6	1.0	.0	.0	2.0	.1	4.0	1.0	4.0	1.0	10.0	.0	6.0	N
27	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	1.0	1.0	13.2	5.3	363.2	.0	20.0	.2	20.2	69.2	1.0	.0	.0	2.0	.3	6.0	14.0	4.0	7.0	4.0	.0	2.0	N
28	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
29	.0	1.0	.0	1.0	.0	.0	1.0	.3	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	13.5	2.0	54.0	.0	9.0	.5	3.0	27.0	1.0	.0	.0	2.0	.2	4.0	5.0	4.0	4.0	5.0	33.3	3.0	N
30	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
31	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
32	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
33	.0	1.0	1.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	2.0	1.0	.0	2.0	.0	.0	.0	11.3	6.1	423.8	.0	20.0	.2	23.5	69.2	1.0	.0	.0	3.0	.3	7.0	13.0	4.0	7.0	4.0	.0	2.0	N
34	4.0	7.0	3.0	4.0	2.0	12.0	4.0	.2	6.0	3.0	.8	17.0	1.0	.0	18.0	.0	.0	.0	20.4	21.9	9749.9	.2	90.0	.1	541.7	445.9	.3	3.0	6.0	15.0	.1	41.0	49.0	15.0	16.0	30.0	25.0	22.0	N
35	.0	7.0	1.0	.0	.0	10.0	4.0	.8	2.0	4.0	1.0	12.0	1.0	.0	5.0	.0	4.0	1.0	16.9	9.0	1371.3	.1	39.0	.1	76.2	152.4	.3	4.0	5.0	10.0	.6	12.0	27.0	6.0	9.0	7.0	.0	5.0	N
36	2.0	1.0	1.0	1.0	.0	.0	1.0	.1	.0	1.0	1.0	2.0	1.0	.0	6.0	.0	1.0	1.0	19.9	6.5	838.9	.0	36.0	.2	46.6	129.1	1.0	.0	.0	3.0	.1	13.0	23.0	6.0	6.0	11.0	14.3	7.0	N
37	6.0	1.0	1.0	1.0	.0	.0	1.0	.1	.0	1.0	1.0	2.0	1.0	.0	18.0	.0	1.0	1.0	13.9	27.8	10733.2	.1	99.0	.0	596.3	386.8	1.0	.0	.0	3.0	.0	37.0	62.0	6.0	9.0	27.0	5.3	19.0	N
38	4.0	1.0	2.0	1.0	3.0	.0	1.0	.1	.0	1.0	1.0	3.0	1.0	.0	7.0	.0	1.0	1.0	5.0	27.0	3642.8	.0	39.0	.0	202.4	134.9	1.0	.0	.0	4.0	.1	12.0	27.0	2.0	9.0	17.0	36.4	8.0	N
39	2.0	1.0	1.0	1.0	.0	.0	1.0	.1	.0	1.0	1.0	2.0	1.0	.0	6.0	.0	1.0	1.0	19.9	6.5	838.9	.0	36.0	.2	46.6	129.1	1.0	.0	.0	3.0	.1	13.0	23.0	6.0	6.0	11.0	14.3	7.0	N
40	1.0	3.0	.0	.0	.0	4.0	2.0	.2	2.0	1.0	.5	5.0	1.0	.0	9.0	.0	2.0	1.0	12.9	9.9	1267.2	.0	32.0	.1	70.4	128.0	.5	1.0	2.0	5.0	.2	9.0	23.0	5.0	11.0	12.0	.0	9.0	N
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
16990	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
16991	.0	1.0	.0	.0	1.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.2	1.0	3.0	1.0	3.0	5.0	33.3	2.0	N
16992	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
16993	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
16994	.0	1.0	.0	.0	.0	.0	1.0	.5	.0	1.0	1.0	1.0	1.0	.0	2.0	.0	.0	.0	5.3	1.5	12.0	.0	4.0	.7	.7	8.0	1.0	.0	.0	2.0	.3	1.0	3.0	1.0	3.0	4.0	.0	2.0	N
16995	.0	8.0	1.0	.0	.0	8.0	5.0	.2	4.0	2.0	.4	16.0	1.0	.0	27.0	.0	4.0	.8	25.2	13.6	4646.9	.1	69.0	.1	258.2	341.8	.2	2.0	4.0	13.0	.2	29.0	40.0	16.0	15.0	29.0	.0	27.0	N
16996	7.0	13.0	5.0	1.0	3.0	12.0	8.0	.2	6.0	7.0	.9	33.0	1.0	.0	50.0	.0	4.0	.5	45.7	22.1	22252.6	.3	176.0	.1	1236.3	1008.1	.1	3.0	6.0	27.0	.1	79.0	97.0	34.0	19.0	63.0	7.4	51.0	N
16997	4.0	11.0	8.0	.0	3.0	16.0	1.0	.0	8.0	1.0	1.0	31.0	1.0	.0	39.0	.0	1.0	1.0	68.6	11.7	9324.1	.3	144.0	.1	518.0	799.9	1.0	4.0	8.0	27.0	.0	68.0	76.0	35.0	12.0	48.0	7.1	39.0	N
16998	.0	3.0	3.0	.0	.0	4.0	1.0	.1	2.0	1.0	1.0	9.0	1.0	.0	8.0	.0	1.0	1.0	33.7	5.4	977.4	.1	42.0	.2	54.3	181.5	1.0	1.0	2.0	9.0	.1	20.0	22.0	13.0	7.0	10.0	.0	8.0	N
16999	.0	1.0	.0	.0	.0	.0	1.0	.2	.0	1.0	1.0	1.0	1.0	.0	6.0	.0	1.0	1.0	7.6	1.3	11.9	.0	6.0	.8	.7	9.5	1.0	.0	.0	2.0	.1	5.0	1.0	2.0	1.0	8.0	.0	6.0	N
17000	1.0	1.0	4.0	3.0	.0	.0	1.0	.1	.0	1.0	1.0	5.0	1.0	.0	7.0	.0	1.0	1.0	43.9	3.6	582.5	.1	37.0	.3	32.4	159.9	1.0	.0	.0	6.0	.1	17.0	20.0	14.0	6.0	13.0	30.0	10.0	N
17001	10.0	17.0	8.0	3.0	9.0	32.0	9.0	.2	16.0	8.0	.9	44.0	1.0	.0	47.0	.0	9.0	1.0	113.1	10.8	13115.7	.4	207.0	.1	728.7	1217.7	.1	8.0	16.0	37.0	.1	94.0	113.0	48.0	11.0	71.0	20.3	50.0	N

Lampiran 11 Source Code Performance Controller

```
package controller;

import java.io.File;
import java.text.NumberFormat;
import javax.swing.JOptionPane;
import view.MainForm;
import model.DataSet;
import model.LogisticRegressionClassifier;

/**
 * @author harsih
 */
public class PerformanceController {
   private final DataSet dataSet = new DataSet();
   public void performance(String fileName, int dataFilter, int foldCrossValidation) throws Exception
{

      if (!fileName.equals("")) {
         File file = new File(fileName);
         if ((file.isFile())) {
            NumberFormat numberFormatter = NumberFormat.getNumberInstance();
            numberFormatter.setMinimumFractionDigits(3);
            numberFormatter.setMaximumFractionDigits(3);
            NumberFormat percentFormatter = NumberFormat.getPercentInstance();
            percentFormatter.setMinimumFractionDigits(3);
            percentFormatter.setMaximumFractionDigits(3);
            //Cekdata
              dataSet.setFileName(fileName);
            dataSet.readData();
            if ((dataSet.getAttributList() != null) && (dataSet.getAttributList().size() > 0)) {
               if (dataSet.getDataList().size() > 0) {
                  LogisticRegressionClassifier logreg = new LogisticRegressionClassifier();
                  logreg.ProsesClassifierLogReg(dataFilter, fileName, foldCrossValidation);
                    //  System.out.println(numberFormatter.format(logreg.getResult().getGMeanValue()));

MainForm.performanceForm.setConfusionMatrixValue(Double.toString(logreg.getResult().getTruePo
sitive()),Double.toString(logreg.getResult().getFalsePositive()),Double.toString(logreg.getResult().get
FalseNegative()),Double.toString(logreg.getResult().getTrueNegative()),percentFormatter.format(logre
g.getResult().getPositivePredictiveValue()),percentFormatter.format(logreg.getResult().getNegativePre
dictiveValue()),percentFormatter.format(logreg.getResult().getSensitivity()),percentFormatter.format(l
ogreg.getResult().getSpecificity()));
MainForm.performanceForm.setAccuracyTextField(percentFormatter.format(logreg.getResult().getAc
curacy()));
MainForm.performanceForm.setSensitivityTextField(percentFormatter.format(logreg.getResult().getS
ensitivity()));
MainForm.performanceForm.setSpecificityTextField(percentFormatter.format(logreg.getResult().getS
pecificity()));
MainForm.performanceForm.setAucTextField(numberFormatter.format(logreg.getResult().getAucVal
ue()));
MainForm.performanceForm.setPositivePredictiveValueTextField(percentFormatter.format(logreg.get
Result().getPositivePredictiveValue()));
MainForm.performanceForm.setNegativePredictiveValueTextField(percentFormatter.format(logreg.ge
tResult().getNegativePredictiveValue()));
MainForm.performanceForm.setFMeasureTextField(numberFormatter.format(logreg.getResult().getF
MeasureValue()));
MainForm.performanceForm.setGMeanTextField(numberFormatter.format(logreg.getResult().getGMe
anValue()));
               } else {
                  JOptionPane.showMessageDialog(null, "Training file does not contain the data",
"Error", JOptionPane.ERROR_MESSAGE);
```

```
                }
            } else {
                JOptionPane.showMessageDialog(null, "Training file does not contain the attributes", "Error", JOptionPane.ERROR_MESSAGE);
            }
        } else {
            JOptionPane.showMessageDialog(null, "Training file not found", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } else {
        JOptionPane.showMessageDialog(null, "No training file selected", "Error", JOptionPane.ERROR_MESSAGE);
    }
  }
}
```

Lampiran 12 Source Code Logistic Regression Classifier

```
package model;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.Random;
import weka.classifiers.Classifier;
import weka.classifiers.Evaluation;
import weka.classifiers.evaluation.NominalPrediction;
import weka.classifiers.functions.Logistic;
import weka.core.FastVector;
import weka.core.Instances;
import weka.filters.supervised.instance.Resample;
import weka.filters.supervised.instance.SMOTE;
import weka.filters.supervised.instance.SpreadSubsample;
import weka.filters.supervised.instance.StratifiedRemoveFolds;

/**
 * @author harsih
 */
public class LogisticRegressionClassifier {
    private HasilValidasi result;
    private String fileDataSource = "";
    public String getFileDataSource() {
        return fileDataSource;
    }
    public void setFileDataSource(String fileDataSource) {
        this.fileDataSource = fileDataSource;
    }
    public HasilValidasi getResult() {
        return result;
    }
    public static BufferedReader readDataFile(String filename) {
        BufferedReader inputReader = null;
        try {
            inputReader = new BufferedReader(new FileReader(filename));
        } catch (FileNotFoundException ex) {
            System.err.println("File not found: " + filename);
        }
        return inputReader;
    }
    public static Evaluation classify(Classifier model,
            Instances trainingSet, Instances testingSet) throws Exception {
        Evaluation evaluation = new Evaluation(trainingSet);
        model.buildClassifier(trainingSet);
        evaluation.evaluateModel(model, testingSet);
        return evaluation;
    }
    public static double calculateAccuracy(FastVector predictions) {
        double correct = 0;
        for (int i = 0; i < predictions.size(); i++) {
            NominalPrediction np = (NominalPrediction) predictions.elementAt(i);
            if (np.predicted() == np.actual()) {
                correct++;
            }
        }
        return 100 * correct / predictions.size();
    }
    public static Instances[][] crossValidationSplit(Instances data, int numberOfFolds) {
        Instances[][] split = new Instances[2][numberOfFolds];
        for (int i = 0; i < numberOfFolds; i++) {
```

```
            split[0][i] = data.trainCV(numberOfFolds, i);
            split[1][i] = data.testCV(numberOfFolds, i);
        }
        return split;
    }
    public void ProsesClassifierLogReg(int dataFilter, String fileName, int foldValidation) throws
Exception {
        double truePositive;
        double falseNegative;
        double falsePositive;
        double trueNegative;
        if (this.getFileDataSource().equals(fileName)) {
            this.setFileDataSource(fileName);
        }
        BufferedReader dataFile = readDataFile(fileName);
        Instances data = new Instances(dataFile);
        data.setClassIndex(data.numAttributes() - 1);
        Logistic logisticRegression = new Logistic();
        FastVector fastVector = new FastVector();
        Classifier classfierLogistic = new Logistic();
        Evaluation evaluation = new Evaluation(data);
        Random random = new Random(1);
        Instances sampleData2 = null;
        if (dataFilter == 1) {
            // pake resample
            Instances sampleData = null;

            if (this.getFileDataSource().equals(fileName)) {
                evaluation.crossValidateModel(classfierLogistic, sampleData2, foldValidation, random);
                System.out.println("ini samp2:" + this.getFileDataSource());
            } else {
                String filteroptions = "-S 1 -B 0.5 -Z 100.0";
                Resample sampler = new Resample();
                sampler.setInputFormat(data);
                sampler.setOptions(weka.core.Utils.splitOptions(filteroptions));
                sampler.setRandomSeed((int) System.currentTimeMillis());
                sampleData = data;
                sampleData = Resample.useFilter(sampleData, sampler);
                sampleData2 = sampleData;
                evaluation.crossValidateModel(classfierLogistic, sampleData, foldValidation, random);
                System.out.println("ini masih asli:" + this.getFileDataSource());
            }
        } else if (dataFilter == 2) {
            // pake resample
            Instances sampleData = null;
            String filteroptions = "-C 0 -K 5 -P 100.0 -S 1";
            SMOTE sampler = new SMOTE();
            sampler.setInputFormat(data);
            sampler.setOptions(weka.core.Utils.splitOptions(filteroptions));
            sampler.setRandomSeed((int) System.currentTimeMillis());
            sampleData = data;
            sampleData = Resample.useFilter(sampleData, sampler);
            evaluation.crossValidateModel(classfierLogistic, sampleData, foldValidation, random);
        } else if (dataFilter == 3) {
            // pake resample
            Instances sampleData = null;
            String filteroptions = "-M 0.0 -X 0.0 -S 1";
            SpreadSubsample sampler = new SpreadSubsample();
            sampler.setInputFormat(data);
            sampler.setOptions(weka.core.Utils.splitOptions(filteroptions));
            sampler.setRandomSeed((int) System.currentTimeMillis());
            sampleData = data;
            sampleData = Resample.useFilter(sampleData, sampler);
            evaluation.crossValidateModel(classfierLogistic, sampleData, foldValidation, random);
```

```
        } else if (dataFilter == 4) {
            // pake resample
            Instances sampleData = null;
            String filteroptions = "-S 0 -N 10 -F ";
            StratifiedRemoveFolds sampler = new StratifiedRemoveFolds();
            sampler.setInputFormat(data);
            sampler.setOptions(weka.core.Utils.splitOptions(filteroptions));
            // sampler.setRandomSeed((int)System.currentTimeMillis());
            sampleData = data;
            sampleData = Resample.useFilter(sampleData, sampler);
            evaluation.crossValidateModel(classfierLogistic, sampleData, foldValidation, random);
        } else {
            // None
            evaluation.crossValidateModel(classfierLogistic, data, foldValidation, random);
        }
        result = new HasilValidasi();
        truePositive = evaluation.confusionMatrix()[0][0];
        trueNegative = evaluation.confusionMatrix()[1][1];
        falsePositive = evaluation.confusionMatrix()[1][0];
        falseNegative = evaluation.confusionMatrix()[0][1];
        //System.out.println(truePositive);
        //System.out.println(falseNegative);
        double truePositiveRate = (double) truePositive / (truePositive + falseNegative);
        double falsePositiveRate = (double) falsePositive / (falsePositive + trueNegative);
        double recall = (double) truePositive / (truePositive + falseNegative);
        double precision = (double) truePositive / (truePositive + falsePositive);
        double accuracy = (double) (truePositive + trueNegative) / (truePositive + trueNegative +
falsePositive + falseNegative);
        double sensitivity = recall;
        double specificity = (double) trueNegative / (trueNegative + falsePositive);
        double positivePredictiveValue = precision;
        double negativePredictiveValue = (double) trueNegative / (trueNegative + falseNegative);
        double fMeasure = (2 * (recall * precision)) / (recall + precision);
        double gMean = Math.sqrt(sensitivity * specificity);
        double auc = evaluation.areaUnderROC(1); //(1 + truePositiveRate - falsePositiveRate)/2;
        // double auc = (1 + truePositiveRate - falsePositiveRate)/2;
        double rmse = evaluation.rootMeanSquaredError();
        result.setTruePositive(truePositive);
        result.setTrueNegative(trueNegative);
        result.setFalsePositive(falsePositive);
        result.setFalseNegative(falseNegative);
        result.setAccuracy(accuracy);
        result.setSensitivity(sensitivity);
        result.setSpecificity(specificity);
        result.setPositivePredictiveValue(positivePredictiveValue);
        result.setNegativePredictiveValue(negativePredictiveValue);
        result.setFMeasureValue(fMeasure);
        result.setGMeanValue(gMean);
        result.setAucValue(auc);
        System.out.println(evaluation.toSummaryString());
        System.out.println(evaluation.toClassDetailsString());
        System.out.println(evaluation.toMatrixString());

    }
}
```

Lampiran 13 Source Code Data Controller

```
package controller;
import java.io.File;
import java.text.NumberFormat;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import model.Attribute;
import model.RenderTable;
import view.DataViewForm;
import model.DataSet;
import model.Label;
import model.Statistic;
/**
* @author harsih
 */
public class DataController {
    private final DataSet dataSet = new DataSet();
    private DataViewForm dataViewForm;
    public void showDataView(String fileName) {
        if (!fileName.equals("")) {
            File file = new File(fileName);
            if ((file.isFile())) {
                dataViewForm = new DataViewForm(null, true);
                dataSet.setFileName(fileName);
                dataSet.readData();
                if ((dataSet.getAttributList() != null) && (dataSet.getAttributList().size() > 0)) {
                    int[] columnWidth = new int[dataSet.getAttributList().size() + 1];
                    columnWidth[0] = Integer.toString(dataSet.getDataList().size()).length();
                    if (columnWidth[0] < 3) {
                        columnWidth[0] = 3;
                    }
                    dataViewForm.addAttribut("<html><center>No.<br> </center>");
                    for (int i = 0; i < dataSet.getAttributList().size(); i++) {
                        Attribute dataAttribute = dataSet.getAttributList().get(i);
                        dataViewForm.addAttribut("<html><center>" + dataAttribute.name + "<br>(" +
dataAttribute.type + ")</center>");
                        /*if (dataAttribute.type.equalsIgnoreCase("NUMERIC")){
dataViewForm.addAttribut("<html><center>"+dataAttribute.name+"<br>("+dataAttribute.type+")</ce
nter>");
                        } else {
dataViewForm.addAttribut("<html><center>"+dataAttribute.name+"<br>("+dataAttribute.type+")</ce
nter>");
                        }*/
                        columnWidth[i + 1] = dataAttribute.name.length();
                        if (columnWidth[i + 1] < (dataAttribute.type.length() + 2)) {
                            columnWidth[i + 1] = (dataAttribute.type.length() + 2);
                        }
                    }
                    dataViewForm.setAlignment(0, RenderTable.RIGHT, columnWidth[0] * 9);
                    for (int i = 0; i < dataSet.getAttributList().size(); i++) {
                        Attribute dataAttribute = dataSet.getAttributList().get(i);
                        if (dataAttribute.type.equalsIgnoreCase("NUMERIC")) {
                            dataViewForm.setAlignment(i + 1, RenderTable.RIGHT, columnWidth[i + 1] * 9);
                        } else {
                            dataViewForm.setAlignment(i + 1, RenderTable.CENTER, columnWidth[i + 1] * 9);
                        }
                    }
                    Object[] data = new Object[dataSet.getAttributList().size() + 1];
                    ArrayList<ArrayList> dataRecord = dataSet.getDataList();
                    for (int j = 0; j < dataRecord.size(); j++) {
                        ArrayList dataArray = dataRecord.get(j);
                        data[0] = j + 1;
                        for (int i = 0; i < dataArray.size(); i++) {
```

```
                    data[i + 1] = dataArray.get(i);
                }
                dataViewForm.appendData(data);
            }
        }
        dataViewForm.setTitle("Data View");
        dataViewForm.setVisible(true);
    } else {
        JOptionPane.showMessageDialog(null, "Training file not found", "Error",
JOptionPane.ERROR_MESSAGE);
    }
  } else {
    JOptionPane.showMessageDialog(null, "No training file selected", "Error",
JOptionPane.ERROR_MESSAGE);
  }
 }
 public void showMetaDataView(String fileName) {
  if (!fileName.equals("")) {
    File file = new File(fileName);
    if ((file.isFile())) {
      NumberFormat numberFormatter = NumberFormat.getNumberInstance();
      numberFormatter.setMinimumFractionDigits(3);
      numberFormatter.setMaximumFractionDigits(3);
      dataViewForm = new DataViewForm(null, true);
      dataSet.setFileName(fileName);
      dataSet.readData();
      if ((dataSet.getAttributList() != null) && (dataSet.getAttributList().size() > 0)) {
        String[] header = {"No.", "Attribute", "Role", "Type", "Statistic", "Range"};
        int[] columnWidth = new int[header.length];
        String dataLabel = dataSet.getAttributList().get(dataSet.getAttributList().size() - 1).type;
        String[] labelName = dataLabel.substring(1, dataLabel.length() - 1).split(",");
        Label[] label = new Label[labelName.length];
        for (int i = 0; i < labelName.length; i++) {
          label[i] = new Label();
          label[i].name = labelName[i];
          label[i].count = 0;
        }
        for (int i = 0; i < header.length; i++) {
          columnWidth[i] = header[i].length();
        }
        for (String item : header) {
          dataViewForm.addAttribut("<html><center>" + item + "</center>");
        }
        Statistic[] dataStatistic = new Statistic[dataSet.getAttributList().size()];
        ArrayList<ArrayList> dataRecord = dataSet.getDataList();
        for (int j = 0; j < dataRecord.size(); j++) {
          ArrayList dataArray = dataRecord.get(j);
          for (int i = 0; i < dataArray.size(); i++) {
            if (j == 0) {
              dataStatistic[i] = new Statistic();
            }
            if (dataSet.getAttributList().get(i).type.equalsIgnoreCase("NUMERIC")) {
              double value = 0;
              try {
                value = Double.parseDouble(dataArray.get(i).toString());
              } catch (NumberFormatException ex) {
              }
              if (j == 0) {
                dataStatistic[i].sum = value;
                dataStatistic[i].min = value;
                dataStatistic[i].max = value;
                dataStatistic[i].count = 1;
                dataStatistic[i].variant = 0;
              } else {
```

```
                        dataStatistic[i].sum += value;
                        if (dataStatistic[i].min > value) {
                            dataStatistic[i].min = value;
                        }
                        if (dataStatistic[i].max < value) {
                            dataStatistic[i].max = value;
                        }
                        try {
                            dataStatistic[i].count++;
                        } catch (Exception ex) {
                        }
                    }
                } else {
                    boolean found = false;
                    int n = 0;
                    while (!found && (n < label.length)) {
                        if (dataArray.get(i).equals(label[n].name)) {
                            found = true;
                            label[n].count++;
                        }
                        n++;
                    }
                }
            }
        }
        for (ArrayList dataArray : dataRecord) {
            for (int i = 0; i < dataArray.size(); i++) {
                if (dataSet.getAttributList().get(i).type.equalsIgnoreCase("NUMERIC")) {
                    double value = 0;
                    try {
                        value = Double.parseDouble(dataArray.get(i).toString());
                    } catch (NumberFormatException ex) {
                    }
                    dataStatistic[i].variant += Math.pow((value - (dataStatistic[i].sum / dataStatistic[i].count)), 2);
                }
            }
        }
        String role, type, statistic, range, avg, sd;
        for (int i = 0; i < dataSet.getAttributList().size(); i++) {
            Attribute dataAttribute = dataSet.getAttributList().get(i);
            if (dataAttribute.type.equalsIgnoreCase("NUMERIC")) {
                role = "Regular";
                type = "Numeric";
                avg = (dataStatistic[i].count == 0) ? "~" : numberFormatter.format(dataStatistic[i].sum / dataStatistic[i].count);
                sd = (dataStatistic[i].count == 1) ? "~" : numberFormatter.format(Math.sqrt(dataStatistic[i].variant / (dataStatistic[i].count - 1)));
                statistic = "avg = " + avg + " +/- " + sd;
                range = "[" + (dataStatistic[i].min) + ";" + dataStatistic[i].max + "]";
            } else {
                role = "Label";
                type = "Nominal";
                Label tempLabel = new Label();
                tempLabel.name = label[0].name;
                tempLabel.count = label[0].count;
                for (int n = 0; n < label.length - 1; n++) {
                    for (int m = n + 1; m < label.length; m++) {
                        if (label[n].count < label[m].count) {
                            tempLabel.name = label[n].name;
                            tempLabel.count = label[n].count;
                            label[n].name = label[m].name;
                            label[n].count = label[m].count;
                            label[m].name = tempLabel.name;
```

```
                            label[m].count = tempLabel.count;
                        }
                    }
                }
                range = "";
                for (Label item : label) {
                    range += range.equals("") ? item.name + " (" + item.count + ")" : ", " + item.name
+ " (" + item.count + ")";
                }
                statistic = "mode = " + label[0].name + " (" + label[0].count + "), least = " +
label[label.length - 1].name + " (" + label[label.length - 1].count + ")";
            }
            Object[] metaDataRecord = new Object[]{i + 1, dataAttribute.name, role, type, statistic,
range};
            dataViewForm.appendData(metaDataRecord);

            for (int j = 0; j < metaDataRecord.length; j++) {
                if (columnWidth[j] < metaDataRecord[j].toString().length()) {
                    columnWidth[j] = metaDataRecord[j].toString().length();
                }
            }
        }
        for (int i = 0; i < header.length; i++) {
            dataViewForm.setAlignment(i, RenderTable.LEFT, columnWidth[i] * 9);
        }
    }
    dataViewForm.setTitle("Meta Data View");
    dataViewForm.setVisible(true);
  } else {
    JOptionPane.showMessageDialog(null, "Training file not found", "Error",
JOptionPane.ERROR_MESSAGE);
  }
 } else {
  JOptionPane.showMessageDialog(null, "No training file selected", "Error",
JOptionPane.ERROR_MESSAGE);
 }
 }
}
```