SELEKSI FITUR MENGGUNAKAN SYMMETRICAL UNCERTAINTY PADA PREDIKSI CACAT PERANGKAT LUNAK

SKRIPSI

Oleh :
AHMAD ROIHAN
NIM. 14650082

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2018

SELEKSI FITUR MENGGUNAKAN SYMMETRICAL UNCERTAINTY PADA PREDIKSI CACAT PERANGKAT LUNAK

SKRIPSI

Diajukan Kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
AHMAD ROIHAN
NIM. 14650082

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK BRAHIM MALANG
2018

LEMBAR PERSETUJUAN

SELEKSI FITUR MENGGUNAKAN SYMMETRICAL UNCERTAINTY PADA PREDIKSI CACAT PERANGKAT LUNAK

SKRIPSI

Oleh :
AHMAD ROIHAN
NIM. 14650082

Telah Diperiksa dan Disetujui untuk diuji:
Tanggal, 13 September 2018

Dosen Pembimbing I	Dosen Pembimbing II
Fatchurrochman, M.Kom	Irwan Budi Santoso, M.Kom
NIP. 19700731 200501 1 002	NIP. 19770103 201101 1 004

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PENGESAHAN

SELEKSI FITUR MENGGUNAKAN SYMMETRICAL UNCERTAINTY PADA PREDIKSI CACAT PERANGKAT LUNAK

SKRIPSI

Oleh:
AHMAD ROIHAN
NIM. 14650082

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 13 September 2018

Susunan Dewan Penguji			Tanda Tangan
Penguji Utama	:	Supriyono, M.Kom NIDT. 19841010 20160801 1 078	( )
Ketua Penguji	:	A'la Syauqi, M.Kom NIP. 19771201 200801 1 007	( )
Sekretaris Penguji	:	Fatchurrochman, M.Kom NIP. 19700731 200501 1 002	( )
Anggota Penguji	:	Irwan Budi Santoso, M.Kom NIP. 19770103 201101 1 004	( )

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini,

Nama : Ahmad Roihan

NIM : 14650082

Jurusan : Teknik Informatika

Fakultas : Sains dan Teknologi

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambilan tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 29 Agustus 2018
Yang membuat pernyataan,

Ahmad Roihan
NIM. 14650082

MOTTO

"Profesional itu Berat"

HALAMAN PERSEMBAHAN

Bismillahirrahmanirrahim

Segala puji bagi Allah atas ridho-Nya dan juga dukungan serta doa dari orang-orang tercinta akhirnya skripsi ini dapat terselesaikan dengan baik. Oleh karena itu kupersembahkan karya sederhana ini kepada:

Ayah dan ibu tersayang yang selalu memberikan motivasi, pengorbanan dan doa yang tiada henti untuk kesuksesan saya.

Bapak Fatchurrochman, M.kom dan Bapak Irwan Budi Santoso, M.Kom selaku dosen pembimbing yang selama ini tulus, sabar serta ikhlas meluangkan waktunya untuk memberikan bimbingan, selain itu juga untuk seluruh dosen Teknik Informatika yang selama ini telah mendidik dan menyalurkan segala ilmu informatika dan juga pengalaman yang sangat berarti untuk saya. Bapak dan Ibu dosenku jasamu akan selalu terpatri dihati.

Teman-teman lencu geng, Fajrul Iqbal Mubarok (Kisun) yang telah memberikan banyak bantuan dari semester 1 kuliah sampai semester akhir di bidang teknis, Iqbal Chalif Maulana (Calip) yang telah memberikan bantuan di bidang multimedia selama kuliah dan Diko Andri Vidian (Dick) yang telah memberikan waktu luangnya membantu saya di bidang teknis, kosnya untuk tempat inkubasi skripsi saya dan selalu memberikan tekanan saat mengerjakan skripsi.

Teman-teman Biner (TI 2014) sifa, afrizal, habibil, arif, nindi , ami, ima dst, terima kasih atas semangat, dukungan dan bantuan serta kekompakan selama ini, terima kasih atas canda tawa dan kenangan manis yang telah terukir selama ini. Sekaligus teman di saat jurusan telah sepi dari angkatan 2014. Semoga suatu hari ada moment bertemu berkumpul bersama lagi.

Seluruh teman-temanku yang senantiasa selalu memotivasi dan menjadi tempat untuk saling belajar.

Serta seluruh pihak yang tidak dapat disebutkan satu persatu, terima kasih.

KATA PENGANTAR

Assalamualaikum Wr. Wb.

Segala puji bagi Allah SWT tuhan semesta alam, karena atas segala rahmat dan karunia-Nya sehingga penulis mampu menyelesaikan skripsi ini dengan baik dan lancar. Shalawat serta salam selalu tercurahkan kepada tauladan terbaik Nabi Muhammad SAW yang telah membimbing umatnya dari zaman kebodohan menuju Islam yang rahmatan lil alamin.

Selanjutnya penulis haturkan ucapan terima kasih karena dalam penyelesaian skripsi ini tidak lepas dari bantuan, bimbingan serta dukungan dari beberapa pihak. Ucapan terima kasih ini penulis sampaikan kepada:

1. Prof. DR. H. Abd. Haris, M.Ag, selaku rektor UIN Maulana Malik Ibrahim Malang beserta seluruh staf. Bakti Bapak dan Ibu sekalian terhadap UIN Maliki Malang yang menaungi segala kegiatan di kampus UIN Maliki Malang.
2. Dr. Sri Harini, M.Si selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang beserta seluruh staf. Bapak dan ibu sekalian sangat berjasa memupuk dan menumbuhkan semangat untuk maju kepada penulis.
3. Bapak Dr. Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang, yang sudah memberi banyak menginspirasi dan memotivasi untuk terus berkembang.

4. Bapak Fatchurrochman, M.Kom selaku dosen pembimbing I yang telah bersedia meluangkan waktu untuk membimbing, memotivasi dan memberi arahan kepada penulis dalam pengerjaan skripsi ini hingga akhir.
5. Bapak Irwan Budi Santoso, M.T selaku dosen pembimbing II yang juga senantiasa memberi masukan dan nasihat serta petunjuk dalam penyusunan skripsi ini.
6. Ibu Linda Salma Angreani, M.T selaku dosen wali yang telah dan selalu memberi arahan dan bimbingan hingga saat ini.
7. Ayah, Ibu dan Kakak serta keluarga besar tercinta yang selalu memberi dorongan dan doa yang senantiasa mengiringi setiap langkah penulis.
8. Seluruh Dosen Teknik Informatika yang telah memberikan keilmuan serta pengalaman yang berarti kepada penulis selama ini.
9. Teman-teman yang telah memotivasi dan membantu banyak hal selama ini.
10. Seluruh teman-teman Teknik Informatika UIN Maulana Malik Ibrahim Malang yang telah banyak berbagi ilmu, pengalaman dan menjadi inspirasi untuk terus semangat belajar.
11. Teman-teman seperjuangan Teknik Informatika 2014 yang telah berjuang bersama dan saling mendukung selama ini.
12. Para peneliti yang telah melakukan penelitian tentang prediksi cacat perangkat lunak dan seleksi fitur yang menjadi acuan penulis dalam pembuatan skripsi ini. Serta semua pihak yang telah membantu yang tidak bisa disebutkan satu persatu. Terimakasih banyak.

Penulis menyadari bahwa skripsi ini masih jauh dari sempurna dikarenakan terbatasnya pengalaman dan pengetahuan yang dimiliki penulis. Oleh karena itu, penulis mengharapkan segala bentuk saran dan masukan serta kritik yang membangun dari berbagai pihak. Terlepas dari berbagai kekurangan tersebut, semoga skripsi ini dapat memberikan manfaat bagi pembaca dan mendorong peneliti selanjutnya dalam menyempurnakannya. Amin.

Wassalamualaikum Wr. Wb.

Malang, 29 Agustus 2018

Penulis

DAFTAR ISI

HALAMAN JUDUL ..... **i**
LEMBAR PERSETUJUAN ..... **ii**
HALAMAN PENGESAHAN ..... **iii**
HALAMAN PERNYATAAN KEASLIAN TULISAN ..... **iv**
MOTTO ..... **v**
HALAMAN PERSEMBAHAN ..... **vi**
KATA PENGANTAR ..... **vii**
DAFTAR ISI ..... **x**
DAFTAR GAMBAR ..... **xii**
DAFTAR TABEL ..... **xiii**
ABSTRAK ..... **xiv**
ABSTRACT ..... **xv**
ملخص ..... **xvi**
BAB I PENDAHULUAN **1**
1.1. Latar Belakang Masalah 1
1.2. Pernyataan Masalah 5
1.3. Tujuan Penelitian 5
1.4. Manfaat Penelitian 5
1.5. Batasan Masalah 5
1.6. Sistematika Penulisan 6
BAB II STUDI PUSTAKA **7**
2.1 Penelitian Terkait 7
2.2 Cacat Perangkat Lunak 8
2.3 Prediksi Cacat Perangkat Lunak 9
2.4 Diskritisasi Data 16
2.5 Diskritisasi Data dengan *Equal Width Binning* 17
2.6 Seleksi Fitur 18
2.7 Seleksi Fitur dengan *Symmetrical Uncertainty* 18
2.8 *Split Percentage* 21
2.9 *Naïve Bayes* 21
2.10 Pengujian Hasil Seleksi Fitur dengan *Naïve Bayes* 24
2.11 Evaluasi Klasifikasi 25
BAB III METODOLOGI PENELITIAN **28**
3.1 Analisis Sistem 28

3.1.1 Studi Literatur 29
3.1.2 Koleksi Data 30
3.2 Perancangan Sistem 30
3.2.1. Proses Diskritisasi dengan *Equal Width Binning* 32
3.2.2. Proses Seleksi Fitur dengan *Symmetrical Uncertainty* 35
3.2.3. Proses Pengujian 40
3.2.3.1. Split Percentage 40
3.2.3.2. Naïve Bayes Classifier 41
3.2.3.3. Confusion Matrix 45
BAB IV UJI COBA DAN PEMBAHASAN 47
4.1 Uji Coba 47
4.1.1 Lingkungan Uji Coba 47
4.1.2 Data Uji coba 48
4.1.3. Tampilan Sistem 51
4.1.4 Hasil Pengujian 52
4.1.4.1. Seleksi Fitur denganSymmetrical Uncertainty 52
4.1.4.2. Uji coba Klasifikasi 58
4.2 Pembahasan 64
4.2.1 Pembahasan hasil pengujian *dataset* CM1 64
4.2.2 Pembahasan hasil pengujian *dataset* JM1 65
4.2.3 Pembahasan hasil pengujian *dataset* KC1 66
4.2.4 Pembahasan hasil pengujian *dataset* KC2 67
4.2.5 Pembahasan hasil pengujian *dataset* PC1 68
4.3 Integrasi dengan Islam 70
BAB V KESIMPULAN DAN SARAN 75
5.1 Kesimpulan 75
5.2 Saran 76
DAFTAR PUSTAKA 77

DAFTAR GAMBAR

Gambar 3.1 Prosedur Penelitian 28
Gambar 3.2 Desain Sistem Yang Diusulkan 31
Gambar 3.3 Flowchart Equal Width Binning 33
Gambar 3.4 Flowchart Algoritma Symmetrical Uncertainty 37
Gambar 4.1 Tampilan Sistem 51
Gambar 4.2 Grafik akurasi klasifikasi NB *dataset* CM1 64
Gambar 4.3 Grafik akurasi klasifikasi NB *dataset* JM1 65
Gambar 4.4 Grafik akurasi klasifikasi NB *dataset* KC1 66
Gambar 4.5 Grafik akurasi klasifikasi NB *dataset* KC2 67
Gambar 4.6 Grafik akurasi klasifikasi NB *dataset* PC1 68

DAFTAR TABEL

Tabel 2.1 Dataset CM1 11
Tabel 2.2 Dataset CM1 (Lanjutan)..... 12
Tabel 2.3 Ukuran Evaluasi Model Klasifikasi 26
Tabel 2.4 Tabel Confusion Matrix 27
Tabel 3.1 Potongan Dataset CMI 33
Tabel 3.2 Hasil Diskritisasi Dataset CM1 Fitur Ke-1 (Loc) 35
Tabel 3.3 Hasil Diskritisasi Semua Fitur Dataset CM1 35
Tabel 3.4 Hasil Perangkingan Sesuai Nilai Symmetrical Uncertainty (SU)..... 39
Tabel 3.5 Hasil Seleksi 5 Fitur Pada Dataset CM1 40
Tabel 3.6 Baris Ke-48 Set Pertama Dataset CM1..... 43
Tabel 3.7 Confusion Matrix Klasifikasi Dataset CM1 Dengan Seleksi Fitur..... 45
Tabel 4.1 *Dataset* CM1 48
Tabel 4.2 *Dataset* JM1 48
Tabel 4.3 *Dataset* KC1..... 48
Tabel 4.4 *Dataset* KC2..... 49
Tabel 4.5 *Dataset* PC1 49
Tabel 4.6 *Dataset* Penelitian 49
Tabel 4.7 Fitur setiap *dataset* 50
Tabel 4.8 Perankingan Fitur *dataset* CM1 Berdasarkan nilai *Symmetrical Uncertainty*..... 52
Tabel 4.9 Perankingan Fitur *dataset* CM1 Berdasarkan nilai *Symmetrical Uncertainty* (Lanjutan)..... 53
Tabel 4.10 Perankingan Fitur *dataset* JM1 Berdasarkan nilai *Symmetrical Uncertainty*..... 54
Tabel 4.11 Perankingan Fitur *dataset* KC1 Berdasarkan nilai *Symmetrical Uncertainty*..... 55
Tabel 4.12 Perankingan Fitur *dataset* KC2 Berdasarkan nilai *Symmetrical Uncertainty*..... 56
Tabel 4.13 Perankingan Fitur *dataset* PC1 Berdasarkan nilai *Symmetrical Uncertainty*..... 57
Tabel 4.14 Hasil Uji Coba Klasifikasi *Dataset* CM1..... 59
Tabel 4.15 Hasil Uji Coba Klasifikasi *Dataset* JM1..... 60
Tabel 4.16 Hasil Uji Coba Klasifikasi *Dataset* KC1 61
Tabel 4.17 Hasil Uji Coba Klasifikasi *Dataset* KC2 62
Tabel 4.18 Hasil Uji Coba Klasifikasi *Dataset* PC1..... 63
Tabel 4.19 Fitur yang relevan berdasarkan seleksi fitur *Symmetrical Uncertainty* 69

ABSTRAK

Roihan, Ahmad. 2018. **Seleksi Fitur Menggunakan Symmetrical Uncertainty Pada Prediksi Cacat Perangkat Lunak.** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Fatchurrochman, M.Kom. (II) Irwan Budi Santoso, M.Kom.

Kata kunci: prediksi cacat perangkat lunak, seleksi fitur *Symmetrical Uncertainty.*

Cacat perangkat lunak didefinisikan sebagai cacat pada perangkat lunak seperti cacat pada dokumentasi, pada kode program, pada desain dan hal – hal lain yang menyebabkan kegagalan perangkat lunak. Salah satu cara yang bisa dilakukan untuk meminimalisir terjadinya cacat perangkat lunak adalah teknik prediksi cacat perangkat lunak menggunakan *dataset Metrics Data Program* (MDP). Namun tidak semua fitur yang ada pada *dataset* memiliki pengaruh besar terhadap prediksi cacat perangkat lunak karena *dataset* tersebut dibuat tidak khusus untuk pediksi cacat perangkat lunak. Oleh karena itu, seleksi fitur diperlukan untuk mendapatkan fitur yang berpengaruh terhadap prediksi cacat perangkat lunak.

Penelitian ini melakukan pemilihan fitur menggunakan seleksi fitur *Symmetrical Uncertainty* dengan jumlah pengambilan fitur yang berbeda-beda dengan tujuan mendapatkan fitur yang paling berpengaruh atau relevan. Hasil dari penelitian ini menyimpulkan bahwa fitur yang relevan terhadap prediksi cacat perangkat lunak berdasarkan seleksi fitur *Symmetrical Uncertainty* pada *dataset CM1 adalah fitur Time to write program (t). Dataset JM1 adalah fitur total operand (total_Opnd), unique operands (uniq_Opnd), total operator (total_Op), unique operators (uniq_Op), Intelligence (i), Programlength (l), Difficulty (d), Volume (v), Error estimate (b), line count of code (loc), Cyclomatic complexity (v(g)), Essential complexit (ev(g)), Totaloperator+operand (n), Count of Statement Lines (IOCode), Time to write program (t), Effort to write program (e), Design complexity (iv(g)), Branch count (branchCount) dan lOComment. Dataset KC1 adalah fitur Difficulty (*d). Dataset PC1 adalah fitur *Count of lines of comments* (lOComment). erdasarkan uji coba klasifikasi fitur-fitur tersebut menghasilkan akurasi terbaik.

ABSTRACT

Roihan, Ahmad. 2018. **Feature Selection Uses Symmetrical Uncertainty in Software Defect Prediction.** Undergraduate Thesis. Department of Informatic Engineering Faculty of Science and Technology Maulana Malik Ibrahim State Islamic University, Malang. Supervisor: (I) Fatchurrochman, M.Kom. . (II) Irwan Budi Santoso, M.Kom.

Keywords: software defect prediction, feature selection Symmetrical Uncertainty.

Software defects are defined as defects in software such as defects in documentation, program code, design and other things that cause software failure. One way that can be done to minimize the occurrence of software defects is the software defect prediction technique using the dataset Metrics Data Program (MDP). However, not all features in the dataset have a major influence on the prediction of software defects because the dataset is not specifically for software defects. Therefore, feature selection is needed to get features that affect the prediction of software defects.

This study selected features using the Symmetrical Uncertainty feature selection with the number of different features taken with the aim of getting the most influential or relevant features. The results of this study concluded that the relevant feature of the prediction of software defects based on the Symmetrical Uncertainty feature selection on the CM1 dataset is the Time to write program (t) feature. The JM1 dataset is a feature of total operand (total_Opnd), unique operands (uniq_Opnd), total operator (total_Op), unique operators (uniq_Op), Intelligence (i), Programlength (l), Difficulty (d), Volume (v), Error estimate (b), line count of code (loc), Cyclomatic complexity (v (g)), Essential complexit (ev (g)), Total operator + operand (n), Count of Statement Lines (IOCode), Time write program (t), Effort to write program (e), complexity design (iv (g)), Branch count (branchCount) and lOComment. The KC1 dataset is a Difficulty feature (the KC2 dataset is Difficulty (d). The PC1 dataset is a feature of Count of lines of comments (lOComment). Based on the classification trials these features produce the best accuracy.

ملخص

ريحان، أحمد. 2018. **اختيار الميزة باستخدام أداة إبهام المتماثل في التنبؤ من عيوب البرمجيات**، البحث العلمي. قسم المعلوماتية كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية **مالانج. المشرف** : (1)**فتح الرحمن** الماجستير، (2) إروان بودي سانتوسا الماجستير.

الكلمات الرئيسية : تنبؤ عيوب البرمجيات، اختيار الميزة إبهام المتماثل.

توصف عيوب البرمجيات أنها عيوب في البرمجيات كالعيوب الموجودة في الوثيقة ورمز برنامج والتخطيط وغيرها من الأشياء التي تؤدي إلى فشل البرمجيات. ومن الطرق التي يمكن القيام بها للتقصير عن حدوث عيوب البرمجيات هي تقنية تنبؤ عيوب البرمجيات باستخدام برنامج بيانات المقاييس (*Metrics Data Program* (MDP)). ولكن ليس لكل الميزات في مجموعة البيانات تأثير كبير على تنبؤ عيوب البرمجيات إذ لا تخصص تلك مجموعة البيانات لتنبؤ عيوب البرمجيات. لذلك، تعد عملية اختيار الميزة ضروريا للحصول على الميزات المؤثرة على التنبؤ من عيوب البرمجيات.

يقوم هذا البحث باختيار الميزات باستخدام اختيار ميزة إبهام المتماثل مع عدد أخذ الميزات المختلفة التي تهدف إلى الحصول على الميزات الأكثر تأثيراً أو الأصوب. واستنتجت نتيجة هذا البحث إلى أن الميزة الصائبة لتنبؤ عيوب البرمجيات اعتمادا على اختيار ميزة إبهام المتماثل لمجموعة البيانات *CM1* هي ميزة *Time to write program (t)*. ولمجموعة البيانات *JM1* هي ميزات *total operand* (*total_Opnd*)، و*unique operands* (*uniq_Opnd*)، و*total operator* (*total_Op*)، و *unique operators* (*uniq_Op*)، و*Intelligence* (*i*)، و*Programlength* (*l*)، و*Difficulty* (*d*)، و *Volume* (*v*)، و *Error estimate* (*b*)، و*line count of code* (*loc*)، و *Cyclomatic complexity* (*v(g)*)، و*Essential complexity* (*ev(g)*)، و*Totaloperator+operand* (*n*)، و *Lines Count of Statement* (*IOCode*)، و*Time to write program* (*t*)، و *Effort to write program* (*e*)، و *Design complexity* (*iv(g)*)، و*Branch count* (*branchCount*)، و *lOComment*. ولمجموعة البيانات *KC1* هي ميزة *Difficulty* (d). ولمجموعة البيانات PC1 هي ميزة *Count of lines of comments* (lOComment). واعتمادا على تجربة التصنيف فتكون هذه الميزات المذكورة تحصل على أحسن الدقة.

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Ayat-ayat Al-Qur'an menyimpan berbagai kunci untuk pegangan hidup kita. Dengan memegang kunci-kunci tersebut, maka semua masalah kita dapat teratasi. Berawal dari kajian yang dilakukan penulis pada potongan Ayat Al-Qur'an surat al-ma'idah ayat 2 berikut;

يَا أَيُّهَا الَّذِينَ آمَنُوا لَا تُحِلُّوا شَعَائِرَ اللَّهِ وَلَا الشَّهْرَ الْحَرَامَ وَلَا الْهَدْيَ وَلَا الْقَلَائِدَ وَلَا آمِّينَ الْبَيْتَ الْحَرَامَ يَبْتَغُونَ فَضْلًا مِنْ رَبِّهِمْ وَرِضْوَانًا ۚ وَإِذَا حَلَلْتُمْ فَاصْطَادُوا ۚ وَلَا يَجْرِمَنَّكُمْ شَنَآنُ قَوْمٍ أَنْ صَدُّوكُمْ عَنِ الْمَسْجِدِ الْحَرَامِ أَنْ تَعْتَدُوا ۘ وَتَعَاوَنُوا عَلَى الْبِرِّ وَالتَّقْوَىٰ ۖ وَلَا تَعَاوَنُوا عَلَى الْإِثْمِ وَالْعُدْوَانِ ۚ وَاتَّقُوا اللَّهَ ۖ إِنَّ اللَّهَ شَدِيدُ الْعِقَابِ

Artinya : "*Hai orang-orang yang beriman, janganlah kamu melanggar syi'ar-syi'ar Allah, dan jangan melanggar kehormatan bulan-bulan haram, jangan (mengganggu) binatang-binatang had-ya, dan binatang-binatang qalaa-id, dan jangan (pula) mengganggu orang-orang yang mengunjungi Baitullah sedang mereka mencari kurnia dan keridhaan dari Tuhannya dan apabila kamu telah menyelesaikan ibadah haji, maka bolehlah berburu. Dan janganlah sekali-kali kebencian(mu) kepada sesuatu kaum karena mereka menghalang-halangi kamu dari Masjidilharam, mendorongmu berbuat aniaya (kepada mereka).* ***Dan tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan pelanggaran***. *Dan bertakwalah kamu kepada Allah, sesungguhnya Allah amat berat siksa-Nya.*" (al-ma'idah:2).

Pada potongan ayat tersebut Allah SWT memerintahkan untuk saling tolong-menolong dalam aktivitas kebaikan dan dilarang untuk tolong-menolong

dalam perbuatan yang salah dan mengakibatkan dosa, potongan ayat tersebut dalam Tafsir Ibnu Katsir dijelaskan Allah SWT memerintahkan kepada hamba-hamba- Nya yang beriman untuk saling menolong dalam berbuat kebaikan yaitu kebajikan dan meninggalkan hal-hal yang mungkar, hal ini dinamakan ketakwaan. Allah SWT melarang mereka bantu-membantu dalam kebatilan serta tolong-menolong dalam perbuatan dosa dan hal- hal yang diharamkan. Ibnu Jarir mengatakan bahwa dosa itu ialah meninggalkan apa yang diperintahkan oleh Allah untuk dikerjakan.

Berdasarkan kajian Al-Qur'an surah al-ma'idah ayat 2, kita diperintahkan agar saling tolong menolong sesama muslim dalam hal kebaikan, hal itu sama halnya dengan definisi teknologi yaitu untuk memberikan kenyamanan serta kemudahan bagi manusia lain. Pada penelitian ini penulis melakukan penelitian terkait *Software Development Life Cycle* yaitu pada tahap pengujian (*testing*) karena tahap pengujian harus dioperasikan secara efektif untuk merilis perangkat lunak untuk pengguna yang bebas dari *bug*. Bebas dari *bug* adalah hal yang paling penting untuk diperhatikan dalam pengembangan perangkat lunak untuk keefektifan dan efisiensi kinerja perangkat lunak bagi pengguna serta mempermudah pekerjaan para pengembang perangkat lunak.

Cacat perangkat lunak didefinisikan sebagai cacat pada perangkat lunak seperti cacat pada dokumentasi, pada kode program, pada desain dan hal – hal lain yang menyebabkan kegagalan perangkat lunak. Dalam Proses pengembangan perangkat lunak yang berkualitas dan kompleks membutuhkan biaya yang tinggi, Oleh karena itu dibutuhkan cara yang efektif untuk meminimalkan biaya dan usaha dalam membangun perangkat lunak. Salah satu cara untuk mengembangkan

perangkat lunak berkualitas adalah dengan meminimalkan terjadinya cacat ketika perangkat lunak telah dijalankan menggunakan teknik prediksi cacat perangkat lunak. Prediksi cacat perangkat lunak dapat mendeteksi modul terkecil perangkat lunak yang memiliki kecenderungan cacat.

Menurut Wahono (2015 : 5) terdapat lima topik ilmu *machine learning* yang sering digunakan untuk memprediksi cacat perangkat lunak, kelima topik tersebut yaitu *Estimation, Association, dataset Analysis, Clustering* dan *Classification.* Proses prediksi cacat perangkat lunak dengan *machine learning* tersebut dilakukan dengan menggunakan *dataset* NASA MDP (*Metric Data Program*) yang berisi sekumpulan fitur-fitur *log* data cacat perangkat lunak, namun perlu diketahui bahwa dari keseluruhan fitur yang ada tidak semua fitur tersebut memiliki pengaruh signifikan terhadap proses prediksi cacat perangkat lunak karena yang digunakan dibuat tidak khusus untuk prediksi cacat perangkat lunak (Arar& Ayan, 2017). Oleh karena itu fitur-fitur yang tidak mempunyai pengaruh yang besar terhadap proses prediksi cacat perangkat lunak akan diabaikan atau dihapus untuk memaksimalkan proses prediksi cacat perangkat lunak.

Fokus penelitian ini adalah untuk menentukan fitur yang relevan atau yang berpengaruh signifikan terhadap proses prediksi cacat perangkat lunak, fitur yang berpengaruh signifikan memungkinkan proses prediksi cacat perangkat lunak akan menjadi lebih akurat, berdasarkan penelitian-penelitian yang telah dilakukan oleh beberapa peneliti, melakukan proses seleksi fitur sebelum melakukan proses klasifikasi dapat meningkatkan akurasi atau prediksi. Seperti penelitian yang dilakukan oleh Akbar (2017), yang berkontribusi pada proses pemilihan fitur dengan metode *Gain Ratio* dan melakukan normalisasi data sebelum data tersebut

di klasifikasi untuk mengetahui akurasi prediksi dan hasil penelitiannya nilai akurasi dari prediksi cacat perangkat lunak meningkat. Perbedaan dengan penelitian yang akan dilakukan ini adalah pada metode seleksi fitur dan metode *preprocessing data* yang sebelumnya menggunakan *Gain Ratio* dan normalisasi data, pada penelitian kali ini penulis menggunakan seleksi fitur *Symmetrical Uncertainty* dan untuk tahap *preprocessing* data penulis menggunakan metode diskritisasi data.

Penelitian tentang seleksi fitur lainnya dilakukan oleh Kumar dan Sree (2014), mereka melakukan penelitian pada beberapa metode seleksi fitur berbasis perangkingan pada 5 *dataset Automated Student Assessment Prize* (ASAP) dan melakukan uji coba dengan menggunakan model *Sequential Minimal Optimization* pada hasil akhirnya metode seleksi fitur *Symmetrical Uncertainty* (SU) menjadi metode terbaik dibandingkan beberapa metode seleksi fitur lainnya.

Berdasarkan latar belakang diatas dan penelitian-penelitian yang telah dilakukan oleh para peneliti yang meneliti tentang prediksi cacat perangkat lunak, seleksi fitur merupakan fokus dari penelitian ini. Fitur yang relevan memungkinkan proses prediksi cacat perangkat lunak akan menjadi lebih akurat (Arar & Ayan, 2017) dan (Akbar, 2017). Penelitian ini memilih seleksi fitur dengan algoritma *Symmetrical Uncertainty* berdasarkan penelitian yang dilakukan oleh Kumar dan Sree (2014) yang menunjukkan bahwa seleksi fitur dengan menggunakan algoritma *Symmetrical Unscertainty* merupakan algoritma terbaik dibandingkan beberapa algoritma seleksi fitur lainnya.

1.2. Pernyataan Masalah

Berdasarkan uraian pada latar belakang diatas, maka dapat dirumuskan masalah yang dijadikan sebagai fokus penelitian ini yaitu menentukan fitur apasaja yang memiliki pengaruh yang signifikan atau relevan terhadap prediksi cacat perangkat lunak berdasarkan seleksi fitur *Symmetrical Uncertainty*.

1.3. Tujuan Penelitian

Berdasarkan rumusan masalah, maka tujuan dari penelitian ini adalah untuk menentukan fitur-fitur dari *dataset* MDP yang memiliki pengaruh signifikan terhadap prediksi cacat perangkat lunak berdasarkan seleksi fitur *Symmetrical Uncertainty*.

1.4. Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian ini adalah dapat meningkatkan kualitas perangkat lunak melalui aktivitas pengujian dengan cara prediksi cacat perangkat lunak yang lebih akurat sehingga cacat perangkat lunak bisa terdeteksi lebih dini.

1.5. Batasan Masalah

Untuk menjaga fokus dari penelitian ini, maka batasan masalah dari penelitian ini adalah data yang digunakan berupa 5*datasetlog* cacat perangkat lunakyang diunduh dari PROMISE (*Predictor Models in Software Engineering*) *repository* melaluihttp://promise.site.uottawa.ca/SERepository/s-page.html pada hari Senin, 5 Maret 2018 yaitu CM1, JM1, KC1, KC2 dan PC1.

1.6. Sistematika Penulisan

Dalam penulisan skripsi ini, secara keseluruhan terdiri dari lima bab yang masing-masing bab disusun dengan sistematika sebagai berikut :

BAB I PENDAHULUAN

Bab ini merupakan bagian awal, dalam bab ini berisi latar belakang, pernyataan masalah, tujuan penelitian, batasan masalah, manfaat penelitiaan dan sistematika penulisan laporan.

BAB II STUDI PUSTAKA

Bab ini berisi tentang teori-teori yang berhubungan dengan permasalahan yang diangkat dari penelitian ini, antara lain penelitian-penelitian terkait, cacat perangkat lunak, prediksi cacat perangkat lunak, algoritma diskritisasi data *Equal Width Binning*, algoritma seleksi fitur *Symmetrical Uncertainty*, algoritma pemecahan data *Split Percentage*, algoritma klasifikasi *Naïve Bayes*.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan tentang analisis sistem dan perancangan sistem yang akan dibuat sekaligus batasan-batasan sistem serta di dalamnya juga terdapat beberapa hitungan manual algoritma-algoritma yang dipakai.

BAB IV UJI COBA DAN PEMBAHASAN

Bab ini berisi mengenai pengujian dan analisis dari hasil pengujian dari sistem yang telah dibangun berdasarkan hasil perancangan pada bab 3 sebelumnya.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran seluruh penelitian yang telah dilakukan.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Terdapat beberapa penelitian terdahulu yang mendukung antara lain sebagaiberikut;

Penelitian di bidang prediksi cacat perangkat lunak menggunakan *framework* dengan tiga komponen utama yaitu *data preprocessor* dengan algoritma *log-filtering*, seleksi fitur dengan algoritma *Info Gain* dan *learning algorithm* pada 13 *dataset* NASA MDP dan 4 AR *dataset* kemudian melakukan pengujian dengan menggunakan tiga *classifier* yaitu *Naïve Bayes*, J48 dan *OneR* dengan 12 skema pengujian. Hasil dari penelitian tersebut *Naïve Bayes* merupakan algoritma dengan akurasi tertinggi (Song dkk.,2011). *Framework* Penelitian ini merupakan dasar penelitian ini dalam melakukan tahapan-tahapan berjalannya sistem yang akan dibuat.

Penelitian lain dilakukan oleh Karabulut dkk. (2012), pada penelitian tersebut Karabulut dkk melakukan penelitian terhadap beberapa metode seleksi fitur, beberapa seleksi fitur antara lain *Information Gain* (IG), *Gain ratio* (GR), *Symmetrical Uncertainty* (SU), *Relief-F* (RF), *One-R*, *Chi-Square* (CS) yang diterapkan untuk mendapatkan fitur yang relevan pada 15 *dataset* dan pengukuran performa dengan menggunakan algoritma klasifikasi antara lain; *Naïve Bayes* (NB), *Multilayer Perceptron* (MLP) dan *decision tree J48*. Kesimpulan dari penelitiannya adalah seleksi fitur dapat meningkatkan akurasi dari sebuah *classifier*, karena seleksi fitur dapat mengurangi jumlah dimensi sehingga

penggunaan prosesor dan memori berkurang, data menjadi lebih mudah dipahami dan lebih mudah untuk dipelajari.

Penelitian tentang seleksi fitur lainnya dilakukan oleh Kumar dan Sree (2014), pada penelitian tersebut Kumar dan Sree membandingkan beberapa metode seleksi fitur berbasis perangkingan pada 5 *dataset Automated Student Assessment Prize* (ASAP) dan melakukan uji coba dengan menggunakan model *sequential minimal optimization*. Pada hasil akhirnya metode seleksi fitur *Symmetrical Uncertainty* (SU) menjadi metode terbaik dibandingkan beberapa metode seleksi fitur lainnya.

2.2 Cacat Perangkat Lunak

Secara umum ada beberapa tahap dalam *Software Development Life Cycle* (SDLC) yaitu analisa kebutuhan, desain, implementasi, ujicoba dan *release* (Arar & Ayan,2015). Tahap pengujian harus dioperasikan secara efektif untuk merilis perangkat lunak untuk pengguna yang bebas dari *bug*. Bebas dari *bug* adalah hal yang paling penting untuk diperhatikan dalam pengembangan perangkat lunak untuk keefektifan dan efisiensi kinerja perangkat lunak bagi pengguna.

Cacat pada perangkat lunak didefinisikan sebagai cacat pada perangkat lunak yang mungkin terjadi seperti cacat pada kode program, dokumentasi, pada desain, dan hal-hal lain yang menyebabkan kegagalan perangkat lunak. Cacat perangkat lunak dapat muncul pada berbagai tahap proses pengembangan perangkat lunak (Pressman, 2001). Cacat perangkat lunak merupakan faktor penting yang mempengaruhi kualitas perangkat lunak. Kualitas perangkat lunak dapat ditingkatkan dengan mencegah munculnya cacat perangkat lunak melalui

perbaikan yang mungkin menghasilkan cacat perangkat lunak pada proses pengembangan perangkat lunak (Boehm & Basili, 2001). Menurut Runeson dkk.(2006 : 83) cacat perangkat lunak terdiri dari cacat kebutuhan, cacat desain dan cacat kode.

Untuk mempertahankan kualitas perangkat lunak perlu adanya pengawasan atau pengujian seperti mendeteksi kecacatan perangkat lunak tersebut. Tahap ini merupakan tahap yang paling penting dalam SDLC, tahap ini melakukan pengujian perangkat lunak dan pemeriksaan validasi, setelah produk dinyatakan tidak cacat maka persetujuan diberikan dan perangkat lunak bisa digunakan pada kebutuhan. Biaya untuk memperbaiki cacat yang terdeteksi ketika masih pada tahap pengembangan jauh lebih sedikit dibandingkan ketika perangkat lunak sudah digunakan pada kebutuhan (Pelayo & Dick, 2007 : 69-72). Salah satu cara yang bisa dilakukan untuk pengujian adalah teknik prediksi cacat perangkat lunak, teknik tersebut dapat mendeteksi hingga modul terkecil perangkat lunak yang memiliki kecenderungan cacat.

2.3 Prediksi Cacat Perangkat Lunak

Prediksi cacat perangkat lunak atau *Software Defect Prediction* (SDP) adalah salah satu kegiatan yang sangat membantu pada tahap pengujian SDLC. Teknik pencegahan cacat perangkat lunak pertama kali diusulkan oleh IBM *corporation* dan dapat digunakan untuk mencegah munculnya cacat perangkat lunak ditahap lanjut pada proses pengembangan perangkat lunak (Chang dkk.,2009). Setelah itu, muncul beberapa pendekatan untuk mencegah munculnya cacat perangkat lunak seperti *causal analysis* dan *prediction model*.

Teknik pemodelan prediksi cacat perangkat lunak berusaha untuk menemukan sejumlah pola cacat perangkat lunak yang dapat digunakan untuk memprediksi kemunculan cacat perangkat lunak dimasa mendatang. Sebuah pola cacat perangkat lunak didefinisikan sebagai sekumpulan fitur yang dapat digunakan untuk mendeskripsikan dan memprediksikan kemunculan dari cacat perangkat lunak. Pola cacat perangkat lunak dapat diturunkan dengan menerapkan model statistik pada perangkat lunak yang mengandung cacat. Untuk menentukan apakah sebuah perangkat lunak mengandung cacat perangkat lunak, fitur-fitur dari perangkat lunak dibandingkan dengan pola cacat perangkat lunak. Fitur-attribuat yang digunakan untuk mengukur produk perangkat lunak antara lain ukuran, kompleksitas, usaha dan sejarah perubahan.

Terdapat beberapa penelitian telah dilakukan terkait topik prediksi cacat perangkat lunak dengan menerapkan algoritma *machine learning* yang berbeda-beda. Menurut Wahono (2015 : 5), pada *literature review-nya* mengungkapkan bahwa prediksi cacat perangkat lunak saat ini berfokus pada lima topik yaitu :

1. *Estimation*
2. *Association.*
3. *Classification.*
4. *Clustering.*
5. *Dataset Analysis.*

Dari lima topik diatas, prediksi cacat perangkat lunak pada tahun 2000 sampai 2013 mengungkapkan bahwa *classification* atau klasifikasi merupakan pilihan dari kebanyakan peneliti yang melakukan penelitian dibidang prediksi cacat perangkat lunak dengan prosentase 77,46% dari semua topik yang

disebutkan diatas, terdapat tiga alasan mengapa kebanyakan peneliti memilih klasifikasi yaitu :

1. Klasifikasi sesuai dengan kebutuhan industri yang memerlukan beberapa metode untuk memprediksi modul mana yang lebih cenderung rentan atau rawan. Dengan demikian, hasil prediksi dapat digunakan untuk mendukung sumber daya pengujian yang lebih tepat sasaran.
2. *Dataset* NASA MDP yang memang disiapkan untuk klasifikasi.
3. *Clustering* dan *association* biasanya menghasilkan kinerja yang tidak diinginkan dan tidak dapat dipublikasikan dalam literatur.

Proses prediksi cacat perangkat lunak dengan teknik klasifikasi tersebut dilakukan dengan menggunakan *dataset* dari NASA MDP (*Metric Data Program*). *Dataset* ini berisi data-data berupa *log* cacat perangkat lunak. Setiap baris menunjukkan modul terkecil dari sebuah program yaitu berupa *method* atau *function* dan setiap kolom menunjukkan nilai metrik.

Dataset yang digunakan pada penelitian adalah 5 *dataset* NASA MDP yaitu CM1, JM1, KC1, KC2 dan PC1.Berikut merupakan isi dari *dataset* CM1:

Tabel 2.1 Dataset CM1

No.	Simbol	Keterangan
1.	loc	**McCabe's *"line count of code"***
2.	v(g)	**McCabe "*Cyclomatic complexity*"**
3.	ev(g)	**McCabe "*Essential complexity*"**
4.	iv(g)	**McCabe "*Design complexity*"**
5.	UniqOp	***unique operators***
6.	UniqOpnd	***unique operands***

Tabel 2.2 Dataset CM1 (Lanjutan)

7.	TotalOp	**total operator**
8.	TotalOpnd	**total operand**
9.	n	**Halstead "Total operator + operand"**
10.	v	**Halstead "Volume"**
11.	L	**Halstead "*Programlength*"**
12.	d	**Halstead "*Difficulty*"**
13.	i	**Halstead "*Intelligence*"**
14.	e	**Halstead " *Effort to write program*"**
15.	b	**Halstead "*Error estimate*"**
16.	t	**Halstead "*Time to write program*"**
17.	IOCode	***Count of Statement Lines***
18.	IOComment	***Count of lines of comments***
19.	IOBlank	***Count of blank lines***
20.	lOCodeAndComment	***Count of Code and Comments Lines***
21.	branchCount	***Branch count***
22.	*defect*	**Hasil prediksi {*false,true*}**

Berdasarkan tabel 2.1 diatas terlihat bahwa fitur atau metrik yang digunakan berjumlah 21 fitur, berikut adalah penjelasan dari setiap fitur yang digunakan (Akbar, 2017);

a. *Line of code* (loc)

Metrik loc ini merupakan metrik yang umum digunakan untuk mengukur kompleksitas sebuah kode program dengan cara menghitung jumlah baris kode program suatu perangkat lunak.

b. *Cyclomatic complexity* (v(g))

Metrik v(g) adalah nilai metrik kompleksitas yang digunakan untuk menilai struktur logika keputusan sebuah kode program. Logika keputusan

dapat berupa statemen *if* atau *looping* yang diilustrasikan sebagai sebuah *graph* yang terdiri dari *arc*(e) dan *node*(n). Rumus perhitungannya adalah v(g) = e – n + 2.

c. *Essential complexity* (ev(g))

Metrik ev(g) adalah nilai metrik kompleksitas yang dihitung dengan mengurangi nilai *cyclomatix complexity* v(g) dengan jumlah *subflowgraph* yang terdiri dari *single entry* dan *single exit*. Rumus perhitungan *Essential complexity* adalah ev(g) = v(g) – m, m adalah jumlah *subflowgraph* yang merupakan *single entry* dan *single exit.*

d. *Design complexity*(iv(g))

Design complexity adalah *cyclomatic complexity* dari suatu modul yang mengurangi *flowgraph*. *Flowgraph* "g" dari sebuah modul dikurangi untuk menghilangkan kompleksitas yang tidak mempengaruhi keterkaitan antara modul desain. *Design complexity* dihitung dengan mengurangi modul pada *flowgraph* dengan cara menghilangkan *node decisicon* yang tidak memiliki dampak pada kontrol sebuah program.

e. *Unique operators* (UniqOp)

Metrik UniqOp merupakan nilai metrik yang didapat dengan cara menghitung jumlah operator yang berbeda dalam sebuah kode program.

f. *Unique operands* (UniqOpnd)

Metrik UniqOpnd merupakan nilai metrik yang didapat dengan cara menghitung jumlah *operand* yang berbeda dalam sebuah kode program.

g. Total operator (TotalOp)

Metrik TotalOp merupakan nilai metrik yang didapat dengan cara menghitung jumlah operator dalam sebuah kode program.

h. Total operand (TotalOpnd)

Metrik uniq_Op merupakan nilai metrik yang didapat dengan cara menghitung jumlah operator yang berbeda dalam sebuah kode program. Sebagai contoh perhitungan perhatikan kode program berikut;

```
main(){ int a, b, c, avg;
     scanf("%d %d %d", &a, &b, &c);
     avg = (a + b + c) / 3;
     printf("avg = %d", avg);}
```

uniq_Op pada program tersebut adalahmain, (), {}, int, scanf, &, =, +, /, printf sehingga nilai uniq_Op = 10.

i. Total operator + operand (n)

Metrik “n” merupakan nilai metrik yang didapat dengan cara menghitung jumlah TotalOp dengan TotalOpnd dalam sebuah kode program. Rumus untuk menghitung “n” adalah n = TotalOp + TotalOpnd.

j. Volume (v)

Metrik “v” merupakan nilai metrik yang didapat dengan cara menghitung sesuai. Rumus v = n * $log_2(\text{UniqOp} + \text{UniqOpnd})$

k. *Program length* (L)

Metrik L merupakan metrik yang digunakan untuk menghitung rasio antara metrik *volume* dengan total operators.Rumus dari Ladalah V/N.

l. *Difficulty* (d)

Metrik d merupakan invers dari metrik program length. Nilai metrik difficulty akan bertambah saat jumlah operators dan operands bertambah. Rumus dari metrik d = 1/I.

m. *Intelligence* (i)

Metrik i digunakan untuk mengukur kompleksitas algoritma yang diterapkan pada kode program dan tidak bergantung pada bahasa pemrograman yang dipakai. Rumus dari metrik i = v / d.

n. *Effort to write program* (e)

Metrik e adalah metrik yang digunakan untuk mengukut *effort* atau usaha untuk menulis sebuah kode program. Rumus untuk menghitung nilai metrik ini dengan persamaan e = d * v.

o. *Error estimate* (b)

Metric b merupakan metric yang menunjukkan jumlah bug validasi, perhitungan b dilakukan dengan rumus b = v / 3000.

p. *Time to write program* (t)

Metrik T merupakan metrik yang digunakan untuk mengukur perkiraan waktu yang proportional dari setiap effort. Rumus untuk menghitung nilai T dengan T = e / 18(seconds).

q. *Count of Statement Lines* (IOCode)

Metrik IOCode merupakan metrik yang menghitung baris statement (if, while, for) dalam baris kode program.

r. *Count of lines of comments* (IOComment)

Metrik IOComment merupakan metrik yang menghitung jumlah baris komen dalam kode program.

s. *Count of blank lines* (IOBlank)

Metrik IOBlank merupakan metrik yang menghitung jumlah baris yang kosong dalam kode program.

t. *Count of Code and Comments Lines* (lOCodeAndComment)

Metrik IOCodeAndComment merupakan metrik yang menghitung jumlah baris kode yang terdapat komen.

u. *Branch count*(branchCount)

Metrik branchCount adalah metrik yang menghitung jumlah percabangan dalam kode program. Jumlah percabangan oleh statement *if* dan case dalam kode program.

2.4 Diskritisasi Data

Diskritisasi data merupakan proses mereduksi sekumpulan nilai yang terdapat pada fitur kontinu, dengan membagi *range* dari fitur ke dalam *interval*. menurut Suyanto (2017 : 112). Diskritisasi adalah salah satu tahap pada praproses data yang paling penting. Sebagian besar algoritma *machine learning* yang ada mampu mengekstrak pengetahuan dari *database* yang menyimpan fitur diskrit.Jika fitur kontinyu, algoritma dapat diintegrasikan dengan algoritma diskritisasi yang mengubahnya menjadi fitur diskrit. Pada *dataset* NASA MDP data berupa data kontinu, menurut Santoso (2013 : 9) dalam bukunya menjelaskan bahwa data/variabel kontinu merupakan data yang nilainya dapat menepati semua nilai diantara dua titik atau interval, pada umumnya nilai dari variabel/data

kontinu diperoleh dari hasil pengukuran, sehingga dapat dijumpai nilai-nilai pecahan ataupun nilai bulat. Pada penelitian yang dilakukan oleh Singh dan Verma (2009 : 837-839) Diskritisasi data dapat meningkatkan akurasi klasifikasi *Naïve Bayes* dan *J48* pada prediksi cacat perangkat lunak. Terdapat banyak metode diskritisasi data, contoh metode dari diskritisasi data, diantaranya *binning, histogram, clustering, decision tree* dan analisis korelasi.

2.5 Diskritisasi Data dengan *Equal Width Binning*

Metode ini adalah salah satu metode diskritisasi *unsupervised* yang *simple* dan termasuk kategori *binning*, metode ini pun banyak digunakan karena *simpel* sehingga mudah diterapkan. Yang dkk. (2010 : 106) menjelaskan alur proses diskritisasi dengan metode *Equal Width Binning* sebagai berikut;

a. Mendifinisakan nilai K, K adalah jumlah interval. K akan membagi garis bilangan antara Vmin dan Vmax menjadi interval K dengan lebar yang sama. Vmin adalah nilai terkecil dari fitur dan Vmax adalah nilai terbesar dari fitur. Dougherty *dkk.* (1995 :194–202) mendefinisikan nilai $K = max\{1,2\ log\ L\}$, nilai L adalah jumlah nilai pengamatan yang berbeda untuk setiap fitur.

b. Menghitung lebar setiap interval dengan rumus $W = \frac{Vmax - Vmin}{K}$

c. Menentukan titik potong yang didapat dengan menghitung,

   Vmin + W, Vmin + 2W,…,Vmin + (K-1)W.

2.6 Seleksi Fitur

Seleksi Fitur merupakan salah satu teknik dari reduksi data artinya konsep dasar dari seleksi fitur adalah menghilangkan data yang tidak begitu berpengaruh dalam suatu. Seleksi Fitur merupakan proses menghapus data yang redundan atau data yang kiranya tidak begitu berpengaruh dari suatu *dataset*, dari proses seleksi fitur dapat meningkatkan akurasi dari klasifikasi karena fitur-fitur yang sebelumnya tidak begitu berpengaruh atau redundan akan terseleksi terlebih dahulu termasuk data-data yang mengandung *noisy* (Karabulut dkk., 2012 : 323-327).

Banyak fitur yang tidak relevan pada data yang akan di proses, Jadi mereka harus dihapus, terdapat banyak algoritma *data mining* yang tidak bekerja dengan baik pada data yang fitur yang banyak. Karena itu teknik pemilihan fitur perlu diterapkan sebelum semua jenis algoritma *data mining* diterapkan. Itu tujuan utama pemilihan fitur (Beniwal & Arora, 2012 : 1).

2.7 Seleksi Fitur dengan *Symmetrical Uncertainty*

Symmetrical Uncertainty merupakan salah satu metode seleksi fitur yang berbasis *entropy*, arti dari *entropy* sendiri adalah keberagaman, dalam *data mining* ,*entropy* digunakan untuk mengukur *heterogenitas* (keberagaman) dalam suatu himpunan data. Semakin heterogen suatu himpunan data, semakin besar pula nilai *entropy*-nya.

Pada penelitian ini menggunakan metode diskritisasi untuk mengubah data pada menjadi data kategoris dengan algoritma *Equal Width Binning* sebagaimana

sudah dijelaskan di sub bab sebelumnya.Terdapat beberapa tahapan untuk menghitung nilai *Symmetrical Uncertainty*diantaranya sebagai berikut :

a. Menghitung nilai *Entropy.*
b. Menghitung nilai *Information Gain.*
c. Menghitung nilai *Symmetrical Uncertainty.*

Secara ringkas perhitungan nilai *Symmetrical Uncertainty*adalah sebagai berikut;

a. Penghitungan nilai *Entropy*

$$Entropy(S) = \sum_{i}^{C} -p^i \log^2 p^i$$

Keterangan :

C : jumlah nilai yang ada pada fitur target (jumlah kelas)

Pi : rasio antara jumlah sampel pada kelas i terhadap semua sampel pada himpunan data.

Dari diatas dapat dicermati bahwa apabila hanya terdapat 2 kelas dan dari kedua kelas tersebut memiliki komposisi jumlah sampel yang sama, maka entropynya = 0.

b. Penghitungan nilai *Information Gain*

Setelah mendapatkan nilai *entropy*, maka langkah selanjutnya adalah melakukan perhitungan nilai *Information Gain*. *Information Gain* didefinisikan sebagai ukuran efektvitas suatu fitur dalam mengklasifikasikan data. Perhitungan nilai *Information Gain* dilakukan setelah perhitungan nilai *Entropy* karena pada perhitungan nilai *Information Gain* memerlukan inputan berupa nilai *Entropy dan* nilai *Information Gain* hasil perhitungan ini nantinya akan menjadi input

perhitungan *Gain ratio*. Berdasarkan perhitungan matematis *Information Gain* dari suatu fitur A dapat dilihat pada rumus berikut:

fitur A dapat dilihat pada rumus

$$Gain(S, A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} entr_{(S_v)} \quad S^v)$$

Keterangan :

A : Fitur

V : menyatakan suatu nilai yang mungkin untuk fitur A

Values (A) : himpunan nilai-nilai yang mungkin untuk fitur A

$|^{alue}_{S^v}|$: jumlah smpel untuk nilai v

$|^{S_v|}_{S|}$: jumlah seluruh sampel data

Entropy ($_{S^v)}$: entropy untuk sampel-sampel yang memiliki nilai v

c. Penghitungan nilai *Symmetrical Uncertainty*

Setelah mendapat nilai *Information Gain* langkah selanjutnya yaitu menghitung nilai *Symmetrical Uncertainty* dengan menggunakan rumus berikut;

; nilai Sym

$$SU(S, A) = 2 * \frac{Gain(S, A)}{H(S) + H(A)}$$

Keterangan :

Gain (S,A) : Nilai *Information Gain.*

H(S) : *Entropy* seluruh sampel data.

H(A) : *entropy* fitur.

SU : *Symmetrical Uncertainty*

2.8 *Split Percentage*

Split Percentage merupakan salah satu teknik testing dari pencacahan data klasifikasi, sesuai dengan namanya metode *split percentage* mempartisi himpunan data menjadi 2 bagian yaitu data *training* (data latih) dan data *testing* (data uji) berdasarkan berapa persen pembagiannya, dalam penelitian ini data dibagi menjadi 2 bagian dengan pembagian 60 persen (data *training*) banding 40 persen (data *testing*) berdasarkan penelitian yang dilakukan oleh Sastry dan Prasad (2015 (Sastry & Prasad , 2015) akurasi dari klasifikasi mengalami kenaikan dengan menggunakan metode pencacahan data *split percentage* 60 persen.

2.9 *Naïve Bayes*

Naive Bayes adalah salah satu algoritma yang paling banyak digunakan dalam masalah klasifikasi karena kesederhanaan, keefektifan, dan ketahanannya (Arar & Ayan, 2017) dan Menurut Akbar (2017 : 13) *Naive Bayes* merupakan metode yang digunakan dalam statistika untuk menghitung peluang dari suatu hipotesis, *Naive Bayes* menghitung peluang suatu kelas berdasarkan pada fitur yang dimiliki dengan menganggap setiap fitur bersifat *independent* dan penentuan kelas berdasarkan nilai probabilitas yang paling tinggi.

Metode klasifikasi ini didasarkan pada teorema *Bayes* yang ditemukan oleh Thomas Bayes pada abad ke-18 dengan mengasumsikan independensi atau ketidaktergantungan yang kuat (naif) pada fitur yang ada artinya sebuah fitur pada sebuah data tidak saling berkaitan atau tidak memiliki hubungan apapun antara fitur yang satu dengan fitur lainnya. *Naive Bayes* memiliki keunggulan untuk pengembangan *data mining*, yaitu kemudahan konstruksinya dan tidak

membutuhkan parameter skema pengulangan yang kompleks sehingga mudah dalam membaca data dalam jumlah yang besar. Hal ini terjadi karena desain rancangan penuntunan klasifikasi terhadap data. Selain itu, metode ini dinyatakan sebagai algoritma yang mempunyai sifat *simplicity*, *elegance* dan *robustness* (Subiyakto, 2008), berikut adalah rumus umum teorema *Bayes* (Prasetyo, 2012:59-62; Suyanto, 2017:126-129);

$$P(Y|X) = \frac{P(Y)P(X|Y)}{P(X)}$$

$P(Y|X)$: Probabilitas akhir bersyarat, suatu hipotesis Y terjadi jika diberikan bukti X terjadi

$P(X|Y)$: Probabilitas sebuah bukti X terjadi akan mempengaruhi hipotesis Y

$P(Y)$: Probabilitas awal hipotesis Y terjadi tanpa memandang bukti apapun

$P(X)$: Probabilitas awal bukti X terjadi tanpa memandang hipotesis/bukti yang lain

Dasar dari aturan *bayes* adalah bahwa hasil dari hipotesis atau peristiwa (Y) dapat diperkirakan berdasarkan pada beberapa bukti (X) yang diamati, misalnya ada beberapa bukti sebagai berikut X_1, X_2,..,X_n, sehingga probabilitas akhir dituliskan sebagai;

$$P(Y|X^1, X^2, \ldots, X^n) = \frac{P(X_1, X_2, \ldots, X_n|Y)P(Y)}{P(X^1, X^2, \ldots, X^n)}$$

Karena *NaïveBayes* mengasumsikan bahwa setiap fitur independen maka rumus diatas dapat diubah menjadi;

$$P(Y|X^1, X^2, \ldots, X^n) = \frac{P(X^1|Y)}{P(X^1)} x \frac{P(X_2|Y)}{x\ P(X^2)} \frac{x}{x} \frac{\ldots\ x}{\ldots} \frac{P(X_n|Y)}{x\ P(X^n)} x\ P(Y)$$

Berdasarkan rumus diatas dengan k kelas maka rumus *Naïve Bayes* untuk klasifikasi adalah sebagai berikut;

asi adalah sebagai berikut

$$P(Y^k|X) = \frac{P(Y^k) \prod_{i=1}^{n} P(X^i|Y^k)}{P(X)}$$

$P(Y_k|X)$ merupakan probabilitas data dengan vektor X pada kelas Y_k. $P(Y_k)$probabilitas awal kelas Y_k. $\prod_{i=1}^{n} P(X_i|Y_k)$ probabilitas independen kelas Y_k dari semua fitur (n) dalam vector X. Karena nilai $P(X)$ selalu tetap (artinya, tuple X memiliki probabilitas yang sama untuk masuk kedalam kelas manapun) sehingga dalam perhitungan prediksi akan memaksimalkan $P(Y_k) \prod_{i=1}^{n} P(X_i|Y_k)$. Dalam perhitungan $P(X_i|Y_k)$ umumnya menggunakan fitur bertipe kategoris namun untuk fitur bertipe numerik (kontinu) ada 2 perlakuan yaitu;

a. Melakukan diskritisasi terhadap fitur kontinu, mengganti nilai fitur kontinu dengan nilai interval diskrit.
b. Membiarkan fitur tetap bertipe kontinu dan melakukan perhitungan sesuai dengan distribusi gaussian yang didefinisiakan sebagai;

$$P(X^i|Y^k) = \frac{1}{\sigma_{Y_k}\sqrt{2\pi}} e^{-\frac{(x - \mu_{Y_k})^2}{2\sigma_{Y_k}^2}}$$

Yang diketahui bahwa σ_{Y_k} dan μ_{Y_k}adalah standart deviasi dan rata-rata dari nilai-nilai pada fitur kontinu X_i untuk kelas Y_k. Penentuan kelas hasil prediksi dengan memilih nilai terbesar dari probabilitas kelas hasil perhitungan $P(Y_k) \prod_{i=1}^{n} P(X_i|Y_k)$, jika ditulis dalam rumus tampil sebagai berikut;

$$arg\ max\ P(Y^k) \prod_{i=1}^{n} P(X^i | Y^k)$$

2.10 Pengujian Hasil Seleksi Fitur dengan *Naïve Bayes*

Prediksi cacat perangkat lunak dengan menggunakan algoritma *naïve bayes* merupakan algoritma yang sangat sering digunakan pada prediksi cacat perangkat lunak, menurut Wahono (2015) penelitian terkait prediksi cacat perangkat lunak menggunakan teknik klasifikasi dari tahun 2000 sampai 2013 menggunakan algoritma *naïve bayes* memiliki prosentase 26,42% dan merupakan algoritma yang paling sering digunakan dan menurut Song dkk. (2011) dan Menzies dkk. (2007) prediksi cacat perangkat lunak menggunakan *Naïve Bayes* memiliki preforma yang relatif baik.

Terdapat beberapa penelitian terdahulu yang menggunakan algoritma klasifikasi *Naïve Bayes* diantaranya :

Penelitian pertama dilakukan oleh Song dkk. (2011), pada penelitian tersebut Song dkk melakukan prediksi cacat perangkat lunak menggunakan *framework* yang mereka buat dengan tiga komponen utama yaitu *data preprocessor* dengan algoritma *log-filtering*, seleksi fiturdengan algoritma *Info Gain* dan *learning algorithm* pada 13 *dataset* NASA MDP dan 4 AR *dataset*kemudian melakukan pengujian dengan menggunakan tiga *classifier* yaitu naïve bayes, J48 dan OneR dengan 12 skema pengujian, hasil dari penelitian tersebut naïve bayes merupakan algoritma dengan akurasi tertinggi.

Penelitian kedua dilakukan oleh Kumaresh dkk. (2014), pada penelitian tersebut prediksi cacat perangkat lunak menggunakan 7 *dataset* NASA MDP dan

melakukan pengujian dengan dua klasifier yaitu *NaïveBayes* dan *BayesNet*. Hasil dari penelitian tersebut algoritma *Naïve Bayes* memiliki akurasi yang lebih tinggi dibandingkan algoritma *Bayes Net*.

Penelitian ketiga dilakukan oleh Akbar (2017), pada penelitian tersebut prediksi cacat perangkat lunak menggunakan 5 *dataset NASA MDP* dan melakukan kontribusi berupa pemilihan fitur dengan metode *Gain Ratio*, melakukan normalisasi data sebelum data tersebut di klasifikasi untuk mengetahui akurasi prediksi. Hasil penelitiannya menujukkan nilai akurasi dari prediksi cacat perangkat lunak meningkat.

2.11 Evaluasi Klasifikasi

Evaluasi terhadap suatu *classifier* umumnya dilakukan menggunakan sebuah himpunan data uji, yang tidak digunakan dalam pelatihan *classifier* tersebut, dengan suatu ukuran tertentu. Terdapat sejumlah ukuran yang dapat digunakan untuk menilai atau mengevaluasi model klasifikasi, diantaranya adalah *accuracy* atau tingkat pengenalan, *error rate* atau tingkat kesalahan atau kekeliruan klasifikasi, *recall* atau *sensitivity* atau *true positive rate*, *specificity* atau *true negative rate*, *precision*, *F-measure* atau F_1 atau *F-score* atau rata-rata harmonik dan *precision* dan *recall*, dan F_β (J. Han dkk., 2012), yang secara ringkas diilustrasikan pada Tabel 2.3

Tabel 2.3 Ukuran Evaluasi Model Klasifikasi

No	Ukuran	Rumus
1	*Accuracy* atau tingkat pengenalan	
2	*Error rate* atau tingkat kesalahan atau kekeliruan klasifikasi	
3	*Recall* atau *sensitivity* atau *true positive rate*	
4	*Specificity* atau *true negative rate*	
5	*Precision*	
6	*F* atau F_1atau *F-score* atau rata-rata harmonik dari *precision* dan *recall*	$\frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 \times precision \times recall}{precision + recall}$
7	F_β, dimana β adalah sebuah bilangan *riil nonnegative*	$\frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$

Empat istilah sangat penting untuk memahami semua ukuran evaluasi dalam tabel tersebut adalah sebagai berikut:

- ***TP*** atau *True Positives* adalah jumlah tuple positif yang dilabeli dengan benar oleh classifier. Yang dimaksud tuple positif adalah tuple actual yang berlabel positif, seperti tuple dengan label Bonus= ‘Ya’.
- ***TN*** atau *True Negatives* jumlah tuple negative yang dilabeli dengan benar oleh classifier. Yang dimaksud tuple negative adalah tuple actual yang berlabel negative, seperti tuple dengan label Bonus = ‘Tidak’.
- ***FP*** atau *False Positives* adalah jumlah tuple negative yang salah dilabeli oleh classifier. Misalnya, sebuah tuple pelanggan yang berlabel Bonus = ‘Tidak’ tetapi oleh classifier dilabeli Bonus =’Ya’.
- ***FN*** atau *False Negatives* adalah jumlah tuple positif yang salah dilabeli oleh classifier. Misalnya, sebuah tuple pelanggan yang berlabel Bonus= ‘Ya’ tetapi oleh classifier dilabelu Bonus=’Tidak’.

Keempat istilah diatas dapat digambarkan sebagai confusion matrix (Suyanto,2017 : 236)seperti diilustrasikan pada tabel 2.4

Tabel 2.4 Tabel Confusion Matrix

		Kelas hasil prediksi		
		Ya	Tidak	Jumlah
Kelas aktual	Ya	*TP*	*FN*	*P*
	Tidak	*FP*	*TN*	*N*
	Jumlah	*P'*	*N'*	*P+N*

Confusion matrix sangat berguna untuk menganalisis kualitas *classifier* dalam mengenali tuple-tuple dari kelas yang ada. *TP* dan *TN* menyatakan bahwa *classifier* mengenali tuple dengan benar, artinya tuple positif dikenali sebagai positif dan tuple negative dikenali sebagai negative. Sebalikanya, *FP* dan *FN* menyatakan bahwa *classifier* salah dalam mengenali tuple, tuple negative dikenali sebagai postif dan tuple negative dikenali sebagai postif. *P'* adalah jumlah tuple yang diberi label positif (*TP+FP*) sedangkan *N'* adalah jumlah tuple yang diberi label negative (*TN + FN*). Jumlah keseluruhan tuple dapat dinyatakan sebagai label negative (*TN + FN*). Jumlah ke

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

(*TP+TN+FP+FN*) atau (*P+N*) atau (*P'+N'*) (Suyanto,2017, p.234-236). Untuk mengetahui nilai akurasi suatu prediksi, dapat digunakan rumus :

BAB III

METODOLOGI PENELITIAN

3.1 Analisis Sistem

Pada subbab ini akan dijelaskan tentang analisis dan perancangansistem yang akan dibuat, analisis sistem penelitian merupakan rangkaian tahapan penelitian yang tersusun secara sistematis. Tujuan dari analisis dan perancangansistem ini adalah agar pelaksanaan penelitian mendapatkan hasil yang sesuai dengan tujuan penelitian. Adapun rangkaian tahapan yang akan dilakukan dapat dilihat pada gambar 3.1 Prosedur Penelitian.

Gambar 3.1 Prosedur Penelitian

Berdasarkan Gambar 3.1 terdapat beberapa proses yang harus dilakukan untuk menyelesaikan penelitian ini. Penelitian dimulai dengan studi literatur yang dilakukan untuk menemukan informasi kasus dari referensi-referensi terkait.

Referensi–referensi ini dapat berupa buku-buku, jurnal-jurnal atau tulisan-tulisan penelitian tentang prediksi cacat perangkat lunak dan penentuan fitur dengan berbagai seleksi fitur atau artikel-artikel yang membahas kasus yang sama dengan kasus dalam penelitian ini. Tahap selanjutnya yaitu mengidentifikasi masalah dengan menentukan pertanyaan-pertanyaan penelitian dalam penelitian. Setelah menentukan pertanyaan-pertanyaan penelitian, dilakukan proses pengumpulan data berupa data data *log* cacat perangkat lunak untuk nantinya akan ditentukan fitur-fitur mana saja yang memiliki pengaruh besar dan akan dilakukan pengecekan akurasi prediksi cacat perangkat lunak pada proses klasifikasi. Setelah mendapatkan data data *log* cacat perangkat lunak untuk testing akurasi, dilanjutkan untuk perancangan sistem untuk menentukan alur proses jalannya sebuah sistem yang akan di buat, selanjutnya adalah melakukan penghitungan akurasi sistem terhadap data data khusus yang telah terkumpul, dilanjutkan dengan analisa setiap hasil penghitungan dan pada akhir penelitian di dapatkan beberapa kesimpulan.

3.1.1 Studi Literatur

Pada sub bab ini penelitian ini dilakukan beberapa studi *literature* untuk menemukan referensi-referensi mengenai prediksi cacat perangkat lunak, dengan studi literature ini diharapkan dapat memberikan gambaran bagi pembaca mengenai penelitian yang akan dilakukan, berikut hasil dari studi literature yang di dapatkan dari beberapa referensi :

1. Beberapa penelitian tentang prediksi cacat perangkat lunak yang telah di teliti oleh beberapa peneliti terdahulu berdasarkan kode program.

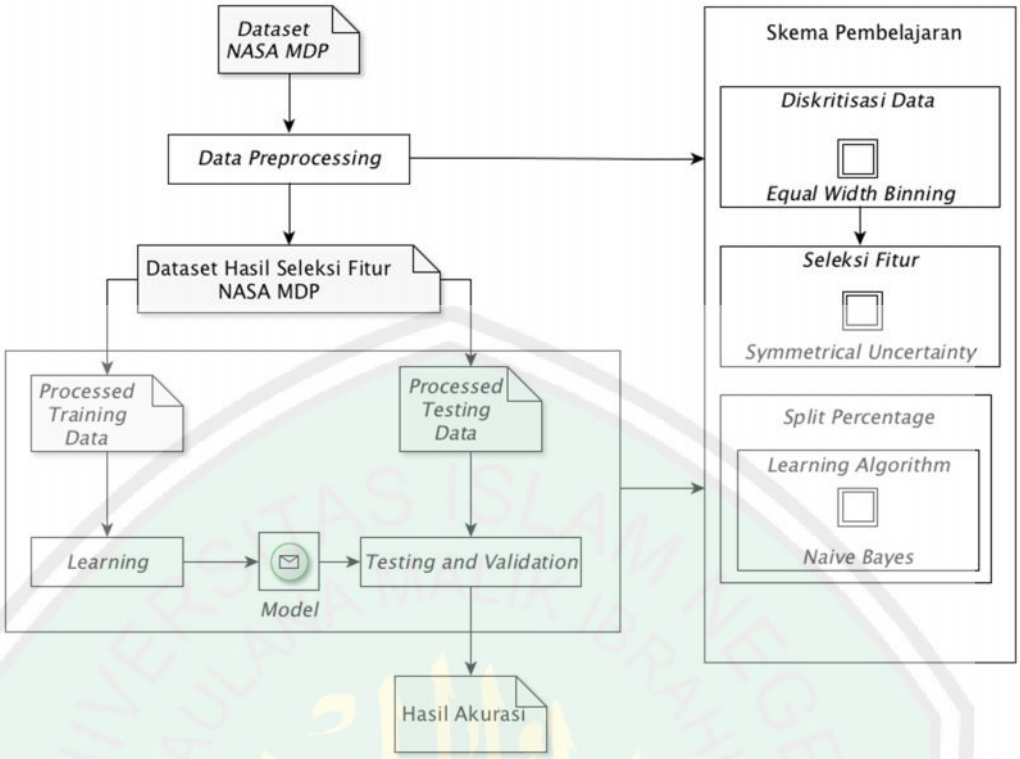
2. Metode diskritisasi dengan *Equal Width Binning*.
3. Metode seleksi fitur dengan *Symmetrical Uncertainty*.
4. Metode pencacahan data *Split Percentage.*
5. Metode klasifikasi *Naïve Bayes* dan teknik evaluasi klasifikasi.

3.1.2 Koleksi Data

Pada tahap ini dilakukan pengumpulan *dataset* yang menyimpan *log* cacat perangkat lunak. Dari *dataset* tersebut bisa terlihat setiap baris menunjukkan modul terkecil dari sebuah program tersebut diunduh dari website PROMISE (*Predictor Models in Software Engineering Repository*) melalui hhtp://promise.site.uottawa.ca/SERepository/-page.html pada hari Senin, 5 Maret 2018. Setiap baris dari tersebut menunjukkan modul terkecil dari sebuah program yaitu berupa *method* atau *function* dan setiap kolom menunjukkan nilai metric, berikut 5 yang akan digunakan pada penelitian ini yaitu CM1, JM1, KC1, KC2 dan PC1.

3.2 Perancangan Sistem

Pada subbabpenelitian ini dipaparkan tentang perancangan sistem yang diusulkan, untuk lebih jelasnya terkait alur sistem bisa dilihat pada gambar 3.2.



Gambar 3.2 Desain Sistem Yang Diusulkan

Proses seleksi fitur menggunakan algoritma *Symmetrical Uncertainty*, yang mana tahapan ini termasuk dalam tahapan *preprocessing* data seperti pada gambar 3.2 diatas, sebelum proses seleksi fitur dilakukan, *dataset* NASA MDP didiskritisasi terlebih dahulu dengan *algoritma Equal Width Binning,* sehingga dataset yang awalnya berisi data kontinu menjadi data diskrit, kemudian dilakukan proses seleksi fitur dengan algoritma *Symmetrical Uncertainty* sehingga didapatkan nilai *Symmetrical Uncertainty* untuk masing-masing fitur dan dirangking menurut nilai *Symmetrical Uncertainty*, setelah dirangking tahapan berikutnya adalah tahap pengujian, pada tahapan pengujian terdapat 3 proses, yaitu proses pencacahan data dengan *algoritma Split Percentage*, proses klasifikasi dengan *algoritma Naïve Bayes* dan proses penghitungan akurasi dengan *Confusion Matrix.*

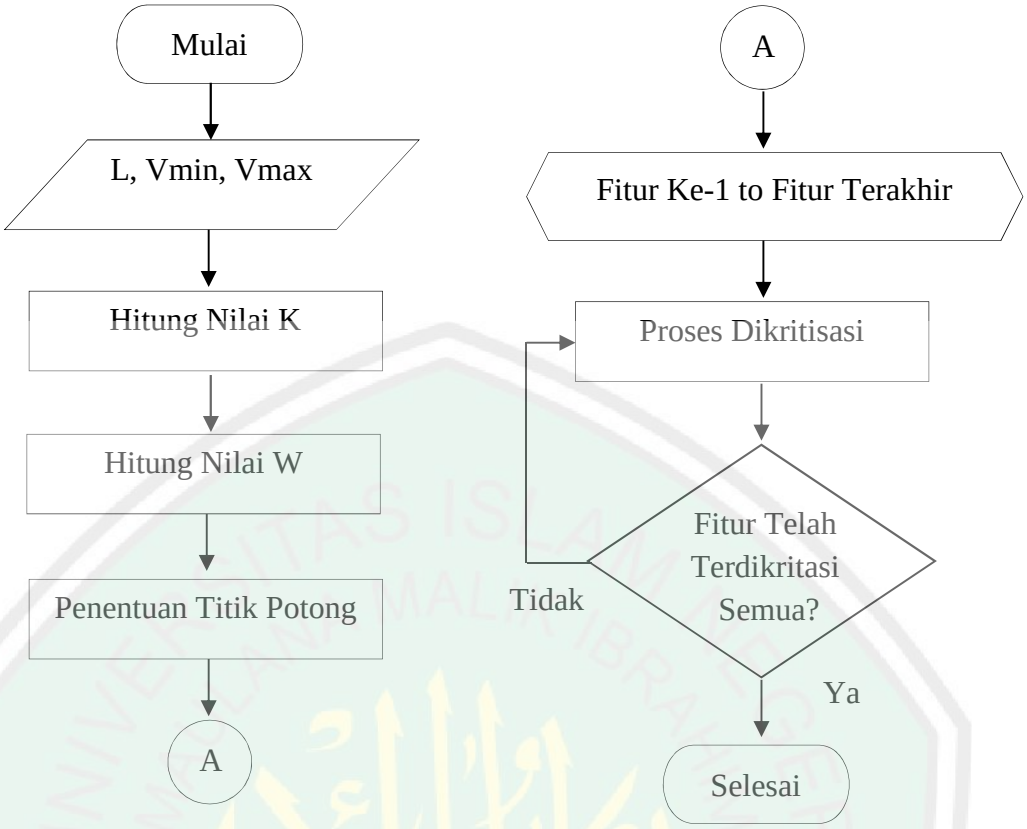
3.2.1. Proses Diskritisasi dengan *Equal Width Binning*

Umumnya, *Naïve Bayes* mudah dihitung untuk fitur bertipe kategoris, namun pada terdapat beberapa fitur bertipe kontinu (numerik) sehingga ada perlakuan khusus sebelum dimasukkan dalam *Naïve Bayes*. Untuk mengubah fitur bertipe numerik menjadi kategoris dapat dilakukan dengan cara diskritisasi pada setiap fitur kontinu tersebut dengan nilai interval diskrit (Essra dkk., 2016 : 11), selain itu menurut Singh dan Verma (2009 : 839) bahwa klasifikasi *Naïve Bayes* memiliki akurasi yang lebih baik jika menggunakan data diskrit.

Seperti yang telah dijelaskan pada bab 2 tentang diskritisasi dengan metode *equal width binning*, adapaun tahapan melakukan diskritisasi data meliputi;

a. Tentukan nilai $K = max\{1,2\ log\ L\}$ jika K bernilai 1 maka diskritisasi data menjadi *all*.
b. Hitung W = (Vmax – Vmin) / K.
c. Tentukan titik potong dengan perhitungan;
   Vmin+W, Vmin+2W,…, Vmin+(K-1)W
d. Tentukan nilai dari termasuk dalam rentang yang sesuai.

Berdasarkan alur tahap perhitungan yang telah dijelaskan diatas, berikut pada gambar 3.3 merupakan *flowchart* algoritma *equal width binning.*



Gambar 3.3 Flowchart Equal Width Binning

Berikut Tabel 3.1merupakan potongan dari *dataset* CM1 yang akan diproses diskritisasi dengan algoritma *equal width binning.*

Tabel 3.1 Potongan Dataset CMI

No.	loc	v(g)	ev(g)	iv(g)	…	*defects*
1.	1,1	1,4	1,4	1,4	…	***false***
2.	1	1	1	1	…	***true***
3.	24	5	1	3	…	***false***
…	…	…	…	…	…	…
498.	28	6	5	5	…	***true***

Berikut contoh perhitungan secara manual dari proses diskritisasi dengan algoritma *Equal Width Binning.* Pada contoh perhitungan manual penelitian ini menggunakan *dataset* CM1.

a. Menghitung nilai K.

Nilai K didapat dari rumus :

$$K = max\{1{,}2\ log\ L\}$$

Keterangan :

L = (Jumlah Nilai berbeda) untuk fitur ke-1=102, ke-2=34, ke-3=18, .., ke-n

jadi, K =max(1,2 *log* 102)→ K = max(2.41) → K = **3**

b. Menghitung nilai W.

Nilai W didapat dari rumus :

$$W = (Vmax - Vmin) / K$$

Keterangan :

Vmin = 1, ke-2 = 1, ke-3 = 1, …, ke-N

Vmax = 423, ke-2 = 96, ke-3 = 30, …, ke-N

jadi, W = (423 – 1) / 3 → W = **140.66666666666666**

c. Tentukan titik potong.

Titik potong di dapata dari rumus :

$$Vmin+W,\ Vmin+2W,\ldots,\ Vmin+(K-1)W$$

Hasilnya :

Vmin + W → 1 + 140.66666666666666 = **141.66667**

Vmin + 2W → 1 + (2 x 140.66666666666666) = **282.33334**

Karena K – 1 = 2 titik potong sudah terhitung semua.

Interval (-inf – **141.66667**)

Interval (**141.66667 - 282.33334**)

Interval (**282.33334** – inf)

d. Tentukan nilai dari termasuk dalam rentang yang sesuai.

Pada table 3.2 penulis melakukan implementasi dari hasil perhitungan diskritisasi pada *dataset* CM1 fitur ke-1 (loc).

Tabel 3.2 Hasil Diskritisasi Dataset CM1 Fitur Ke-1 (Loc)

No.	loc	v(g)	ev(g)	iv(g)	…	*defects*
1.	(-inf-141.66667)	1,4	1,4	1,4	…	***false***
2.	(-inf-141.66667)	1	1	1	…	***true***
3.	(-inf-141.66667)	5	1	3	…	***false***
…	…	…	…	…	…	**…**
498.	(-inf-141.66667)	6	5	5	…	***true***

Ulangi proses diskritisasi untuk semua fitur, pada tabel 3.3 penulis melakukan perhitungan semua fitur pada *dataset* CM1.

Tabel 3.3 Hasil Diskritisasi Semua Fitur Dataset CM1

No.	loc	v(g)	ev(g)	iv(g)	…	defects
1.	(-inf-141.66667)	(-inf-48.5)	(-inf-15.5)	(-inf-32.0)	…	***false***
2.	(-inf-141.66667)	(-inf-48.5)	(-inf-15.5)	(-inf-32.0)	…	***true***
3.	(-inf-141.66667)	(-inf-48.5)	(-inf-15.5)	(-inf-32.0)	…	***false***
…	…	…	…	…	…	**…**
498.	(-inf-141.66667)	(-inf-48.5)	(-inf-15.5)	(-inf-32.0)	…	***true***

3.2.2. Proses Seleksi Fitur dengan *Symmetrical Uncertainty*

Dari sekian banyak fitur, tidak semua fitur yang memberikan pengaruh besar terhadap hasil prediksi, pada tahap ini akan dilakukan proses seleksi fitur yaitu pemilihan fitur-fitur mana saja yang mempunyai nilai pengaruh yang besar dan nantinya akan di rangking dari yang memiliki nilai *Symmetrical Uncertainty*

tertinggi sampai terendah, berikut merupakan tahapan dari perhitungan nilai *Symmetrical Uncertainty*.

a. Menghitung nilai *Entropy* setiap fitur dengan rumus :

$$Entropy(S) = \sum_{i}^{c} -p_i \log_2 p_i$$

b. Menghitung nilai *Information Gain* setiap fitur dengan rumus :

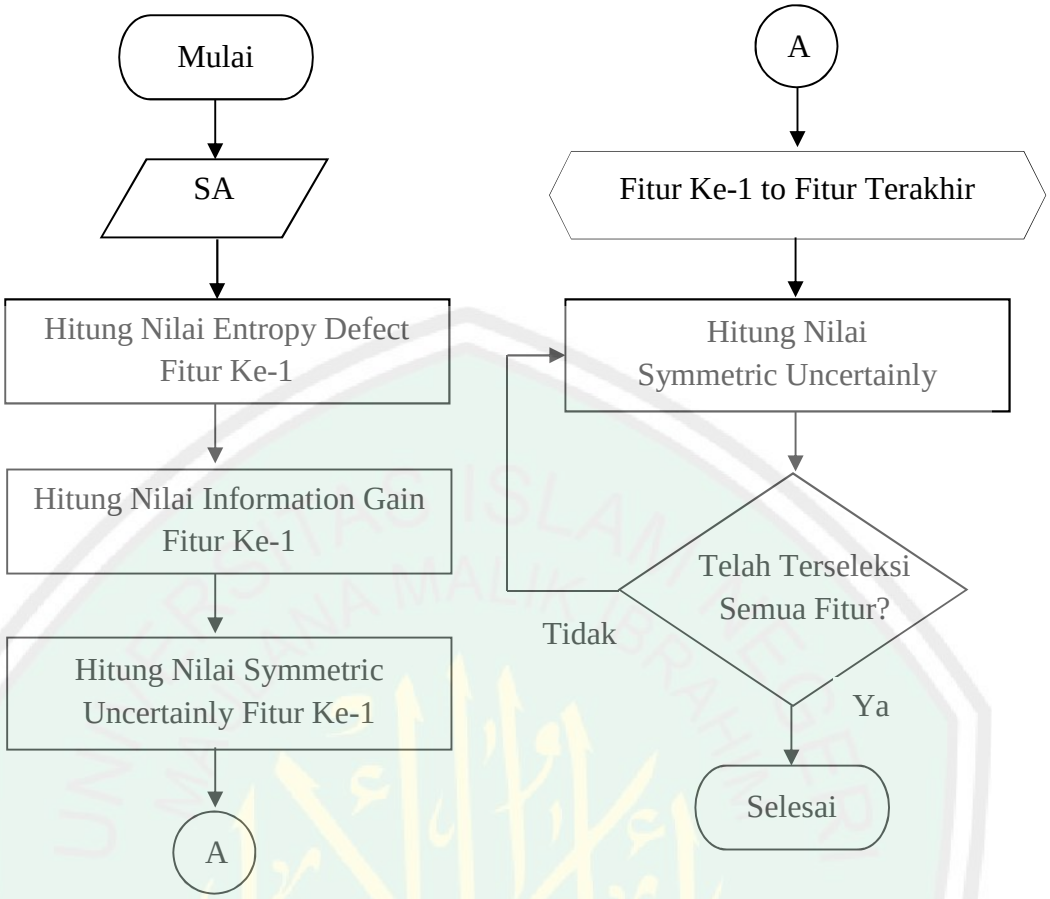
$$Gain(S,A) = Entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|} entropy\ (S_v)$$

c. Menghitung nilai *Symmetrical Uncertainty*setiap fitur dengan rumus :

$$SU(S,A) = 2 * \frac{Gain(S,A)}{H(S) + H(A)}$$

d. Perangkingan fitur berdasarkan nilai *Symmetrical Uncertainty*.Tahap terakhir adalah melakukan perankingan berdasarkan nilai *Symmetrical Uncertainty* dan mengambil beberapa fitur yang dianggap memiliki pengaruh terhadap proses prediksi, jumlah fitur yang dipilih sesuai dengan keinginan *user*.

Berdasarkan alur tahapan yang telah dijelaskan jika dibentuk dalam sebuah flowchart tampil seperti berikut.



Gambar 3.4 Flowchart Algoritma Symmetrical Uncertainty

Berikut merupakan contoh perhitungan dari algoritma *Symmetrical Uncertainty* berdasarkan dari data yang telah didiskritisasi pada proses sebelumnya.

a. Menghitung nilai *entropy*.

***Entropy* (Defect)**

$$= \left(\left(\frac{-449}{498}\right) log_2 \frac{449}{498}\right) + \left(\left(\frac{-49}{498}\right) log_2 \frac{49}{498}\right) = \mathbf{0.46388254}$$

***Entropy* Fitur loc**

Entropy((-inf – 141.66667))

$$= \left(\left(\frac{-440}{485}\right) \log_2 \frac{440}{485}\right) + \left(\left(\frac{-45}{485}\right) \log_2 \frac{45}{485}\right) = \mathbf{0.4456932}$$

Entropy((141.66667 - 282.33334))

$$= \left(\left(\frac{-7}{10}\right)\log_2\frac{7}{10}\right) + \left(\left(\frac{-3}{10}\right)\log_2\frac{3}{10}\right) = \mathbf{0.8812909}$$

Entropy((282.33334 – inf))

$$= \left(\left(\frac{-2}{3}\right)\log_2\frac{2}{3}\right) + \left(\left(\frac{-1}{3}\right)\log_2\frac{1}{3}\right) = \mathbf{0.91829586}$$

Ulangi langkah 1 untuk fitur selanjutnya hingga mendapat nilai *entropy* seluruh fitur.

b. Menghitung nilai *Information Gain.*

***Information Gain*Fitur loc**

$$Gain(\text{Defect,loc}) = 0.46388254 - \left(\left(\frac{485}{498}\right)0.4456932 + \left(\frac{10}{498}\right)0.8812909 + \left(\frac{3}{498}\right)0.91829586\right)$$

$$= \mathbf{0.006595403}$$

c. Menghitung nilai *Symmetrical Uncertainty.*

$$SU = 2 * \frac{0.006595403}{0.46388254 + (0.0371648244 + 0.1132143341 + 0.0444279484)}$$

$$= \mathbf{0.0200258286}$$

Ulangi Perhitungan *Symmetrical Uncertainty*fitur selanjutnya hingga mendapat nilai *Symmetrical Uncertainty*seluruh fitur.

d. Perangkingan

Setelah didapat nilai *Symmetrical Uncertainty* dari setiap fitur makan langkah 4 atau langkah terakhir yang dilakukan adalah merangking berdasar kan nilai *Symmetrical Uncertainty*, semakin besar nilai *Symmetrical Uncertainty* maka semakin besar pulan pengaruh fitur

tersebut pada proses prediksi cacat perangkat lunak. Pada Tabel 3.4 yang menunjukkan hasil perangkingan fitur sesuai dengan nilai *Symmetrical Uncertainty* pada *dataset* CM1.

Tabel 3.4 Hasil Perangkingan Sesuai Nilai Symmetrical Uncertainty (SU)

No.	Nilai SU	Fitur
1.	0.051976	T
2.	0.051976	E
3.	0.04639	uniq_Opnd
4.	0.040624	I
5.	0.040076	uniq_Op
6.	0.036229	V
7.	0.028595	lOComment
8.	0.027274	B
9.	0.025103	D
10.	0.022167	N
11.	0.020026	Loc
12.	0.020026	total_Opnd
13.	0.017623	total_Op
14.	0.01304	ev(g)
15.	0.011744	lOBlank
16.	0.006972	v(g)
17.	0.006972	iv(g)
18.	0.006972	branchCount
19.	0.000716	lOCode
20.	0.000337	L
21.	0	locCodeAndComment

Menurut (Wang *dkk*., 2012 : 124-132) Jumlah seleksi fitur tergantung yang diinginkan oleh *user*, namun pada penelitian ini penulis menggunakan asumsi seperti pada penelitian terdahulu yang menjelaskan bahwa jumlah seleksi fitur yang direkomendasikan adalah $log_2 n$ dengan nilai n adalah jumlah seluruh fitur.

Pada perhitungan manual ini penulis mengambil jumlah fitur sebanyak 5 (warna hijau pada table 4.4) fitur karena $log_2 21 = 4.4$, hasil seleksi fitur dari keseluruhan fiturpada *dataset* CM1 hasilnya ditunjukkan pada tabel 3.5.

Tabel 3.5 Hasil Seleksi 5 Fitur Pada Dataset CM1

No.	Nilai SU	Fitur	Keterangan
1.	0.051976	T	Halstead "*Time to write program*"
2.	0.051976	E	Halstead " *Effort to write program* "
3.	0.04639	uniq_Opnd	*unique operands*
4.	0.040624	I	Halstead "*Intelligence*"
5.	0.040076	uniq_Op	*unique operators*

3.2.3. Proses Pengujian

Pada tahap ini 5 fitur yang telah terseleksi oleh metode *Symmetrical Uncertainty* akan di uji menggunakan sebuah *classifier*. Berdasarkan studi literature yang dilakukan pada penelitian ini, pada tahap pengujian ini terdapat beberapa proses yaitu :

1. *Split Percentage.*
2. *Naïve Bayes classifier.*
3. *Confusion Matrix.*

3.2.3.1. *Split Percentage*

Pada tahap pertama data akan di pecah menjadi data *testing* dan data *training* dengan teknik split percentage yang telah dijelaskan pada bab 2. Pada penelitian ini dilakukan dengan *split percentage* 60 persen, sehingga dari *dataset* CM1 yang memiliki jumlah data 498 baris data dibagi menjadi 2 bagian *testing*

dan *training*. Pada kasus ini terdapat 5 fitur dengan 498 baris data. Dari 498 data tersebut dibagi menjadi 2 bagian, dengan perbandingan 60 persen banding 40 persen dimana 60% * 498 = 299, dikarenakan posisi pembagian label pada *dataset* tidak rata, maka dari itu penulis mengambil maka baris data 300 - 498 dijadikan data *training* dan sisanya 199 data yaitu baris 1 - 199 data menjadi data *testing*.

3.2.3.2. *Naïve Bayes Classifier*

Pada tahap ini data *training* pada baris data 300 – 498 dicari nilai probabilitasnya untuk setiap fitur. Dikarenakan data yang digunakan merupakan data numerik kontinu maka perhitungan sesuai dengan distribusi gaussian dengan rumus :

$$P(X_i|Y_k) = \frac{1}{\sigma_{Y_k}\sqrt{2\pi}} e^{-\frac{(x-\mu_{Y_k})^2}{2\sigma_{Y_k}^2}}$$

Langkah pertama yang dilakukan adalah mencari nilai rata-rata (*mean*) dan standar deviasi sebagai berikut :

1. Mencari nilai probabilitas rata-rata fitur T berdasarkan kelas *defect* "*TRUE*".

   Rata-Rata = $\frac{991{,}46 + 439{,}7 + \ldots + 504{,}35}{48}$ = 4228,398

   Standar Deviasi = $\sqrt{54987108{,}05}$ = 7415,329261

2. Mencari nilai probabilitas rata-rata fitur T berdasarkan kelas *defect* "*FALSE*".

   Rata-Rata = $\frac{6{,}5 + 2{,}73 + \ldots + 131{,}62}{251}$ = 1807,451155

   Standar Deviasi = $\sqrt{65755168{,}98}$ =8108,956097

3. Mencari nilai probabilitas rata-rata fitur E berdasarkan kelas *defect* "*TRUE*".

   Rata-Rata $= \frac{117 + 49{,}13 + \ldots + 2369{,}07}{48} = 32534{,}11398$

   Standar Deviasi = $\sqrt{17815825255}$ =133475,9351

4. Mencari nilai probabilitas rata-rata fitur E berdasarkan kelas *defect* "*FALSE*".

   Rata-Rata $= \frac{17846{,}19 + 7914{,}68 + \ldots + 9078{,}38}{251} = 76111{,}15354$

   Standar Deviasi = $\sqrt{17815825255}$ =133475,9351

5. Mencari nilai probabilitas rata-rata fitur uniq_Opnd berdasarkan kelas *defect* "*TRUE*".

   Rata-Rata $= \frac{32 + 33 + \ldots + 23}{48} = 53{,}35416667$

   Standar Deviasi = $\sqrt{3026{,}191046} = 55{,}01082663$

6. Mencari nilai probabilitas rata-rata fitur uniq_Opnd berdasarkan kelas *defect* "*FALSE*".

   Rata-Rata $= \frac{4 + 2 + \ldots + 12}{251} = 22{,}27888446$

   Standar Deviasi = $\sqrt{808{,}3939124} = 28{,}43226886$

7. Mencari nilai probabilitas rata-rata fitur I berdasarkan kelas *defect* "*TRUE*".

   Rata-Rata = $\frac{38{,}55 + 52{,}03 + \ldots + 35{,}08}{48} = 69{,}816875$

   Standar Deviasi = $\sqrt{3541{,}011435} = 59{,}50639827$

8. Mencari nilai probabilitas rata-rata fitur I berdasarkan kelas *defect* "*FALSE*".

Rata-Rata $= \frac{13 + 7{,}86 + \ldots + 21{,}83}{251} = 33{,}90545817$

Standar Deviasi $= \sqrt{956{,}4819961} = 30{,}92704312$

9. Mencari nilai probabilitas rata-rata fitur uniq_Op berdasarkan kelas *defect* "*TRUE*".

   Rata-Rata $= \frac{27 + 22 + \ldots + 20}{48} = 22{,}89583333$

   Standar Deviasi $= \sqrt{103{,}0314716} = 10{,}15044194$

10. Mencari nilai probabilitas rata-rata fitur uniq_Op berdasarkan kelas *defect* "*FALSE*".

    Rata-Rata $= \frac{4 + 5 + \ldots + 10}{251} = 14{,}84860558$

    Standar Deviasi $= \sqrt{87{,}96898805} = 9{,}379178431$

Setelah mendapatkan semua nilai rata-rata dan standar deviasi setiap fitur, langkah selanjutnya adalah melakukan pengujian dengan menggunakan data *testing*. Data *testing* didapat dari baris 1 - 199, untuk data *testing* penulis mengambil contoh data baris ke-1. Data baris ke-1 di tunjukkan pada tabel berikut :

Tabel 3.6 Baris Ke-48 Set Pertama Dataset CM1

t	e	uniq_Opnd	i	uniq_Op	defects
1,3	1,3	1,2	1,3	1,2	?

1. Menghitung nilai probabilitas menggunakan rumus distribusi gaussian :

$$P(t = 1{,}3|TRUE) = \frac{1}{7415{,}329261\sqrt{2\pi}} e^{\frac{-(1{,}3 - 4228{,}397708)^2}{(2 \times 7415{,}329261)^2}} = 4{,}57432$$

$$P(e = 1{,}3|TRUE) = \frac{1}{133475{,}9351\sqrt{2\pi}} e^{\frac{-(1{,}3 - 76111{,}15354)^2}{(2\times 133475{,}9351)^2}} = 2{,}54105$$

$$P(\text{uniq\_Opnd} = 1{,}2|TRUE) = \frac{1}{55{,}01082663\sqrt{2\pi}} e^{\frac{-(1{,}2 - 53{,}35416667)^2}{(2\times 55{,}01082663)^2}} = 0{,}0046279$$

$$P(\text{i} = 1{,}3|TRUE) = \frac{1}{59{,}50639827\sqrt{2\pi}} e^{\frac{-(1{,}3 - 69{,}816875)^2}{(2\times 59{,}50639827)^2}} = 0{,}003455966$$

$$P(\text{uniq\_Op} = 1{,}2|TRUE) = \frac{1}{10{,}15044194\sqrt{2\pi}} e^{\frac{-(1{,}3 - 22{,}895833338)^2}{(2\times 10{,}15044194)^2}} = 0{,}00400384\sout{4}$$

$$P(t = 1{,}3|FALSE) = \frac{1}{8108{,}956097\sqrt{2\pi}} e^{\frac{-(1{,}3 - 1807{,}451155)^2}{(2\times 8108{,}956097)^2}} = 4{,}80045$$

$$P(e = 1{,}3|FALSE) = \frac{1}{145961{,}2084\sqrt{2\pi}} e^{\frac{-(1{,}3 - 32534{,}11398)^2}{(2\times 145961{,}2084)^2}} = 2{,}66683$$

$$P(\text{uniq\_Opnd} = 1{,}2|FALSE) = \frac{1}{28{,}43226886\sqrt{2\pi}} e^{\frac{-(1{,}3 - 22{,}27888446)^2}{(2\times 28{,}43226886)^2}} = 0{,}010662463$$

$$P(\text{i} = 1{,}3|FALSE) = \frac{1}{30{,}92704312\sqrt{2\pi}} e^{\frac{-(1{,}3 - 33{,}90545817)^2}{(2\times 30{,}92704312)^2}} = 0{,}007401603$$

$$P(\text{uniq\_Op} = 1{,}2|FALSE) = \frac{1}{9{,}379178431\sqrt{2\pi}} e^{\frac{-(1{,}3 - 14{,}84860558)^2}{(2\times 9{,}379178431)^2}} = 0{,}014757784$$

2. Menghitung probabilitas akhir setiap kelas.

$$\text{P(X} \mid TRUE) = P(I = 1{,}3|TRUE) \text{ X } P(E = 1{,}3|TRUE) \text{ X } P(T = 1{,}3|TRUE)$$

$$\text{X } P(\text{uniq\_Op} = 1{,}2|TRUE) \text{ X } P(\text{uniq\_Opnd} = 1{,}2|TRUE)$$

$$= 1{,}19495$$

$$\text{P(X} \mid FALSE) = P(I = 1{,}3|FALSE) \text{ X } P(E = 1{,}3|FALSE) \text{ X}$$

$$P(T = 1{,}3|FALSE) \text{ X } P(\text{uniq\_Op} = 1{,}2|FALSE) \text{ X}$$

$$P(\text{uniq\_Opnd} = 1{,}2|FALSE)$$

$$= 1{,}25165$$

3. Menentukan kelas berdasarkan nilai probabilitas tertinggi.

P(*DEFECT* = *TRUE*) = 48/299= 0,160535117

P(*DEFECT* = *FALSE*) = 251/299= 0,839464883

P(*X* |*TRUE*) = 1,19495 X 0,160535117 = 1,19495

P(*X* |*FALSE*) = 1,25165 X 0,839464883 = 1,25165

Berdasarkan hasil perhitungan yang telah dilakukan, data *testing* baris ke-1 termasuk dalam DEFECT = *FALSE* ,karena nilai Nilai (DEFECT = FALSE) lebih besar dari (DEFECT = TRUE). Proses ini dilakukan sampai baris data ke 199.

3.2.3.3. *Confusion Matrix*

Dari proses perhitungan diatas hasil dari *classifier* akan di bandingkan dengan hasil *dataset* asli dan seperti yang telah dijelaskan pada bab 2 bahwa proses evaluasi klasifikasi dilakukan dengan melakukan perhitungan berdasarkan *confusion matrix*. Pada tahap ini melakukan perhitungan sesuai dengan *confusion matrix* untuk mendapatkan nilai akurasi dari proses klasifikasi pada dengan seleksi fitur *Symmetrical Uncertainty*. Berikut salah satu contoh perhitungan akurasi dari proses prediksi cacat perangkat lunak pada *dataset* CM1.

Tabel 3.7 Confusion Matrix Klasifikasi Dataset CM1 Dengan Seleksi Fitur

		Kelas hasil prediksi		
		Ya	Tidak	Jumlah
Kelas aktual	Ya	19	10	*P*
	Tidak	10	160	*N*
	Jumlah	29	170	*P+N*

Sesuai dengan rumus perhitungan *confusion matrix* maka perhitungan akurasi sebagai berikut:

Akurasi dengan seleksi fitur SU= $\frac{160 + 19}{160 + 19 + 10 + 10} = 0.8995$

BAB IV

UJI COBA DAN PEMBAHASAN

Pada bab ini akan dibahas terkait uji coba dari rancangan atau desain sistem yang telah dibuat pada bab sebelumnya. Penjelasan pertama pada bab ini adalah tentang uji coba penelitian yang berisi tentang lingkungan uji coba, data yang digunakan, tampilan sistem yang berhasil dibuat, hasil uji coba seleksi fitur dan hasil uji coba klasifikasi.

4.1 Uji Coba

Sebelum menjelaskan proses uji coba terlebih dahulu akan dijelasan hal-hal yang berkaitan terhadap proses uji coba yaitu lingkungan uji coba dan juga data yang digunakan untuk uji coba.

4.1.1 Lingkungan Uji Coba

Lingkungan uji coba menjelaskan tentang spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam penelitian ini, adapun spesifikasi perangkat keras yang digunakan adalah sebagai berikut;

a. *Processor 1,6 GHz Intel Core i5.*
b. *Memory 8 GB 1600 MHz DDR3.*
c. *Solid State Drive 128 GB*

Sedangkan spesifikasi perangkat lunak yang digunakan adalah sebagai berikut;

a. Sistem operasi *macOS High Sierra 10.13.3 (17D47).*
b. *Weka* versi 3.7.2
c. *Netbeans 8.2* dan *Ms.excel 2010*

4.1.2 Data Uji coba

Data yang digunakan pada penelitian ini berupa *dataset* yang berjumlah 5 *dataset* yaitu CM1, JM1, KC1, KC2 dan PC1. Berikut adalah 5 *dataset* yang digunakan pada penelitian ini;

Tabel 4.1 *Dataset* CM1

No	loc	v(g)	ev(g)	iv(g)	….	defect
1	1.1	1.4	1.4	1.4	…	False
2	1	1	1	1	…	True
3	24	5	1	3	…	False
…	…	…	…	…	…	…
498	28	6	5	5	…	True

Tabel 4.2 *Dataset* JM1

No	loc	v(g)	ev(g)	iv(g)	….	defect
1	1.1	1.4	1.4	1.4	…	False
2	1	1	1	1	…	True
3	72	7	1	6	…	True
…	…	…	…	…	…	…
10885	19	3	1	1	…	False

Tabel 4.3 *Dataset* KC1

No	loc	v(g)	ev(g)	iv(g)	….	defect
1	1.1	1.4	1.4	1.4	…	False
2	1	1	1	1	…	True
3	83	11	1	11	…	True
…	…	…	…	…	…	…
2109	11	2	1	2	…	False

Tabel 4.4 *Dataset* KC2

No	loc	v(g)	ev(g)	iv(g)	….	defect
1	1.1	1.4	1.4	1.4	…	no
2	1	1	1	1	…	yes
3	415	59	50	51	…	yes
…	…	…	…	…	…	…
522	3	1	1	1	…	yes

Tabel 4.5 *Dataset* PC1

No	loc	v(g)	ev(g)	iv(g)	….	defect
1	1.1	1.4	1.4	1.4	…	False
2	1	1	1	1	…	True
3	91	9	3	2	…	True
…	…	…	…	…	…	…
1109	26	18	13	6	…	False

Berdasarkan data setiap *dataset* maka Tabel 4.6 dan tabel 4.7 menunjukkan informasi 5 *dataset* yang digunakan pada penelitian ini. Informasi yang ditunjukkan antara lain jumlah total modul setiap *dataset*, jumlah modul yang cacat dan jumlah modul yang dinyatakan tidak cacat serta informasi tentang fitur-fitur yang ada pada setiap *dataset*.

Tabel 4.6 *Dataset* Penelitian

Dataset	Jumlah total modul	Jumlah modul cacat	Jumlah modul tidak cacat	Jumlah fitur
CM1	498	449	49	21
JM1	10885	8779	2106	21
KC1	2109	1783	326	21
KC2	522	415	107	21
PC1	1109	1032	77	21

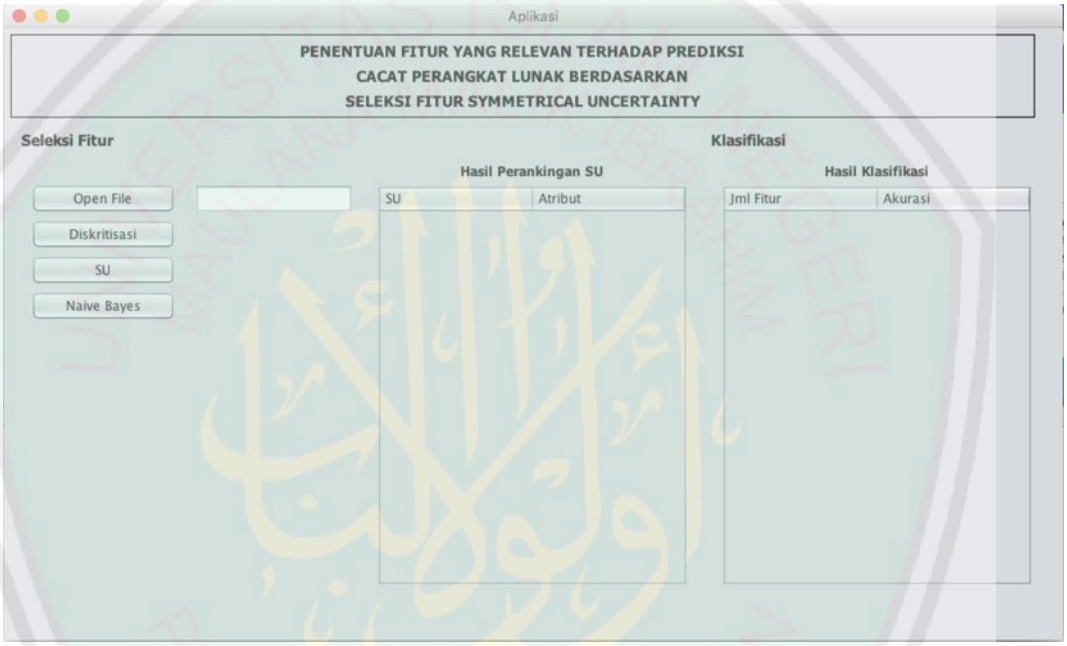
Tabel 4.7 Fitur setiap *dataset*

No	Fitur	*Dataset*				
		CM1	**JM1**	**KC1**	**KC2**	**PC1**
1	loc	√	√	√	√	√
2	v(g)	√	√	√	√	√
3	ev(g)	√	√	√	√	√
4	iv(g)	√	√	√	√	√
5	uniq_Op	√	√	√	√	√
6	uniq_Opnd	√	√	√	√	√
7	total_Op	√	√	√	√	√
8	total_Opnd	√	√	√	√	√
9	n	√	√	√	√	√
10	v	√	√	√	√	√
11	l	√	√	√	√	√
12	d	√	√	√	√	√
13	i	√	√	√	√	√
14	e	√	√	√	√	√
15	b	√	√	√	√	√
16	t	√	√	√	√	√
17	lOCode	√	√	√	√	√
18	lOComment	√	√	√	√	√
19	lOBlank	√	√	√	√	√
20	locCodeAndComment	√	√	√	√	√
21	branchCount	√	√	√	√	√

Berdasarkan tabel 4.7 dapat diketahui bahwa semua *dataset* yang digunakan menggunakan fitur yang sama dalam melakukan prediksi cacat perangkat lunak dengan jumlah 21 fitur yaitu fitur loc, v(g), ev(g), iv(g), uniq_Op, uniq_Opnd, total_Op, total_Opnd, n, v, l, d, I, e, b, t, IOCode, IOComment, IOBlank, lOCodeAndComment dan branchCount.

4.1.3. Tampilan Sistem

Berdasarkan dari rencana yang telah dijelaskan pada desain sistem yang proses diskritisasi *Equal Width Binning*, proses seleksi fitur *Symmetrical Uncertainty*, proses testing menggunakan klasifikasi dengan *Naïve Bayes* dengan metode pencacahan data *Split Percentage* dan proses perhitungan akurasi. Proses-proses tersebut diimplementasikan dalam bahasa pemrograman java yang dibuat dengan tampilan ditunjukkan pada gambar 4.1.



Gambar 4.1 Tampilan Sistem

Berdasarkan gambar 4.1 dapat dilihat bahwa terdapat beberapa *button* yang berguna untuk menjalankan sebuah proses sesuai dengan algoritma yang telah dijelaskan pada bab sebelumnya. Alur untuk menjalankan sistem tersebut dimulai dari klik *button* "open file" untuk mengambil file yang akan digunakan ,kemudian klik *button* "diskritisasi" untuk menjalankan proses diskritisasi tunggu hingga muncul pemberitahuan proses selesai. Selanjutnya klik *button* "*Symmetrical Uncertainty*" untuk menjalankan proses perangkingan sesuai nilai *Symmetrical*

Uncertainty, tunggu hingga muncul proses selesai. Langkah selanjutnya adalah proses klasifikasi dengan cara klik *button* "Naïve Bayes", hasil dari proses klasifikasi berupa akurasi klasifikasi.

4.1.4 Hasil Pengujian

Uji coba pada penelitian ini dilakukan terhadap lima*dataset* yaitu *dataset* CM1, JM1, KC1, KC2 dan PC1. Tahap pertama dalam pengujian ini adalah menentukan urutan fitur yang relevan dari setiap *dataset* berdasarkan seleksi fitur *Gain Ratio* dan tahap kedua adalah melakukan pengujian akurasi klasifikasi Naïve Bayes pada setiap *dataset* dengan seleksi fitur *Symmetrical Uncertainty*, teknik pencacahan data pada klasifikasi yang digunakan adalah *Split Percentage*.

4.1.4.1. Seleksi Fitur dengan*Symmetrical Uncertainty*

Fitur yang relevan didapat dengan melakukan seleksi fitur menggunakan metode seleksi fitur *Symmetrical Uncertainty*.Hasil dari seleksi fitur ini berupa urutan atau perangkingan fitur sesuai dengan nilai *Symmetrical Uncertainly*. Tabel 4.8 hingga tabel 4.12 menunjukkan urutan atau perankingan fitur berdasarkan nilai *Symmetrical Uncertaily* dari kelima *dataset* yang digunakan.

Tabel 4.8 Perankingan Fitur *dataset* CM1 Berdasarkan nilai *Symmetrical Uncertainty*

SU	Fitur
0.051975742	t
0.051975742	e
0.04638984	uniq_Opnd
0.040624384	i
0.040075373	uniq_Op
0.036228858	v
0.028594445	lOComment

Tabel 4.9 Perankingan Fitur *dataset* CM1 Berdasarkan nilai *Symmetrical Uncertainty* (Lanjutan)

0.027273918	b
0.025103398	d
0.02216654	n
0.020025829	total_Opnd
0.020025829	loc
0.017623467	total_Op
0.013039448	ev(g)
0.0117442105	lOBlank
0.0069715413	branchCount
0.0069715413	iv(g)
0.0069715413	v(g)
7.1619457E-4	lOCode
3.369857E-4	l
0.0	locCodeAndComment

Berdasarkan Tabel 4.8 dapat diketahui bahwa urutan fitur berdasarkan seleksi fitur *Symmetrical Uncertainty* dari keseluruhan fitur pada *dataset* CM1 adalah fitur *Time to write program* (t), *Effort to write program* (e), *Unique operands* (uniq_Opnd), *Intelligence* (i), *Unique operators* (uniq_Op), *Volume* (v), *Count of lines of comments* (IOComment) dan seterusnya hingga urutan paling terahir adalah fitur *Count of Code and Comments Lines* (locCodeAndComment)

Tabel 4.10 Perankingan Fitur *dataset* JM1 Berdasarkan nilai *Symmetrical Uncertainty*

SU	Fitur
0.16210316	total_Opnd
0.16210316	uniq_Opnd
0.16110688	total_Op
0.16110688	uniq_Op
0.024624571	i
0.013672832	l
0.006988602	d
0.0055682273	v
0.0054560886	b
0.005453203	loc
0.005016318	v(g)
0.004823109	ev(g)
0.004766534	n
0.0042555523	lOCode
0.00386886	t
0.00386886	e
0.0028005422	iv(g)
0.0019884203	branchCount
0.0017856058	lOComment
9.5930917E-4	lOBlank
6.131259E-4	locCodeAndComment

Tabel 4.10 menunjukkan urutan fitur hasil proses seleksi fitur *Symmetrical Uncertainty* pada *dataset* JM1 adalah fitur *total operand* (total_Opnd), *unique operands* (uniq_Opnd), total operator (total_Op), unique operators (uniq_Op), *Intelligence* (i) dan seterusnya hingga urutan paling akhir adalah fitur *Count of Code and Comments Lines* (locCodeAndComment).

Tabel 4.11 Perankingan Fitur *dataset* KC1 Berdasarkan nilai *Symmetrical Uncertainty*

SU	Fitur
0.09793076	d
0.07879831	uniq_Op
0.066774994	i
0.0604467	l
0.051130842	uniq_Opnd
0.04660237	loc
0.03870525	lOComment
0.037232142	v
0.03202861	b
0.0319969	total_Op
0.031011138	lOCode
0.027737021	total_Opnd
0.026975641	n
0.02155451	t
0.02155451	e
0.010434293	ev(g)
0.008201639	branchCount
0.008201639	v(g)
0.007534064	lOBlank
0.0031398684	iv(g)
7.846477E-6	locCodeAndComment

Berdasarkan Tabel 4.11 dapat diketahui bahwa urutan fitur berdasarkan seleksi fitur *Symmetrical Uncertainty* pada *dataset* KC1 adalah fitur *Difficulty* (d), *Unique* operators (uniq_Op), *Intelligence* (i), *Program length* (l), *unique operands* (uniq_Opnd) dan seterusnya hingga urutan paling akhir adalah fitur *Count of Code and Comments Lines* (locCodeAndComment).

Tabel 4.12 Perankingan Fitur *dataset* KC2 Berdasarkan nilai *Symmetrical Uncertainty*

SU	Fitur
0.08707828	d
0.06490186	i
0.064838625	lOCodeAndComment
0.053137675	uniq_Op
0.04477855	lOComment
0.0335845	total_Opnd
0.0335845	uniq_Opnd
0.0335845	lOCode
0.0335845	t
0.0335845	e
0.0335845	n
0.0335845	loc
0.022919482	lOBlank
0.022805717	total_Op
0.022805717	b
0.022805717	v
0.011679214	branchCount
0.011679214	iv(g)
0.011679214	ev(g)
0.011679214	v(g)
2.6498653E-6	l

Urutan fitur yang relevan pada *dataset* KC2 berdasarkan seleksi fitur *Symmetrical Uncertainty* ditunjukkan pada Tabel 4.12 yaitu fitur *Difficulty* (d),*Intelligence* (i), *Count of Code and Comments Lines* (lOCodeAndComment), *Unique operands* (Uniq_Opnd), *Count of lines of comments* (lOComment), *total operand* (total_Opnd) dan seterusnya hingga urutan paling akhir adalah fitur *Program length* (l).

Tabel 4.13 Perankingan Fitur *dataset* PC1 Berdasarkan nilai *Symmetrical Uncertainty*

SU	Fitur
0.063341364	lOComment
0.036421258	uniq_Opnd
0.03592117	lOCode
0.03592117	B
0.03592117	loc
0.035757195	T
0.035757195	E
0.03552261	V
0.03458709	branchCount
0.03458709	v(g)
0.0303572	I
0.023435665	iv(G)
0.023350235	total_Op
0.023282481	lOBlank
0.022954492	total_Opnd
0.020104924	N
0.01781536	uniq_Op
0.014149548	locCodeAndComment
0.009230957	ev(g)
0.006033182	L
0.0041150427	D

Pada *dataset* PC1 urutan fitur yang relevan berdasarkan seleksi fitur *Symmetrical Uncertainty*ditunjukkan pada Tabel 4.13, urutan fitur tersebut adalah fitur*Count of lines of comments* (lOComment), *Unique operands* (uniq_Opnd), *Count of Statement Lines* (IOCode), *Error estimate* (b), *Line of* code (loc), *Time to write program* (t) dan seterusnya hingga urutan terakhir adalah fitur *Difficulty* (d).

4.1.4.2. Uji coba Klasifikasi

Pada bagian ini dilakukan pengujian klasifikasi prediksi cacat perangkat lunak dengan fitur hasil seleksi fitur *Symmetrical Uncertainty* pada setiap *dataset*. Hasil dari proses pengujian ini berupa akurasi hasil klasifikasi dengan algoritma klasifikasi Naïve Bayes dengan teknik *Split Percentage*. Proses uji coba klasifikasi akan dilakukan dengan beberapa jumlah fitur yang berbeda untuk mendapatkan jumlah fitur yang paling relevan, pada penelitian ini penulis melakukan uji coba dengan jumlah fitur sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur.

Hasil dari proses uji coba klasifikasi adalah akurasi klasifikasi yang diperoleh dari perhitungan *confusion matrix*, sebagai gambaran perhitungan akurasi klasifikasi penulis melampirkan perhitungan akurasi klasifikasi *dataset* CM1 ditunjukkan pada bagian lampiran I. Perlu diketahui bahwa nilai yang ditunjukkan pada table *confusion matrix* merupakan nilai dari hasil perhitungan kecocokan antara hasil prediksi terhadap kenyataan atau data.

Hasil dari proses uji coba klasifikasi ditunjukkan mulai tabel 4.14 hingga tabel 4.18, tabel-tabel tersebut menunjukkan hasil dari proses uji coba kelima dataset yaitu dataset CM1, JM1, KC1, KC2 dan PC1.

Tabel 4.14 Hasil Uji Coba Klasifikasi *Dataset* CM1

Jumlah fitur	Akurasi klasifikasi NB(%)
1	**99.50%**
2	**99.50%**
3	93.97%
4	90.95%
5	89.95%
6	89.95%
7	90.95%
8	91.96%
9	90.95%
10	90.45%
11	90.45%
12	90.95%
13	90.45%
14	90.45%
15	90.95%
16	90.95%
17	90.95%
18	90.95%
19	90.95%
20	90.95%
21	88.94%

Berdasarkan pada Tabel 4.14 dapat diketahui bahwa hasil kasifikasi prediksi cacat perangkat lunak pada *dataset* CM1 dengan fitur hasil seleksi fitur *Symmetrical Uncertainty* didapatkan akurasi paling baik dengan jumlah fitur sebanyak 2 dengan akurasi klasifikasi *Naive Bayes* sebesar 99.50%**.**

Tabel 4.15 Hasil Uji Coba Klasifikasi *Dataset* JM1

Jumlah fitur	Akurasi klasifikasi NB(%)
1	80.29%
2	80.25%
3	80.27%
4	80.20%
5	80.02%
6	79.86%
7	79.65%
8	79.86%
9	79.86%
10	79.95%
11	80.18%
12	80.25%
13	80.29%
14	80.34%
15	80.41%
16	80.50%
17	80.41%
18	**80.62%**
19	80.55%
20	80.48%
21	80.59%

Berdasarkan pada tabel 4.15 dapat diketahui bahwa hasil kasifikasi prediksi cacat perangkat lunak pada *dataset* JM1 dengan fitur hasil seleksi fitur *Symmetrical Uncertainty* didapatkan akurasi klasifikasi *Naïve Bayes* paling baik sebesar 80.62% dengan jumlah fitur sebanyak 18.

Tabel 4.16 Hasil Uji Coba Klasifikasi *Dataset* KC1

Jumlah fitur	Akurasi klasifikasi NB(%)
1	**85.55%**
2	84.12%
3	84.60%
4	83.06%
5	83.18%
6	83.06%
7	83.29%
8	83.65%
9	83.77%
10	83.77%
11	83.53%
12	83.53%
13	83.65%
14	83.89%
15	84.00%
16	83.77%
17	83.89%
18	83.89%
19	83.77%
20	83.65%
21	83.77%

Berdasarkan pada Tabel 4.16 dapat diketahui bahwa hasil kasifikasi prediksi cacat perangkat lunak pada *dataset* KC1 dengan fitur hasil seleksi fitur *Symmetrical Uncertainty* didapatkan akurasi klasifikasi Naïve Bayes paling baik sebesar 85.55% dengan jumlah fitur sebanyak 1.

Tabel 4.17 Hasil Uji Coba Klasifikasi *Dataset* KC2

Jumlah fitur	Akurasi klasifikasi NB(%)
1	**96.65%**
2	91.39%
3	92.82%
4	91.39%
5	92.34%
6	93.78%
7	93.30%
8	93.78%
9	95.22%
10	96.17%
11	95.69%
12	96.17%
13	96.17%
14	96.17%
15	95.69%
16	96.17%
17	96.17%
18	96.17%
19	**96.65%**
20	96.17%
21	83.73%

Tabel 4.17 menunjukkan hasil kasifikasi prediksi cacat perangkat lunak *dataset* KC2 dengan fitur hasil seleksi fitur *Symmetrical Uncertainty*, akurasi klasifikasi *Naïve Bayes* paling baik sebesar 96.65% dengan jumlah fitur sebanyak 1 dan 19, namun pada percobaan ini terdapat beberapa fitur dengan hasil akurasi yang sama nantinya akan dipilih jumlah fitur yang paling kecil yaitu jumlah fitur 1 karena dengan jumlah fitur yang sedikit bisa menghasilkan akurasi yang sama besarnya.

Tabel 4.18 Hasil Uji Coba Klasifikasi *Dataset* PC1

Jumlah fitur	Akurasi klasifikasi NB(%)
1	**91.22%**
2	90.77%
3	90.09%
4	88.96%
5	88.29%
6	89.19%
7	89.41%
8	89.41%
9	89.19%
10	89.19%
11	88.29%
12	88.96%
13	88.51%
14	88.51%
15	88.29%
16	87.84%
17	87.84%
18	88.06%
19	88.51%
20	88.51%
21	87.61%

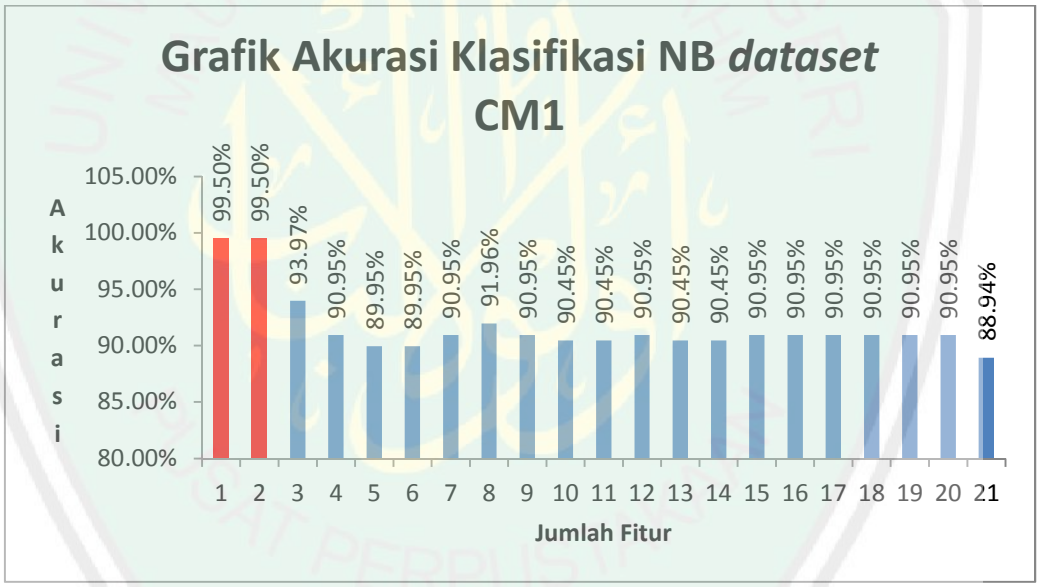
Berdasarkan pada Tabel 4.18 dapat diketahui bahwa hasil klasifikasi prediksi cacat perangkat lunak pada *dataset* PC1 dengan fitur hasil seleksi fitur *Symmetrical Uncertainty* didapatkan akurasi klasifikasi *Naïve Bayes* paling baik sebesar 91.22% dengan jumlah fitur sebanyak 1.

4.2 Pembahasan

Pada bagian ini akan dibahas mengenai hasil dari pengujian yang telah dilakukan, pembahasan dilakukan terhadap semua hasil pengujian yang meliputi pengujian pada *dataset* CM1, JM1, KC1, KC2 dan PC1.

4.2.1 Pembahasan hasil pengujian *dataset* CM1

Sesuai hasil pengujian yang telah dilakukan pada *dataset* CM1 yang ditunjukkan pada tabel 4.14 maka gambar 4.2 menunjukkan grafik nilai akurasi hasil dari pengujian *dataset* CM1 dengan jumlah fitur sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur.



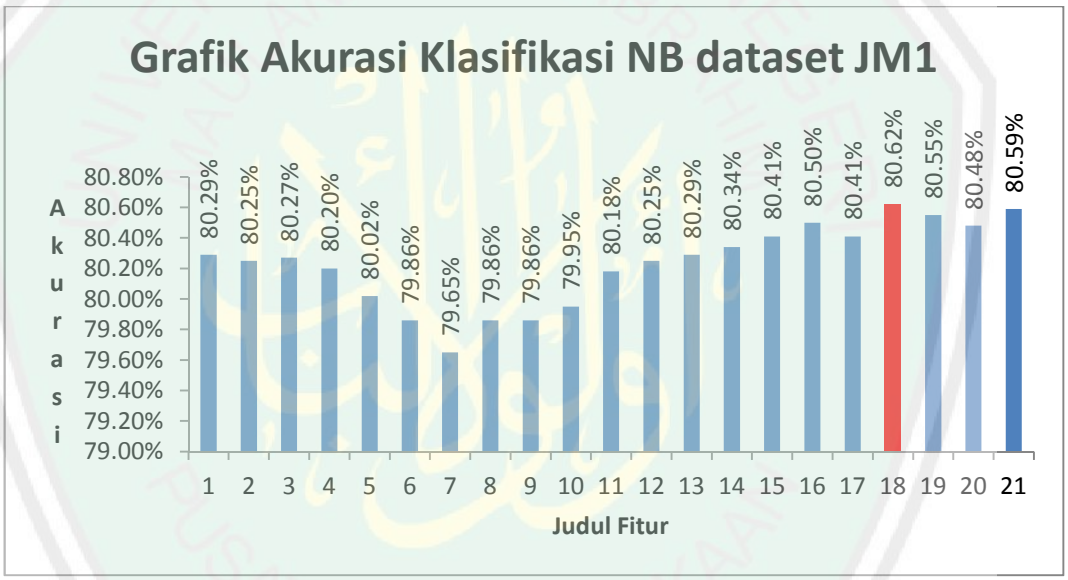
Gambar 4.2 Grafik akurasi klasifikasi NB *dataset* CM1

Pada grafik 4.2 terlihat bahwa akurasi paling baik ditunjukkan pada jumlah fitur sebanyak 1 fitur dan 2 fitur yaitu sebesar 99.50%, namun pada percobaan ini terdapat beberapa fitur dengan hasil akurasi yang sama nantinya akan dipilih jumlah fitur yang paling kecil yaitu jumlah fitur 1 karena dengan jumlah fitur yang sedikit bisa menghasilkan akurasi yang sama besarnya. sehingga fitur yang

memiliki pengaruh besar terhadap prediksi cacat perangkat lunak untuk *dataset* CM1 berdasarkan seleksi fitur *Symmetrical Uncertainty* yaitu fitur *Time to write program* (t).

4.2.2 Pembahasan hasil pengujian *dataset* JM1

Hasil pengujian yang telah dilakukan pada *dataset* JM1 yang ditunjukkan pada Tabel 4.15 maka gambar 4.3 menunjukkan grafik nilai akurasi hasil dari pengujian *dataset* JM1 dengan jumlah fitur sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur.



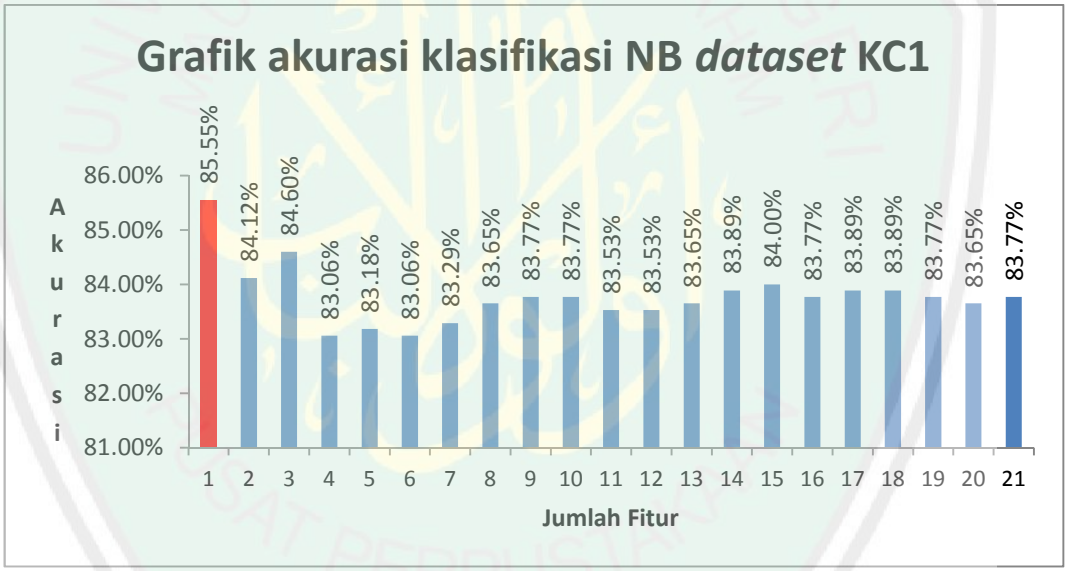
Gambar 4.3 Grafik akurasi klasifikasi NB *dataset* JM1

Pada grafik 4.3 terlihat bahwa akurasi paling baik ditunjukkan pada jumlah fitur sebanyak 18 fitur yaitu sebesar 80.62%, sehingga fitur yang memiliki pengaruh besar terhadap prediksi cacat perangkat lunak untuk *dataset* JM1 berdasarkan seleksi fitur *SymmetricalUncertainty* berjumlah 18 fitur yaitu fitur *total operand* (total_Opnd), *unique operands* (uniq_Opnd) , *total operator* (total_Op), *unique operators* (uniq_Op), *Intelligence* (i), *Programlength* (l),

Difficulty (d), *Volume* (v), *Error estimate* (b), *line count of code* (loc), *Cyclomatic complexity* (v(g)), *Essential complexit* (ev(g)), *Totaloperator+operand* (n), *Count of Statement Lines* (IOCode), *Time to write program* (t), *Effort to write program* (e), *Design complexity* (iv(g)), *Branch count* (branchCount) dan lOComment.

4.2.3 Pembahasan hasil pengujian *dataset* KC1

Sesuai hasil pengujian yang telah dilakukan pada *dataset* KC1 yang ditunjukkan pada Tabel 4.16 maka Gambar 4.4 menunjukkan grafik nilai akurasi hasil dari pengujian *dataset* KC1 dengan jumlah fitur sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur.

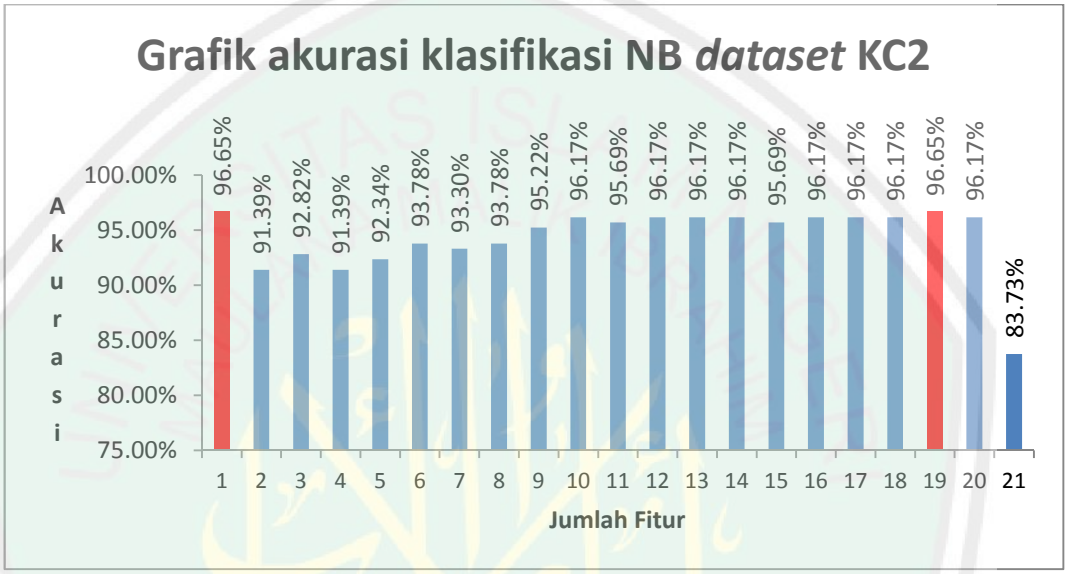


Gambar 4.4 Grafik akurasi klasifikasi NB *dataset* KC1

Pada grafik 4.4 terlihat bahwa akurasi paling baik ditunjukkan pada jumlah fitur sebanyak 1 fitur yaitu sebesar 85.55%, sehingga fitur yang memiliki pengaruh besar terhadap prediksi cacat perangkat lunak untuk *dataset* KC1 berdasarkan seleksi fitur *Symmetrical Uncertainty* berjumlah 1 fitur yaitu fitur *Difficulty* (d).

4.2.4 Pembahasan hasil pengujian *dataset* KC2

Sesuai hasil pengujian yang telah dilakukan pada *dataset* KC2 yang ditunjukkan pada Tabel 4.17 maka Gambar 4.5 menunjukkan grafik nilai akurasi hasil dari pengujian *dataset* KC2 dengan jumlah fitur sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur.

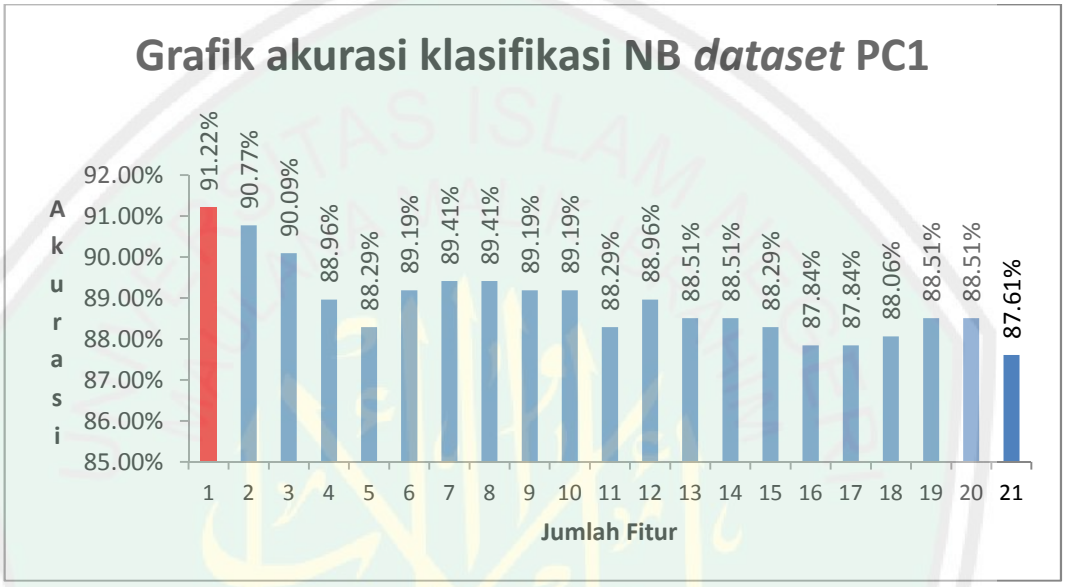


Gambar 4.5 Grafik akurasi klasifikasi NB *dataset* KC2

Pada grafik 4.5 terlihat bahwa akurasi paling baik ditunjukkan pada jumlah fitur sebanyak 1 dan 19 fitur yaitu sebesar 96.65%, namun pada percobaan ini terdapat beberapa fitur dengan hasil akurasi yang sama nantinya akan dipilih jumlah fitur yang paling kecil yaitu jumlah fitur 1 karena dengan jumlah fitur yang sedikit bisa menghasilkan akurasi yang sama besarnya. sehingga fitur yang memiliki pengaruh besar terhadap prediksi cacat perangkat lunak untuk *dataset* KC2 berdasarkan seleksi fitur *Symmetrical Uncertainty* yaitu fitur *Difficulty* (d).

4.2.5 Pembahasan hasil pengujian *dataset* PC1

Sesuai hasil pengujian yang telah dilakukan pada *dataset* PC1 yang ditunjukkan pada Tabel 4.18 maka Gambar 4.6 menunjukkan grafik nilai akurasi hasil dari pengujian *dataset* PC1 dengan jumlah fitur sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur.



Gambar 4.6 Grafik akurasi klasifikasi NB *dataset* PC1

Pada grafik 4.6 terlihat bahwa akurasi paling baik ditunjukkan pada jumlah fitur sebanyak 1 fitur yaitu sebesar 91.22%, sehingga fitur yang memiliki pengaruh besar terhadap prediksi cacat perangkat lunak untuk *dataset* PC1 berdasarkan seleksi fitur *SymmetricalUncertainty* berjumlah 1 fitur yaitu fitur *lOComment*.

Rangkuman dari keseluruhan pembahasan hasil pengujian yang telah dilakukan terhadap lima *dataset* yaitu *dataset* CM1, JM1, KC1, KC2 dan PC1 dengan jumlah fitur yang diambil sebanyak 1 fitur, 2 fitur, 3 fitur, dan seterusnya sampai keseluruhan fitur maka tabel 4.19 menunjukkan fitur yang relevan

terhadap prediksi cacat perangkat lunak berdasarkan seleksi fitur *Symmetrical Uncertainty* yang dibuktikan dengan hasil dari pengujian klasifikasi menunjukkan tingkat akurasi paling besar.

Tabel 4.19 Fitur yang relevan berdasarkan seleksi fitur *Symmetrical Uncertainty*

Dataset	Fitur	Akurasi NB
CM1	• *Time to write program* (t)	99.50%
JM1	• *total operand* (total_Opnd) • *unique operands* (uniq_Opnd) • *total operator*(total_Op) • *unique operators* (uniq_Op) • *Intelligence* (i) • *Programlength* (l) • *Difficulty* (d) • *Volume* (v) • *Error estimate* (b) • *line count of code* (loc) • *Cyclomatic complexity* (v(g)) • *Essential complexit* (ev(g)) • *Totaloperator+operand* (n) • *Count of Statement Lines* (IOCode) • *Time to write program* (t) • *Effort to write program* (e) • *Design complexity* (iv(g)) • *Branch count* (branchCount). • lOComment	80.62%
KC1	• *Difficulty* (d)	85.55%
KC2	• *Difficulty* (d)	96.65%
PC1	• lOComment	91.22%

Berdasarkan Tabel 4.19 maka fitur yang relevan terhadap prediksi cacat perangkat lunak untuk *dataset* CM1 adalah fitur *Time to write program* (t) dengan akurasi sebesar 99.50%. *Dataset* JM1 adalah fitur *total operand* (total_Opnd), *unique operands* (uniq_Opnd) , *total operator* (total_Op), *unique operators* (uniq_Op), *Intelligence* (i), *Programlength* (l), *Difficulty* (d), *Volume* (v), *Error estimate* (b), *line count of code* (loc), *Cyclomatic complexity* (v(g)), *Essential complexit* (ev(g)), *Totaloperator+operand* (n), *Count of Statement Lines* (IOCode), *Time to write program* (t), *Effort to write program* (e), *Design complexity* (iv(g)), *Branch count* (branchCount), lOComment dengan akurasi sebesar 80.62%. *Dataset* KC1 adalah fitur *Difficulty* (d) dengan akurasi sebesar 85.55%. *Dataset* KC2 adalah *Difficulty* (d) dengan akurasi sebesar 96.65%. *Dataset* PC1 adalah fitur *Count of lines of comments* (lOComment) dengan akurasi sebesar 91.22%.

4.3 Integrasi dengan Islam

Ayat-ayat Al-Qur'an menyimpan berbagai kunci untuk pegangan hidup kita.Dengan memegang kunci-kunci tersebut, maka semua masalah kita dapat teratasi.Sebagimana dalam islam diperintahkan untuk saling tolong-menolong dalam hal kebaikan, penejelasan tersebut ada pada potongan Ayat al-quran surat al-ma'idah ayat 2 berikut;

يَا أَيُّهَا الَّذِينَ آمَنُوا لَا تُحِلُّوا شَعَائِرَ اللَّهِ وَلَا الشَّهْرَ الْحَرَامَ وَلَا الْهَدْيَ وَلَا الْقَلَائِدَ وَلَا آمِّينَ الْبَيْتَ الْحَرَامَ يَبْتَغُونَ فَضْلًا مِنْ رَبِّهِمْ وَرِضْوَانًا ۚ وَإِذَا حَلَلْتُمْ فَاصْطَادُوا ۚ وَلَا يَجْرِمَنَّكُمْ شَنَآنُ **قَوْمٍ أَنْ صَدُّوكُمْ عَنِ الْمَسْجِدِ** الْحَرَامِ أَنْ تَعْتَدُوا ۘ **وَتَعَاوَنُوا عَلَى الْبِرِّ وَالتَّقْوَىٰ ۖ وَلَا تَعَاوَنُوا عَلَى الْإِثْمِ وَالْعُدْوَانِ** ۚ وَاتَّقُوا اللَّهَ ۖ إِنَّ اللَّهَ شَدِيدُ **الْعِقَابِ**

Artinya : "*Hai orang-orang yang beriman, janganlah kamu melanggar syi'ar-syi'ar Allah, dan jangan melanggar kehormatan bulan-bulan haram, jangan (mengganggu) binatang-binatang had-ya, dan binatang-binatang qalaa-id, dan jangan (pula) mengganggu orang-orang yang mengunjungi Baitullah sedang mereka mencari kurnia dan keridhaan dari Tuhannya dan apabila kamu telah menyelesaikan ibadah haji, maka bolehlah berburu. Dan janganlah sekali-kali kebencian(mu) kepada sesuatu kaum karena mereka menghalang-halangi kamu dari Masjidilharam, mendorongmu berbuat aniaya (kepada mereka).* ***Dan tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan pelanggaran****. Dan bertakwalah kamu kepada Allah, sesungguhnya Allah amat berat siksa-Nya.*" (al-ma'idah:2).

Pada potongan ayat tersebut Allah SWT memerintahkan untuk saling tolong-menolong dalam aktivitas kebaikan dan dilarang untuk tolong-menolong dalam perbuatan yang salah dan mengakibatkan dosa, potongan ayat tersebut dalam Tafsir Ibnu Katsir dijelaskan Allah SWT memerintahkan kepada hamba-hamba- Nya yang beriman untuk saling menolong dalam berbuat kebaikan yaitu kebajikan dan meninggalkan hal-hal yang mungkar; hal ini dinamakan ketakwaan. Allah SWT melarang mereka bantu-membantu dalam kebatilan serta tolong-menolong dalam perbuatan dosa dan hal- hal yang diharamkan. Ibnu Jarir mengatakan bahwa dosa itu ialah meninggalkan apa yang diperintahkan oleh Allah untuk dikerjakan. Pelanggaran itu artinya melampaui apa yang digariskan oleh Allah dalam agama kalian, serta melupakan apa yang difardukan oleh Allah atas diri kalian dan atas diri orang lain. Penjelasan tersebut kemudian diperkuat

dengan hadits yang diriwayatkan oleh Imam Bukhari secara munfarid melalui hadis Hasyim dengan sanad yang sama dan lafaz yang semisal. Keduanya mengetengahkan hadis ini melalui jalur Sabit, dari Anas yang menceritakan bahwa Rasulullah Saw.telah bersabda:

انْصُرْ أَخَاكَ ظَالِمًا أَوْ مَظْلُومًا". قِيلَ: يَا رَسُولَ اللَّهِ، هَذَا نَصَرْتُهُ مَظْلُومًا، فَكَيْفَ أَنْصُرُهُ ظَالِمًا؟ قَالَ:

"تَمْنَعُهُ مِنَ الظُّلْمِ، فَذَاكَ نَصْرُكَ إِيَّاهُ

Artinya :*"Tolonglah saudaramu, baik dia berbuat aniaya ataupun dianiaya." Ditanyakan, "Wahai Rasulullah, orang ini dapat aku tolong bila dalam keadaan teraniaya, tetapi bagaimana menolongnya jika dia berbuat aniaya?" Rasulullah Saw. menjawab, "Kamu cegah dia dari perbuatan aniaya, itulah cara kamu menolongnya.".*

Menurut kajian yang penulis lakukan pada al-qur'an potongan surah al-ma'idah yang artinya "***Dan tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan pelanggaran***" dan telah dijelaskan di atas dengan rujukan tafsir Ibnu Katsir yang menjelaskan bahwasannya Allah SWT memerintahkan kepada hamba-hamba- Nya yang beriman untuksaling menolong dalam berbuat kebaikan yaitu kebajikan dan meninggalkan hal-hal yang mungkar; hal ini dinamakan ketakwaan, adapun kajian yang dilakukan penulis terhadap kata "takwa". takwa dalam penjelasan hadits Ar-Ba'in Imam Nawawi ke-18, menjelaskan bahwa :

عَنْ أَبِيْ ذَرٍّ جُنْدُبِ بن جُنَادَةَ وَأَبِي عَبْدِ الرَّحْمَن مُعَاذِ بن جَبَلٍ رَضِيَ اللهُ عَنْهُمَا عَنْ رَسُول اللهِ صلى

روه – (اتَّقِ اللهَ حَيْثُمَا كُنْتَ، وَأَتْبِعِ السَّيِّئَةَ الحَسَنَةَ تَمْحُهَا، وَخَالِقِ الناس بِخُلُقٍ حَسَنٍ) :الله عليه وسلم قَالَ

حسنٌ صحيح :وفي بعض النسخ .حديث حسن :الترمذي وقال

Dari Abu Dzar, Jundub bin Junadah dan Abu ‘Abdurrahman, Mu’adz bin Jabal radhiyallahu ‘anhuma, dari Rasulullah Shallallahu ‘alaihi wa Sallam, beliau bersabda : “Bertaqwalah kepada Allah di mana saja engkau berada dan susullah sesuatu perbuatan dosa dengan kebaikan, pasti akan menghapuskannya dan bergaullah sesama manusia dengan akhlaq yang baik”.

(HR. Tirmidzi, ia telah berkata : Hadits ini hasan, pada lafazh lain derajatnya hasan shahih).

Berdasarkan kajian al-quran surah surat al-ma’idah ayat 2 dan hadits Ar-Ba’in Imam Nawawi ke-18, Allah SWT menyuruh kita untuk saling tolong menolong dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan pelanggaran,serta susullah sesuatu perbuatan dosa dengan kebaikan, pasti akan menghapuskannya dan bergaullah sesama manusia dengan akhlaq yang baik, itu semua merupakan salah satu ciri-ciri dari orang yang bertakwa kepada Allah SWT. Potongan surah al-ma’idah ayat 2 dan hadits Ar-Ba’in Imam Nawawi ke-18 itupun memiliki hubungan dengan surah Al-Hujurat Ayat 13 yang berbunyi,

يَٰٓأَيُّهَا ٱلنَّاسُ إِنَّا خَلَقْنَٰكُم مِّن ذَكَرٍ وَأُنثَىٰ وَجَعَلْنَٰكُمْ شُعُوبًا وَقَبَآئِلَ لِتَعَارَفُوٓا۟ ۚ إِنَّ أَكْرَمَكُمْ عِندَ ٱللَّهِ أَتْقَىٰكُمْ ۚ إِنَّ ٱللَّهَ عَلِيمٌ خَبِيرٌ

Artinya : “Hai manusia, sesungguhnya Kami menciptakan kamu dari seorang laki-laki dan seorang perempuan dan menjadikan kamu berbangsa-bangsa dan bersuku-suku supaya kamu saling kenal-mengenal. Sesungguhnya orang yang

paling mulia diantara kamu disisi Allah ialah orang yang paling takwa diantara kamu.Sesungguhnya Allah Maha Mengetahui lagi Maha Mengenal".

Berdasarkan kajian beberapa potongan surah pada al-quran, tafsir ibnu katsir dan hadits Ar-Ba'in Imam Nawawi ke-18 diatas, kualitas dari manusia dapat dilihat dari ketakwaannya, dan terdapat beberapa ciri-ciri dari orang yang bertakwa diantaranya adalah orang yang saling tolong-menolong dalam hal kebaikan dan tidak tolong menolong dalam hal berbuat dosa dan pelanggaran. Kemudian susullah sesuatu perbuatan dosa dengan kebaikan, pasti kebaikan itu akan menghapuskannya dan bergaullah sesama manusia dengan akhlaq yang baik. hal itu sama halnya dengan beberapa ini dari definisi teknologi yaitu untuk memberikan kenyamanan serta kemudahan bagi manusia.Pada penelitian ini penulis melakukan penelitian terkait pengujian (*testing*) karena tahap pengujian harus dioperasikan secara efektif untuk merilis perangkat lunak untuk pengguna yang bebas dari *bug*. Bebas dari *bug* adalah hal yang paling penting untuk diperhatikan dalam pengembangan perangkat lunak untuk keefektifan dan efisiensi kinerja perangkat lunak bagi pengguna serta mempermudah pekerjaan para pengembang perangkat lunak.Secara umum ada beberapa tahap dalam *Software Development Life Cycle* (SDLC) yaitu analisa kebutuhan, desain, implementasi, pengujian (*testing*) dan *release*.Pada penelitian ini penulis melakukan penelitian terkait pengujian (*testing*) karena tahap pengujian harus dioperasikan secara efektif untuk merilis perangkat lunak untuk pengguna yang bebas dari *bug*. Bebas dari *bug* adalah hal yang paling penting untuk diperhatikan dalam pengembangan perangkat lunak untuk keefektifan dan efisiensi kinerja perangkat lunak.

BAB V

KESIMPULAN DAN SARAN

Pada bab penutup ini menjelaskan tentang kesimpulan dari penelitian ini serta memberi saran bagi pembaca untuk pengembangan penelitian.

5.1 Kesimpulan

Berdasarkan dari hasil penelitian yang telah dilakukan, maka dapat disimpulkan bahwa fitur yang relevan berdasarkan seleksi fitur *Symmetrical Uncertainty* untuk setiap *dataset* yaitu *dataset* CM1 adalah fitur *Time to write program* (t) dengan akurasi sebesar 99.50%. *Dataset* JM1 adalah fitur *total operand* (total_Opnd), *unique operands* (uniq_Opnd) , *total operator* (total_Op), *unique operators* (uniq_Op), *Intelligence* (i), *Programlength* (l), *Difficulty* (d), *Volume* (v), *Error estimate* (b), *line count of code* (loc), *Cyclomatic complexity* (v(g)), *Essential complexit* (ev(g)), *Totaloperator+operand* (n), *Count of Statement Lines* (IOCode), *Time to write program* (t), *Effort to write program* (e), *Design complexity* (iv(g)), *Branch count* (branchCount), lOComment dengan akurasi sebesar 80.62%. *Dataset* KC1 adalah fitur *Difficulty* (d) dengan akurasi sebesar 85.55%. *Dataset* KC2 adalah *Difficulty* (d) dengan akurasi sebesar 96.65%. *Dataset* PC1 adalah fitur *Count of lines of comments* (lOComment) dengan akurasi sebesar 91.22%. Fitur-fitur tersebut dikatakan sebagai fitur yang relevan untuk prediksi cacat perangkat lunak karena berdasarkan uji coba klasifikasi yang menghasilkan akurasi terbaik dari uji coba yang telah dilakukan.

5.2 Saran

Peneliti menyadari bahwa dalam penelitian ini masih belum sempurna, dengan demikian perlu adanya pengembangan untuk mendapatkan hasil yang lebih baik, beberapa saran dari peneliti antara lain:

a. Melakukan seleksi fitur dengan metode seleksi fitur yang lain seperti *Fast Correlation Based Filter* (FCBF), *Correlation-based Feature Selection* (CFS), *Chi-square* dan *Relief-F* dan metode seleksi fitur lainnya, dengan tujuan mendapatkan fitur yang mungkin lebih relevan terhadap prediksi cacat perangkat lunak.
b. Melakukan uji coba dengan *dataset* yang berbeda.
c. Proses seleksi fitur dan klasifikasi menggunakan model pemecahan data *cross-validation*.
d. Melakukan pembuktian dengan algoritma klasifikasi yang lain seperti *C4.5*, *Random Forest*, *Support Vector Machine* (SVM) atau algoritma klasifikasi yang lain dengan tujuan untuk menguji fitur yang didapat pada seleksi fitur.

DAFTAR PUSTAKA

Akbar , M. S. (2017). *Prediksi Cacat Perangkat Lunak dengan Optimasi Naive Bayes Menggunakan Gain Ratio.* Surabaya, Jawa Timur, Indonesia: Fakultas Teknologi Informasi. Institut teknologi sepuluh november .

Al-Sheikh, A. b. (1994). *Tafsir Ibnu Katsir Jilid 1.* Bogor: Pustaka Imam asy-Syafi'i.

Arar, O. F., & Ayan , K. (2015). Software defect prediction using cost-sensitive neural network. *Applied Soft Computing*.

Arar, O. F., & Ayan, K. (2017, Mei). A Feature Dependent Naive Bayes Approach and Its Application to the Software Defect Prediction Problem. *Soft Computing Journal*.

Beniwal , S., & Arora , J. (2012). Classification and Feature Selection Techniques in Data Mining. *International Journal of Engineering Research & Technology (IJERT)*, 1.

Boehm , B., & Basili , R. V. (2001). Software Defect Reduction Top 10 List .

Chang, C. P., Chu, C. P., & Yeh, Y. F. (2009). Integrating in-process software defect prediction with association mining to discover defect pattern. *Information and Software Technology Journal*.

Dougherty , J., Kohavi , R., & Sahami , M. (1995). Supervised and Unsupervised Discretization of Continuous Features.

Essra , A., Rahmadani , & Safriadi . (2016, Desember). ANALISIS INFORMATION GAIN ATTRIBUTE EVALUATION UNTUK KLASIFIKASI SERANGAN INTRUSI. *Jurnal ISD (Information System Development)*.

Han , J., Kamber , M., & Pei , J. (2012). *Data Mining Concepts and Techniques Third Edition.* United States of America , United States of America: Morgan Kaufmann .

Karabulut, E. M., Özel, S. A., & İbrikçi, T. (2012). A comparative study on the effect of feature selection on classification accuracy. *Procedia Technology*.

Kumar, C. S., & Sree, R. R. (2014, OCTOBER). APPLICATION OF RANKING BASED ATTRIBUTE SELECTION FILTERS TO PERFORM AUTOMATED EVALUATION OF DESCRIPTIVE ANSWERS THROUGH SEQUENTIAL MINIMAL OPTIMIZATION MODELS. *Journal on Soft Computing* .

Kumaresh , S., R , B., & Sivaguru , M. (2014). Software Defect Classification using Bayesian Classification Techniques. *International Conference on Communication, Computing and Information Technology* .

Menzies , Greenwald , J., & Frank , A. (2007, Januari). Data Mining Static Code Attributes to Learn Defect Predictors. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING* .

Menzies, T. (n.d.). *http://promise.site.uottawa.ca/SERepository/datasets-page.html*. Retrieved Maret Senin, 2018, from Promise Software Engineering Repository: http://promise.site.uottawa.ca/

Pelayo , L., & Dick , S. (2007). Applying Novel Resampling Strategies To Software Defect Prediction. *Conference Fuzzy Information Processing Society*.

Prasetyo, E. (2012). *Data Mining - Konsep dan Aplikasi Menggunakan Matlab.* Yogyakarta: Andi.

Pressman , R. S. (2001). *Software Engineering A PRACTITIONER'S APPROACH.* (N. T. C. L. Liu, Ed.) New York San Francisco : McGraw-Hill.

Runeson , P., Andersson , C., Thelin , T., Andrews , A., & Berling , T. (2006). What Do We Know about Defect Detection Methods? *IEEE SOFTWARE* .

Santoso, I. B. (2013). *STATISTIKA UNTUK TEKNIK INFORMATIKA.* Malang, Jawa Timur, Indonesia: UIN-Maliki Press.

Sastry, K., & Prasad , R. S. (2015). *Software Defect Predication using Classifier Mining* . IJRDO - Journal of Computer Science and Engineering.

Singh , P., & Verma , S. (2009). An Investigation of the Effect of Discretization on Defect Prediction Using Static Measures. *2009 International Conference on Advances in Computing, Control, and Telecommunication Technologies*.

Song , Q., Jia , Z., Shepperd , M., Ying , S., & Liu , J. (2011). A General Software Defect-Proneness Prediction Framework. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*.

Subiyakto , A. (2008, April). Penggunaan Algoritma Klasifikasi dalam Data Mining.

Suyanto. (2017). *Data Mining Untuk Klasifikasi dan Klasterisasi Data.* Bandung: Informatika.

Wahono , R. S. (2015). A Systematic Literature Review of Software Defect Prediction: Research Trends, Datasets, Methods and Frameworks. *Software Engineering* .

Wang , H., Khoshgoftaar , T. M., & Napolitano , A. (2012). Software measurement data reduction using ensemble techniques. *Neurocomputing*.

Yang, Y., Webb, G. I., & Wu, X. (2010). Discretization Methods. *Data Mining and Knowledge Discovery Handbook, 2nd ed.*

LAMPIRAN 1

Perhitungan *confusion matrix* dan akurasi klasifikasi

Tabel 1 Hasil Prediksi dataset CM1 dengan jumlah fitur sebanyak 5 fitur

No	Hasil Prediksi	Kenyataan
1	FALSE	FALSE
2	FALSE	FALSE
3	TRUE	FALSE
4	FALSE	FALSE
5	FALSE	FALSE
6	TRUE	FALSE
7	FALSE	FALSE
8	FALSE	FALSE
9	FALSE	FALSE
10	FALSE	FALSE
11	TRUE	FALSE
12	FALSE	FALSE
13	FALSE	FALSE
14	FALSE	FALSE
15	FALSE	FALSE
16	FALSE	FALSE
17	FALSE	FALSE
18	FALSE	FALSE
19	FALSE	FALSE
20	FALSE	FALSE
21	FALSE	FALSE
22	FALSE	FALSE
23	FALSE	FALSE
24	TRUE	FALSE
25	FALSE	FALSE
26	FALSE	FALSE
27	FALSE	FALSE
28	FALSE	FALSE
29	FALSE	FALSE
30	FALSE	FALSE
31	FALSE	FALSE
32	FALSE	FALSE
33	TRUE	FALSE
34	FALSE	FALSE
35	FALSE	FALSE
36	TRUE	FALSE
37	FALSE	FALSE
38	FALSE	FALSE
39	FALSE	FALSE
40	FALSE	FALSE
41	FALSE	FALSE
42	FALSE	FALSE
43	FALSE	FALSE
44	FALSE	FALSE
45	FALSE	FALSE
46	FALSE	TRUE
47	FALSE	TRUE
48	FALSE	TRUE
49	FALSE	TRUE
50	FALSE	TRUE
51	FALSE	FALSE
52	FALSE	FALSE
53	FALSE	FALSE
54	FALSE	FALSE
55	FALSE	FALSE
56	FALSE	FALSE
57	FALSE	FALSE
58	FALSE	FALSE
59	FALSE	FALSE
60	FALSE	FALSE
61	FALSE	FALSE
62	FALSE	FALSE
63	FALSE	FALSE
64	FALSE	FALSE
65	FALSE	FALSE
66	FALSE	FALSE
67	FALSE	FALSE
68	FALSE	FALSE
69	FALSE	FALSE
70	FALSE	FALSE
71	FALSE	FALSE
72	FALSE	FALSE
73	FALSE	FALSE
74	FALSE	FALSE

75	FALSE	FALSE
76	FALSE	FALSE
77	FALSE	FALSE
78	FALSE	FALSE
79	FALSE	FALSE
80	TRUE	FALSE
81	FALSE	FALSE
82	FALSE	FALSE
83	FALSE	FALSE
84	FALSE	FALSE
85	FALSE	FALSE
86	FALSE	FALSE
87	FALSE	FALSE
88	FALSE	FALSE
89	FALSE	FALSE
90	FALSE	FALSE
91	FALSE	FALSE
92	FALSE	FALSE
93	FALSE	FALSE
94	FALSE	FALSE
95	FALSE	FALSE
96	FALSE	TRUE
97	FALSE	TRUE
98	FALSE	TRUE
99	FALSE	TRUE
100	FALSE	TRUE
101	FALSE	FALSE
102	FALSE	FALSE
103	FALSE	FALSE
104	FALSE	FALSE
105	FALSE	FALSE
106	FALSE	FALSE
107	FALSE	FALSE
108	FALSE	FALSE
109	FALSE	FALSE
110	FALSE	FALSE
111	FALSE	FALSE
112	FALSE	FALSE
113	FALSE	FALSE
114	FALSE	FALSE
115	FALSE	FALSE
116	FALSE	FALSE
117	FALSE	FALSE

118	TRUE	FALSE
119	FALSE	FALSE
120	FALSE	FALSE
121	FALSE	FALSE
122	FALSE	FALSE
123	FALSE	FALSE
124	FALSE	FALSE
125	FALSE	FALSE
126	FALSE	FALSE
127	FALSE	FALSE
128	FALSE	FALSE
129	FALSE	FALSE
130	FALSE	FALSE
131	FALSE	FALSE
132	FALSE	FALSE
133	FALSE	FALSE
134	FALSE	FALSE
135	FALSE	FALSE
136	FALSE	FALSE
137	TRUE	FALSE
138	FALSE	FALSE
139	FALSE	FALSE
140	FALSE	FALSE
141	FALSE	FALSE
142	FALSE	FALSE
143	FALSE	FALSE
144	FALSE	FALSE
145	FALSE	FALSE
146	FALSE	TRUE
147	TRUE	TRUE
148	FALSE	TRUE
149	TRUE	TRUE
150	FALSE	TRUE
151	FALSE	FALSE
152	FALSE	FALSE
153	FALSE	FALSE
154	FALSE	FALSE
155	FALSE	FALSE
156	FALSE	FALSE
157	FALSE	FALSE
158	FALSE	FALSE
159	FALSE	FALSE
160	FALSE	FALSE

161	FALSE	FALSE
162	FALSE	FALSE
163	FALSE	FALSE
164	FALSE	FALSE
165	FALSE	FALSE
166	FALSE	FALSE
167	FALSE	FALSE
168	FALSE	FALSE
169	FALSE	FALSE
170	FALSE	FALSE
171	FALSE	FALSE
172	FALSE	FALSE
173	FALSE	FALSE
174	FALSE	FALSE
175	FALSE	FALSE
176	TRUE	FALSE
177	FALSE	FALSE
178	FALSE	FALSE
179	TRUE	FALSE
180	FALSE	FALSE
181	FALSE	FALSE
182	FALSE	FALSE
183	FALSE	FALSE
184	FALSE	FALSE
185	TRUE	FALSE
186	FALSE	FALSE
187	FALSE	FALSE
188	FALSE	FALSE
189	FALSE	FALSE
190	FALSE	FALSE
191	FALSE	FALSE
192	FALSE	FALSE
193	FALSE	FALSE
194	FALSE	FALSE
195	TRUE	FALSE
196	FALSE	TRUE
197	FALSE	TRUE
198	FALSE	TRUE
199	FALSE	TRUE
200	FALSE	TRUE
201	FALSE	FALSE
202	FALSE	FALSE
203	FALSE	FALSE

204	FALSE	FALSE
205	FALSE	FALSE
206	FALSE	FALSE
207	FALSE	FALSE
208	FALSE	FALSE
209	FALSE	FALSE
210	FALSE	FALSE
211	FALSE	FALSE
212	TRUE	FALSE
213	FALSE	FALSE
214	FALSE	FALSE
215	FALSE	FALSE
216	FALSE	FALSE
217	FALSE	FALSE
218	FALSE	FALSE
219	FALSE	FALSE
220	FALSE	FALSE
221	FALSE	FALSE
222	FALSE	FALSE
223	FALSE	FALSE
224	FALSE	FALSE
225	FALSE	FALSE
226	FALSE	FALSE
227	FALSE	FALSE
228	FALSE	FALSE
229	FALSE	FALSE
230	FALSE	FALSE
231	TRUE	FALSE
232	FALSE	FALSE
233	FALSE	FALSE
234	FALSE	FALSE
235	FALSE	FALSE
236	FALSE	FALSE
237	FALSE	FALSE
238	FALSE	FALSE
239	FALSE	FALSE
240	FALSE	FALSE
241	FALSE	FALSE
242	FALSE	FALSE
243	TRUE	FALSE
244	FALSE	FALSE
245	FALSE	FALSE
246	FALSE	TRUE

247	FALSE	TRUE
248	TRUE	TRUE
249	TRUE	TRUE
250	FALSE	TRUE
251	FALSE	FALSE
252	FALSE	FALSE
253	FALSE	FALSE
254	FALSE	FALSE
255	FALSE	FALSE
256	FALSE	FALSE
257	FALSE	FALSE
258	FALSE	FALSE
259	FALSE	FALSE
260	FALSE	FALSE
261	FALSE	FALSE
262	FALSE	FALSE
263	FALSE	FALSE
264	FALSE	FALSE
265	FALSE	FALSE
266	FALSE	FALSE
267	FALSE	FALSE
268	FALSE	FALSE
269	FALSE	FALSE
270	FALSE	FALSE
271	TRUE	FALSE
272	FALSE	FALSE
273	FALSE	FALSE
274	FALSE	FALSE
275	FALSE	FALSE
276	FALSE	FALSE
277	FALSE	FALSE
278	FALSE	FALSE
279	TRUE	FALSE
280	TRUE	FALSE
281	FALSE	FALSE
282	FALSE	FALSE
283	FALSE	FALSE
284	FALSE	FALSE
285	FALSE	FALSE
286	FALSE	FALSE
287	FALSE	FALSE
288	FALSE	FALSE
289	FALSE	FALSE

290	FALSE	FALSE
291	FALSE	FALSE
292	FALSE	FALSE
293	FALSE	FALSE
294	FALSE	FALSE
295	FALSE	FALSE
296	TRUE	TRUE
297	FALSE	TRUE
298	FALSE	TRUE
299	TRUE	TRUE
300	FALSE	TRUE
301	FALSE	FALSE
302	FALSE	FALSE
303	FALSE	FALSE
304	FALSE	FALSE
305	FALSE	FALSE
306	FALSE	FALSE
307	FALSE	FALSE
308	FALSE	FALSE
309	FALSE	FALSE
310	FALSE	FALSE
311	FALSE	FALSE
312	FALSE	FALSE
313	FALSE	FALSE
314	FALSE	FALSE
315	TRUE	FALSE
316	FALSE	FALSE
317	FALSE	FALSE
318	FALSE	FALSE
319	FALSE	FALSE
320	FALSE	FALSE
321	TRUE	FALSE
322	FALSE	FALSE
323	FALSE	FALSE
324	FALSE	FALSE
325	FALSE	FALSE
326	FALSE	FALSE
327	FALSE	FALSE
328	FALSE	FALSE
329	FALSE	FALSE
330	FALSE	FALSE
331	FALSE	FALSE
332	TRUE	FALSE

333	FALSE	FALSE
334	FALSE	FALSE
335	FALSE	FALSE
336	FALSE	FALSE
337	FALSE	FALSE
338	FALSE	FALSE
339	FALSE	FALSE
340	FALSE	FALSE
341	FALSE	FALSE
342	FALSE	FALSE
343	TRUE	FALSE
344	FALSE	FALSE
345	FALSE	FALSE
346	TRUE	TRUE
347	TRUE	TRUE
348	FALSE	TRUE
349	TRUE	TRUE
350	FALSE	TRUE
351	FALSE	FALSE
352	FALSE	FALSE
353	FALSE	FALSE
354	FALSE	FALSE
355	FALSE	FALSE
356	TRUE	FALSE
357	FALSE	FALSE
358	FALSE	FALSE
359	FALSE	FALSE
360	FALSE	FALSE
361	FALSE	FALSE
362	FALSE	FALSE
363	FALSE	FALSE
364	FALSE	FALSE
365	FALSE	FALSE
366	FALSE	FALSE
367	FALSE	FALSE
368	FALSE	FALSE
369	FALSE	FALSE
370	FALSE	FALSE
371	FALSE	FALSE
372	FALSE	FALSE
373	FALSE	FALSE
374	FALSE	FALSE
375	FALSE	FALSE

376	FALSE	FALSE
377	FALSE	FALSE
378	TRUE	FALSE
379	FALSE	FALSE
380	FALSE	FALSE
381	FALSE	FALSE
382	FALSE	FALSE
383	FALSE	FALSE
384	FALSE	FALSE
385	FALSE	FALSE
386	FALSE	FALSE
387	FALSE	FALSE
388	FALSE	FALSE
389	FALSE	FALSE
390	FALSE	FALSE
391	FALSE	FALSE
392	FALSE	FALSE
393	FALSE	FALSE
394	FALSE	FALSE
395	FALSE	FALSE
396	FALSE	TRUE
397	TRUE	TRUE
398	FALSE	TRUE
399	FALSE	TRUE
400	FALSE	TRUE
401	FALSE	FALSE
402	FALSE	FALSE
403	FALSE	FALSE
404	FALSE	FALSE
405	FALSE	FALSE
406	FALSE	FALSE
407	FALSE	FALSE
408	FALSE	FALSE
409	FALSE	FALSE
410	FALSE	FALSE
411	FALSE	FALSE
412	FALSE	FALSE
413	TRUE	FALSE
414	FALSE	FALSE
415	FALSE	FALSE
416	FALSE	FALSE
417	FALSE	FALSE
418	FALSE	FALSE

419	FALSE	FALSE
420	FALSE	FALSE
421	FALSE	FALSE
422	TRUE	FALSE
423	FALSE	FALSE
424	FALSE	FALSE
425	FALSE	FALSE
426	FALSE	FALSE
427	FALSE	FALSE
428	FALSE	FALSE
429	FALSE	FALSE
430	FALSE	FALSE
431	FALSE	FALSE
432	FALSE	FALSE
433	FALSE	FALSE
434	FALSE	FALSE
435	TRUE	FALSE
436	FALSE	FALSE
437	FALSE	FALSE
438	FALSE	FALSE
439	FALSE	FALSE
440	TRUE	FALSE
441	FALSE	FALSE
442	FALSE	FALSE
443	FALSE	FALSE
444	FALSE	FALSE
445	FALSE	FALSE
446	FALSE	TRUE
447	FALSE	TRUE
448	FALSE	TRUE
449	FALSE	TRUE
450	FALSE	FALSE
451	FALSE	FALSE
452	FALSE	FALSE
453	FALSE	FALSE
454	FALSE	FALSE
455	FALSE	FALSE
456	FALSE	FALSE
457	FALSE	FALSE
458	FALSE	FALSE
459	FALSE	FALSE
460	FALSE	FALSE
461	FALSE	FALSE

462	FALSE	FALSE
463	FALSE	FALSE
464	FALSE	FALSE
465	FALSE	FALSE
466	FALSE	FALSE
467	FALSE	FALSE
468	FALSE	FALSE
469	FALSE	FALSE
470	FALSE	FALSE
471	TRUE	FALSE
472	FALSE	FALSE
473	FALSE	FALSE
474	FALSE	FALSE
475	FALSE	FALSE
476	FALSE	FALSE
477	FALSE	FALSE
478	FALSE	FALSE
479	FALSE	FALSE
480	TRUE	FALSE
481	FALSE	FALSE
482	FALSE	FALSE
483	FALSE	FALSE
484	FALSE	FALSE
485	FALSE	FALSE
486	FALSE	FALSE
487	TRUE	FALSE
488	FALSE	FALSE
489	FALSE	FALSE
490	FALSE	FALSE
491	FALSE	FALSE
492	FALSE	FALSE
493	FALSE	FALSE
494	FALSE	TRUE
495	FALSE	TRUE
496	FALSE	TRUE
497	FALSE	TRUE
498	FALSE	TRUE

Berdasarkan tabel 1 yang menunjukkan hasil prediksi dengan kenyataan jika dibentuk dalam *confusion matrix* maka akan tampil seperti tabel 2, setelah *confusion matrix* berhasil dibuat maka nilai akaurasi dapat dihitung.

Tabel 2 Confusion Matrix Klasifikasi Dataset CM1 Dengan Seleksi Fitur

		Kelas hasil prediksi		
		Ya	Tidak	Jumlah
Kelas aktual	Ya	10	32	*P*
	Tidak	39	417	*N*
	Jumlah	49	449	*P+N*

Sesuai dengan rumus perhitungan *confusion matrix* maka perhitungan akurasi sebagai berikut:

Akurasi dengan seleksi fitur SU $= \frac{417+10}{417+10+39+32} = 0.8574$