TESIS – KI 142502

PREDIKSI CACAT PERANGKAT LUNAK DENGAN OPTIMASI NAIVE BAYES MENGGUNAKAN GAIN RATIO

Muhammad Sonhaji Akbar

NRP. 5115201010

DOSEN PEMBIMBING
Dr. Ir. Siti Rochimah, MT
NIP. 1968100 2199403 2 001

PROGRAM MAGISTER

BIDANG KEAHLIAN REKAYASA PERANGKAT LUNAK

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2017

(halaman ini sengaja dikosongkan)

THESIS – KI 142502

SOFTWARE DEFECT PREDICTION WITH NAÏVE BAYES GAIN RATIO OPTIMATION

Muhammad Sonhaji Akbar

NRP. 5115201010

SUPERVISOR
Dr. Ir. Siti Rochimah, MT
NIP. 1968100 2199403 2 001

MASTER PROGRAM

THE EXPERTISE OF SOFTWARE ENGINEERING

DEPARTMENT OF INFORMATICS

FACULTY OF INFORMATION TECHNOLOGY

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2017

(halaman ini sengaja dikosongkan)

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom.)
di
Institut Teknologi Sepuluh Nopember Surabaya

oleh:
Muhammad Sonhaji Akbar
Nrp. 5115201010

Dengan judul :
Prediksi Cacat Perangkat Lunak dengan Optimasi Naive Bayes Menggunakan Gain Ratio

Tanggal Ujian : 10-1-2017
Periode Wisuda : 2016 Gasal

Disetujui oleh:

Dr. Ir. Siti Rochimah, M.T
NIP. 196810021994032001
(Pembimbing 1)

Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng.
NIP. 197411232006041001
(Penguji 1)

Rizky Januar Akbar, S.Kom, M.Eng
NIP. 198701032014041001
(Penguji 2)

Fajar Baskoro, S.Kom, M.T
NIP. 197404031999031002
(Penguji 3)

an. Direktur Program Pascasarjana
Asisten Direktur

Prof. Dr. Ir. Tri Widjaja, M.Eng.
NIP. 19611021 198803 1 001

Direktur Program Pasca Sarjana,

Prof.Ir.Djauhar Manfaat, M.Sc., Ph.D.
NIP. 196012021987011001

(halaman ini sengaja dikosongkan)

Prediksi Cacat Perangkat Lunak dengan Optimasi Naive Bayes Menggunakan Gain Ratio

Nama Mahasiswa : Muhammad Sonhaji Akbar

NRP : 5115201010

Pembibimbing : Dr. Ir. Siti Rochimah, MT

Abstrak

Kesalahan prediksi cacat perangkat lunak merupakan hal yang sangat penting karena data yang salah prediksi dapat menimbulkan pengaruh terhadap perangkat lunak itu sendiri. Kurang optimalnya metode prediksi yang digunakan. Masih terdapat kesalahan dalam memprediksi cacat perangkat lunak. Dapat memperlambat proses prediksi cacat perangkat lunak. Bagaimana mengoptimasi metode prediksi dalam cacat perangkat lunak. Metode yang cocok digunakan dalam menangani kesalahan prediksi dalam cacat perangkat lunak. Tujuan dan keuntungan optimasi untuk meningkatkan keakuratan prediksi kerusakan/cacat sebuah perangkat lunak.

Penelitian ini mengusulkan metode Naïve Bayes untuk meningkatkan meningkatkan akurasi prediksi cacat perangkat lunak dengan menggunakan Gain Ratio. Menurut Zaidi, Serquedes dan Carman, pembobotan atribut pada Naïve Bayes (NB) dapat mengurangi dampak kegagalan prediksi. Salah satu metode yang dapat digunakan untuk pembobotan Naïve Bayes yaitu Gain Ratio. Gain Ratio digunakan untuk memilih atribut terbaik diantara seluruh atribut pada metode prediksi. Penggunaan Gain Ratio diharapkan dapat meningkatkan performa prediksi. Atribut dari Gain Ratio sendiri merupakan hasil bagi dari Information Gain dan Split Information. Hasil pengujian dari penelitian ini yaitu meningkatkan akurasi prediksi cacat perangkat lunak menggunakan metode optimasi Naive Bayes Gain Ratio (NBGR). Hasil precision, recall, f-measure dan akurasi dari metode NBGR berturut-turut yaitu 84,48%, 84,86%, 83,54% dan 84,89. Seluruh hasil akurasi NBGR lebih baik daripada metode NB.

.

Kata Kunci : *Prediksi, Cacat Perangkat Lunak, Naïve Bayes, Gain Ratio.*

(halaman ini sengaja dikosongkan)

Software Defect Prediction with Naïve Bayes Gain Gatio Optimation

Nama Mahasiswa : Muhammad Sonhaji Akbar

NRP : 5115201010

Pembibimbing : Dr. Ir. Siti Rochimah, MT

Abstract

Software defects prediction errors is very important because incorrect data predictions can be impacting on the software itself. Less optimal prediction methods used. Still there are errors in predicting software defects. Can slow down the process of software defects prediction. How to optimize prediction methods in software defects. The method suitable for use in dealing with the prediction error in the software defects. Interest and profit optimization to improve the accuracy of prediction of the damage / defect software.

This study proposes a method to improve the Naïve Bayes improve the accuracy of prediction of software defects by using Gain Ratio. According to Zaidi, Serquedes and Carman, weighting attributes on Naïve Bayes (NB) can reduce the impact of failure prediction. One method that can be used for weighting Naïve Bayes ie Gain Ratio. Gain Ratio is used to select the best among all the attributes attribute on the method of prediction. Use of Gain Ratio is expected to improve the performance of the prediction. Attributes of Gain Ratio itself is the quotient of Information Gain and Information Split. The test results of this research is to improve the prediction accuracy of software defects using optimization methods Naive Bayes Gain Ratio (NBGR). Results of precision, recall, F-measure and accuracy of the method NBGR respectively, are 83.48%, 84.86%, 83.54% and 84.89. All proceeds NBGR accuracy better than NB method.

Keywords : *Prediction, Defect, Software, Naïve Bayes, Gain Ratio.*

(halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur kepada Allah SWT atas segala karunia dan rahmat-Nya sehingga penulis dapat menyelesaikan tesis yang berjudul "Prediksi Cacat Perangkat Lunak Dengan Optimasi Naive Bayes Menggunakan Gain Ratio":

Pada kesempatan ini penulis ingin menyampaikan ucapan terima kasih dan penghormatan yang sebesar-besarnya kepada:

1. Bapak saya Gatot Subrata, Ibu saya Endang Sriwati, adik saya Fatah Kurniawan, serta keluarga yang selalu memberikan dukungan penuh untuk menyelesaikan tesis ini.
2. Ibu Dr. Ir. Siti Rochimah, M.T selaku dosen pembimbing yang telah bersedia meluangkan waktu untuk memberikan ilmu, arahan, petunjuk, dan motivasi selama proses pengerjaan tesis ini.
3. Bapak Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng, Bapak Rizky Januar Akbar, S.Kom, M.Eng, Bapak Fajar Baskoro S.Kom, M.T selaku dosen penguji yang telah memberikan saran, arahan, dan koreksi dalam tesis ini.
4. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya.
5. Seluruh staf dan karyawan Teknik Informatika ITS yang telah banyak memberikan kelancaran administrasi akademik.
6. Rekan-rekan angkatan 2015 Pasca Sarjana Teknik Informatika ITS yang telah menemani perjuangan selama 1,5 tahun ini atas dukungan terhadap pengerjaan tesis ini.
7. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu-persatu.

Surabaya, Januari 2017

Muhammad Sonhaji Akbar

(halaman ini sengaja dikosongkan)

DAFTAR ISI

DAFTAR GAMBAR

(halaman ini sengaja dikosongkan)

DAFTAR TABEL

(halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

Pada Bab ini akan dijelaskan mengenai beberapa hal dasar dalam penelitian yang meliputi latar belakang, perumusan masalah, tujuan, manfaat, kontribusi penelitian, dan batasan masalah.

1.1. Latar Belakang

Mengembangkan perangkat lunak yang berkualitas dan kompleks membutuhkan biaya yang tinggi. 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.

Terdapat beberapa penelitian yang telah dikembangkan untuk membangun metode prediksi cacat perangkat lunak tetapi masih kurang optimal seperti *Relational Association Rule Mining* (Gabriela Czibula, 2014) butuh mencari kombinasi data yang sering muncul sehingga memerlukan waktu komputasi yang lama. Selain itu metode *Support Vector Machin*e dapat dikatakan baik tergantung pada dataset yang digunakan, ketika menggunakan data dengan dua *kelas* cocok tapi tidak cocok ketika data yang diolah besar (Karim O. Elish, 2008). Metode lain yaitu *Cost Sensitive Neural Network* dapat digunakan pada data apa saja, tetapi karena data butuh pelatihan terlebih dahulu maka komputasi yang dihasilkan akan lebih lama (Zheng, 2010). Terdapat satu lagi metode yaitu *Neural Network* (NN) yang digabungkan dengan *Particle swarm optimization* (PSO) dan *Association Rule*, dalam metode ini PSO mudah diimplementasikan akan tetapi pengggunakan PSO ini hanya cocok digunakan untuk metode NN (B. Dhanalaxmia, 2015). Beberapa masalah dalam metode-metode tersebut dapat diatasi menggunakan Naive Bayes dengan Pembobotan (Zhang Jiang, 2016), karena tidak membutuhkan waktu komputasi yang lama, lebih efektif, efisien, dapat mereduksi atribut, stabil dan akurasi dapat meningkat. Salah satu pembobotan menggunakan Gain Ratio (Chen, 2008), Gain Ratio dapat memperbaiki data yang tidak stabil, cocok untuk data numeric dua *kelas*, sederhana sehingga komputasi lebih cepat. Terdapat juga metode Naive Bayes dengan optimasi Gain Ratio (Socrates, 2016) tetapi digunakan dalam prediksi teks berita.

Naive Bayes memiliki beberapa kelebihan (Hamzah, 2012), yaitu algoritma yang sederhana, lebih cepat dalam penghitungan dan berakurasi tinggi. Akan tetapi, pada metode Naive Bayes juga memiliki kelemahan (Socrates, 2016) dimana sebuah probabilitas tidak bisa mengukur seberapa besar tingkat keakuratan sebuah prediksi. Dengan kata lain, jika sebuah probabilitas tidak dapat merepresentasikan sebuah data maka prediksi yang dihasilkan kurang akurat. Selain itu, terdapat permasalahan data yang sering muncul pada kelas lain dan muncul juga pada kelas yang diuji mengakibatkan kesalahan prediksi. Hal ini yang menyebabkan metode Naive Bayes masih belum optimal.

Penelitian ini mengusulkan metode optimasi Naive Bayes mennggunakan Gain Ratio untuk meningkatkan meningkatkan akurasi prediksi cacat perangkat lunak. Menurut Zaidi, Serquedes dan Carman (Zaidi, 2013), pembobotan atribut pada Naive Bayes dapat mengurangi dampak kegagalan prediksi. Salah satu metode yang dapat digunakan untuk pembobotan Naive Bayes yaitu Gain Ratio. Gain Ratio digunakan untuk memilih atribut terbaik di antara seluruh atribut pada metode prediksi. Penggunaan Gain Ratio diharapkan dapat meningkatkan akurasi prediksi. Atribut dari Gain Ratio sendiri merupakan hasil bagi dari Information Gain dan Split Information.

1.2. Perumusan Masalah

Hipotesis dalam penelitian ini adalah prediksi cacat perangkat lunak menggunakan metode optimasi Naive Bayes dengan praproses untuk seleksi fitur dari dataset untuk mengatasi masalah kesalahan prediksi cacat perangkat lunak menggunakan Gain Ratio, diharapkan dapat meningkatkan akurasi prediksi cacat perangkat lunak. Metode Naive Bayes Gain Ratio ini diharapkan lebih baik dibanding dengan metode Naive Bayes. Berdasarkan hipotesis tersebut maka rumusan masalah dalam penelitian ini dapat dijabarkan menjadi beberapa butir berikut ini.

1. Bagaimana melakukan prediksi cacat perangkat lunak menggunakan Naive Bayes?
2. Bagaimana melakukan optimasi metode Naive Bayes menggunakan Gain Ratio pada prediksi cacat perangkat lunak?
3. Bagaimana cara melakukan evaluasi dari metode yang diusulkan?

1.3. Tujuan

Tujuan dari tesis ini adalah mengusulkan optimasi metode prediksi cacat perangkat lunak Naive Bayes dengan Gain Ratio. Diharapkan dengan penggabungan dua metode tersebut dapat meningkatkan akurasi prediksi cacat perangkat lunak yang lebih baik daripada metode yang telah ada sebelumnya.

1.4. Manfaat

Penelitian ini diharapkan dapat melakukan prediksi cacat perangkat lunak dengan akurat yang berguna bagi pengembang untuk meminimalisir terjadinya cacat perangkat lunak.

1.5. Kontribusi Penelitian

Kontribusi dalam penelitian ini terkait dengan prediksi cacat perangkat lunak adalah sebagai berikut:

1. Membangun model baru prediksi cacat perangkat lunak dengan menerapkan pendekatan Naive Bayes Gain Ratio yang berguna untuk mendeteksi modul cacat perangkat lunak.
2. Menambahkan langkah praproses dengan menyeleksi fitur yang berpengaruh terhadap munculnya cacat perangkat lunak menggunakan Gain Ratio.

1.6. Batasan Masalah

Batasan dalam penilitian ini adalah sebagai berikut.

1. Penelitian ini menekankan pada mekanisme prediksi cacat perangkat lunak bukan pada proses perbaikan cacat perangkat lunak.
2. Penelitian ini hanya dilakukan untuk prediksi cacat modul atau kode program perangkat lunak.
3. Penelitian ini menekankan pada prediksi cacat perangkat lunak menggunakan metode optimasi metode Naive Bayes dengan Gain Ratio.
4. Dataset cacat perangkat lunak yang digunakan dalam penelitian ini menggunakan dataset NASA MDP yang dapat diakses di situs PROMISE.

(halaman ini sengaja dikosongkan)

BAB II

KAJIAN PUSTAKA DAN DASAR TEORI

Pada Bab 2 ini dijelaskan konsep dasar tentang teori dan kajian pustaka yang digunakan sebagai landasan dalam melakukan penelitian.

2.1. Data Pengujian Cacat Perangkat Lunak Mc Cabe

Dalam proyek pengembangan perangkat lunak selalu muncul risiko adanya modul cacat yang menyebabkan kegagalan di saat perangkat lunak dieksekusi. Beberapa repositori perangkat lunak menyediakan kode proyek perangkat lunak dan data log atau histori munculnya cacat pada sebuah modul berdasarkan nilai metrik kompleksitas kode program. Data Cacat Perangkat Lunak Mc Cabe atau metrik kompleksitas merupakan sebuah perhitungan matematis yang berguna untuk menilai kompleksitas sebuah kode program. Metrik kompleksitas juga berguna untuk mengidentifikasi modul mana dalam sebuah perangkat lunak yang akan sulit untuk dilakukan pengujian atau pemeliharaan (McCabe, 1976). Untuk ke depannya kita dapat menggunakan nilai met tersebut sebagai atribut atau fitur untuk mengklasifikasi modul cacat perangkat lunak.

Tabel 2.3 Metrik Dataset NASA MDP Software Defect

Simbol	Keterangan
loc	McCabe's line count of code
v(g)	McCabe "cyclomatic complexity"
ev(g)	McCabe "essential complexity"
iv(g)	McCabe "design complexity"
n	Halstead total operators + operands
v	Halstead "volume"
l	Halstead "program length"
d	Halstead "difficulty"
i	Halstead "intelligence"
e	Halstead "effort"
b	Halstead "effort"
t	Halstead's time estimator
lOCode	Halstead's line count
lOComment	Halstead's count of lines of comments
lOBlank	Halstead's count of blank lines
lOCodeAndComment	Line of comment and code
uniq_Op	Halstead Unique operators
uniq_Opnd	Halstead unique operands
total_Op	Halstead Total operators
total_Opnd	Halstead Total operands
branchCount	Branch count of the flowgraph
defects	Class {false,true}

Penelitian ini menggunakan metrik yang terdiri dari 4 metrik McCabe (McCabe, 1976), 16 metrik Halstead (Halstead, 1977) dan 1 metrik *branch count*. Metrik-metrik tersebut juga digunakan dalam dataset NASA MDP. Tabel 2.1 menunjukkan metrik yang digunakan dalam dataset NASA MDP. Tabel 2.2 menunjukkan contoh data cacat perangkat lunak berdasarkan metrik kompleksitas.

Tabel 2.4 Contoh data Cacat Perangkat Lunak

loc	v(g)	ev(g)	iv(g)	n	...	Cacat
1,1	1,4	1,4	1,4	1,3	**...**	False
1	1	1	1	1	**...**	False
24	5	1	3	63	**...**	True
20	4	4	2	47	**...**	True
24	6	6	2	72	**...**	True

2.1.1. Metric Mc Cabe

Metrik McCabe (McCabe, 1976) adalah metrik kompleksitas yang diperkenalkan oleh McCabe pada tahun 1976. Terdapat beberapa macam perhitungan metrik yang diajukan oleh McCabe dalam menilai kompleksitas sebuah kode program yaitu: *line of code*, *cyclomatic complexity*, *essential complexity*, dan *design complexity*.

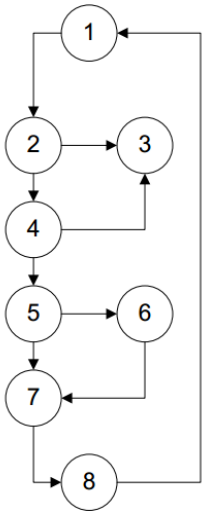
1. Line of Codes (loc)

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

2. Cyclomatic Complexity (v(g))

Metrik v(g) adalah metrik kompleksitas yang dihitung berdasarkan aliran atau *flow* untuk menilai struktur logika keputusan atau *decision logic* sebuah kode program. Logika keputusan dapat berupa statemen *if* atau *looping*. *Flow* sebuah program dapat diilustrasikan sebagai sebuah *graph* yang terdiri dari *edges* dan *vertices*. *Cyclomatic complexity* v(g) dengan n *vertices*, e *edges*, dan p komponen dapat dihitung seperti pada persamaan 2.1

$$v(g) = e-n+p \quad (2.1)$$



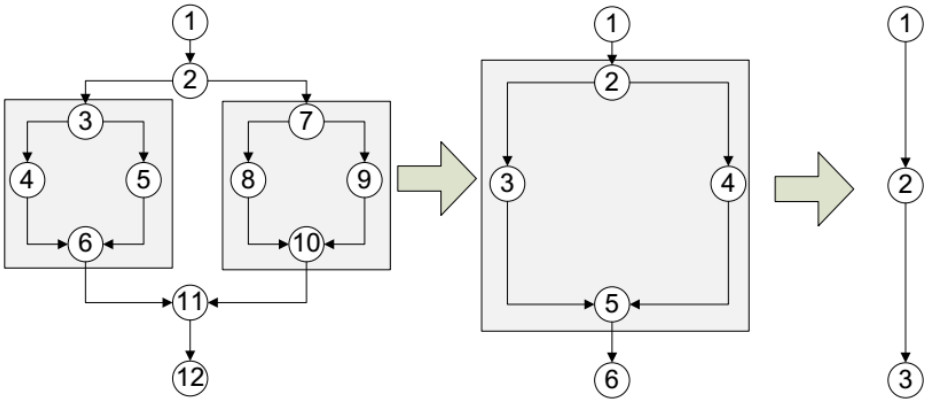
Gambar 2.1. Aliran kode program

Gambar 2.1 merupakan sebuah contoh aliran sebuah kode program yang memiliki nilai v(g) = 10 – 8 + 1 = 3.

3. Essential Complexity (ev(g))

Metrik ev(g) adalah nilai metrik kompleksitas yang digunakan untuk menilai struktur dari sebuah kode program. *Essential complexity* dihitung dengan mengurangi nilai *cyclomatix complexity* v(g) dengan jumlah *subgraph* yang terdiri dari *single entry* dan *single exit*. Rumus dari *essential complexity* dapat dilihat pada persamaan 2.2. Gambar 2.2 menunjukkan contoh perhitungan *essential complexity*. ev(g) = 4 – 3 = 1.

$$ev(g) = v(g) - m \tag{2.2}$$



Gambar 2.2. Contoh Graph Penghitungan Essential

4. Design Complexity iv(g)

Metrik iv(g) merupakan metrik yang digunakan untuk menghitung jumlah logika keputusan atau *decision logic*. Metrik ini mengidikasikan *reliability*, *integrity* dan *testability* sebuah perangkat lunak. *Design complexity* dihitung dengan mengurangi modul pada flowgraph dengan cara menghilangkan *node decisicon* yang tidak memiliki dampak pada kontrol sebuah program.

2.1.2. Metrik Halstead

Metrik Halstead adalah metrik kompleksitas yang diperkenalkan oleh Maurice Howard Halstead (Halstead, 1977). Tujuan dari Halstead adalah mengidentifikasi kompleksitas kode program berdasarkan perhitungan sains.

1. Total Operators dan Operands (n)

Metrik n menghitung jumlah total *operators* (TotalOp) dan *operands* (TotalOpnd) dalam sebuah kode program. Rumus untuk menghitung metrik N dapat dilihat pada persamaan 2.3.

n = TotalOp + TotalOpnd (2.3)

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

Contoh *operators* dalam kode program tersebut adalah `main, (), {}, int, scanf, &, =, +, /, printf`. Sedangkan contoh *operands* adalah `a`, `b`, `c`, `avg`, `"%d %d %d"`, `3`, `"avg = %d"`. Maka n = 16 + 15 = 31.

2. Volume (v)

Metrik v menghitung jumlah bit yang dibutuhkan oleh program dengan total *operators* dan *operands* n. Rumus dari metrik v dapat dilihat pada persamaan 2.4.

v = n * log2(TotalOp + TotalOpnd) (2.4)

3. Program Length (l)

Metrik l merupakan metrik yang digunakan untuk menghitung rasio antara metrik *volume* dengan *total operators and operands*. Rumus metrik L dapat dilihat pada persamaan 2.5.

$$l = v/n \tag{2.5}$$

4. 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 dapat dilihat pada persamaan 2.6.

$$d = l/l \tag{2.6}$$

5. 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 dapat dilihat pada persamaan 2.7.

$$i = v/d \tag{2.7}$$

6. Effort to Write Program (e)

Metrik e adalah metrik yang digunakan untuk mengukut effort atau usaha untuk menulis sebuah kode program. Metrik ini merupakan rasio antara metric volume dan intelligence. Rumus dari metrik E Program dapat dilihat pada persamaan 2.8.

$$e = v/i \tag{2.8}$$

7. Effort Estimate (b)

Metrik b merupakan metrik yang digunakan untuk mengukur *effort* atau usaha yang dibutuhkan oleh *author* untuk memahami kode program. Metrik *effort* merupakan perkalian dari metrik *difficulty* dan *volume*. Rumus metrik B dapat dilihat pada persamaan 2.9.

$$b = d * v \tag{2.9}$$

8. Time Estimator (t)

Metrik T merupakan metrik yang digunakan untuk mengukur perkiraan waktu yang proportional dari setiap *effort*. Rumus dari metrik T dapat dilihat pada persamaan 2.10.

$$t = e/18 \text{ seconds} \quad (2.10)$$

9. Count of Statement Lines (lOCode)

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

10. Count of Lines of Comments (lOComment)

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

11. Count of Blank Lines (lOBlank)

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

12. Count of Code and Comments Lines (lOCodeAndComment)

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

13. Unique Operators (UniqOp)

Metrik UniqOp merupakan metrik yang digunakan untuk menghitung jumlah *operators* yang berbeda dalam sebuah kode program. Metrik *unique operators* disimbolkan sebagai n_1.

14. Unique Operands (UniqOpnd)

Metrik UniqOpnd merupakan metrik yang digunakan untuk menghitung jumlah *operands* yang berbeda dalam sebuah kode program. Metrik *unique operands* disimbolkan sebagai n_2.

15. Total Operators (TotalOp)

Metrik TotalOp merupakan metrik yang digunakan untuk menghitung jumlah jumlah total *operators* dalam sebuah kode program. Metrik TotalOp disimbolkan sebagai N_1.

16. Total Operands (TotalOpnd)

Metrik TotalOpnd merupakan metrik yang digunakan untuk menghitung jumlah total *operands* dalam sebuah kode program. Metrik TotalOpn disimbolkan sebagai N_2.

2.1.3. Metrik Branch Count (branchCount)

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

2.2. Pemilihan Fitur

Tidak semua metrik dari dataset mempengaruhi munculnya cacat pada perangkat lunak. Untuk mengatasi hal ini maka akan dilakukan pemilihan atau seleksi fitur menggunakan metode Gain Ratio Attribute Eval (GR).

2.2.1. Gain Ratio

Pemilihan fitur atau pembobotan dapat dirumuskan menggunakan Gain Ratio (Rafik Khairul Amin, 2015). Dimana dari setiap atribut Gain Ratio dikali jumlah data n kemudian dibagi dengan rata-rata Gain Ratio semua atribut. Parameter yang tepat digunakan untuk mengukur efektifitas suatu atribut dalam melakukan teknik pengklasifikasian sampel data, salah satunya adalah dengan menggunakan information gain.Sebelum mencari nilai gain, terlebih dahulu mencari peluang kemunculan suatu record dalam atribut (entropy) (Rafik Khairul Amin, 2015).

1. Penghitungan Nilai Entropy

Untuk mendapatkan nilai information gain, terlebih dahulu kita harus mengetahui parameter lain yang mempengaruhi nilai gain, dimana parameter ini sangat diperlukan untuk mendapatkan nilai gain. Parameter tersebut adalah entropy. Parameter ini sering digunakan untuk mengukur heterogenitas suatu kumpulan sampel data. Secara matematis nilai entropy dapat dihitung dengan menggunakan persamaan 2.11.

$$Entropy(S) = \sum_{i}^{c} - \ p_i \log 2 p_i \qquad (2.11)$$

C = jumlah nilai yang ada pada atribut target (jumlah kelas)

Pi = jumlah sampel pada kelas i

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

2. Penghitungan *Information Gain*

Ketika kita sudah mendapatkan nilai entropy, maka langkah selanjutnya adalah melakukan perhitungan terhadap information gain.Berdasarkan perhitungan matematis information gain dari suatu atribut A dapat dilihat pada persamaan 2.12.

$$Gain(S, A) = Entropy(S) - \sum_{Vevslution(A)} \frac{|S_v|}{S} entropy(s_v) \qquad (2.12)$$

A : atribut

V : menyatakan suatu nilai yang mungkin untuk atribut A

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

|Sv| : jumlah smpel untuk nilai v

|S| : jumlah seluruh sampel data

Entropy (S) : entropy untuk sampel-sampel yang memiliki nilai v

3. Gain Ratio

Untuk menghitung gain ratio diperlukan split information. Split information dihitung dengan persamaan 2.13.

$$SplitInformation \ = -\sum_{t=1}^{c} \frac{si}{s} \log 2 \frac{si}{s} \qquad (2.13)$$

Dimana S1 sampai Sc adalah c subset yang dihasilkan dari pemecahan S dengan menggunakan atribut A yang mempunyai banyak C nilai. Selanjutnya gain ratio dihitung dengan persamaan 2.14.

$$GainRatio \ = -\frac{Gain(S, A)}{SplitInformation(S, A)} \qquad (2.14)$$

2.3. Permasalahan Kesalahan Prediksi

Penelitian mengenai prediksi cacat perangkat lunak pada metode Naive Bayes juga memiliki kelemahan (Socrates, 2016) dimana sebuah probabilitas tidak bisa mengukur seberapa besar tingkat keakuratan sebuah prediksi sehingga akan terjadi miss prediksi. Dengan kata lain, jika sebuah probabilitas tidak dapat merepresentasikan sebuah data maka prediksi yang dihasilkan kurang akurat. Selain itu, terdapat permasalahan data yang sering muncul pada kelas lain dan muncul juga pada kelas yang diuji mengakibatkan kesalahan prediksi. Hal ini yang menyebabkan metode Naive Bayes masih belum optimal.

2.4. Prediksi Cacat Perangkat Lunak

Dalam tahap ini akan dilakukan mengenali pola cacat perangkat lunak yang selanjutnya akan digunakan untuk melakukan prediksi cacat perangkat lunak. Pengenalan pola sekaligus prediksi ini menggunakan gabungan Naive Bayes dan Gain Ratio.

2.4.1. Naive Bayes Gain Ratio

Naive Bayes adalah metode yang digunakan dalam statistika untuk menghitung peluang dari suatu hipotesis, Naive Bayes menghitung peluang suatu kelas berdasarkan pada atribut yang dimiliki dan menentukan kelas yang memiliki probabilitas paling tinggi. Naive Bayes mengprediksikan kelas berdasarkan pada probabilitas sederhana dengan mangasumsikan bahwa setiap atribut dalam data tersebut bersifat saling terpisah. Metode Naive Bayes merupakan salah satu metode yang banyak digunakan berdasarkan beberapa sifatnya yang sederhana, metode Naive Bayes mengprediksikan data berdasarkan probabilitas P atribut x dari setiap kelas y data. Pada model probablitas setiap kelas k dan jumlah atribut a yang dapat dituliskan seperti persamaan 2.15 berikut.

$$\boldsymbol{P}(\boldsymbol{y}_k \mid \boldsymbol{x}_1, \boldsymbol{x}_2, ... \boldsymbol{x}_a) \tag{2.15}$$

Penghitungan Naive Bayes yaitu probabilitas dari kemunculan dokumen xa pada kategori kelas yk P(xa|yk), dikali dengan probabilitas kategori kelas P(yk). Dari hasil kali tersebut kemudian dilakuan pembagian terhadap probabilitas kemunculan dokumen P(xa). Sehingga didapatkan rumus penghitungan Naive Bayes dituliskan pada persamaan 2.16..

$$\boldsymbol{P}(\boldsymbol{y}_k \mid \boldsymbol{x}_a) = \frac{\boldsymbol{P}(\boldsymbol{y}_k)\boldsymbol{P}(\boldsymbol{x}_a \mid \boldsymbol{y}_k)}{\boldsymbol{P}(\boldsymbol{x}_a)} \tag{2.16}$$

Kemudian dilakukan proses pemilihan kelas yang optimal maka dipilih nilai peluang terbesar dari setiap probabilitas kelas yang ada. Sehingga didapatkan rumus untuk memilih nilai terbesar. Pembobotan Naive Bayes dihitung dengan cara menambahkan bobot Gain Ratio w_i pada setiap atribut. Sehingga didapatkan rumus untuk pembobotan Naive Bayes Gain Ratio dituliskan pada Persamaan 2.17.

$$P(y,x) = P(y)\prod_{i=1}^{q} C * P(x_i \mid y) + w^i \quad (2.17)$$

2.5. Evaluasi Penelitian

Tahap evaluasi bertujuan untuk mengetahui tingkat akurasi dari hasil penggunaan metode Weighted Naive Bayes. Dari evaluasi akan tersedia informasi mengenai seberapa besar akurasi yang telah dicapai. Pada proses pengujian dikenal sebagai Matriks Confusion yang merepresentasikan kebenaran dari sebuah prediksi. Tabel Matriks Confusion dapat dilihat pada Tabel 1.

Tabel 2.5 Penghitungan Prediksi

		Hasil Prediksi	
		+	**-**
Kenyataan	**+**	***True Positive***	***False Positive***
	-	***False Negative***	***True Negative***

- True Positive (TP) menunjukkan bahwa dokumen yang termasuk dalam hasil penge lompokkan oleh sistem memang merupakan anggota kelas.
- False Positive (FP) menunjukkan bahwa dokumen yang termasuk dalam hasil penge-lompokkan oleh sistem ternyata seharusnya bukan merupakan anggota kelas.
- False Negative (FN) menunjukkan bahwa dokumen yang tidak termasuk dalam hasil pengelompokkan oleh sistem ternyata seharusnya merupakan anggota kelas.
- True Negative (TN) menunjukkan bahwa dokumen yang tidak termasuk dalam hasil pengelompokkan oleh sistem ternyata seharusnya bukan merupakan anggota kelas.

Akurasi menunjukkan kedekatan nilai hasil pengukuran dengan nilai sebenarnya. Untuk menentukan tingkat akurasi perlu diketahui nilai sebenarnya dari parameter yang diukur. Akurasi, Precision, Recall dan F-Measures didefinisikan dengan Persamaan 2.19 hingga Persamaan 2.22.

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.19)$$

$$precision = \frac{TP}{TP + FP} \quad (2.20)$$

$$recall = \frac{TP}{TP + FN} \quad (2.21)$$

$$f - measure = \frac{2 * precision * recall}{precision + recall} \quad (2.22)$$

2.6. Dataset Penelitian

Dataset yang digunakan dalam penelitian ini adalah histori cacat perangkat lunak yang berasal dari NASA Metric Data Program (MDP). Dataset perangkat lunak yang pernah dibuat akan digunakan untuk memprediksi terdapat cacat atau tidak pada perangkat lunak sebelum dibuat yang mirip dengan sebelumnya. Di dalam dataset NASA MDP terdapat beberapa studi kasus cacat perangkat lunak. Berikut contoh data set dan atributnya:

2.6.1. Dataset CM1

1. Jenis dataset : CM1/software defect prediction
2. Jumlah instan : 498
3. Jumlah atribut : 22 (5 different lines of code measure, 3 McCabe metrics, 4 base Halstead measures, 8 derived Halstead measures, a branch-count, and 1 goal field)
4. Contoh Dataset :

Tabel 2.6. Dataset CM1

No	loc	b	IOComment	Uniq_Op	Uniq_Opnd	...	Defect
1	1.1	1.3	2.0	1.2	1.2	...	FALSE
2	1.0	1.0	1.0	1.0	1.0	...	TRUE
3	24.0	0.1	0.0	15.0	15.0	...	FALSE
4	20.0	0.07	0.0	16.0	8.0	...	FALSE
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
498	6.0	0.02	1.0	8.0	5.0	...	FALSE

2.6.2. Dataset JM1

1. Jenis dataset : JM1/software defect prediction
2. Jumlah instan : 10885
3. Jumlah atribut : 22 (5 different lines of code measure, 3 McCabe metrics, 4 base Halstead measures, 8 derived Halstead measures, a branch-count, and 1 goal field)
4. Contoh Dataset :

Tabel 2.7. Dataset JM1

No	loc	B	IOComment	Uniq_Op	Uniq_Opnd	...	Defect
1	1.1	1.4	1.3	2.0	1.4	...	FALSE
2	1.0	1.0	1.0	1.0	1.0	...	TRUE
3	72.0	7.0	198.0	51.0	13.0	...	TRUE
4	190.0	3.0	600.0	129.0	5.0	...	TRUE
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
10885	85.0	9.0	277.0	69.0	13.0	...	TRUE

2.6.3. Dataset KC1

1. Jenis dataset : KC1/software defect prediction
2. Jumlah instan : 2109
3. Jumlah atribut : 22 (5 different lines of code measure, 3 McCabe metrics, 4 base Halstead measures, 8 derived Halstead measures, a branch-count, and 1 goal field)
4. Contoh Dataset :

Tabel 2.8. Dataset KC1

No	loc	b	IOComment	Uniq_Op	Uniq_Opnd	...	Defect
1	1.4	1.3	1.3	1.2	1.4	...	FALSE
2	1.0	1.0	1.0	1.0	1.0	...	TRUE
3	11.0	927.89	23.04	25.0	21.0	...	TRUE
4	8.0	769.78	14.86	28.0	15.0	...	TRUE
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
20	1.0	85.84	3.33	12.0	1.0	...	TRUE

2.6.4. Dataset KC2

1. Jenis dataset : KC2/software defect prediction
2. Jumlah instan : 522
3. Jumlah atribut : 22 (5 different lines of code measure, 3 McCabe metrics, 4 base Halstead measures, 8 derived Halstead measures, a branch-count, and 1 goal field)
4. Contoh Dataset :

Tabel 2.9. Dataset KC2

No	loc	b	IOComment	Uniq_Op	Uniq_Opnd	...	Defect
1	1.1	1.4	1.3	1.3	1.3	...	no
2	1.0	1.0	1.0	1.0	1.0	...	yes
3	415.0	50.0	8411.31	103.53	2.8	...	yes
4	230.0	10.0	3732.82	39.82	1.24	...	yes
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
20	9.0	1.0	87.57	4.0	0.03	...	no

2.6.5. Dataset PC1

1. Jenis dataset : PC1/software defect prediction
2. Jumlah instan : 1109
3. Jumlah atribut : 22 (5 different lines of code measure, 3 McCabe metrics, 4 base Halstead measures, 8 derived Halstead measures, a branch-count, and 1 goal field)
4. Contoh Dataset :

Tabel 2.10. Dataset PC1

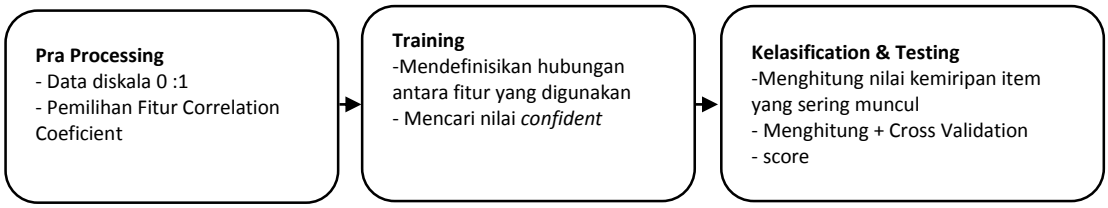
No	loc	b	IOComment	Uniq_Op	Uniq_Opnd	...	Defect
1	1.3	2.0	2.0	2.0	1.2	...	FALSE
2	1.0	1.0	1.0	1.0	1.0	...	TRUE
3	75.47	44.0	11.0	31.0	66.0	...	TRUE
4	89.79	41.0	12.0	24.0	75.0	...	TRUE
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
20	43.78	27.0	11.0	23.0	38.0	...	TRUE

2.7. Penelitian Terkait Metode Prediksi Cacat Perangkat Lunak

Cacat perangkat lunak (Software Defect) didefinisikan sebagai cacat pada perangkat lunak seperti cacat pada dokumentasi, pada kode program, pada desain dan hal – hal lain yang menyebabkan kegagalan perangkat lunak. Penelitian ini hanya melakukan prediksi cacat perangkat lunak pada modul atau kode programmnya saja. 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 aksi yang mungkin menghasilkan cacat perangkat lunak pada proses pengembangan perangkat lunak (Boehm, 2001).

2.7.1. Metode *Relational Association Rule Mining (RARM)*

Dalam melakukan prediksi cacat perangkat lunak, metode RARM ini melakukan kombinasi data yang paling sering muncul, mendefinisikan kondisi data yang telah dikombinasi sehingga membutuhkan waktu komputasi yang lebih lama (Gabriela Czibula, 2014).



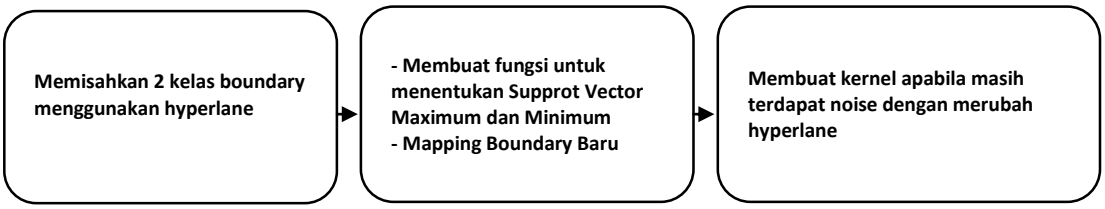
Gambar 2.3. Alur Metode RARM

Langkah pertama metode RARM ini adalah melakukan skala data 0:1 dan melakukakn pemilihan fitur menggunakan Koefisien Korelasi. Setelah itu, dilakukan pelatihan dengan mendefinisikan hubungan antara fitur-fitur yang digunakan dan mencari nilai koefifisiennya. Terakhir melakukan prediksi dan pengujian dengan menghitung nilai kemirian item yang sering muncul, menghitungnya serta menggunakan Penilaian Validasi Silang. Alur metode RARM dapat dilihat pada Gambar 2.3.

2.7.2. *Support Vector Machine (SVM)*

Dalam melakukan prediksi cacat perangkat lunak, metode SVM ini memliki kelebihan yaitu ketika karakteristik data yang diolah tepat maka metode ini akan berjalan dengan optimal, penggunaan data dalam metode ini cocok untuk data dengan kelas berjumlah dua. Metode ini

juga memiliki kelemahan yaitu ketika skala masalah terlalu besar atau jumlah data yang diolah besar maka komputasi yang dibutuhkan cukup lama (Karim O. Elish, 2008).

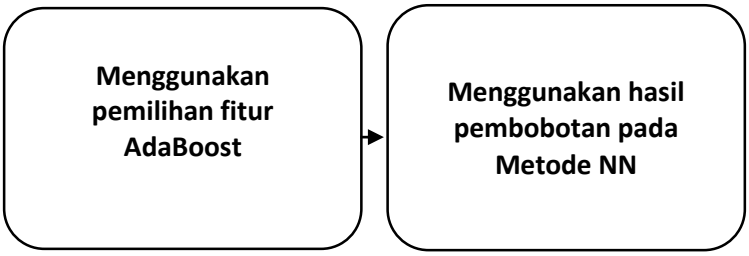


Gambar 2.4. Alur Metode SVM

Berdasarkan Gambar 2.4. langkah pertama dalam metode SVM untuk Prediksi Cacat Perangkat Lunak ini adalah memisahkan dua kelas data menjadi dua batas menggunakan *hyperlane* khas metode SVM. Selanjutnya, membuat fungsi untuk menentukan nilai maksimum dan minimum titik item partikel sehingga dapat dilakukakan pemetaan dan membentuk batas yang baru. Terakhir membuat kernel untuk mengoptimalkan hasil batas yang baru apabila masih terdapat item yang diluar prediksi dengan merubah *hyperlane.*

2.7.3. *Cost Sensitive Neural Network (NN)*

Dalam melakukan prediksi cacat perangkat lunak, metode NN ini dapat digunakan untuk jenis data apa saja, akan tetapi data yang akan diolah membutuhkan proses pelatihan terlebih dahulu. Sama halnya ketika membutuhkan proses yang lebih banyak maka waktu komputasi metode ini juga cukup memakan waktu yang lama (Zheng, 2010).

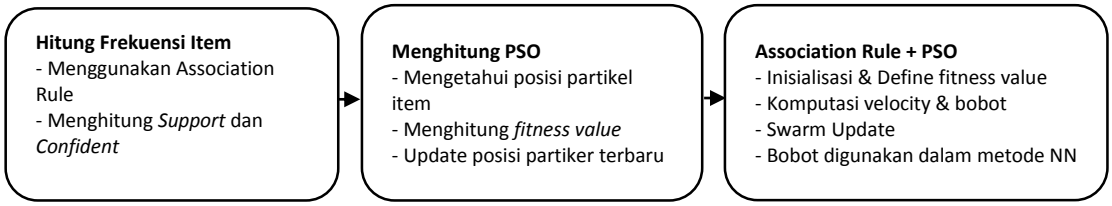


Gambar 2.5. Alur Metode NN

Dalam metode NN ini langkah yang dilakukan seperti metode NN pada umumnya, tetapi pada metode ini, pertama dilakukan pemilihan fitur menggunakan Adabost, selanjutnya diolah menggunakan metode NN dan dilakukan pembobotan sehingga dapat menghasilkan hasil prediksi. Dapat dilihat pada Gambar 2.5.

2.7.4. *Particle Swarm Optimization with Association Rule Mining for ANN*

Dalam melakukan prediksi cacat perakngkat lunak, metode PSO ARM ANN ini mudah diimplementasikan. Karakter dari metode PSO tidak membutuhkan banyak parameter dalam mengolah data maka metode ini mudah diimplementasikan untuk data apa saja. Metode ini lebih efisien atau lebih cepat dalam hal komputasi, tetapi metode ini hanya cocok ketika metode yang digunakan adalah metode NN (B. Dhanalaxmia, 2015).



Gambar 2.6. Alur Metode APSO ANN

Berdasarkan Gambar 2.6. Langkah pertama metode APSO ANN yaitu menghitung Frekuensi item menggunakan *Assoiciation Rule* khususnya menghitung *Support* dan *Confident.* Selanjutnya menghtung PSO yang sebelumnya sudah diketahui posisi partikel item yang akan diolah, lalu menghitung *fitness value* dan melakukan update posisi partikel yang baru. Terakhir yaitu melakukan komputasi velocity dan bobot sehingga didapatkan swarm update dan bobot yang digunakan pada metode NN, setelah itu diolah menggunakan NN dan dapat di prediksikan.

2.7.5. Pembobotan *Naive Bayes* menggunakan Gain Ratio untuk *Text Classification*

Dalam melakukan prediksi, metode Naive Bayes dengan Gain Ratio ini memperbaiki data yang tidak stabil, cocok digunakan untuk data numerik dengan dua kelas, sederhana sehinggga waktu komputasi lebih cepat. kombinasi data yang paling sering muncul, mendefinisikan kondisi data yang telah dikombinasi sehingga membutuhkan waktu komputasi yang lebih lama. Selain itu metode ini lebih efektif, efisien karena dapat mereduksi fitur atau atribut yang ada tetapi tetap stabil sehingga akurasi yang didapatkan dapat meningkat dan waktu komputasi lebih cepat (Zhang Jiang, 2016) (Chen, 2008).

Gambar 2.7. Alur Metode NB GR Teks Berita

Pada Gambar 2.7 terdapat langkah-lagkah metode NB GR pada data teks berita. Langkah pertama yang dilakukan adalah melakukakn pengolahan data teknya menggunakan TF-IDF. Selanjutnya melakukan pembobotan Gain Ratio pada metode Naive Bayes dan didapatkan hasil prediksinya.

2.7.6. Penelitian Prediksi Cacat Perangkat Lunak pada Metode Naive Bayes, Decision Tree dan Neural Network

Topik bahasan yang sering dilakukakn peneliti yaitu prediksi, dataset yang digunakan adalah public dataset spesisik menggunakan Software Defect NASA Dataset, metode yang paling sering digunakan adalah Neural Network dan Naïve Bayes. Terakhir improvement dan akurasi yang sering paling baik dalam Prediksi cacat perangkat lunak adalah Gabungan Metode PSO Rule Mining dan ANN dengan tingkat akurasi mencapai 94%. Lebih rincinya dapat dilihat pada Tabel 2.9.

Tabel 2.11 Hasil Akhir

RQ	*Research Question*	Jawaban
1	Topik	Prediksi
2	Dataset	Software Defect NASA Dataset
3	Metode	Neural Network dan Naïve Bayes
4	Imprrovement Metode dan Akurasi	Gabungan PSO, Rule Mining dan ANN

Dari hasil analisis untuk penelitian selanjutnya beberapa hal yang harus dapat diperhatikan dalam Prediksi cacat perangkat lunak atau Prediksi Cacat Perangkat Lunak: Topik yang sedang berkembang yaitu tentang Prediksi, hampir semua penelitian Prediksi Cacat Perangkat Lunak menggunakan dataset public dataset yaitu Software Defect NASA Dataset. Selanjutnya metode yang paling sering digunakan karena efektif dalam penggunaannya yaitu Neural Network dan Naive Bayes, dengan improvement atau optimasi metode menggunakan PSO dan Rule Mining. Dengan adanya hal ini maka masih dapat dilakukan improvement lain dalam mengoptimasi metode-metode yang sudah ada.

Tabel 2.12. Ringkasan Hasil Studi Literatur

	1 (Czibula, 2014)	2 (Elish, 2008)	3 (Zheng, 2010)	4 (Dhanalaxmi, 2015)
Metode Prediksi	Relational Association Rule Mining	Support Vector Machine	Cost Sensitive Neural Network	PSO + Association Rule Mining + NN
Dataset	NASA MDP (KC1, KC2, CM1, PC1, dan JM1)	NASA MDP (CM1, PC1, KC1 and KC3)	NASA MDP (CM1, KC1, KC2, PC1, dan JM1)	NASA MDP (CM1, KC1, KC2, PC1, dan JM1)
Mekanisme Umum	Mencari kombinasi data yang paling sering muncul, mendefinisikan kombinasi data yang telah ada.	Metode ini melakukan menemukan hyperplane terbaik yang memisahkan dua buah kelas pada *inputspace*	Menggunakan metode yang menyerupai jaringan saraf manusia yang terdapat banyak layer atau neuron tergantung pada kompleksitas data yang akan diolah	Metode NN dengan jaringan sarafnya dioptimasi menggunakan PSO dengan cara menggerakan partikel calon solusi di dalam ruang permasalahan menggunakan fungsi tertentu untuk posisi dan kecepatan dari partikel.
Hasil Evaluasi	Akurasi (Acc), Probablity of False Alarm (Pf).	Specifity, Presisi, Sensitivitas.	Sensitivitas, Probability of False Alarm (Pf)	Akurasi
Kelebihan	Mencari kombinasi data sehingga hasil prediksi akan akurat	Karakteristik data yang digunakan sesuai, cocok digunakan untu data dua kelas	Metode ini dapat digunakan pada data apa saja. Fleksibel.	Metode PSO ini mudah diimplementasikan karena parameter yang dimiliki hanya sedikit. Waktu komputasi lebih cepat.
Kekurangan	Peluang kombinasi kemiripan pada item membutuhkan waktu yang lama karena butuh banyak kombinasi. **Akurasi masih rendah**	Dibutuhkan pembuatan hyperlane dan kernel agar item terprediksi, pembuatan keduanya membutuhkan waktu yang cukup lama. **Akurasi masih rendah**	Sulit dalam melakukan pemodelan menggunakan jaringan syaraf, berkaibat pada sulitnya mendapatkan **hasil prediksi yang akurat**. **Akurasi masih rendah**	Metode PSO lebih cocok digunakan untuk optimasi metode NN karena item berupa partikel. Karena gabungan tiga metode maka waktu komputasi lama. **Akurasi masih rendah**

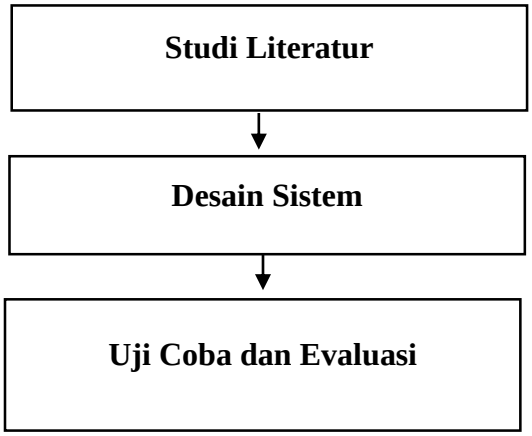
2.8. Metode yang diusulkan

Beberapa metode yang pernah dilakukan dalam prediksi cacat perangkat lunak masih memliki kelemahan diantaranya: Pertama, pada metode RARM Peluang kombinasi kemiripan pada item membutuhkan waktu yang lama karena butuh banyak kombinasi. Kedua, pada metode SVM masih dibutuhkan pembuatan hyperlane dan kernel agar item terprediksi, pembuatan keduanya membutuhkan waktu yang cukup lama. Ketiga, metode NN dalam kasus ini masih sulit dalam melakukakan pemodelan menggunakan jaringan syraraf, berakibat pada sulitnya mendapatkan hasil prediksi yang akurat. Terakhir, gabungan tiga metode PSO AR NN, Metode PSO lebih cocok digunakan untuk optimasi metode NN karena item berupa partikel. Terdapat satu metode gabungan anatara Naive Bayes dengan Gain Ratio yang sebelumnya pernah diterapkan pada data teks dengan melalkukan pengolahan TF-IDF terlebih dahulu. Metode ini mudah, simple dan cocok digunakan untuk karakteristik data numeric dengan dua kelas dengan hasil yang lebih akurat.

(halaman ini sengaja dikosongkan)

BAB III
METODE PENELITIAN

Pada Bab ini akan dijelaskan Metodologi Penelitian. Langkah-langkah yang akan dilakukan pada penelitian ini dapatr dilihat pada Gambar 3.1.

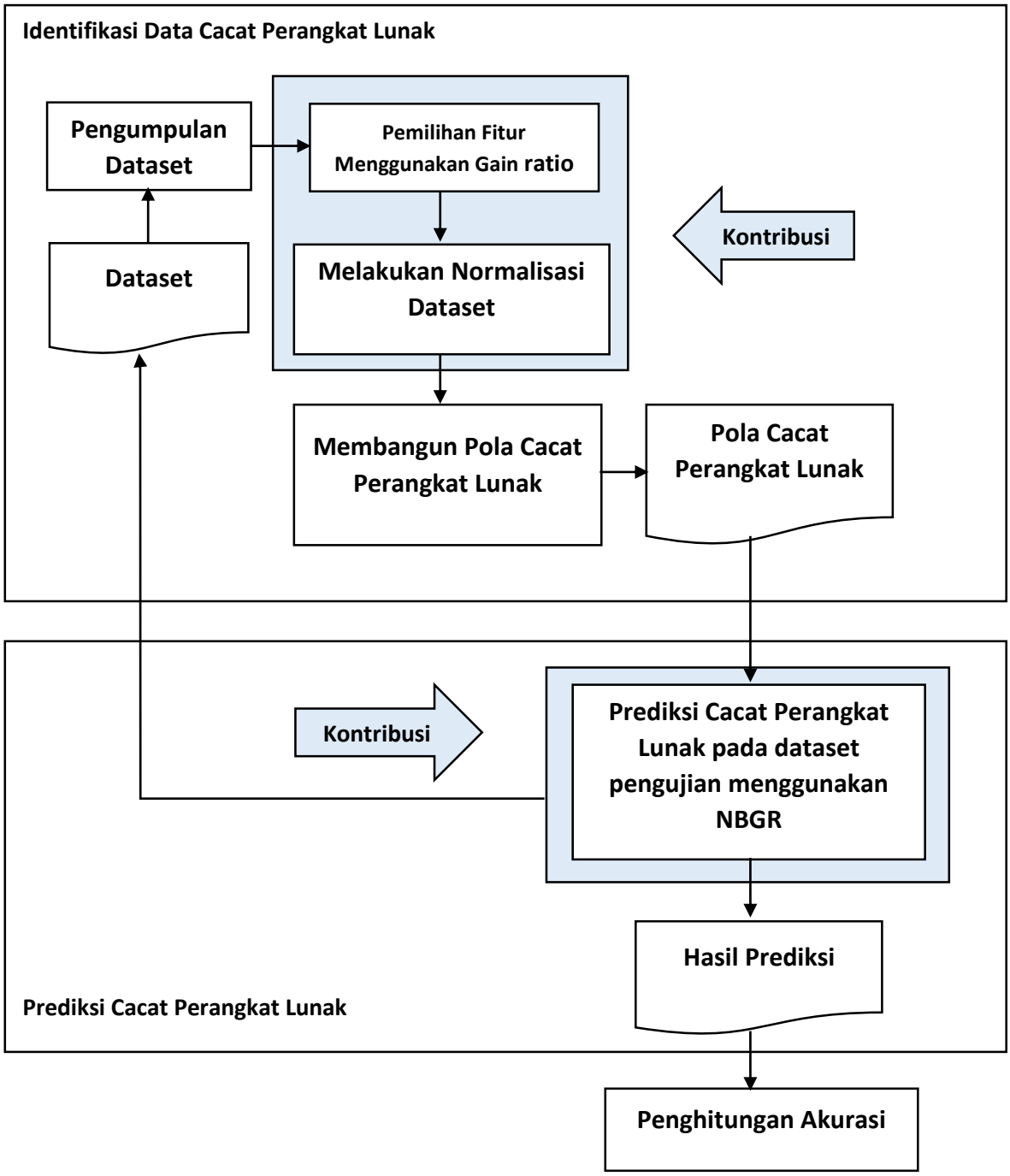


Gambar 3.1 Tahapan Metodologi Penelitian

3.1. Studi Literatur

Pada tahap studi litelatur, dikaji berbagai referensi yang berkaitan dengan klasifikasi cacat perangkat lunak, baik dalam perkembangan pendekatan metode yang telah digunakan serta informasi tentang kelebihan dan kekurangan dari masing – masing penelitian sebelumnya. Studi literatur yang dilakukan diharapkan dapat memberikan gambaran secara lengkap dan dapat memberikan dasar kontribusi tentang identifikasi pola dan prediksi cacat perangkat lunak yang akan dilakukan pada penelitian ini. Studi literatur yang dilakukan adalah mempelajari beberapa referensi sebagai berikut:

1. Penelitian terdahulu yang telah melakukan klasifikasi cacat perangkat lunak berdasarkan dataset kode program.
2. Dasar teori tentang dataset metrik kompleksitas kode program.
3. Metode pemilihan Fitur Gain Ratio (GR).
4. Metode prediksi Naive Bayes Gain Ratio (NBGR).



Gambar 3.2. Desain Sistem Metode yang Diusulkan

3.2. Desain Sistem

Pada subbab ini akan dibahas perancangan dari solusi yang diusulkan. Fitur utama yang disediakan dari solusi ini adalah sebagai berikut:

1. Kemampuan untuk mengidentifikasi pola cacat perangkat lunak berdasarkan nilai metrik kompleksitas.

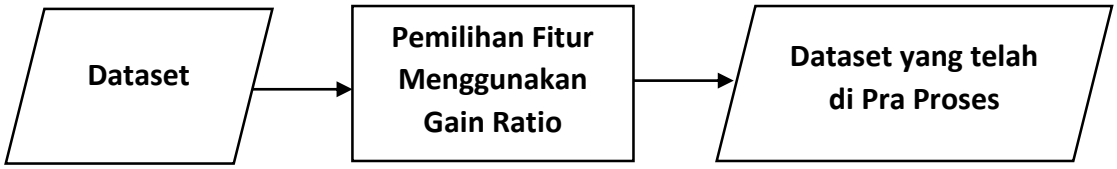
2. Kemampuan melakukan prediksi kemunculan cacat perangkat lunak berdasarkan metrik kompleksitas.
3. Secara umum desain sistem yang diajukan dapat dilihat pada Gambar 3.2

3.2.1. Pengumpulan Dataset

Pada tahapan ini dilakukan untuk mengambil dataset pada repositori PROMISE NASA yang menyimpan 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.

3.2.2. Praproses

Pada tahap praproses dataset cacat perangkat lunak yang akan dilakukan uji coba terlebih dahulu dilakukan seleksi fitur metrik menggunakan metode Gain Ratio. Gambar 3.3 menunjukkan alur pra proses dataset.



Gambar 3.3. Alur Pra Proses

3.2.2.1.Pemilihan Fitur Menggunakan Gain Ratio

Pada tahap ini dilakukan pemilihan fitur dataset metrik kompleksitas menggunakan Gain Ratio. Dari 21 data metrik kompleksitas dipilih beberapa metrik atau atribut yang memiliki ranking bobot tertinggi atau terbesar dengan kelas cacat perangkat lunak. Penghitungan Gain Ratio telah dijelaskan pada sub bab 2.3.1. Penghitungan Gain Ratio dilakukan dengan menghitung bobot sesuai dengan rumus Gain Ratio. Urutan atribut akan menjadi peringkat dan akan diambil lima atribut terbaik. Tabel 3.1. bobot gain ratio yang diperingkat, sebelum terseleksi berdasarkan dataset salah satu studi kasus yaitu CM1. Pada Tabel 3.2. ditampilkan tabel dataset yang terpiluh dan akan digunakan pada proses prediksi cacat perangkat lunak.

Tabel 3.1. Ilustrasi Bobot Gain Ratio Dataset CM1

Bobot	Ranked attributes
0.0574	**14 lOComment**
0.0531	**17 uniq_Op**
0.0526	**18 uniq_Opnd**
0.052	**11 b**
0.0515	**1 loc**
0.0466	9 i
0.0455	6 v
0.0453	4 iv(g)
0.0447	5 n
0.0437	19 total_Op
0.0408	15 lOBlank
0.0393	10 e
0.0393	12 t
0.0363	20 total_Opnd
0.0319	8 d
0.0306	7 l
0.0282	21 branchCount
0.0275	2 v(g)
0.0203	13 lOCode
0	3 ev(g)
0	16 locCodeAndComment
Selected attributes: 14,17,18,11,1	

Tabel 3.2. Ilustrasi Dataset Hasil Pemilihan Fitur

Atribut/ Metrik
lOComment
uniq_Op
uniq_Opnd
b
loc

Tabel 3.1 dan 3.2 menunjukkan hasil pemilihan fitur untuk dataset CM1 menggunakan Gain Ratio. Jumlah fitur yang akan digunakan tergantung dataset yang digunakan. Selanjutnya akan dilakukakn prediksi menggunakan atribut yang telah terpilih tersebut.

3.2.2.2.Normalisasi Dataset

Tipe data yang digunakan adalah kategorikal. Sebelumnya data masih berupa numerik dan harus dirubah menjadi kategorikal terlebih dahulu seperti yang telah dijelaskan pada bab 2 Tabel 3.3 menunjukkan nilai metrik kompleksitas yang belum diubah ke dalam data yang akan diolah pada Naïve Bayes Gain Ratio dalam setiap modul. Nilai kategori 1, 2, 3, 4, 5 pada Tabel 3.3 dan Tabel 3.4. masing-masing merupakan nilai minimum hingga maksimum data setiap tipe metrik.

Tabel 3.3. Nilai Data Sebelum dinormalisasi

ID	**loc**	**b**	**lOComment**	**uniq_Op**	**uniq_Opnd**	**Defect**
1	1,1	1,3	2	1,2	1,2	FALSE
2	1	1	1	1	1	TRUE
3	24	0,1	0	15	15	FALSE
4	20	0,07	0	16	8	FALSE
5	24	0,12	0	16	12	FALSE
6	24	0,12	0	16	12	FALSE
7	7	0,01	0	4	5	FALSE
8	12	0,03	0	10	7	FALSE
9	25	0,18	16	15	20	FALSE
10	46	0,45	35	15	37	FALSE
…	**…**	**…**	**…**	**…**	**…**	**…**

Tabel 3.4. Nilai Minimum Maximum Dan Interval Kategori

ID	**MIN**	**MAX**	**Kategori**				
			1	**2**	**3**	**4**	**5**
Atribut 1	1	423	0-80	81-160	161-240	241-320	321-423
Atribut 2	0	5,7	0-1	1,1-2,0	2,1-3,0	3,1-4,0	4,1-5,7
Atribut 3	0	339	0-60	61-120	121-180	181-240	241-339
Atribut 4	1	72	0-15	16-30	31-45	46-50	51-72
Atribut 5	0	314	0-60	61-120	121-180	181-240	241-314

Pada Tabel 3.4 menampilkan nilai minimum dan maksimum dari masing-masing atribut data yang digunakan. Pada atribut 1 nilai minimumnya adalah 1 dan nilai maksimumnya 423, setelah itu langsung dibagi menjadi lima kategori. Atribut 1 kategori 1 dengan interval mulai dari 0 hingga 80, kategori 2 mulai 81 hingga 160, kategori 3 mulai 161 hingga 240, kategori 4 mulai 241 hingga 320, kategori 5 mulai 321 hingga 423. Pada atribut 2 nilai minimumnya adalah 0 dan nilai maksimumnya 5,7, setelah itu langsung dibagi menjadi lima kategori. Atribut 1 kategori 1 dengan interval mulai dari 0 hingga 1, kategori 2 mulai 1,1 hingga 2, kategori 3 mulai 2,1 hingga 3, kategori 4 mulai 3,1 hingga 4, kategori 5 mulai 4,1 hingga 5,7. Dan sama untuk membaca data yang lainnya.

3.2.2.3.Mengidentifikasi Pola Cacat Perangkat Lunak

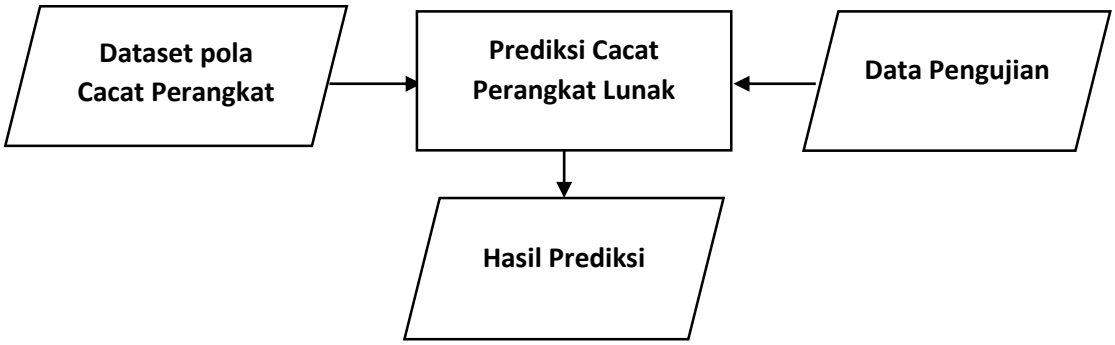
Pada langkah ini akan dilakukan pembangunan pola cacat perangkat lunak atau dolakukan normalisasi data berdasarkan metrik kompleksitas yang berguna untuk melakukan prediksi pada tahap pengujian. Hasil Normalisasi data hingga dapat diolah dalam proses prediksi dapat dilihat pada Tabel 3.5.

Tabel 3.5. Nilai Data Setelah dinormalisasi

ID	**loc**	**b**	**lOComment**	**uniq_Op**	**uniq_Opnd**	**Defect**
1	1	2	1	1	1	FALSE
2	1	1	1	1	1	TRUE
3	1	1	1	1	1	FALSE
4	1	1	1	2	1	FALSE
5	1	1	1	2	1	FALSE
6	1	1	1	2	1	FALSE
7	1	1	1	1	1	FALSE
8	1	1	1	1	1	FALSE
9	1	1	1	1	1	FALSE
10	1	1	1	1	1	FALSE
…	**…**	**…**	**…**	**…**	**…**	**…**

3.2.3. Prediksi Cacat Perangkat Lunak

Prediksi cacat perangkat lunak adalah proses yang dilakukan untuk melakukan prediksi cacat perangkat lunak pada dataset testing berdasarkan pola-pola kemunculan cacat perangkat lunak yang telah didapatkan pada proses sebelumnya. Alur klasifikasi cacat perangkat lunak dapat dilihat pada Gambar 3.6.



Gambar 3.4. Alur Prediksi Cacat Perangkat Lunak

Tahap prediksi dilakukan dengan memilih dan menyeleski fitur terbaik dari dataset terlebih dahulu dan mengubah bentuk data menjadi lima kategori. Setelah itu dilakukakan prediksi dengan dataset yang terpilih dengan dataset pengujian dan akan muncul hasil prediksi.

Sebelum melakukan penghitungan menggunakan rumus Naive Bayes Gain Ratio dilakukakn penghitungan jumlah kemunculan pada masing-masing kelas. Salah satu contoh penghitungan kemunculan dapat dilihat pada Tabel 3.6. Pada Tabel 3.7 ditampilkan jumlah kelas cacat maupun tidak cacat.

Tabel 3.6. Kemunculan tiap kelas

data	CACAT	TIDAK CACAT
1	161	1975
2	63	220
3	13	39
4	3	4
5	5	7

Tabel 3.7. Jumlah Detail Kelas

CACAT	49
TIDAK CACAT	449

Penghitungan Naive Bayes langkah pertama yaitu hitung probabilitias kelas pertama terhadap lima atribut terpilih. Dapat dilihat pada persamaan 3.1.

$$\boldsymbol{P}(\boldsymbol{y_k} \mid \boldsymbol{x_1}, \boldsymbol{x_2}, \ldots . \boldsymbol{x_a}) \tag{3.1}$$

$$P(y, x) = 63/49 \tag{3.2}$$

Tabel 3.8. Ilustrasi penghitungan Probabilitas Class Cacat

p(x1=2,y=cacat)
p(x2=3,y=cacat)
p(x3=4,y=cacat)
p(x4=5,y=cacat)
p(x5=1,y=cacat)

Tabel 3.9.13 Ilustrasi penghitungan Probabilitas Class Tidak Cacat

P(X1=2|Y=tidakcacat)
P(X2=3|Y=tidakcacat)
P(X3=4|Y=tidakcacat)
P(X4=5|Y=tidakcacat)
P(X5=1|Y=tidakcacat)

Setelah didapatkan Probabilitas masing-masing atribut terhadap kelas masing-masing, selanjutnya dilakukan Penghitungan Gain Ratio untuk digunakan menjadi bobot dari masing-masing atribut. Pertama menghitung Information Gain. Penghitungan Information Gain dapat dilihat pada Tabel. 3.10. Untuk mendapatkan nilai information gain, terlebih dahulu kita harus mengetahui parameter lain yang mempengaruhi nilai gain, dimana parameter ini sangat diperlukan untuk mendapatkan nilai gain. Parameter tersebut adalah entropy. Parameter ini sering digunakan untuk mengukur heterogenitas suatu kumpulan sampel data. Ilustrasi penghitungan nilai entropy dapat dilihat pada persamaan 3.3.

$$Entropy(S) = \sum_{i}^{c} - \ p_i \log 2 p_i \qquad (3.3)$$

Ketika kita sudah mendapatkan nilai entropy, maka langkah selanjutnya adalah melakukan perhitungan terhadap information gain. Berdasarkan perhitungan matematis information gain dari suatu atribut A dapat diformulasikan sebagai berikut :

(3.4)

Untuk menghitung Gain Ratio diperlukan split information. Split information dihitung dengan formula sebagai berikut:

$$SplitInformation \ = -\sum_{t=1}^{c} \frac{si}{s} \log 2 \frac{si}{s} \qquad (3.5)$$

Dimana S1 sampai Sc adalah c subset yang dihasilkan dari pemecahan S dengan menggunakan atribut A yang mempunyai banyak C nilai. Selanjutnya gain ratio dihitung dengan cara sebagai berikut:

$$GainRatio \ = -\frac{Gain(S, A)}{SplitInformation(S, A)} \qquad (3.6)$$

Hasil dari penghitungan Gain Ratio menghasilkan bobot yang akan berguna dalam pembobotan pada Naive Bayes Gain Ratio. Nilai Gain Ratio dapat dilihat pada Tabel 10.

Tabel 3.10. Nilai Bobot Gain Ratio yang didapatkan

Atribut	**Gain Ratio**
loc	0,0574
b	0,0531
lOComment	0,0526
uniq_Op	0,052
uniq_Opnd	0,0515

Setelah didapatkan nilai Gain Ratio selanjutnya hitung nilai Naive Bayes Gain Ratio masing-masing kelas. Penghitungan Naive Bayes Gain Ratio dapat dilihat pada persamaan 3.7.

$$P(y, x) = 1{,}28 + 0{,}0574 \qquad (3.7)$$

Setelah Probabilitas masing-masing kelas terhadap masing-masing atribut didapatkan maka selanjutnya menentukan termasuk kelas Cacat atau Tidak Cacat. Penentuan Cacat atau Tidak Cacat-nya data dapat dilihat pada Tabel 3.11. Pada persamaan 3.8 dan persamaan 3.9

dapat dilahat mana yang lebih besar yaitu yang dominan sehingga pada ilustrasi tersebut data termasuk pada kelas x1 atau Cacat.

P(y,x1) = 11,8 (3.8)

P(y,x2) = 0,5 (3.9)

Selanjutnya dilakukan penghitungan atau Evaluasi prediksi menggunakan akurasi, precison, recall dan f-measure.

3.3. Penghitungan Akurasi / Evaluasi

Proses yang dilakukan dalam tahap ini adalah menguji akurasi pendekatan yang diajukan dalam melakukan prediksi cacat perangkat lunak. Seperti yang telah dijelaskan pada bab 2 akurasi yang akan diukur meliputi precision, recall, f-measure, dan akurasi dari cacat perangkat lunak. Contoh penghitungan akurasi, precision, recall, f-measure seperti Tabel 3.11 dan persamaan 3.8. hingga persamaan 3.11.

Tabel 3.11. Matrik Akurasi

		Fakta	
		CACAT	**TIDAK CACAT**
Sistem	**CACAT**	442	7
	TIDAK CACAT	49	0

$$precision = \frac{442}{442 + 7} = 0{,}90 \quad (3.8)$$

$$recall = \frac{442}{442 + 49} = 0{,}95 \quad (3.9)$$

$$f - measure = \frac{2 * 0{,}90 * 0{,}95}{0{,}90 + 0{,}95} = 0{,}94 \quad (3.10)$$

$$Akurasi = \frac{442 + 0}{442 + 0 + 7 + 49} = 0{,}88 \quad (3.11)$$

Berdasarkan Tabel 3.11. dapat dilihat nilai *True Positive (TP)* 442, nilai False Positive (FP) 7, nilai False Negative (FN) 49 dan nilai True Negative (TN) 0. Sehingga dapat dihitumg nilai precision pada persamaan 3.8 sebesar 0,90. Nilai Recall pada persamaan 3.9 sebesar 0,95. Nilai f-measure pada persamaan 3.10 sebesar 0,94 dan nilai akurasi pada persamaan 3.11 sebesar 0,88 atau 88%. Nilai precision, recall, f-measure dan akurasi inilah yang akan dibandingkan dengan hasil dari metode-metode lain pada prediksi cacat perangkat lunak.

(halaman ini sengaja dikosongkan)

BAB IV

PENGUJIAN DAN EVALUASI

Pada bab ini akan dibahas tentang evaluasi hasil pengujian. Pembahasan pertama adalah lingkungan pengujian yang terdiri dari perangkat keras dan perangkat lunak. Pembahasan kedua adalah skenario pengujian dan evaluasi dari setiap skenario untuk mendapatkan kinerja terbaik melalui perbandingan performa akurasi, precision, recall, dan f-measure. Pembahasan terakhir adalah tentang analisa hasil pengujian.

4.1. Lingkungan Uji Coba

Lingkungan pengujian pada penelitian ini meliputi spesifikasi perangkat keras yang digunakan ditunjukkan pada Tabel 4.1 dan spesifikasi perangkat lunak yang digunakan dan ditunjukkan pada Tabel 4.2.

Tabel 4.1. Spesifikasi Hardware

No	Nama	Spesifikasi
1	**Processor**	Intel Celeron CPU 847 @1.10 GHz
2	**Memori**	4,00 GB
3	**Harddisk**	500 GB
4	**VGA**	Intel HD Graphics 1696 MB

Tabel 4.2. Spesifikasi Software

No	Nama	Spesifikasi
1	**Sistem Operasi**	Windows 7 Ultimate
2	**Proses Data**	Ms. Excel 2013
3	**Weka**	v. 3.6

4.2. Skenario Pengujian

Penelitian ini mennguankan dataset pengujian yang didapat dari NASA Metric Data Program (MDP). Terdapat lima dataset pengujian. Pada Tabel 4.3 menunjukkan spesifikasi dataset pengujian yang digunakan dalam penelitian ini.

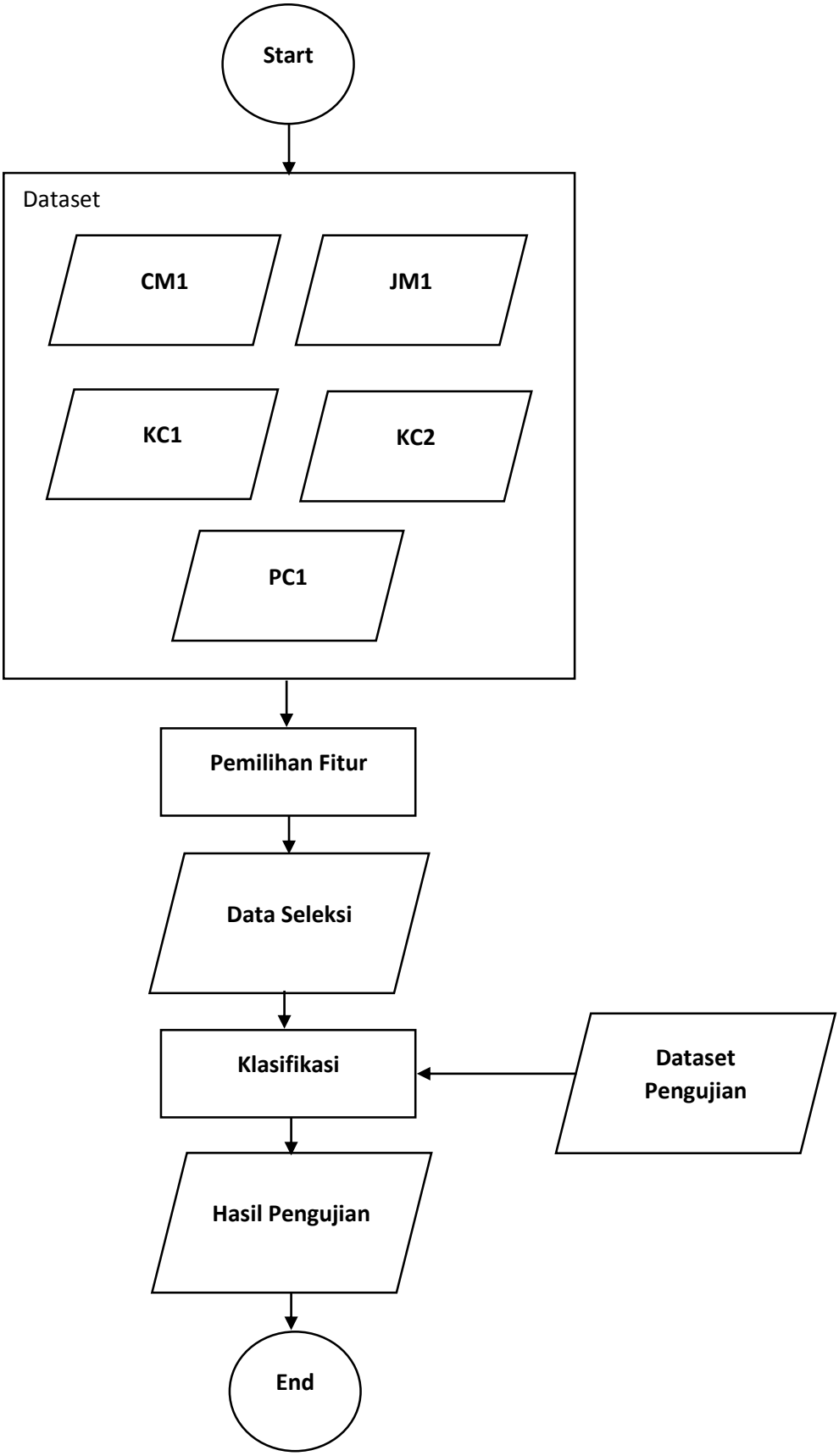
Tabel 4.3. Dataset Pengujian

Modul	Jumlah Total Modul	Jumlah Modul Tidak Cacat	Jumlah Modul Cacat	Presentase Modul Cacat (%)	Bahasa
CM1	498	449	49	9,83	C
JM1	10885	8779	2106	19,35	C
KC1	2109	1783	326	15,45	C++
KC2	522	415	107	20,5	C++
PC1	1109	1032	77	6,94	C

Pada tabel 4.4 ditampilkan rekap distribusi data yang digunakan mulai dari dataset CM1, JM, KC1, KC2 dan PC1 dengan jumlah kelas masing maisng-masing dataset. CM1 dengan 9,83% cacat dan 90,16% tidak cacat. Dataset JM1 dengan 80,6% cacat dan 19,35% tidak cacat. Dataset KC1 dengan 15,45% cacat dan 84,54% tidak cacat. Dataset KC2 dengan 20,5% cacat dan 79,5% tidak cacat. Dataset PC1 dengan 93,05% cacat dan 6,94% tidak cacat.

Tabel 4.4. Rekap Distribusi Kelas

Dataset	Yes (%)	No (%)
CM1	9.83	90.16
JM1	80.65 (true)	19.35 (false)
KC1	15.45	84.54
KC2	20.5	79.5
PC1	93.05 (true)	6.94 (false)



Gambar 4.1. Metode Pengujian

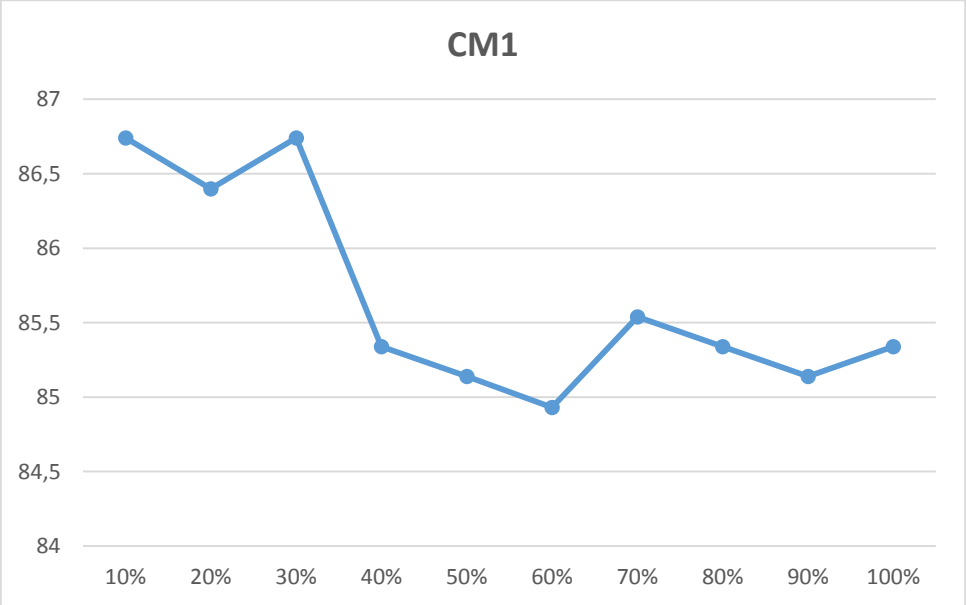
4.3. Pemilihan Fitur

Skenario pengujian 1 dan pengujian 2 memerlukan dataset yang telah terpilih sebelumnya. Pada Tabel 4.5 hingga Tabel 4.9 menampilkan akurasi metode disteipa distribusi modul cacat perangkat lunak pada dataset CM1, JM1, KC1, KC2 dan PC1.

Tabel 4.5. Akurasi metode setiap atribut yang terseleksi pada dataset CM1

Jumlah Atribut	Presntase Modul (%)	Presentase Akurasi (%)
3	10	86,74
5	20	86,40
7	**30**	**86,74**
9	40	85,34
11	50	85,14
13	60	84,93
15	70	85,54
17	80	85,34
19	90	85,14
21	100	85,34

Pada Tabel 4.5 dan Gambar 4.2 menunjukkan bahwa dominasi dan persebaran data atau atribut terbaik pada dataset CM1 terdapat pada tujuh atribut terbaik, dengan presentase modul 30% dengan akurasi modul 86,74%.

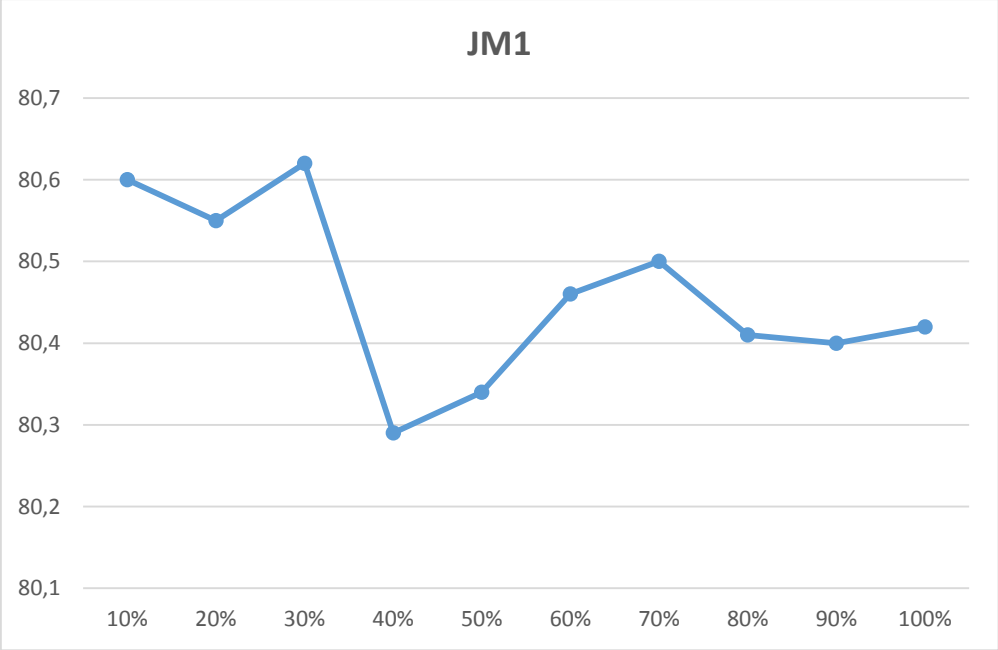


Gambar 4.2 Grafik akurasi metode setiap atribut yang terseleksi pada dataset CM1

Tabel 4.6. Akurasi metode setiap atribut yang terseleksi pada dataset JM1

Jumlah Atribut	Presentase Modul (%)	Presentase Akurasi (%)
3	10	80,6
5	20	80,55
7	**30**	**80,62**
9	40	80,29
11	50	80,34
13	60	80,46
15	70	80,5
17	80	80,41
19	90	80,4
21	100	80,42

Pada Tabel 4.6 dan Gambar 4.3 menunjukkan bahwa dominasi dan persebaran data atau atribut terbaik pada dataset JM1 terdapat pada tujuh atribut terbaik, dengan presentase modul 30% dengan akurasi modul 80,62%.

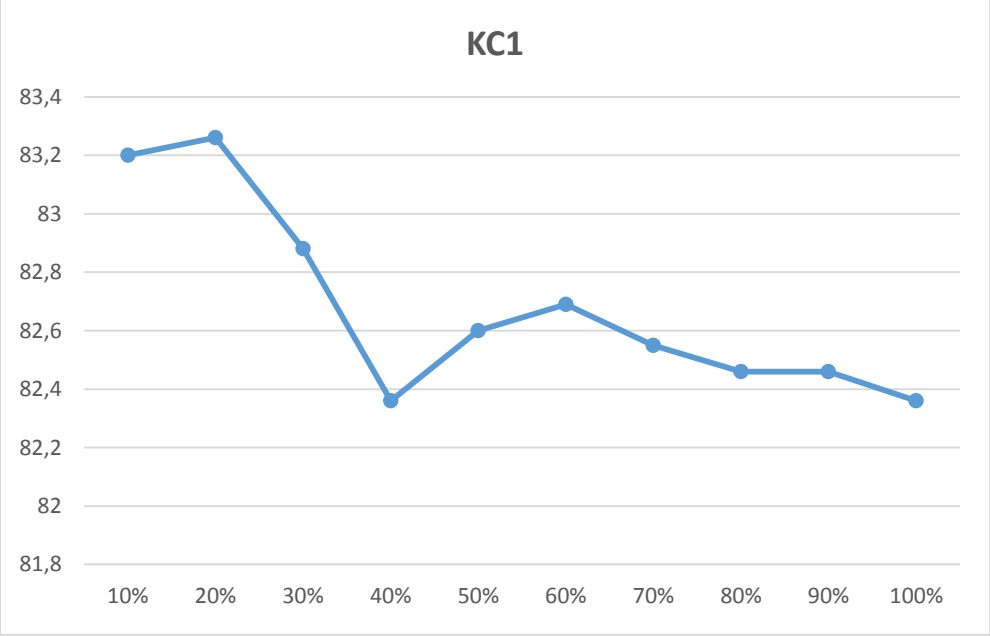


Gambar 4.3 Grafik akurasi metode setiap atribut yang terseleksi pada dataset JM1

Tabel 4.7. Akurasi metode setiap atribut yang terseleksi pada dataset KC1

Jumlah Atribut	Presntase Modul (%)	Presentase Akurasi (%)
3	10	83,2
5	**20**	**83,26**
7	30	82,88
9	40	82,36
11	50	82,6
13	60	82,69
15	70	82,55
17	80	82,46
19	90	82,46
21	100	82,36

Pada Tabel 4.7 dan Gambar 4.4 menunjukkan bahwa dominasi dan persebaran data atau atribut terbaik pada dataset CM1 terdapat pada lima atribut terbaik, dengan presentase modul 32% dengan akurasi modul 83,26%.

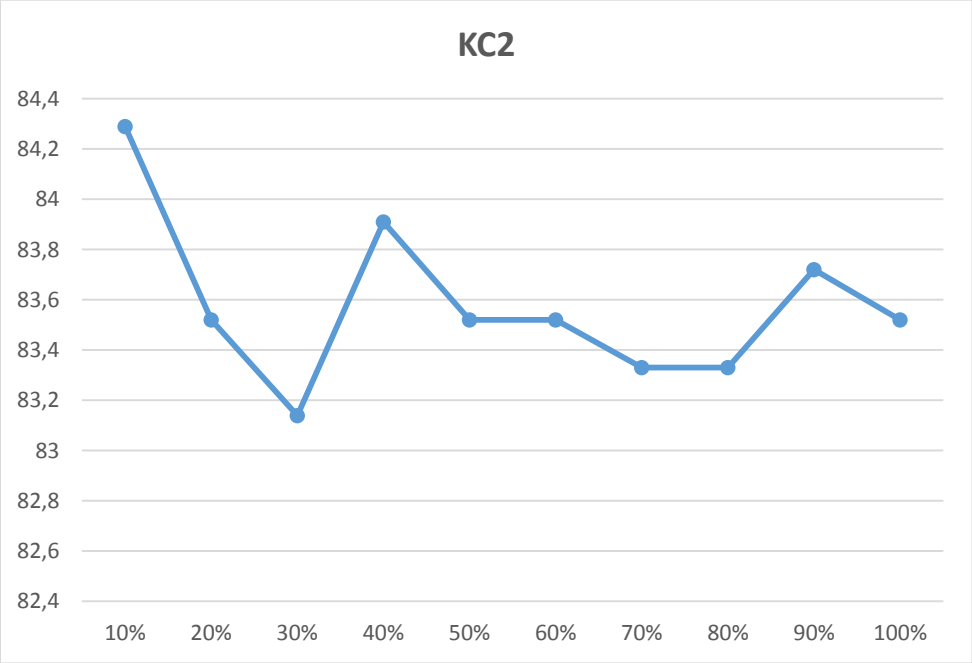


Gambar 4.4 Grafik akurasi metode setiap atribut yang terseleksi pada dataset KC1

Tabel 4.8. Akurasi metode setiap atribut yang terseleksi pada dataset KC2

Jumlah Atribut	Presntase Modul (%)	Presentase Akurasi (%)
3	**10**	**84,29**
5	20	83,52
7	30	83,14
9	40	83,91
11	50	83,52
13	60	83,52
15	70	83,33
17	80	83,33
19	90	83,72
21	100	83,52

Pada Tabel 4.8 dan Gambar 4.5 menunjukkan bahwa dominasi dan persebaran data atau atribut terbaik pada dataset KC2 terdapat pada tiga atribut terbaik, dengan presentase modul 10% dengan akurasi modul 84,29%.

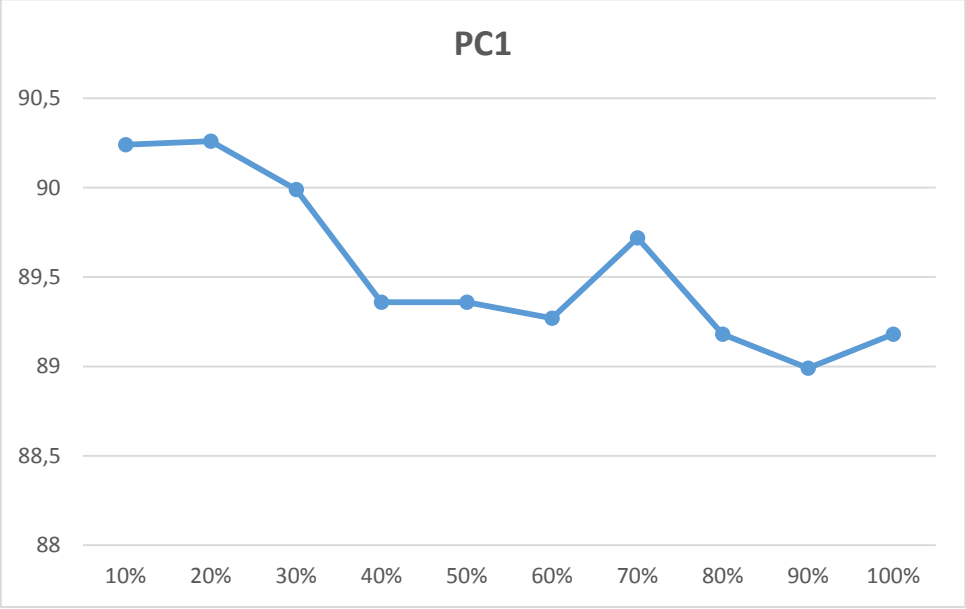


Gambar 4.5 Grafik akurasi metode setiap atribut yang terseleksi pada dataset KC2

Tabel 4.9. Akurasi metode setiap atribut yang terseleksi pada dataset PC1

Jumlah Atribut	Presntase Modul (%)	Presentase Akurasi (%)
3	10	90,24
5	**20**	**90,26**
7	30	89,99
9	40	89,36
11	50	89,36
13	60	89,27
15	70	89,72
17	80	89,18
19	90	88,99
21	100	89,18

Pada Tabel 4.9 dan Gambar 4.6 menunjukkan bahwa dominasi dan persebaran data atau atribut terbaik pada dataset PC1 terdapat pada lima atribut terbaik, dengan presentase modul 30% dengan akurasi modul 90,26%.



Gambar 4.6 Grafik akurasi metode setiap atribut yang terseleksi pada dataset PC1

Setelah dilakukan pemlihan fitur didapatkan fitur pada masing-masing dataset dengan jumlah atribut yag digunakan berbeda-beda. Pada tabel 4.10 menampilkan hasil atribut terpilih pada masing-masing dataset.

Tabel 4.10. Daftar Metrik atau dataset yang terseleksi

CM1	JM1	KC1	KC2	PC1
lOComment	loc	uniq_Opnd	ev(g)	lOBlank
uniq_Op	v(g)	branchCount	uniq_Opnd	lOComment
uniq_Opnd	branchCount	d	loc	I
b	lOCode	v	v	uniq_Opnd
loc	N	v(g)	b	locCodeAndComment
i	iv(g)		n	
v	e		t	
			e	
			lOCode	

4.2.1. Skenario Pengujian 1 Tanpa Naive Bayes Tanpa Gain Ratio

Dalam skenario 1 dilakukan pengujian prediksi data tanpa melibatkan proses pemilhan fitur Gain Ratio. Skenario pertama uji coba metode Naive Bayes tanpa Gain Ratio diterapkan pada Dataset CM1, JM1, KC1, KC2, dan PC1. Pada Tabel 4.11 menunjukkan nbahwa penerapan Naive Bayes memiliki Precision, Recall, F-Measure dan Akurasi terendah adalah Dataset JM1 dan tertinggi adalah dataset PC1.

Tabel 4.11. Hasil Pengujian Skenario 1

NB	CM1	JM1	KC1	KC2	PC1
Precision	86,20	76,50	81,60	82,00	89,90
Recall	85,30	80,40	82,40	83,50	89,20
F-Measure	85,80	77,00	82,00	82,10	89,50
Accuracy	85,34	80,42	82,36	83,52	89,18

4.2.2. Skenario Pengujian 2 Naive Bayes dengan Gain Ratio

Dalam skenario 2 dilakukan pengujian prediksi data dengan melibatkan proses pemilhan fitur Gain Ratio. Atribut terpilih Setelah dilakukan pemilihan fitur Gain Ratio pada dataset CM1 yaitu loc, b, lOComment, uniq_Op, uniq_Opnd, dan kelasnya defects {false,true}. Hasil pemilihan fitur pada dataset JM1 yaitu loc, v(g), n, lOCode, branchCount, dan kelasnya defects {false,true}. Hasil pemilihan fitur pada dataset KC1 yaitu v(g), v, d, uniq_Opnd, branchCount, dan kelasnya defects {false,true}. Hasil pemilihan fitur pada dataset KC2 yaitu loc, ev(g), v, d, b, dan problems {no,yes}. Hasil pemilihan fitur pada dataset PC1 yaitu I, lOComment,

locCodeAndComment, lOBlank, uniq_Opnd, dan kelasnya defects {false,true}. Pada Tabel 4.12 juga menunjukkan bahwa penerapan Naive Bayes Gain Ratio memiliki Precision, Recall, F-Measure dan Akurasi terendah masih pada Dataset JM1 dan tertinggi adalah dataset PC1.

Tabel 4.12. Hasil Pengujian Skenario 2

NBGR	CM1	JM1	KC1	KC2	PC1
Precision	86,3	76,2	81,4	83,2	90,3
Recall	86,7	80,5	82,9	84,5	89,7
F-Measure	86,5	76,4	82	82,8	90
Accuracy	86,74	80,51	82,88	84,48	89,72

4.4. Evaluasi Hasil Pengujian

Pada subbab ini dibahas tentang hasil pengujian dan evaluasi dari setiap skenario pengujian yang telah dilakukan. Hasil prediksi setiap modul dicocokkan dengan status cacat perangkat lunak yang terdapat pada dataset NASA MDP. Perhitungan performa menggunakan nilai akurasi, *precision*, *recall*, dan *f-measure.*

4.3.1. Evaluasi Hasil Pengujian Skenario 1 dan Skenario 2

Tabel 4.13 menunjukkan performa prediksi pada skenario 1 dan skenario 2 berdasarkan parameter dari dataset CM1, JM1, KC1, KC2, dan PC1. Berikut hasil yang didapat setelah dilakukakn pengujian Prediksi Cacat Perangkat Lunak Tanpa Gain Ratio (Skenario 1) dan dengan Gai Ratio (Skenario 2).

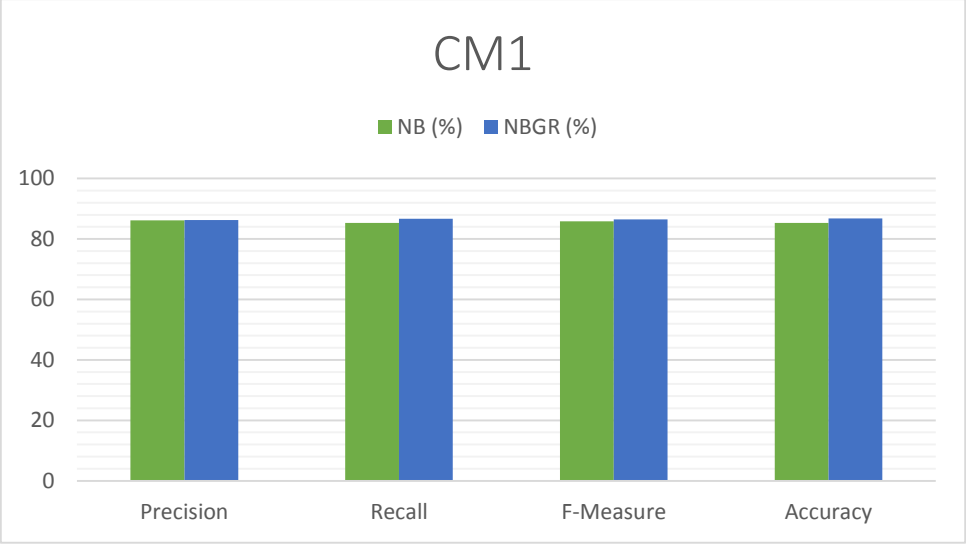
4.3.1.1.Evaluasi Hasil Pengujian Skenario 1 dan 2 Dataset CM1

Tabel 4.13 menunjukkan nilai precision, recall, f-measure dan akurasi NB dan NBGR pada dataset CM1. Pada Tabel 4.13 terlihat bahwa nilai precision, recall, f-measure dan akurasi lebih tinggi metode NBGR dari pada NB. Masing-masing parameter menunjukkan bahwa selisih precision, recall, f-measure dan akurasi cukup signifikan masing-masing yaitu 0,1; 1,4; 0,7; dan 1,4. Hal ini menunjukkan bahwa metode NBGR lebih baik dibandingkan NB untuk dataset CM1.

Tabel 4.13. Hasil Pengujian CM1

Dataset		NB (%)	NBGR (%)	Selisih
CM1	**Precision**	86,20	**86,3**	0,1
	Recall	85,30	**86,7**	1,4
	F-Measure	85,80	**86,5**	0,7
	Accuracy	85,34	**86,74**	1,4

Pada grafik Gambar 4.7. menunjukkan bahwa peningkatan menggunakan metode NBGR sangat signifikan pada Recall dan Akurasi data banyak yang sesuai dengan permintaan sistem dan akan otomatis berdampak pada meningkatnya akurasi.



Gambar 4.7. Grafik Hasil Pengujian CM1

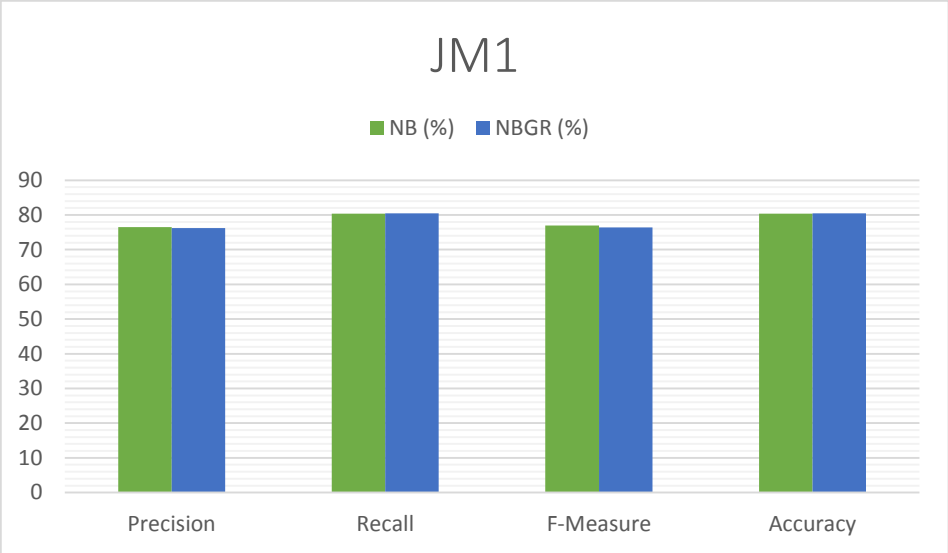
4.3.1.2. Evaluasi Hasil Pengujian Skenario 1 Dataset JM1

Tabel 4.14 menunjukkan nilai precision, recall, f-measure dan akurasi NB dan NBGR pada dataset JM1. Pada Tabel 4.14 terlihat bahwa nilai precision, recall, f-measure dan akurasi lebih tinggi metode NBGR dari pada NB. Masing-masing parameter menunjukkan bahwa selisih precision, recall, f-measure dan akurasi cukup signifikan masing-masing yaitu -0,3; 0,1; -0,6; dan 0,09; kecuali pada nilai precision dan f-measure lebih rendah karena persebaran data setiap atribut tidak sama (Diana, 2011) dan nilai presisi, recall tidak begitu besar sehingga hal ini mempengaruhi kestabilan nilai f-measure. Secara keseluruhan metode NBGR masih lebih baik dibandingkan NB untuk dataset JM1.

Tabel 4.14. Hasil Pengujian JM1

Dataset		NB (%)	NBGR (%)	Selisih
JM1	**Precision**	**76,50**	76,2	-0,3
	Recall	80,40	**80,5**	0,1
	F-Measure	**77,00**	76,4	-0,6
	Accuracy	80,42	**80,51**	0,09

Pada grafik Gambar 4.8 menunjukkan bahwa peningkatan menggunakan metode NBGR sangat signifikan pada precision, recall dan akurasi. Sedangkan nilai f-measure rendah, karena terjadi kesalahan klasifikasi, sehingga data yang terambil tidak sesuai dengan data aslinya.



Gambar 4.8. Grafik Hasil Pengujian JM1

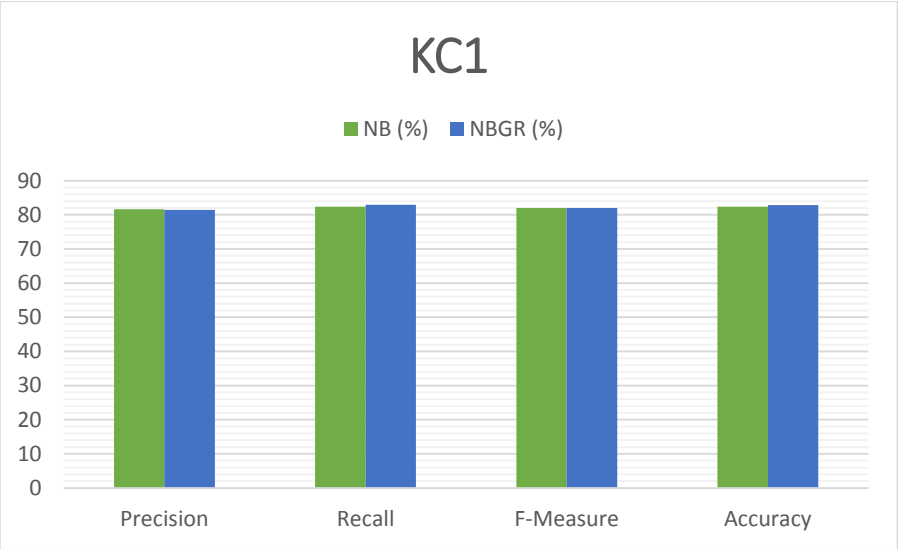
4.3.1.3. Evaluasi Hasil Pengujian Skenario 1 Dataset KC1

Tabel 4.15 menunjukkan nilai precision, recall, f-measure dan akurasi NB dan NBGR pada dataset KC1. Pada Tabel 4.15 terlihat bahwa nilai precision, recall, f-measure dan akurasi lebih tinggi metode NBGR dari pada NB. Masing-masing parameter menunjukkan bahwa selisih precision, recall, f-measure dan akurasi cukup signifikan masing-masing yaitu -0,2; 0,5; 0,0; dan 0,52; kecuali pada nilai precision lebih rendah tetapi hal ini tidak mempengaruhi nilai presisi dan recall sehingga hasil akurasi tetap mengalami peningkatan. Secara keseluruhan metode NBGR masih lebih baik dibandingkan NB untuk dataset KC1.

Tabel 4.15. Hasil Pengujian KC1

Dataset		NB (%)	NBGR (%)	Selisih
KC1	**Precision**	**81,60**	81,40	-0,20
	Recall	82,40	**82,90**	0,50
	F-Measure	82,00	82,00	0,00
	Accuracy	82,36	**82,88**	0,52

Pada grafik Gambar 4.9 menunjukkan bahwa peningkatan menggunakan metode NBGR sangat signifikan pada recall dan akurasi. Meskipun nilai akurasi, precision tinggi namun recall yang tinggi karena ketidakseimbangan distribusi modul cacat dan tidak cacat dalam dataset menyebabkan pendeteksian modul cacat yang memiliki distribusi lebih sedikit dinilai lebih penting dibandingkan modul tidak cacat.



Gambar 4.9. Grafik Hasil Pengujian KC1

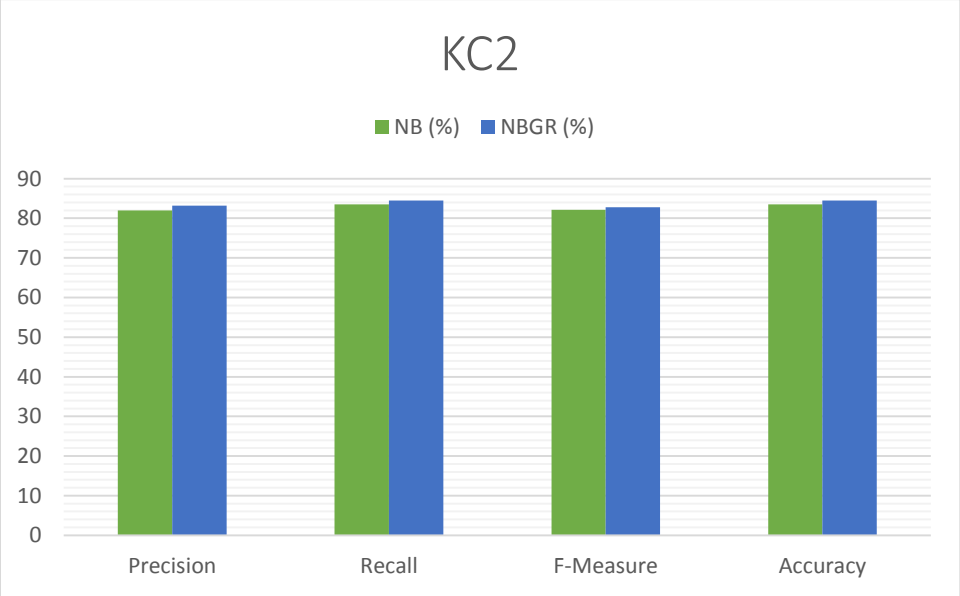
4.3.1.4. Evaluasi Hasil Pengujian Skenario 1 Dataset KC2

Tabel 4.16 menunjukkan nilai precision, recall, f-measure dan akurasi NB dan NBGR pada dataset KC2. Pada Tabel 4.16 terlihat bahwa nilai precision, recall, f-measure dan akurasi lebih tinggi metode NBGR dari pada NB. Masing-masing parameter menunjukkan bahwa selisih precision, recall, f-measure dan akurasi cukup signifikan masing-masing yaitu 1,2; 1; 0,7; dan 0,96. Hasil semua pengolahan data tetap stabil dan megalami peningkatan hasil akurasi prediksi. Secara keseluruhan metode NBGR masih lebih baik dibandingkan NB untuk dataset KC2.

Tabel 4.16. Hasil Pengujian KC2

Dataset		NB (%)	NBGR (%)	Selisih
KC2	**Precision**	82,00	**83,2**	1,2
	Recall	83,50	**84,5**	1
	F-Measure	82,10	**82,8**	0,7
	Accuracy	83,52	**84,48**	0,96

Pada grafik Gambar 4.10 menunjukkan bahwa peningkatan menggunakan metode NBGR sangat signifikan pada Precision, Recall dan Akurasi. Sedangkan nilai f-measure tidak mengalami perubahan, hal ini tidak akan mempengaruhi hasil prediksi.



Gambar 4.10. Grafik Hasil Pengujian KC2

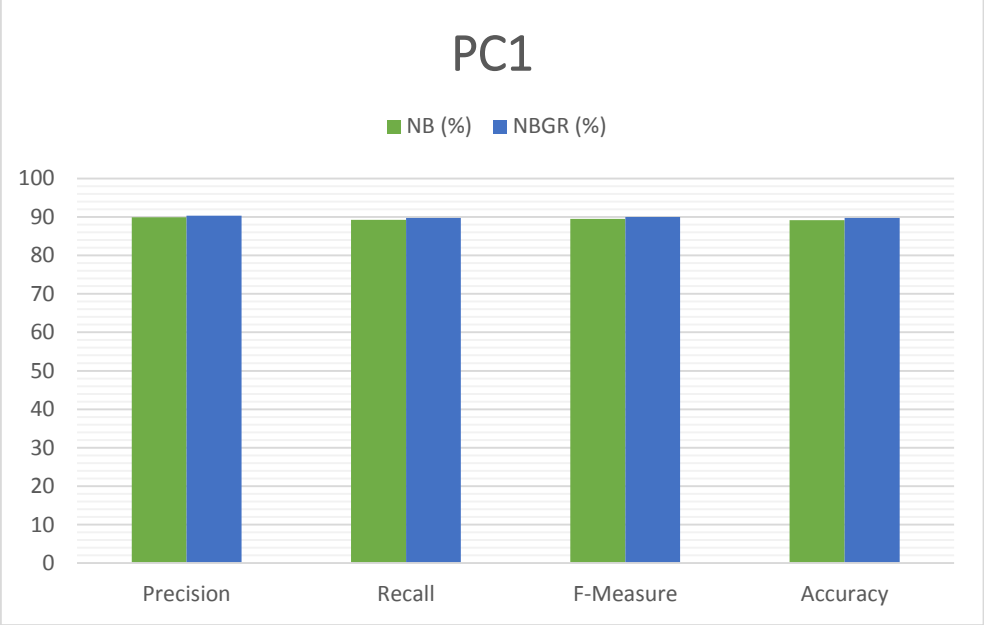
4.3.1.5. Evaluasi Hasil Pengujian Skenario 1 Dataset PC1

Tabel 4.17 menunjukkan nilai precision, recall, f-measure dan akurasi NB dan NBGR pada dataset PC1. Pada Tabel 4.17 terlihat bahwa nilai precision, recall, f-measure dan akurasi lebih tinggi metode NBGR dari pada NB. Masing-masing parameter menunjukkan bahwa selisih precision, recall, f-measure dan akurasi cukup signifikan masing-masing yaitu 0,4; 0,5; 0,5; dan 0,54; ha ini menandakan metode NBGR lebih baik dibandingkan NB untuk dataset PC1.

Tabel 4.17. Hasil Pengujian PC1

Dataset		NB (%)	NBGR (%)	Selisih
PC1	**Precision**	89,90	**90,3**	0,4
	Recall	89,20	**89,7**	0,5
	F-Measure	89,50	**90**	0,5
	Accuracy	89,18	**89,72**	0,54

Pada grafik Gambar 4.11. menunjukkan bahwa peningkatan menggunakan metode NBGR sangat signifikan pada semua parameter hal ini terjadi karena NBGR memang cocok digunakan pada dataset PC1.



Gambar 4.11. Grafik Hasil Pengujian PC1

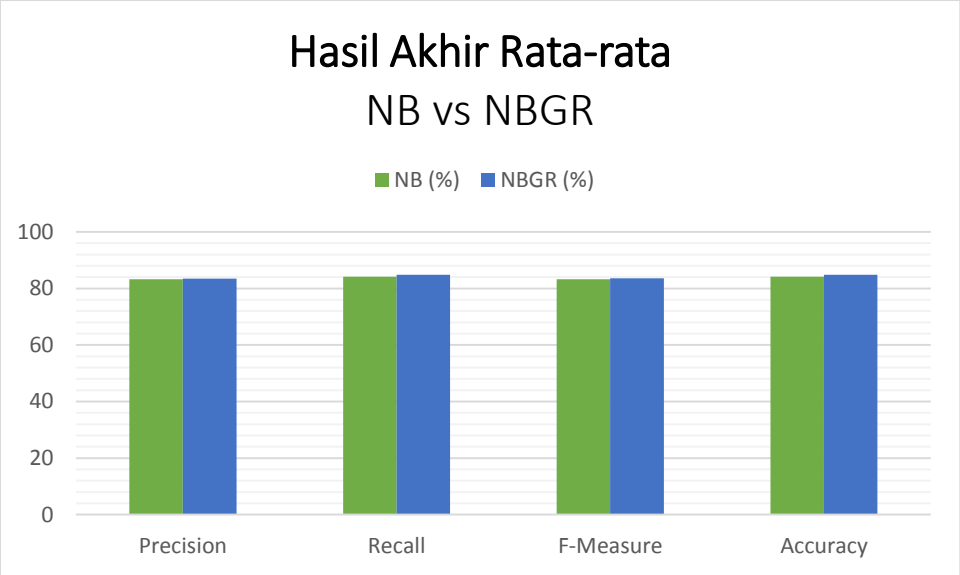
4.3.2. Evaluasi Perbandingan Hasil Pengujian Metode Naive Bayes dengan Hasil Pengujian Naive Bayes Gain Ratio

Tabel 4.18 menunjukkan perbandingan rata-rata performa akurasi, *precision*, *recall*, dan *f-measure* dari metode yang diajukan dengan penelitian sebelumnya pada dataset CM1, JM1, KC1, KC2, dan PC1. Sedangkan Gambar 4.12 menunjukkan grafik perbandingan performa metode yang diajukan dengan metode sebelumnya.

Tabel 4.18. Rata-rata Hasil Pengujian

Parameter	NB (%)	NBGR (%)	Selisih
Precision	83,24	**83,48**	0,24
Recall	84,16	**84,86**	0,7
F-Measure	83,28	**83,54**	0,26
Accuracy	84,17	**84,866**	0,67

Tabel 4.18 menunjukkan nilai rata-rata precision, recall, f-measure dan akurasi NB dan NBGR pada dataset CM1, JM1, KC1, dan KC2. Pada Tabel 4.18 terlihat bahwa nilai precision, recall, f-measure dan akurasi lebih tinggi metode NBGR dari pada NB. Masing-masing parameter menunjukkan bahwa selisih precision, recall, f-measure dan akurasi cukup signifikan masing-masing yaitu 0,24; 0,7; 0,24; dan 0,67. Secara keseluruhan metode NBGR masih lebih baik dibandingkan NB untuk semua dataset yang digunakan.



Gambar 4.12. Grafik Rata-rata Hasil Pengujian

4.3.3. Evaluasi Perbandingan Hasil Pengujian Metode yang diusulkan dengan Hasil Pengujian Penelitian Sebelumnya

Tabel 4.19 hingga Tabel 4.22 menunjukkan perbandingan performa akurasi, *precision, recall dan f-measure* dari metode yang diajukan dengan penelitian sebelumnya pada dataset CM1, JM1, KC1, KC2, dan PC1. Sedangkan Gambar 4.13 menunjukkan grafik perbandingan performa metode yang diajukan dengan metode-metode sebelumnya.

Tabel 4.19 Hasil Perbandingan Precision metode usulan dengan metode sebelumnya

Metode	Precision (%)
NB	83,24
NBGR	83,48
RARM	42,14
SVM	**99,2**
ANN ABC	37,11
D Tree	82,82

Pada Tabel 4.19 menampilkan hasil perbandingan metode usulan (NBGR) dengan metode lain. Nilai Precision untuk metode NB sebesar 83,24%. Metode RARM sebesar 42,14%, metode SVM 99,2%, metode ANN ABC 37,11%, metode D Tree 82,82%. Sedangkan untuk metode yang diusulkan NBGR sebesar 83,42%.

Tabel 4.20 Hasil Perbandingan Recall metode usulan dengan metode sebelumnya

Metode	Recall (%)
NB	84,16
NBGR	**84,86**
RARM	50,01
SVM	54,36
ANN ABC	79,24
D Tree	84,34

Pada Tabel 4.20 menampilkan hasil perbandingan metode usulan (NBGR) dengan metode lain. Nilai Recall untuk metode NB sebesar 84,16%. Metode RARM sebesar 50,01%, metode SVM 54,36%, metode ANN ABC 79,24%, metode D Tree 84,34%. Sedangkan untuk metode yang diusulkan NBGR sebesar 84,9%

Tabel 4.21 Hasil Perbandingan F-Measure metode usulan dengan metode sebelumnya

Metode	F-Measure (%)
NB	83,28
NBGR	83,54
RARM	13,81
SVM	**91,92**
ANN ABC	18,28
D Tree	83,68

Pada Tabel 4.21 menampilkan hasil perbandingan metode usulan (NBGR) dengan metode lain. Nilai F-Measure untuk metode NB sebesar 83,28%. Metode RARM sebesar 13,81%, metode SVM 91,92%, metode ANN ABC 18,28%, metode D Tree 83,68%. Sedangkan untuk metode yang diusulkan NBGR sebesar 83,62%.

Tabel 4.22 Hasil Perbandingan Akurasi metode usulan dengan metode sebelumnya

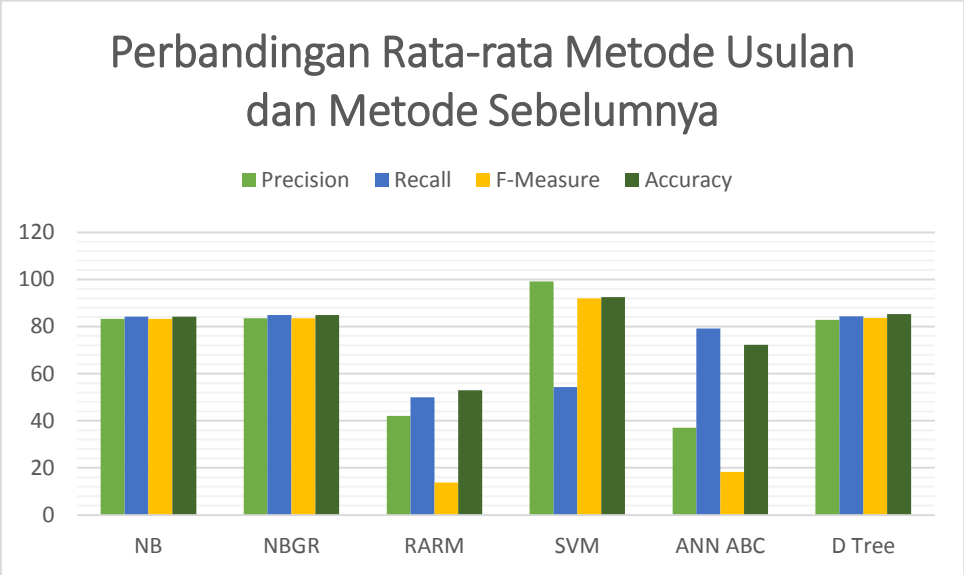
Metode	Akurasi (%)
NB	84,17
NBGR	84,87
RARM	52,95
SVM	**92,45**
ANN ABC	72,24
D Tree	85,32

Pada Tabel 4.22 menampilkan hasil perbandingan metode usulan (NBGR) dengan metode lain. Nilai Akurasi untuk metode NB sebesar 84,17%. Metode RARM sebesar 52,95%, metode SVM 92,45%, metode ANN ABC 72,24%, metode D Tree 85,32%. Sedangkan untuk metode yang diusulkan NBGR sebesar 84,89%.

Tabel 4.23 Hasil Perbandingan Rata-rata metode usulan dengan metode sebelumnya

Parameter	NB (%)	NBGR (%)	RARM (%)	SVM (%)	ANN ABC (%)	D Tree (%)
Precision	83,24	83,48	42,14	**99,2**	37,11	82,82
Recall	84,16	**84,86**	50,01	54,36	79,24	84,34
F-Measure	83,28	83,54	13,81	**91,92**	18,28	83,68
Accuracy	84,17	84,87	52,95	**92,45**	72,24	85,32

Pada Tabel 4.23 menampilkan hasil perbandingan Rata-rata metode usulan (NBGR) dengan metode lain. Nilai Precision teringgi menggunakan metode SVM dengan 99,2%. Nilai Recall teringgi menggunakan metode usulasn NBGR dengan 84,86%. Nilai F-measure teringgi menggunakan metode SVM dengan 91,92%. Nilai Akurasi teringgi menggunakan metode SVM dengan 99,45%. Akan tetapi, hasil precision, recall, f-measure dan akurasi metode usulan semua sudah lebih baik daripada metode Naive Bayes biasa.



Gambar 4.13. Grafik Rata-rata Metode Usulan dan Metode Sebelumnya

Tabel 4.23 dan Gambar 4.13 menunjukkan bahwa performa metode usulan NBGR memiliki nilai rata-rata recall tertinggi dibandingkan metode yang lainnya yaitu sebesar 84.90%, meskipun nilai rata-rata akurasi NBGR bukan yang terbaik namun selisih akurasi tidak begitu signifikan dibanding dengan metode lain dan metode usulan masih lebih baik dalam performa precision, recall, f-measure maupun akurasi, hal ini sesuai dengan hipotesa awal bahwa metode NBGR lebih baik dibandingkan dengan metode NB.

SVM memiliki nilai akurasi, *precision*, dan *f-measure* terbaik, namun SVM memiliki nilai recall rendah yaitu 54.36%. Sehingga performa NBGR tetap dinilai yang terbaik dikarenakan dapat mendeteksi modul cacat perangkat lunak dengan lebih baik. Atau dengan kata lain NBGR memiliki nilai *True Positive* (modul cacat yang benar terdeteksi sebagai modul cacat) lebih tinggi dibandingkan dengan metode lain.

4.4. Analisa Hasil Pengujian

Analisa hasil pengujian yang telah didapatkan dari skenario 1 dan 2 serta perbandingannya metode yang diajukan pada penelitian sebelumnya adalah sebagai berikut:

1. Prasyarat yang harus dipenuhi sebelum melakukan prediksi cacat perangkat lunak adalah melakukan pemilihan fitur terbaik dengan masing-masing dataset terpilih 5 atribut terbaik, sehingga dari total 5 dataset terdapat 25 atribut terbaik untuk selanjutnya dilakukan proses prediksi.
2. Penambahan proses pemilihan fitur Gain Ratio pada Naive Bayes meningkatkan nilai akurasi sebesar 0.72% dibandingkan dengan Naive Bayes sebelumnya. Hal ini menunjukkan bahwa penambahan proses pemilihan fitur menggunakan Gain Ratio dapat meningkatkan pendeteksian modul cacat perangkat lunak.
3. NBGR memiliki nilai *precision, recall, dan f-measure* lebih baik dibandingkan NB biasa masing-masing sebesar 0,24; 0,7; dan 0,24. Meskipun demikian penambahan proses pemilihan fitur pada Naive Bayes tetap dinilai lebih baik dibandingkan Naive Bayes biasa dalam mendeteksi modul cacat perangkat lunak yang memiliki distribusi lebih rendah daripada modul tidak cacat dalam dataset.
4. NBGR memiliki nilai recall paling tinggi dibandingkan dengan metode prediksi lain NB, SVM, dll yaitu sebesar 84,86%. Penambahan proses pemilihan fitur meningkatkan kemampuan untuk mendeteksi modul cacat perangkat lunak secara signifikan. Pendeteksian modul cacat yang memiliki distribusi lebih sedikit dalam dataset dinilai lebih penting dibandingkan modul tidak cacat. Meskipun nilai akurasi NBGR bukan yang terbaik di antara metode yang lain namun perbedaan tidak terlalu signifikan.

BAB V

PENUTUP

Bab ini membahas kesimpulan dari hasil penelitian dan saran untuk penelitian selanjutnya berdasarkan hasil dan evaluasi pengujian pada bab 4.

5. 1. Kesimpulan

Dari hasil penelitian yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut.

1. Penambahan proses pemilihan fitur NBGR pada prediksi cacat perangkat lunak yang memiliki distribusi lebih rendah dalam dataset pelatihan dapat meningkatkan nilai akurasi yaitu sebesar 0,67% lebih baik dibandingkan tanpa pemilihan fitur.
2. NBGR memiliki nilai *precision, recall, dan f-measure* dan akurasi lebih baik dibandingkan NB dengan selisih masing-masing sebesar 0,24; 0,7; 0,24; dan 0,67. Namun dengan ketidakseimbangan modul cacat dan tidak cacat dalam dataset maka pendeteksian modul cacat lebih diutamakan. Sehingga NBGR dinilai tetap lebih baik dikarenakan lebih mampu mendeteksi modul cacat perangkat lunak. Dengan kata lain NBGR memiliki nilai recall atau sensitivitas lebih besar dibandingkan NB biasa.
3. NBGR memiliki nilai recall tertinggi jika dibandingkan metode prediksi lain seperti menggunakan NB, SVM, dll. Meskipun nilai akurasi, NBGR bukan yang terbaik namun nilai recall tetap diutamakan dalam pendeteksian modul cacat perangkat lunak. Semakin tinggi nilai recall maka akan semakin baik dalam mendeteksi modul cacat yang memiliki distribusi lebih sedikit dalam dataset. Ketidakseimbangan modul cacat dan tidak cacat membuat pendeteksian modul cacat perangkat lunak yang memiliki distribusi yang rendah menjadi lebih essensial. Metode Usulan NBGR sudah sesuai dengan hipotesa awal yakni lebih baik dibanding dengan metode NB sebelumnya.

5.2. Saran

Prediksi cacat perangkat lunak yang dilakukan dalam penelitian ini adalah berdasarkan pada dataset NASA MDP. Berdasarkan hasil pengujian perlu dilakukan perbaikan pada penelitian selanjutnya, sehingga nilai precision tidak turun dan nilai f-measure tidak turun dibandingkan metode sebelumya. Prediksi cacat perangkat lunak ke depan dapat dikembangkan dan diterapkan pada projek pengembangan perangkat lunak yang nyata. Selain itu perlu dilakukan kombinasi metode lain dan menggunakan data terbaru untuk mendapatkan kombinasi performa akurasi yang terbaik.

DAFTAR PUSTAKA

B. Dhanalaxmia, G. N. (2015). Adaptive PSO based Association Rule Mining Technique for Software Defect Classification using ANN. *Procedia Computer Science 46*, 432 – 442.

Boehm, B. B. (2001). Software defect top 10 list. *IEEE Computer 34*, 2-6.

Chen, J. (2008). A selective Bayes Classifier for classifying incomplete data based on gain ratio . *Knowledge-Based Systems 21* , 530–534.

Diana, W. N. (2011). *Penerapan Algoritma Improved K-Nearest Neighbors Untuk Pengkategorian Dokumen Teks Berita Berbahasa Indonesia*. Universitas Brawijaya.

Gabriela Czibula, Z. M. (2014). Software defect prediction using relational association rule mining. *Information Sciences*, 260-278.

Halstead, M. H. (1977). Elements of Software Science (Operating and Programming Systems Series.

Hamzah, A. (2012). Klasifikasi Teks Dengan Naïve Bayes Classifier (Nbc) Untuk Pengelompokan Teks Berita Dan Abstract Akademis . *Seminar Nasional Aplikasi Sains & Teknologi (SNAST) Periode III.*

Karim O. Elish, M. O. (2008). Predicting defect-prone software modules using support vector machines. *The Journal of Systems and Software 81*, 649-660.

McCabe, T. (1976). A Complexity Measure. Software Engineering, SE-2. *IEEE Transaction*, 308-320.

Pressman, R. (2001). *Software Engineering: A Practitioner's Approach.* McGraw-Hill, NY.

Rafik Khairul Amin, D. ,. (2015). Implementasi Klasifikasi Decision Tree Dengan Algoritma C4.5. *Jurnal Tugas Akhir Fakultas Informatika*.

Socrates, A. G. (2016). Optimasi Naive Bayes Dengan Pemilihan Fitur Dan Pembobotan Gain Ratio. . *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi Vol. 7*.

Zaidi, C. C. (2013). Alleviating Naive Bayes Attribute Independence Assumption by Attribute Weighting. *Journal of Machine Learning Research 14* , 1947-1988.

Zhang Jiang, L. K. (2016). Two Feature Weighting Approaches for Naive Bayes Text Kelasifiers. *Knowledge-Based System*.

Zheng, J. (2010). Cost-sensitive boosting neural networks for software defect prediction. *Expert Systems with Applications 37*, 4537–4543.

(halaman ini sengaja dikosongkan)

BIOGRAFI PENULIS

Penulis, Muhammad Sonhaji Akbar, lahir di Nganjuk, 30 April 1994. Penulis menempuh pendidikan dari TK Dharma Wanita Tegalgondo tahun 2000, Sekolah Dasar mulai kelas 1 sampai 6 di SD Laboratorium UM lulus tahun 2005. Untuk pendidikan menengah, penulis tempuh di SMP Negeri 18 Malang lulus tahun 2008 dan selanjutnya di SMA Negeri 9 Malang luus tahun 2011. Pada tahun 2011 penulis melanjutkan pendidikan sarjana di prodi Pendidikan Teknik Informatika Universitas Negeri Malang (UM). Pada tahun 2014 penulis mendapatkan beasiswa Peningkatan Prestasi Akademik (PPA) dari Direktorat Jenderal Pendidikan Tinggi Kementerian Pendidikan dan Kebudayaan untuk biaya perkuliahan. Penulis pernah melakukan praktek industri di PT. Telkom Akses Kebalen Surabaya selama dua bulan. Selain itu penulis juga pernah melaksanakan Praktik Pengalaman Lapangan (PPL) di SMKN 2 Malang selama 1,5 bulan. Penulis menyelesaikan pendidikan Sarjana pada tahun 2015. Penulis aktif kegiatan luar kampus Peduli Pendidikan Pedalaman 1000 Guru Malang sampai buku ini ditulis. Penulis melanjutkan studi program pascasarjana di jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya (ITS) dan mengambil bidang keahlian Rekayasa Perangkat Lunak (RPL). Penulis dapat dihubungi melalui email: mson.akbar@gmail.com dan sonhaji.akbar15@mhs.if.its.ac.id