Tesis

Grammatical Evolution untuk Ekstraksi Fitur dengan Pengukuran *Multi Fitness*

Go Frendi Gunawan

NRP : 5111201033

DOSEN PEMBIMBING

Prof. Dr. Ir. Joko Lianto Buliali, Msc.

PROGRAM MAGISTER

BIDANG KEAHLIAN KOMPUTASI CERDAS DAN VISUALISASI

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2013

GRAMMATICAL EVOLUTION UNTUK EKSTRAKSI FITUR DENGAN PENGUKURAN MULTI FITNESS

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar

Magister Teknik Informatika (M.Kom.)

di

Institut Teknologi Sepuluh Nopember

oleh :

Go Frendi Gunawan

NRP. 5111201033

Tanggal Ujian : 29 Juli 2013

Periode Wisuda : September 2013

Disetujui oleh :

1. Prof. Dr. Ir. Joko Lianto Buliali, M.Sc. …..... (Pembimbing)
   NIP. 196707271992031002
2. Dr. Ir. Raden Venantius Hari Ginardi, M.Sc. …..... (Penguji)
   NIP. 196505181992031003
3. Bilqis Amaliah, S.Kom.,M.Kom. …..... (Penguji)
   NIP. 197509142001122002
4. Isye Arieshanti, S.Kom. M. Phil. …..... (Penguji)
   NIP. 197804122006042001

Direktur Program Pasca Sarjana

Prof.Dr.Ir. Adi Soeprijanto, MT

NIP. 19640405 199002 1 001

GRAMMATICAL EVOLUTION UNTUK EKSTRAKSI FITUR DENGAN PENGUKURAN MULTI FITNESS

Nama mahasiswa : Go Frendi Gunawan
NRP mahasiswa : 5111201033
Pembimbing : Prof. Dr. Ir. Joko Lianto Buliali, M.Sc

ABSTRAK

Ekstraksi fitur merupakan salah satu topik yang berpengaruh dalam penyelesaian masalah klasifikasi. Sampai saat ini, tidak ada cara yang baku untuk menentukan fitur-fitur terbaik dari data. Dalam tesis ini, dikembangkan suatu pendekatan *grammatical evolution* dengan pengukuran multi fitness (disebut GE Tatami) guna memperoleh fitur-fitur terbaik dari data. Metode tersebut bertugas menciptakan *n-1* buah fitur baru yang sanggup memisahkan data secara hirarkikal, di mana *n* adalah jumlah kelas dalam data.

Beberapa metode telah dicoba dalam penelitian ini, antara lain algoritma genetika, *grammatical evolution* dengan pengukuran fitness global, *grammatical evolution* dengan pengkururan multi fitness, *grammatical evolution* dengan pengukuran fitness Tatami, dan *grammatical evolution* yang dikembangkan oleh Gavrilis.

Hasil penelitian menunjukkan bahwa pada data-data sintesis dengan menggunakan *decision tree classifier*, metode Tatami menunjukkan hasil yang lebih baik dari keempat metode lainnya. Data sintesis tersebut dapat dipisahkan secara hirarkikal menggunakan fitur-fitur yang di *generate*. Namun metode Tatami menunjukkan hasil yang kurang baik jika fitur-fitur ideal gagal terbentuk atau digunakan SVM sebagai *classifier*.

Kata Kunci: ekstraksi fitur, *grammatical evolution*, klasifikasi, multi-fitness.

GRAMMATICAL EVOLUTION FOR FEATURE EXTRACTION WITH MULTI FITNESS EVALUATION

Student Name : Go Frendi Gunawan
Student Identity Number : 5111201033
Supervisor : Prof. Dr. Ir. Joko Lianto Buliali, M.Sc

ABSTRACT

Feature Extraction is a significant topic in classification problem solving. Until now, there is no such a standard way to determine the best features of a data. In this thesis, grammatical evolution with multiple fitness evaluation approach (named as GE Tatami) has been developed in order to extract best features of the data. The method generate n-1 features which are able to separate data hierarchically, with n is number of classes.

Some methods has been evaluated in this research, including genetics algorithm, grammatical evolution with global fitness measurement, grammatical evolution with multi fitness measurement, grammatical evolution with Tatami fitness measurement, and Gavrilis's grammatical evolution.

It is shown in the experiment that Tatami method produce a better result compared to the four other methods for synthesis data using decision tree classifier. The synthesis data is hierarchicaly separable. However Tatami method show a bad result when it is failed to determine ideal features or using SVM classifier.

Keywords: feature extraction, grammatical evolution, classification, multi-fitness.

DAFTAR ISI

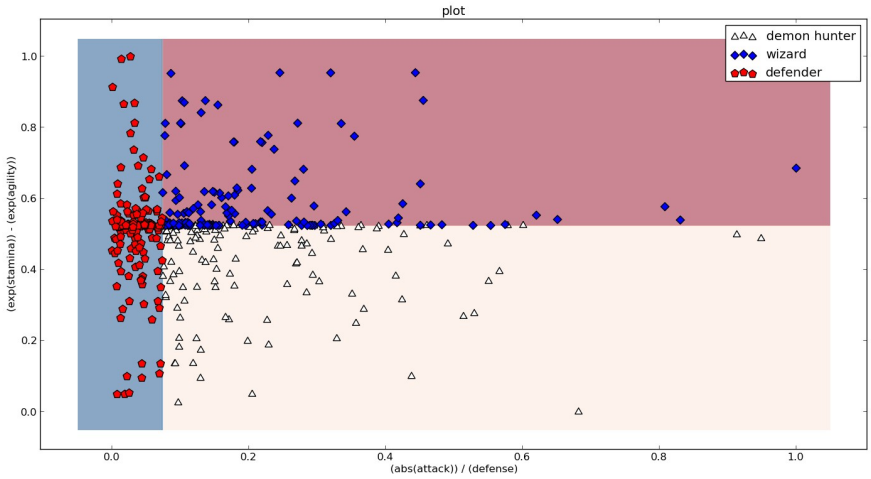
BAB 1
PENDAHULUAN

1.1 Latar Belakang

Ekstraksi fitur merupakan salah satu hal yang paling berpengaruh dalam pemecahan masalah klasifikasi. Pemilihan fitur yang tidak baik akan mengakibatkan kesulitan dalam memisahkan kelas-kelas data. Kegagalan pemisahan kelas-kelas data akan berdampak pada turunnya akurasi dalam proses klasifikasi.

Dalam penelitian sebelumnya (Gunawan, 2012), telah dicoba suatu pendekatan ekstraksi fitur dengan menggunakan *grammatical evolution*. Dalam penelitian tersebut, terdapat sebuah kelemahan dikarenakan hanya dilakukan 1 tolak ukur global untuk pengukuran *fitness value*. Hal ini mengakibatkan fitur-fitur yang sebenarnya cukup baik secara khusus, justru tersingkirkan karena nilai *fitness* globalnya rendah. Penelitian-penelitian lain seperti (Gavrilis, 2006; Gavrilis, 2008) juga menggunakan satu nilai *fitness* terhadap satu set fitur. (Guo, 2011; Li, 2011) juga melakukan hal yang hampir sama terhadap kasus yang berbeda.

Penelitian ini mengusulkan suatu cara baru dalam penilaian *fitness*. Penilaian *fitness* tersebut akan dilakukan dengan cara mengukur keterpisahan satu kelas terhadap kelas-kelas lain pada tiap dimensi. Metode tersebut, selanjutnya dinamakan *Tatami* karena kemiripannya dengan bentuk lantai tradisional Jepang. Dalam metode ini, untuk memisahkan *n* buah kelas, maka dibutuhkan maksimal *n-1* buah fitur (dimensi).



Gambar 1.1 Pemisahan cluster dengan metode Tatami

Pada gambar 1.1, terdapat tiga buah cluster yang masing-masing direpresentasikan dengan warna merah, biru dan putih. Untuk memisahkan ketiga cluster tersebut dibutuhkan 2 buah fitur (dimensi). Dimensi horizontal bertugas untuk memisahkan cluster merah dan kedua cluster lain. Pada dimensi horizontal ini, cluster biru dan putih tidak terpisahkan. Sedangkan pada dimensi vertikal, cluster biru dan putih terpisahkan, walaupun kedua cluster tersebut tidak terpisah dari cluster merah. Dengan menggunakan kedua dimensi ini, maka akan terbentuk ruang fitur baru di mana cluster merah, biru dan putih terpisah secara linear.

Pada akhir proses, diharapkan akan ditemukan sejumlah fitur terbaik, yang dapat membantu *classifier* (*decision tree*) untuk memisahkan kelas-kelas yang ada secara optimal.

1.2 Perumusan Masalah

Dalam penelitian ini, masalah-masalah yang akan diselesaikan dirumuskan sebagai berikut:

1. Bagaimana menentukan fitur-fitur optimal untuk masalah klasifikasi dengan menggunakan *decision tree classifier*.
2. Bagaimana menerapkan skenario pengukuran *fitness* untuk semua kelas
3. Bagaimana melakukan pengujian atas fitur-fitur yang sudah di-*generate*

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah:

1. Data yang diproses adalah data numerik
2. Data yang diproses tidak memiliki *missing attribute*.
3. *Grammar* yang digunakan hanya meliputi operator dan fungsi-fungsi matematika umum (+, -, *, /, *exp*, *abs*, *sqr*, dan *sqrt*).

1.4 Tujuan Penelitian

Menghasilkan dan menguji suatu metode baru (GE Tatami) yang berbasis *grammatical evolution* untuk mengekstrasi fitur pada data numerik, khususnya yang menggunakan aturan percabangan dalam penentuan kelasnya.

1.5 Manfaat Penelitian

Hasil penelitian dapat digunakan sebagai salah satu langkah *preprocessing* sebelum melakukan proses klasifikasi data numerik dengan menggunakan *decision tree*.

Dengan melakukan ekstraksi fitur sebagai bagian dalam tahap *preprocessing*, diharapkan proses klasifikasi data yang tidak memiliki korelasi

langsung terhadap kelas dapat dilakukan dengan lebih baik. Contoh penggunaan disajikan dalam bab empat.

BAB 2
KAJIAN PUSTAKA DAN DASAR TEORI

Pada bab ini akan dibahas beberapa teori dasar yang menunjang dalam pembuatan Tesis.

2.1 Ekstraksi Fitur

Ekstraksi fitur adalah suatu proses untuk mencari transformasi atau pemetaan dari fitur-fitur original ke ruang fitur baru yang dapat memperbesar keterpisahan antar kelas (Guo, 2011).

Banyak peneliti menyetujui bahwa ekstraksi fitur adalah proses terpenting dan tersulit pada masalah pengenalan pola dan klasifikasi. Pemilihan fitur yang paling tepat, mungkin merupakan tugas tersulit dalam pengenalan pola (Micheli-Tzanakou, 2000). Ekstraksi fitur yang ideal akan menghasilkan sebuah representasi yang sangat memudahkan pekerjaan *classifier* (Duda, Hart, & Stork, 2001). Dalam banyak kasus, ekstraksi fitur dilakukan oleh manusia, berdasarkan pengetahuan atau pengalaman, bahkan intuisi para peneliti (Guo, 2011)

Adapun fitur-fitur hasil ekstraksi bisa dikatakan baik, jika berhasil memisahkan data berdasarkan kelas yang diharapkan dengan tingkat kesalahan sekecil mungkin.

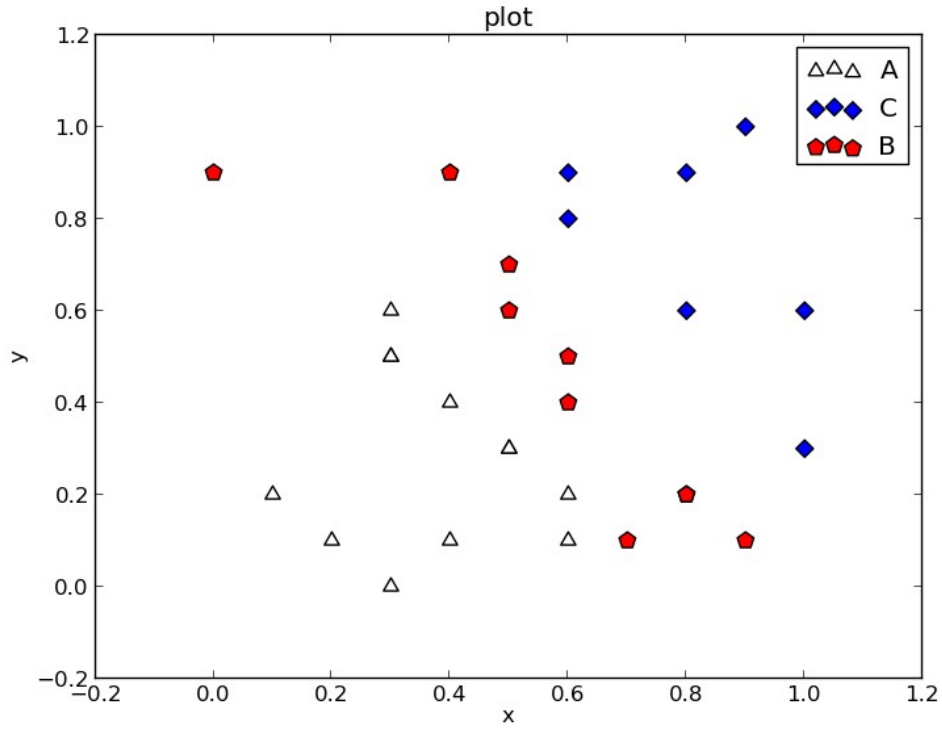
Untuk menjelaskan tujuan dari ekstraksi fitur, pada tabel 2.1 ditampilkan contoh data numerik. Data tersebut terdiri dari 2 fitur original, yakni x dan y. Masing-masing baris dalam tabel digolongkan dalam 3 buah kelas, yakni A, B dan C.

Tabel 2.1. Contoh Data numerik

Fitur original		kelas
x	**y**	
0.3	0.5	A
0.4	0.9	B
0.6	0.2	A
0.9	1.0	C
1.0	0.3	C
0.8	0.2	B
...	...	...

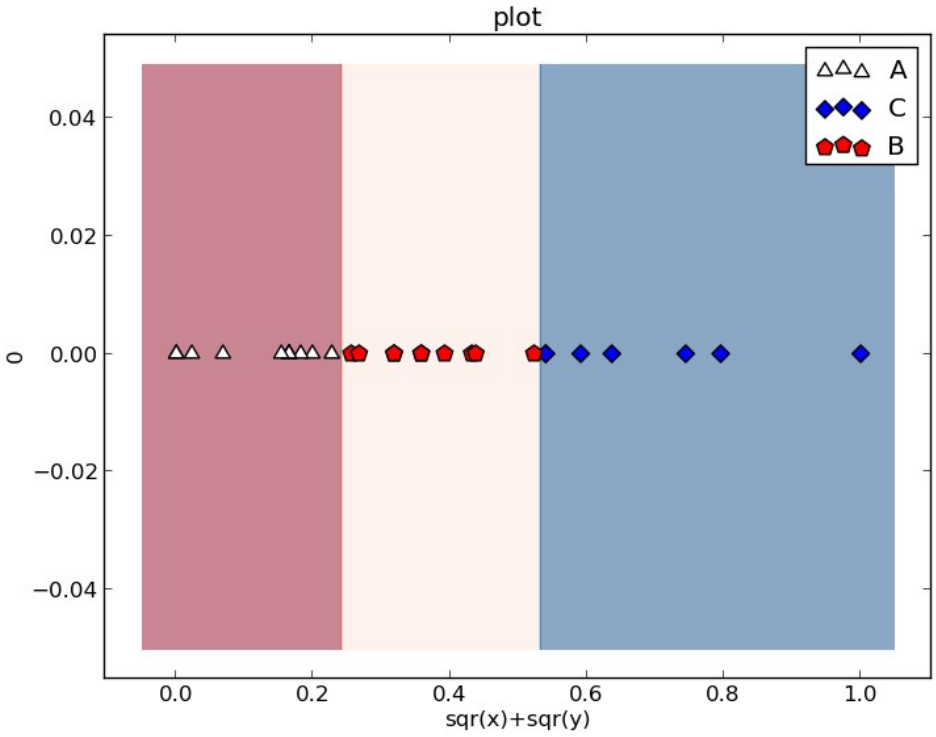
Jika data numerik pada tabel 2.1 dapat pula direpresentasikan dalam bentuk grafis seperti yang disajikan pada gambar 2.1 dengan fitur x sebagai dimensi horizontal, dan y sebagai dimensi vertikal, maka akan tampak bahwa penggunaan dimensi x dan y dapat menciptakan ruang fitur yang sanggup memisahkan kelas A, B dan C.

Adapun demikian, penggunaan dimensi x dan y secara terpisah akan mengakibatkan data-data pada kelas A,B dan C saling overlap (menempati posisi yang sama). Sebagai contoh, terdapat data dari kelas A, B dan C yang sama-sama memiliki nilai x = 0,6



Gambar 2.1 Representasi grafik dari Tabel 2.1

Proses ekstraksi fitur diharapkan mampu menciptakan sesedikit mungkin fitur yang dapat memisahkan kelas-kelas secara cukup baik. Fitur yang dihasilkan dari proses ekstraksi dapat merupakan suatu fungsi matematika yang menggunakan subset dari fitur-fitur original sebagai komponennya, semisal x^2+y^2. Gambar 2.2 menunjukkan bahwa penggunaan fitur x^2+y^2 ternyata telah cukup untuk memisahkan kelas A, B dan C.



Gambar 2.2 Representasi grafik data tabel 2.1 terhadap fitur x^2+y^2

2.2 Grammatical Evolution

Grammatical Evolution adalah pengembangan dari Genetics Programming (yang merupakan pengembangan dari algoritma genetika), yang merupakan suatu algoritma untuk mendapatkan satu set program dalam bahasa terntentu. Dengan memanfaatkan *context-free grammar* yang ditulis dalam *Backus Naur form*, *grammatical evolution* mampu memisahkan representasi fenotip dan genotip dalam individu (Harper, Blair, 2006).

Pada *grammatical evolution*, sebuah individu memiliki dua buah representasi. Representasi yang pertama adalah representasi genotip, sedangkan representasi yang kedua adalah representasi fenotip.

Representasi genotip berupa sekumpulan angka sebagaimana layaknya pada algoritma genetika. Genotip dapat berupa angka biner maupun desimal. Representasi genotip pada *grammatical evolution* akan ditransformasikan menjadi representasi fenotip.

Representasi fenotip pada *grammatical evolution* dapat berupa fungsi matematika, kode program komputer, atau apapun, tergantung pada grammar yang digunakan.

Sama halnya seperti dalam algoritma genetika, pada *grammatical evolution* juga terdapat *fitness function* untuk mengukur kebaikan dari setiap individu. Untuk kasus ekstraksi fitur, umumnya tingkat akurasi *classifier* digunakan sebagai *fitness function*.

2.2.1. Grammar Pada Grammatical Evolution

Untuk mentransformasikan representasi genotip menjadi representasi fenotip dibutuhkan sebuah *grammar*. *Grammar* di sini sebenarnya mirip dengan *grammar* dalam bahasa natural. Hanya saja, direpresentasikan dalam bentuk *backus naur form* (BNF). Dalam sebuah *grammar* terdapat beberapa bagian penting, antara lain:

- T : Terminal set. Merupakan node-node yang sudah tidak mungkin dievolusikan
- N : Non-terminal set. Merupakan node-node yang masih mungkin dievolusikan

- P : Production rules. Merupakan keseluruhan *grammar*
- S :Start symbol. Merupakan salah satu anggota N yang digunakan sebagai node

Tabel 2.2. Contoh Grammar

Node Notation	Node	Aturan Produksi	Notasi Aturan
(A)	<expr>	<expr><op><expr>	(A1)
		<num>	(A2)
		<var>	(A3)
(B)	<op>	+	(B1)
		-	(B2)
		*	(B3)
		/	(B4)
(C)	<var>	x	(C1)
		y	(C2)
(D)	<num>	1	(D1)

Semisal, didefinisikan production rules (P) seperti pada tabel 2.2, maka +, -, *, /, x, y, 1 merupakan anggota dari himpuan Terminal Set (T). Node-node yang menjadi anggota T, merupakan node-node yang sudah tidak mungkin dapat dievolusikan. Sementara itu <expr>, <op>, <var>, <num> digolongkan sebagai Non-terminal Set (N). Node-node tersebut masih mungkin berevolusi menjadi node lain. Node <expr> berfungsi sebagai start symbol (S), artinya node <expr> merupakan node awal.

2.2.2. Transformasi Genotip ke Fenotip pada Grammatical Evolution

Transformasi genotip ke fenotip memanfaatkan *grammar* yang ada dan operasi modulo (sisa bagi) untuk memilih aturan transofrmasi.

Semisal terdapat representasi genotip 11.01.00.10.01, maka proses untuk mendapatkan fenotipnya dapat digambarkan secara lengkap pada tabel 2.3.

Tabel 2.3. Proses Transformasi Genotip ke Fenotip

Before	Gene	Rule	After Transformation
<expr>	11 -> 3	<expr><op><expr>	<expr><op><expr>
<expr>	01 -> 1	<num>	<num><op><expr>
<num>	-	1	1<op><expr>
<op>	00 -> 0	+	1+<expr>
<expr>	10 -> 2	<var>	1+<var>
<var>	01 -> 1	y	1+y

Proses transformasi diawali dengan start symbol (dalam hal ini <expr>). Selanjutnya diambil sebuah segmen dari genotiip (dalam hal ini 11). Segmen tersebut dapat pula dinyatakan dalam bilangan decimal (dalam hal ini 3). Node <expr> memiliki 3 kemungkinan perubahan (A0 : <expr><op><expr>, A1:<num>, dan A2:<var>). Untuk menentukan aturan mana yang akan digunakan, maka dilakukan operasi modulo (sisa bagi), di mana segmen genotip terpilih akan dibagi dengan jumlah kemungkinan evolusi. Karena 3 mod 3 = 0, maka dipilihlah aturan A0, yakni <expr><op><expr>. Proses ini dilanjutkan terus sampai seluruh node telah bertransformasi menjadi anggota terminal set (T).

Dalam contoh transformasi di tabel 2.3, diperoleh representasi fenotip dari 11.01.00.10.01 adalah 1+Y

2.3 Decision Tree

Decision Trees (DTs) merupakan salah satu metode yang umum digunakan untuk masalah regresi dan klasifikasi. Metode ini akan menghasilkan sebuah model yang dapat digunakan untuk menebak nilai variabel target. Model yang dihasilkan *decision tree* sebenarnya merupakan aturan inferensi sederhana yang didapat dari fitur-fitur yang ada (Pedregosa, dkk, 2011).

Decision tree memiliki kecenderungan overfit yang tinggi, selain itu korelasi tiap atribut terhadap keterpisahan kelas memegang peranan yang sangat penting. Karakteristik ini menyebabkan ekstraksi fitur menjadi salah satu hal yang berdampak pada akurasi *decision tree*.

BAB 3
METODE PENELITIAN

3.1 Langkah-Langkah Penelitiann

Pada penelitian ini, terdapat beberapa tahapan penyelesaian yang akan dilakukan, yang masing-masing tahapan menggunakan suatu metode tertentu. Adapun tahapan dan metode yang digunakan adalah sebagai berikut (gambar 3.1):

1. Studi literatur dan pencarian dataset

   Proses ini terdiri atas pencarian referensi-referensi pendukung yang sesuai, baik dari buku, jurnal, maupun artikel. Proses tersebut dilanjutkan dengan pencarian data-data numerik yang tersedia di internet sesuai dengan batasan permasalahan. Selain data-data umum, juga akan dibuat beberapa data sintesis yang dibuat dengan program *spreadsheet*.

2. Menyusun *grammar* dan rancang bangun sistem

   Proses ini terdiri atas perancangan formula *grammar* dan algoritma umum dalam proses ekstraksi fitur. Untuk proses ekstraksi fitur, akan digunakan lima macam metode yang akan dibahas pada subbab 3.2.2:

   1. Metode GA Select
   2. Metode GE Global
   3. Metode GE Multi
   4. Metode GE Tatami
   5. Metode GE Gavrilis

3. Menyusun rancangan pengujian sistem

   Dalam proses ini ditentukan skenario pengujian. Pengujian yang dimaksud dapat berupa pembandingan hasil klasifikasi dengan ekstraksi fitur dalam penelitian ini, ekstraksi fitur dalam penelitian sebelumnya, dan tanpa

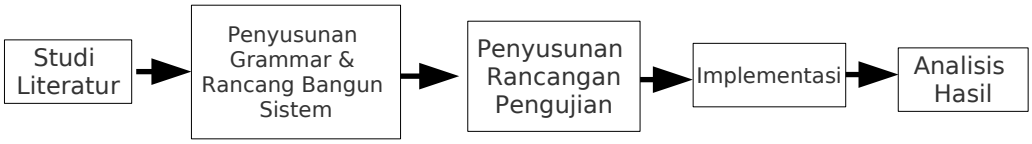
ekstraksi fitur. Akurasi klasifikasi menggunakan *decision-tree classifier* akan digunakan sebagai perbandingan. Khusus pada dataset synthesis 03, akan digunakan SVM dengan kernel linear sebagai pembanding.

4. Mengimplementasikan sistem

   Sistem akan dibuat dalam bahasa pemrograman *Python* yang umum digunakan dalam kepentingan penelitian.

5. Menganalisis hasil yang diperoleh untuk menghasilkan kesimpulan

   Hasil ekstraksi fitur pada langkah nomor 4 akan diuji sesuai dengan rancangan pada langkah no 3. Selanjutnya akan disimpulkan apakah hasil penelitian lebih baik dari penelitian sebelumnya. Jika tidak lebih baik, maka akan dianalisis penyebab kegagalannya.



Gambar 3.1 Skema Metode Penelitian

3.2 Rancangan Sistem

Secara umum, algoritma yang akan digunakan adalah sebagai berikut:

1. Input dataset

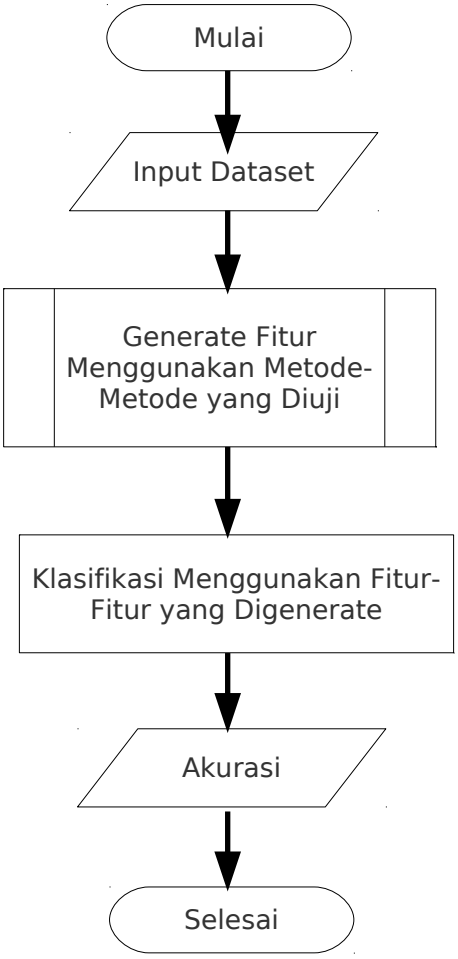
2. Menggunakan metode-metode pada subbab 3.1 langkah ke 2 guna mengekstraksi fitur

3. Generate genotip

   1. Transform genotip menjadi fenotip (fitur-fitur baru), sesuai dengan aturan *grammar* yang disediakan

2. Hitung nilai *fitness* dari setiap fenotip (fitur-fitur baru) terhadap setiap kelas.
3. Pilih fitur-fitur terbaik

4. Membangun *feature space* berdasarkan fitur-fitur terbaik, dan melakukan proses klasifikasi.



Gambar 3.2. Flowchart Sistem

3.2.1. Pembuatan Fitur

Proses pembuatan fitur dilakukan dengan menggunakan *grammatical evolution*. Proses ini ditujukan untuk membuat sebanyak mungkin calon fitur yang akan dinilai tingkat *fitness* nya.

Proses ini dimulai dengan pendefinisian *grammar*. *Grammar* yang telah didefinisikan, kemudian akan digunakan untuk mentransformasi sejumlah genotip yang dihasilkan secara random menjadi sejumlah fenotip. Setiap fenotip akan dihitung nilai *fitness* nya. Selanjutnya semua fenotip akan diurutkan berdasarkan nilai *fitness*. Detail tahapan yang diperlukan untuk pembuatan fitur adalah sebagai berikut:

3.2.1.1. Pendefinisian Grammar

Dalam metode *grammatical evolution*, pendefinisian *grammar* merupakan bagian yang cukup penting. Pendefinisian *grammar* akan menentukan berbagai kemungkinan terciptanya fenotip. Setiap fenotip yang tercipta akan menjadi fitur-fitur baru yang siap dievaluasi berdasarkan nilai *fitness* nya.

Grammar yang digunakan dalam penelitian ini adalah sebagai berikut (diimplementasikan dalam bahasa pemrograman *Python*):

```
1.  self.variables = self.features
2.  self.grammar = {
3.          '<expr>' : ['<var>', '<stmt>'],
4.          '<stmt>' : ['(<expr>) <op>
    (<expr>)','<func>(<expr>)','sqrt(sqr(<expr>+<expr>)/2)'],
5.          '<var>'  : self.variables,
6.          '<op>'   : ['+','-','*','/'],
7.          '<func>' : ['exp','abs','sigmoid','sqr','sqrt','-']
8.      }
```

Gambar 3.3 *Grammar* yang digunakan

Variabel *self.variables* berisi fitur-fitur data original. Pada *self.grammar* didefinisikan bahwa <expr> dapat berevolusi menjadi <var>, (<expr>) <op> (<expr>), atau <func>(<expr>). Sedangkan <var> dapat berevolusi menjadi fitur-fitur original. Demikian pula dengan <op> yang dapat berevolusi menjadi operator-operator matematika dan <func> yang dapat berevolusi menjadi salah satu dari fungsi-fungsi matematika terdefinisi. Proses evolusi sendiri akan bermula dari node <expr>

3.2.1.2. Pembuatan Fitur

Proses pembuatan fitur tak lain adalah transformasi genotip (deretan angka acak yang telah di-*generate*) ke dalam bentuk fenotip menggunakan *grammatical evolution* dengan *grammar* terdefinisi. Proses ini telah dibahas dalam subbab 2.2.2. Pada implementasinya, proses ini didefinisikan dengan sebuah fungsi yang mengembalikan fitur baru dalam format data *string*.

```
1.  def _transform(self, gene):
2.      # caching:
3.      if gene in self.genotype_dictionary:
4.          return self.genotype_dictionary[gene]
5.      # kedalaman maksimum = 20 (mencegah infinite loop)
6.      depth = 20
7.      gene_index = 0
8.      expr = self._start_node
9.      # dimulai dari level 0
10.     level = 0
11.     while level < depth:
12.         i=0
13.         new_expr = ''
14.         # parsing setiap karakter pada expr
15.         while i<len(expr):
16.             found = False
17.             for key in self._grammar:
18.                 # ubah keyword berdasarkan akturan produksi
19.                 if (expr[i:i+len(key)] == key):
20.                     found = True
21.                     # jumlah kemungkinan transformasi
22.                     possibility = len(self._grammar[key])
23.                     # jumlah digit biner utk possibility
```

```
24.                     digit_needed =
    utils.bin_digit_needed(possibility)
25.                     # jika akhir gen sudah tercapai
26.                     if(gene_index+digit_needed)>len(gene):
27.                         # mulai dari depan lagi
28.                         gene_index = 0
29.                     # bagian gen utk transformasi
30.                     used_gene =
    gene[gene_index:gene_index+digit_needed]
31.                     gene_index = gene_index + digit_needed
32.                     rule_index = utils.bin_to_dec(used_gene)
    %possibility
33.                     new_expr += self._grammar[key][rule_index]
34.                     i+= len(key)-1
35.             if not found:
36.                 new_expr += expr[i:i+1]
37.             i += 1
38.         expr = new_expr
39.         level = level+1
40.     # tambahkan ke cache
41.     self.genotype_dictionary[gene] = expr
42.     return expr
```

Gambar 3.4 Fungsi Transformasi untuk Membuat Fitur

Fungsi *_transform* menerima parameter *gene* yang bertipe data string dan berisi deretan angka biner. Kemudian dengan menggunakan parameter *gene* dan *grammar* yang telah didefinisikan sebelumnya, di-*generate* sebuah fitur (fenotip) baru, Fenotip tersebut berupa string yang berisi potongan kode program dalam bahasa Python .

3.2.1.3. Normalisasi Proyeksi Data Berdasarkan Fitur Baru

Fitur yang telah di-*generate* pada subbab sebelumnya, selanjutnya digunakan untuk memproyeksikan data original. Proses ini didefinisikan dalam fungsi get_projection.

Untuk setiap record data yang ada, dilakukan proses evaluasi (didefinisikan pada utils.execute). Proses ini akan mengembalikan sebuah tupple yang berisi angka hasil evaluasi dan status error. Jika status error bernilai benar,

maka ada kemungkinan bahwa hasil evaluasi tidak berupa angka (*Nan* atau *None*). Untuk meminimalisasi error hasil proyeksi, maka jika terjadi error, hasil evaluasi akan diasumsikan sebagai -1.

Selanjutnya, untuk semua data yang berhasil dievaluasi (tidak memunculkan error) akan dilakukan proses normalisasi. Proses normalisasi ini bertujuan untuk mengubah nilai minimum menjadi 0 dan nilai maksimum menjadi 1. Proses normalisasi ini bertujuan untuk memastikan bahwa setiap fitur memiliki *range* yang sama atau hampir sama.

Hasil proyeksi data direpresentasikan dalam bentuk *dictionary* dengan kelas sebagai *key*, dan list hasil proyeksi kelas tersebut sebagai *value*.

```
1.  def get_projection(new_feature, old_features, all_data, used_data =
2.  None, used_target = None):
3.      used_projection, all_result, all_error = [], [], []
4.      # get all result
5.      for data in all_data:
6.          result, error = utils.execute(new_feature, data,
    old_features)
7.          all_error.append(error)
8.          if error:
9.              all_result.append(None)
10.         else:
11.             all_result.append(result)
12.     all_is_none = True
13.     for i in all_result:
14.         if i is not None:
15.             all_is_none = False
16.             break
17.     if all_is_none:
18.         min_result = 0
19.         max_result = 1
20.     else:
21.         min_result = min(x for x in all_result if x is not None)
22.         max_result = max(x for x in all_result if x is not None)
23.     if max_result-min_result>0:
24.         result_range = max_result-min_result
25.     else:
26.         result_range = LIMIT_ZERO
27.     if used_data is None: # include all data
28.         for i in xrange(len(all_result)):
29.             if all_error[i]:
```

```
30.                 all_result[i] = -1
31.             else:
32.                 all_result[i] = (all_result[i]-
    min_result)/result_range
33.         used_projection = all_result
34.     else:
35.         used_result = []
36.         for i in xrange(len(used_data)):
37.             result, error = utils.execute(new_feature, used_data[i],
    old_features)
38.             if error:
39.                 used_result.append(-1)
40.             else:
41.                 used_result.append((result-min_result)/result_range)
42.         used_projection = used_result
43.     # pastikan isi projection hanya berupa angka (int atau float)
44.     for i in xrange(len(used_projection)):
45.         value = used_projection[i]
46.         if (not isinstance(value,float)) and (not
    isinstance(value,int)):
47.             value = -1
48.         if math.isnan(value):
49.             value = -1
50.         used_projection[i] = round(value,2)
51.     if used_target is None:
52.         return used_projection
53.     group_projection = {}
54.     for i in xrange(len(used_projection)):
55.         group = used_target[i]
56.         if not group in group_projection:
57.             group_projection[group]=[]
58.         group_projection[group].append(used_projection[i])
59.     return group_projection
```

Gambar 3.5 Fungsi Proyeksi dan Normalisasi Data

3.2.2. Pemilihan Fitur Terbaik

Pemilihan fitur terbaik diperoleh dengan memanfaatkan lima buah metode. GE Multi dan GE Tatami merupakan metode yang diusulkan, sedangkan metode GA Select, GE Global dan GE Gavrilis digunakan sebagai pembanding.

Untuk pengukuran *fitness* individu, digunakan formula *(true_positive/ (true_positive+false_negative)+ true_negative/(true_negative+false_positive)) – 1*. *True positive* adalah jumlah data yang oleh *classifier* diprediksi berada di dalam kelas tertentu dan ternyata memang benar berada dalam kelas tersebut. *True*

negative adalah jumlah data yang oleh *classifier* diprediksi tidak berada di dalam kelas tertentu dan ternyata memang benar tidak berada dalam kelas tersebut. *False positive* adalah jumlah data yang oleh *classifier* diprediksi berada di dalam kelas tertentu namun ternyata tidak berada dalam kelas tersebut. *False negative* adalah jumlah data yang oleh *classifier* diprediksi tidak berada di dalam kelas tertentu namun ternyata berada dalam kelas tersebut.

Penjelasan rinci mengenai masing-masing metode disajikan dalam subbab berikut:

3.2.2.1. Metode GA Select

Dalam metode GA Select, akan dipilih subset dari fitur original yang paling mampu memisahkan kelas dalam data secara optimum. Penilaian *fitness* dilakukan dengan memanfaatkan akurasi separator. Dalam implementasinya, untuk skenario ini digunakan algoritma genetika biasa.

Setiap individu dalam GA Select terdiri dari binary string. Jika karakter pertama pada binary string adalah 1, maka fitur asli pertama digunakan, sebaliknya jika karakter pertama pada binary string adalah 0, maka fitur pertama tidak digunakan. Demikian untuk karakter-karakter selanjutnya.

Semisal sebuah individu terdiri dari binary string 01110, dan terdapat 5 fitur f1-f5, maka:

- f1 tidak digunakan, karena karakter pertama adalah 0
- f2 digunakan, karena karakter kedua adalah 1
- f3 digunakan, karena karakter ketiga adalah 1
- f4 digunakan, karena karakter keempat adalah 1
- f5 tidak digunakan, karena karakter kelima adalah 0

Banyaknya fitur yang dapat di-*generate* dengan metode ini berkisar antara nol sampai dengan jumlah fitur original.

3.2.2.2. Metode GE Global

Dalam metode GE global, akan dipilih sebuah fitur yang mampu memisahkan semua kelas secara cukup baik. Penilaian *fitness* dilakukan dengan cara mengukur akurasi classifier terhadap data dengan menggunakan fenotip yang di-*generate* oleh grammatical evolution (proses grammatical evolution dijelaskan pada subbab 2.2). Metode GE Global merupakan implementasi dari penelitian sebelumnya (Gunawan, 2012)

Metode ini diharapkan menghasilkan 1 fitur terbaik yang dapat memisahkan semua kelas dalam data.

3.2.2.3. Metode GE Multi

Metode ini merupakan pengembangan dari GE Global. Dalam GE Multi, digunakan pengukuran *multi fitness* untuk memisahkan masing-masing kelas dengan keseluruhan kelas lain. Setiap individu dalam metode ini akan memiliki *n* buah nilai *fitness*, di mana *n* adalah jumlah kelas yang ada.

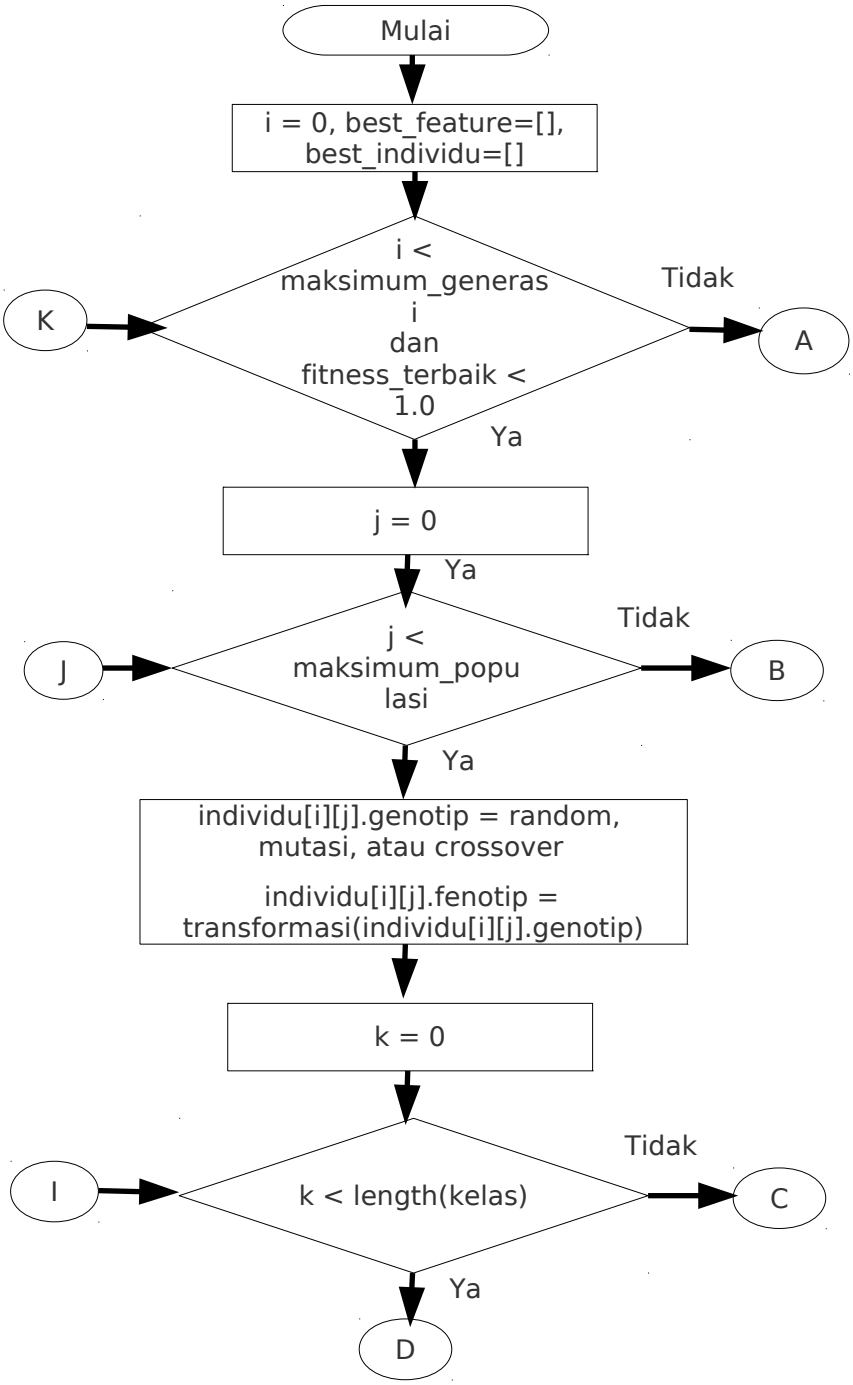
Banyaknya fitur yang bisa di-*generate* dalam metode ini adalah sebanyak jumlah kelas yang ada.

Semisal dalam sebuah dataset terdapat n buah kelas {C1, C2, C3,... Cn}, maka dalam GE Multi, akan terdapat *n fitness value* {f1, f2, f3,... fn}. Masing-masing fitness value menunjukkan keterpisahan sebuah kelas terhadap semua kelas lain. Dalam proses penentuan *fitness value*, kelas-kelas yang tidak selain kelas target akan disatukan kedalam satu kelas. Kemudian dilakukan proses klasifikasi dengan *classifier decision tree*. Proses ini dilakukan sebanyak n kali, sehingga diperoleh n fitness value untuk masing-masing individu.

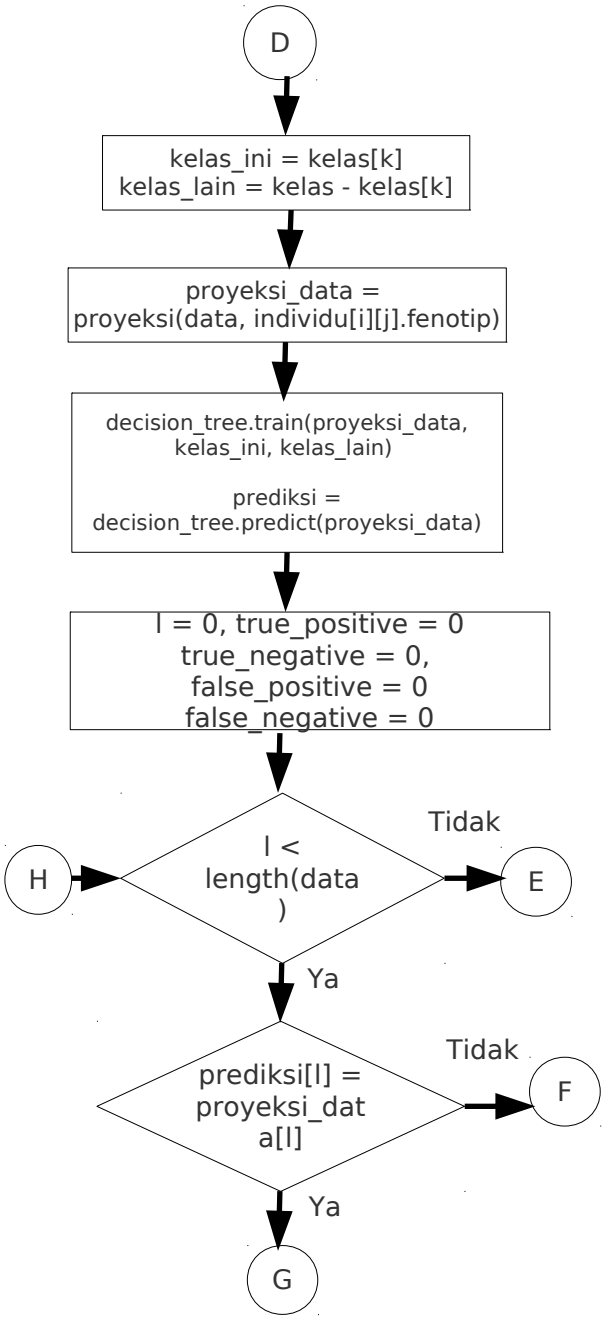
Untuk menghitung f1, maka kelas {C2, C3,.. Cn} digabungkan menjadi C~1. Classifier akan memisahkan C1 dan C~1. Selanjutnya, akurasi classifier akan dijadikan nilai fitness f1. Untuk menghitung f2, maka kelas {C1, C3,.. Cn} digabungkan menjadi C~2. Classifier akan memisahkan C2 dan C~2. Selanjutnya, akurasi classifier akan dijadikan nilai fitness f2. Demikian seterusnya sampai fn.

Di akhir proses GE Multi, dipilih satu individu yang memiliki f1 terbaik, 1 individu yang memiliki f2 terbaik, dan seterusnya, sehingga ditemukan n individu terbaik. Fenotip dari individu-individu terbaik ini selanjutnya digunakan sebagai fitur baru untuk proses klasifikasi.

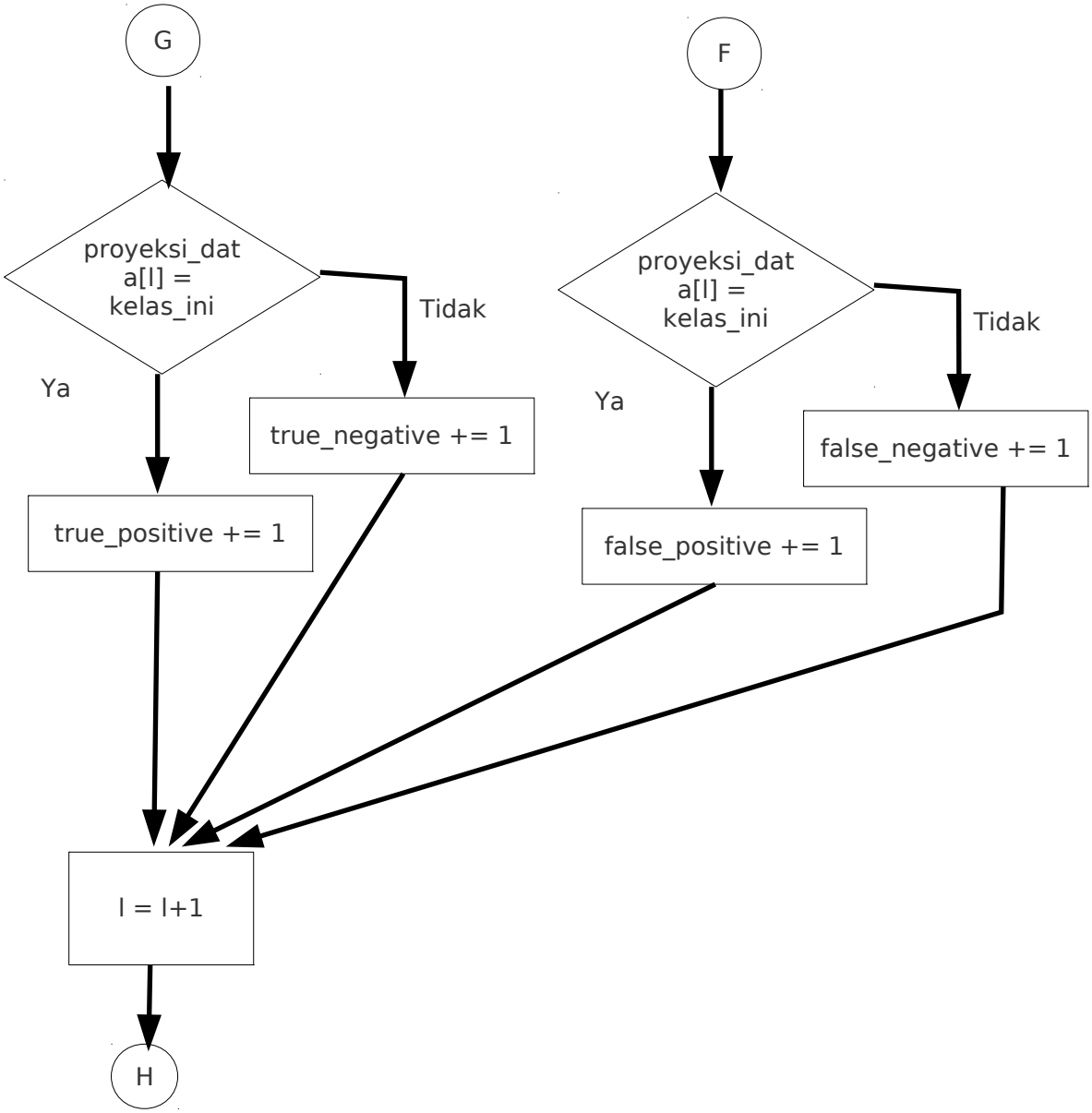
Flowchart metode GE Multi disajikan pada gambar 3.6 sampai gambar 3.9.



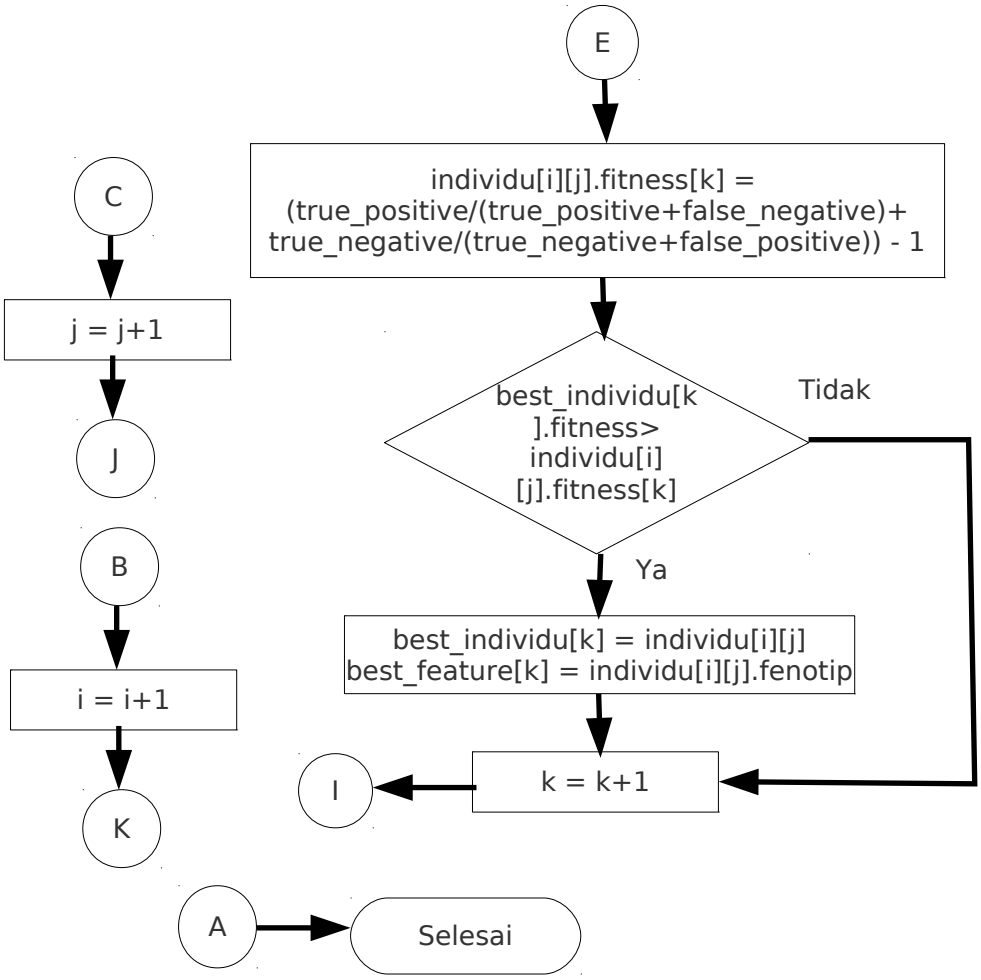
Gambar 3.6 Flowchart GE Multi (Bagian 1)



Gambar 3.7 Flowchart GE Multi (Bagian 2)



Gambar 3.8 Flowchart GE Multi (Bagian 3)



Gambar 3.9 Flowchart GE Multi (Bagian 4)

3.2.2.4. Metode GE Tatami

Metode GE Tatami merupakan pengembangan dari GE Multi. Pengembangan tersebut berasal dari hipotesa bahwa jika sebuah kelas telah terpisah dari semua kelas lainnya, maka pada proses selanjutnya, kelas yang sudah terpisah tersebut bisa diabaikan. Dalam skenario ini, akan tercipta *n-1* buah fitur.

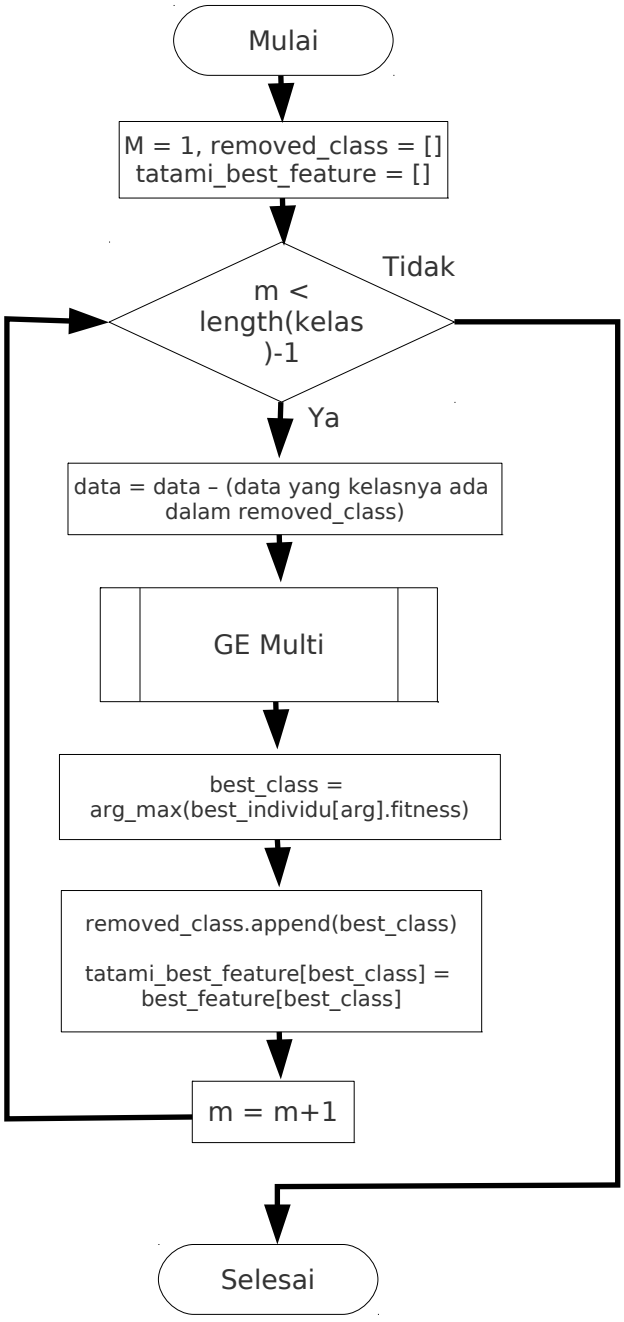
Dasar pemikiran yang melandasi GE Tatami adalah bahwa n-1 buah dimensi sebenarnya telah cukup untuk memisahkan n kelas. Dalam ruang fitur yang terbentuk, akan dibutuhkan n-1 buah garis linear sederhana untuk memisahkan n kelas tersebut. Gambar 1.1 pada Bab I menunjukkan bagaimana 3 buah kelas dapat terpisah dalam sebuah ruang fitur dua dimensi.

Dalam GE Tatami, untuk *n* buah kelas{C1, C2, C3,... Cn}, dilakukan proses GE Multi untuk mencari individu-individu terbaik. Seperti yang telah dibahas dalam subbab sebelumnya, setiap individu dalam GE Multi memiliki n buah fitness value {f1, f2, f3,... fn}. Di sini dipilih satu fitness value terbesar dari semua individu. Fitness value terbesar dari semua individu merepresentasikan kelas yang paling terpisah dari kelas lain. Kelas ini selanjutnya disimbolkan sebagai C*1. Fenotip dari individu terbaik dengan nilai fitness tertinggi selanjutnya disebut F1, dan merupakan bagian dari fitur baru yang akan digunakan dalam proses klasifikasi.

Dalam proses selanjutnya, data-data dalam kelas C*1 dihilangkan, sehingga diperoleh subset data baru yang terdiri dari {C1, C2, C3,... Cn} – C*1. Subset data baru ini akan memiliki n-1 buah kelas. Kemudian kembali dilakukan GE Multi seperti sebelumnya. Perulangan kedua ini akan menghasilkan F2 yang juga merupakan bagian dari fitur baru yang akan digunakan dalam proses klasifkiasi. Perulangan akan dilakukan sebanyak n-1 kali. Dalam setiap

perulangan, jumlah kelas yang terlibat akan berkurang satu, sehingga pada perulangan ke n-1 hanya akan tersisa 2 kelas yang kemudian dipisahkan oleh Fn-1.

Flowchart GE Tatami disajikan pada gambar 3.10.



Gambar 3.10 Flowchart GE Tatami

3.2.2.5. Metode GE Gavrilis

Metode GE Gavrilis merupakan implementasi dari penelitian yang dilakukan oleh Gavrilis (Gavrilis, 2011). Dalam metode ini akan di-*generate* set-set fitur yang masing-masing diwakili oleh satu individu. Berbeda dengan GE Multi dan GE Tatami, di sini satu individu mewakili satu set fitur.

Pengukuran *fitness* dalam metode ini dilakukan dengan mengukur akurasi *classifier* secara empiris.

Jumlah fitur yang di-*generate* dalam GE Gavrilis akan berkisar antara nol sampai tak terhingga.

3.2.3. Pengukuran Performa Fitur

Sebagai *classifier*, digunakan *Decision Tree* yang merupakan salah satu algoritma umum dalam permasalahan klasifikasi. Semua metode yang telah dibahas pada subbab sebelumnya akan digunakan untuk men-*generate* sekumpulan fitur baru. Fitur-fitur tersebut akan digunakan sebagai data baru bagi *Decision Tree*. Diharapkan metode GE Tatami akan memperoleh hasil yang lebih baik dibandingkan dengan metode-metode lain.

Dalam pengujian akan digunakan berbagai macam data. Selain data-data sintesis yang sengaja dibuat untuk menguji hipotesa, percobaan juga akan dilakukan pada dataset iris, e.coli, dan balanced-scale yang telah umum dipakai dalam penelitian-penelitian sejenis. Data-data non-sintesis yang digunakan didapatkan dari website UCI-Machine Learning (http://archive.ics.uci.edu/ml/)

BAB 4
HASIL PENELITIAN DAN PEMBAHASAN

Percobaan diimplementasikan dengan menggunakan bahasa pemrograman Python 2.7 dan beberapa library eksternal. Adapun library eksternal yang digunakan adalah scipy, numpy, matplotlib dan scikit-learn.

Source code program dan hasil lengkap pengujian telah diletakkan di repository *github*. Repository tersebut berlisensi open-source dan bisa diakses secara publik di alamat https://github.com/goFrendiAsgard/feature-extractor dengan lisensi GNU, sehingga bebas dimodifikasi dan digunakan guna penelitian lebih lanjut.

Dalam percobaan yang dilakukan terdapat beberapa metode (GA Select, GE Global, GE Multi, GE Tatami dan GE Gavrilis) yang diujikan pada berbagai macam data. Selain itu, disertakan pula akurasi classifier tanpa ekstraksi fitur sebagai pembanding. Setiap data diuji dengan menggunakan *5 fold cross validation*.

4.1 Pengujian Terhadap Dataset Sintesis 01

Untuk kepentingan uji coba penelitian, maka dibuat beberapa buah dataset sintesis menggunakan aplikasi spreadsheet. Dalam penelitian ini digunakan libre-office.

Pada dataset sintesis 01, terdapat 3 buah kelas, yakni *defender*, *demon hunter* dan *wizard*. Ketiga kelas tersebut didapatkan dengan melakukan kalkulasi berdasarkan 4 fitur (*defense*, *attack*, *agility*, *stamina*). Keempat fitur yang ada bersifat random uniform dan memiliki range antara 0-10 dengan pembulatan satu angka di belakang koma. Dataset sintesis 01 terdiri dari 460 data yang terdiri dari

139 *defender*, 171 *demon hunter* dan 150 *wizard*. Data ini bisa diakses pada https://github.com/goFrendiAsgard/feature-extractor/blob/master/synthesis_01.csv.

Adapun Formula yang digunakan untuk menggolongkan kelas adalah sebagai berikut: *=IF(defense/attack >= 1.4, "defender", IF(agility >= stamina, "demon hunter", "wizard"))*. Pemilihan angka-angka pada formula semata-mata untuk membuat dataset *balanced* (memiliki jumlah data yang hampir sama untuk semua kelas). Formula tersebut dimaksudkan untuk membuat data yang hirarkikal. Melalui perbandingan fitur *defense* dan *attack*, kelas *defender* terpisah dari kedua kelas lainnya (*demon hunter* dan *wizard*). Melalui perbandingan fitur *agility* dan *stamina*, kelas *demon hunter* terpisah dari kelas *wizard*.

Tabel 4.1 Pengujian Pada Dataset Sintesis 01

Percobaan		**Tanpa Ekstraksi Fitur**		**Ekstraksi Fitur: GA Select**		**Ekstraksi Fitur: GE Global**		**Ekstraksi Fiture: GE Multi**		**Ekstraksi Fitur: GE Tatami**		**Ekstraksi Fitur: GE Gavrilis**	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Un-fold	Train	73.0	4	73.0	3	77.8	1	99.3	3	**100**	2	85.6	3
	Test	73.0		73.0		77.8		99.3		**100**		85.6	
	Total	73.0		73.0		77.8		99.3		**100**		85.6	
Fold 1	Train	75.9	4	75.9	3	78.9	1	**100**	3	**100**	2	84.5	61
	Test	67.7		67.7		35.5		76.6		81.1		**83.3**	
	Total	74.3		74.3		70.4		95.4		**96.3**		84.3	
Fold 2	Train	73.7	4	73.7	3	78.9	1	**100**	3	**100**	2	**100**	48
	Test	70.0		70.0		36.7		74.4		65.6		**80.0**	
	Total	73.0		73.0		70.6		95.0		93.2		**96.0**	
Fol	Train	71.6	4	71.6	3	77.5	1	**100**	3	**100**	2	85.1	3

Percobaan		Tanpa Ekstraksi Fitur		Ekstraksi Fitur: GA Select		Ekstraksi Fitur: GE Global		Ekstraksi Fiture: GE Multi		Ekstraksi Fitur: GE Tatami		Ekstraksi Fitur: GE Gavrilis	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
d 3	Test	75.5		75.5		25.5		86.6		**86.7**		**86.7**	
	Total	72.3		72.3		67.3		97.3		**97.4**		85.4	
Fold 4	Train	73.5	4	73.5	3	78.6	1	**100**	3	**100**	2	85.4	3
	Test	74.4		74.4		36.6		**77.7**		**77.7**		76.6	
	Total	73.7		73.7		70.4		**95.6**		**95.6**		83.7	
Fold 5	Train	75.6	4	75.6	3	81.0	1	**100**	3	**100**	2	87.0	2
	Test	70.0		42.2		42.2		**94.4**		82.2		72.2	
	Total	74.5		73.4		73.5		**98.9**		96.5		84.1	

Hasil pengujian menunjukkan bahwa GE Multi dan GE Tatami memeberikan hasil yang cukup baik dengan jumlah fitur yang relatif sedikit. Pada fold 2, GE Gavrilis memperoleh akurasi tertinggi, namun memiliki jumlah fitur yang sangat banyak.

4.2 Pengujian Terhadap Dataset Sintesis 02

Data sintesis 02 memiliki struktur yang hampir sama dengan data sintesis 01. Pada dataset ini terdapat 4 fitur dan 4 kelas.

Pada dataset sintesis 02, terdapat 3 buah kelas, yakni *defender*, *demon hunter*, *monk* dan *wizard*. Ketiga kelas tersebut didapatkan dengan melakukan kalkulasi berdasarkan 4 fitur (*defense*, *attack*, *agility*, *stamina*). Keempat fitur yang ada bersifat random uniform dan memiliki range antara 0-10 dengan pembulatan satu angka di belakang koma. Dataset sintesis 02 terdiri dari 613 data yang terdiri

dari 184 *defender*, 148 *demon hunter*, 135 *monk* dan 146 *wizard*. Data ini bisa diakses pada alamat https://github.com/goFrendiAsgard/feature-extractor/blob/master/synthesis_02.csv

Formula yang digunakan pada dataset sintesis 02 adalah sebagai berikut: *=IF(defense/attack >= 1.6, "defender", IF(agility/stamina >= 1.3, "demon hunter", IF(stamina>defense, "monk", "wizard")))*. Sama seperti pada dataset sintesis 01, formula untuk membuat kelas pada data sintesis 02 juga dimaksudkan untuk membuat data yang hirarkikal. Melalui perbandingan fitur *defense* dan *attack*, kelas *defender* terpisah dari ketiga kelas lainnya (*demon hunter*, *monk* dan *wizard*). Melalui perbandingan fitur *agility* dan *stamina*, kelas *demon hunter* terpisah dari kelas *monk* dan *wizard*. Terakhir, melalui perbandigan fitur *stamina* dan *defense*, kelas *monk* dan *wizard* terpisah satu sama lain. Dibandingkan dengan data sintesis 01, data sintesis 02 ini memiliki struktur hirarkikal yang lebih dalam, dikarenakan jumlah kelasnya lebih banyak.

Tabel 4.2 Pengujian Pada Dataset Sintesis 02

Percobaan		**Tanpa Ekstraksi Fitur**		**Ekstraksi Fitur: GA Select**		**Ekstraksi Fitur: GE Global**		**Ekstraksi Fiture: GE Multi**		**Ekstraksi Fitur: GE Tatami**		**Ekstraksi Fitur: GE Gavrilis**	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Un-fold	Train	77.9	4	77.9	4	70.8	1	**100**	4	**100**	3	90.3	12
	Test	77.9		77.9		70.8		**100**		**100**		90.3	
	Total	77.9		77.9		70.8		**100**		**100**		90.3	
Fold 1	Train	78.7	4	78.7	4	73.2	1	99.4	4	**100**	3	89.8	12
	Test	**76.0**		**76.0**		33.1		46.3		62.8		72.7	
	Total	78.1		78.1		65.2		88.9		**92.6**		86.4	

Percobaan		Tanpa Ekstraksi Fitur		Ekstraksi Fitur: GA Select		Ekstraksi Fitur: GE Global		Ekstraksi Fiture: GE Multi		Ekstraksi Fitur: GE Tatami		Ekstraksi Fitur: GE Gavrilis	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Fold 2	Train	76.4	4	76.4	4	70.5	1	**100**	4	**100**	3	89.4	12
	Test	77.6		77.6		27.2		72.7		**74.3**		85.9	
	Total	76.6		76.6		61.9		94.6		**94.9**		88.7	
Fold 3	Train	79.6	4	79.6	4	71.7	1	99.3	3	**100**	3	90.0	12
	Test	68.6		68.6		34.7		64.4		**83.4**		68.6	
	Total	77.4		77.4		64.4		92.5		**96.7**		85.8	
Fold 4	Train	79.0	4	79.0	4	70.7	1	**100**	4	**100**	3	90.2	12
	Test	72.7		72.7		40.5		71.0		61.1		**80.1**	
	Total	77.8		77.8		64.7		**94.2**		92.3		88.2	
Fold 5	Train	78.0	4	78.0	4	71.7	1	**100**	4	**100**	3	86.9	12
	Test	73.5		73.5		32.2		83.4		**94.2**		70.2	
	Total	77.1		77.1		63.9		96.7		**98.8**		83.6	

Pada Data sintesis 02, GE Multi tidak lagi memberikan performa sebaik pada dataset sintesis 01. Hal tersebut disebabkan karena dengan semakin banyaknya kelas, pemisahan dengan skenario one vs all akan menjadi semakin sulit. Sebaliknya GE Tatami justru menunjukkan hasil yang lebih baik, dikarenakan struktur hirarkikal yang lebih tampak.

4.3 Pengujian Terhadap Dataset Sintesis 03

Dataset sintesis 03 merupakan dataset ideal untuk GE Tatami. Dataset ini terdiri dari 400 data yang terdiri dari 91 kelas A, 81 kelas B, 81 kelas C, 59 kelas

D, dan 88 kelas E. Dalam dataset ini terdapat 5 kelas, A, B, C, D dan E dan 7 fitur (f1, f2, f3, f4, f5, n1, dan n2). Fitur n1 dan fitur n2 adalah fitur noise yang bersifat random uniform dengan ketelitian 1 angka di belakang koma. Keberadaan kedua fitur noise tersebut dimaksudkan untuk menguji apakah metode-metode yang ada dapat memfilter dan menghilangkan noise. Fitur f1 sampai f5 merupakan fitur utama yang diperoleh melalui serangkaian perhitungan.

Pembagian data ke dalam 5 kelas ditentukan berdasarkan 4 fitur tersembunyi m1, m2, m3, dan m4. Fitur m1, m2, m3 dan m4 bersifat random uniform dan memiliki range antara 0-10 dengan ketelitian 1 angka di belakang koma. Penentuan kelas menggunakan formula sebagai berikut: *=IF(m1<0.5, "A", IF(m2<0.5, "B", IF(m3<0.5, "C", IF(m4<0.5, "D","E"))))*. Fitur m1 memisahkan kelas A dengan keempat kelas lainnya (B, C, D, dan E). Fitur m2 memisahkan kelas B dengan ketiga kelas lain (C, D, dan E). Fitur m3 memisahkan kelas C dari kelas D dan E. Terakhir, fitur m4 berfungsi untuk memisahkan kelas D dan E.

Seperti yang telah diungkapkan, fitur m1 sampai m4 tidak ditunjukkan secara eksplisit kepada sistem, melainkan disembunyikan secara implisit melalui formula-formula matematis sederhana ke dalam lima fitur tampak f1 sampai f5. Hal ini dimaksudkan untuk menguji kemampuan GE Tatami dalam memperoleh kembali fitur-fitur utama tersembunyi (m1-m4) berdasarkan fitur-fitur tampak. Proses penyembunyian m1-m4 ke dalam f1-f5 dilakukan sebagai berikut:

- f1 diperoleh secara acak dengan formula =ROUND(RAND()*9.9+0.1,3)
- f2 diperoleh dengan menggunakan rumus f2 = f1/m1
- f3 diperoleh dengan menggunakan rumus f3 = f2/m2
- f4 diperoleh dengan menggunakan rumus f4 = m3/f1
- f5 diperoleh dengan menggunakan rumus f5 = f4/m4

Diharapkan GE Tatami akan berhasil menemukan m1-m4 dengan menggunakan konstruksi matematis dari f1-f5. Iterasi pertama dalam GE Tatami diharapkan mampu menemukan f1/f2 atau bentuk lain yang sebanding dengan m1 (f2 = f1/m1 sehingga m1 = f1/f2). Fitur ini seharusnya mampu memisahkan kelas A dengan keempat kelas lainnya (B, C, D dan E). Pada iterasi kedua, GE Tatami diharapkan mampu menemukan f2/f3 atau bentuk lain yang sebanding dengan m2 (f3 = f2/m2 sehingga m2 = f2/f3). Demikian seterusnya hingga m4-m5 atau fitur-fitur yang sebanding dengan itu ditemukan.

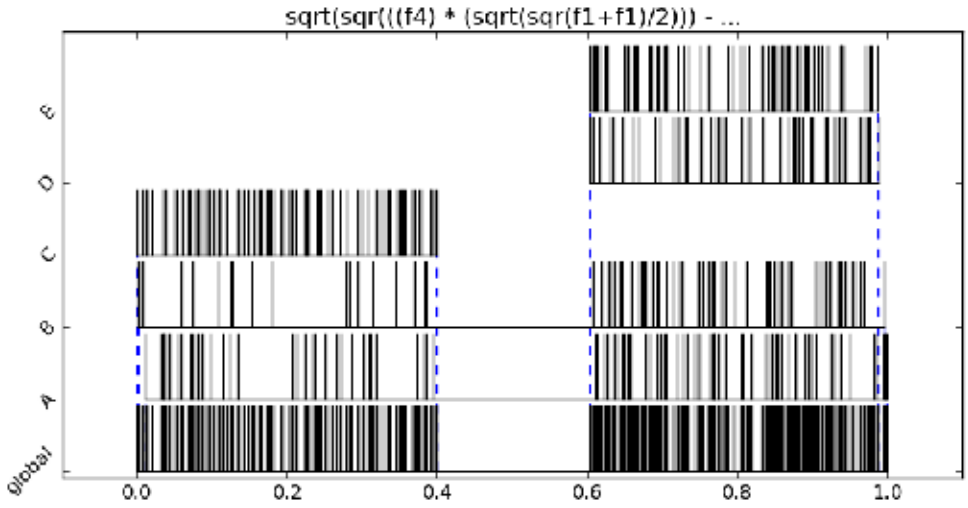
Data sintesis 03 dapat diakses melalui alamat https://github.com/goFrendiAsgard/feature-extractor/blob/master/synthesis_03.csv

Hasil pengujian menunjukkan bahwa untuk dataset sintesis 03, GE Tatami menunjukkan hasil yang sangat baik.

Berikut adalah keempat fitur yang berhasil di generate oleh GE Tatami pada skenario unfold

- (f2) / (f1)
- (f1) / (f3)
- sqrt(sqr(((f4) * (sqrt(sqr(f1+f1)/2))) - (n1)+n1)/2)
- (f5) / (f4)

Tiga dari keempat fitur tersebut bisa dikatakan sebanding dengan m1-m4. m1 = f2/f1, m2 = f1/f3, dan m4 = f5/f4. Sementara itu, fitur *sqrt(sqr((((f4) * (sqrt(sqr(f1+f1)/2))) – (n1)+n1)/2)* walaupun tidak sama dengan m3, namun masih bisa memisahkan kelas C dari kelas D dan E, seperti ditunjukkan dalam gambar 4.1, sehingga fitur ini pun bisa dikatakan baik.



Ga

mbar 4.1 Proyeksi data terhadap fitur

sqrt(sqr(((f4) * (sqrt(sqr(f1+f1)/2))) – (n1)+n1)/2)

Tabel 4.3 Pengujian Pada Dataset Sintesis 03

Percobaan		**Tanpa Ekstraksi Fitur**		**Ekstraksi Fitur: GA Select**		**Ekstraksi Fitur: GE Global**		**Ekstraksi Fiture: GE Multi**		**Ekstraksi Fitur: GE Tatami**		**Ekstraksi Fitur: GE Gavrilis**	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Un-fold	Train	72.5	7	72.5	6	67.0	1	98.2	5	**100**	4	80.7	6
	Test	72.5		72.5		67.0		98.2		**100**		80.7	
	Total	72.5		72.5		67.0		98.2		**100**		80.7	
Fold 1	Train	74.4	7	74.4	6	67.9	1	99.0	5	**100**	4	82.5	6
	Test	26.5		26.5		27.8		45.5		**63.2**		32.9	
	Total	65.0		65.0		60.0		88.5		**92.7**		72.7	
Fol	Train	72.9	7	72.9	6	66.9	1	100.	5	**100.**	4	86.2	47

Percobaan		Tanpa Ekstraksi Fitur		Ekstraksi Fitur: GA Select		Ekstraksi Fitur: GE Global		Ekstraksi Fiture: GE Multi		Ekstraksi Fitur: GE Tatami		Ekstraksi Fitur: GE Gavrilis	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
d 2	Test	62.0		62.0		27.8		68.3		**94.9**		69.6	
	Total	70.7		70.7		59.2		93.7		**99.0**		83.0	
Fold 3	Train	71.3	7	71.3	6	69.4	1	99.6	5	**100**	4	83.1	6
	Test	29.1		29.1		24.0		55.7		**65.8**		46.8	
	Total	63.0		63.0		60.5		91.0		**93.2**		76.0	
Fold 4	Train	72.9	7	72.9	4	66.9	1	98.4	5	**100**	4	73.5	2
	Test	26.5		27.8		26.5		50.6		**78.4**		25.3	
	Total	63.7		64.0		59.0		89.0		**95.7**		64.0	
Fold 5	Train	72.2	7	72.2	6	68.8	1	99.3	5	**100**	4	85.6	47
	Test	24.0		25.3		48.1		59.4		**63.2**		49.3	
	Total	62.7		63.0		64.7		91.5		**92.7**		78.5	

4.4 Pengujian Terhadap Dataset Iris

Dataset iris merupakan dataset yang cukup banyak dipakai dalam penelitian. Data ini terdiri dari 3 kelas (Iris-Setosa, Iris-Versicolor, dan Iris-Virginica) serta 4 atribut fitur original (Sepal Length, Sepal Width, Petal Length, dan Petal Width) yang memiliki ketelitian 1 angka di belakang koma. Dataset iris bersifat multi-variate, terdiri dari 150 data (50 iris-setosa, 50 iris-versicolor, dan 50 iris-virginica). Dataset iris dapat didownload dari website UCI Machine Learning dengan alamat (http://archive.ics.uci.edu/ml/datasets/Iris)

Hasil pengujian terhadap data iris menunjukkan hasil yang hampir seimbang untuk GE Global, GE Multi, GE Tatami dan GE Gavrilis. Namun

tampak bahwa GE Gavrilis memberikan hasil yang sedikit lebih unggul dibandingkan metode-metode lain.

Tabel 4.4 Pengujian Pada Dataset Iris

Percobaan		**Tanpa Ekstraksi Fitur**		**Ekstraksi Fitur: GA Select**		**Ekstraksi Fitur: GE Global**		**Ekstraksi Fiture: GE Multi**		**Ekstraksi Fitur: GE Tatami**		**Ekstraksi Fitur: GE Gavrilis**	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Un-fold	Train	96.0	4	96.0	2	**98.6**	1	**98.6**	3	**98.6**	2	**98.6**	1
	Test	96.0		96.0		**98.6**		**98.6**		**98.6**		**98.6**	
	Total	96.0		96.0		**98.6**		**98.6**		**98.6**		**98.6**	
Fold 1	Train	96.6	4	96.6	2	**99.1**	1	**99.1**	3	98.3	2	**99.1**	1
	Test	86.6		86.6		**96.6**		**96.6**		96.6		**96.6**	
	Total	94.6		94.6		**98.6**		**98.6**		98.0		**98.6**	
Fold 2	Train	95.8	4	95.8	2	**100**	1	99.1	3	99.1	2	**100**	3
	Test	**96.6**		**96.6**		**96.6**		83.3		66.6		**96.6**	
	Total	96.0		96.0		**99.3**		96.0		92.6		**99.3**	
Fold 3	Train	96.6	4	96.6	2	98.3	1	98.3	3	**99.1**	2	98.3	1
	Test	93.3		93.3		76.6		**100**		**100**		**100**	
	Total	96.0		96.0		94.0		98.6		**99.3**		98.6	
Fold 4	Train	95.8	4	95.8	2	**99.1**	1	**99.1**	3	**99.1**	2	**99.1**	1
	Test	96.6		96.6		93.3		**96.6**		**96.6**		**96.6**	
	Total	96.0		96.0		98.0		**98.6**		**98.6**		**98.6**	
Fold 5	Train	96.6	4	96.6	2	98.3	1	98.3	3	**99.1**	2	**99.1**	3
	Test	93.3		93.3		93.3		93.3		**96.6**		**96.6**	
	Total	96.0		96.0		97.3		97.3		**98.6**		**98.6**	

4.5 Pengujian Terhadap Dataset E-Coli

Data E-Coli merupakan dataset yang cukup banyak dipakai dalam penelitian. Data ini terdiri dari 7 atribut (mcg, gvh, lip, chg, aac, alm1, alm2) dan 6 kelas (cp, im, imU, om, omL, pp), yang terdiri dari 335 record (143 cp, 77 im , 52 pp, 35 imU, 20 om, 5 omL, 2 imL, 2 imS). Masing-masing atribut memiliki ketelitian satu angka di belakang koma. Dataset ini merupakan klasifikasi terhadap berbagai varian dari bakteri E-Coli. Dataset E-Coli dapat didownload pada alamat (http://archive.ics.uci.edu/ml/datasets/Ecoli).

Berbeda dengan pengujian-pengujian sebelumnya, di sini justru GA Select menghasilkan akurasi yang paling tinggi dibandingkan keempat skenario lain. Adapun demikian, saat semua data digunakan untuk training sekaligus testing, GE Tatami tampak berhasil memberikan akurasi yang paling tinggi.

Tabel 4.5 Pengujian Pada Dataset E Coli

Percobaan		**Tanpa Ekstraksi Fitur**		**Ekstraksi Fitur: GA Select**		**Ekstraksi Fitur: GE Global**		**Ekstraksi Fiture: GE Multi**		**Ekstraksi Fitur: GE Tatami**		**Ekstraksi Fitur: GE Gavrilis**	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Un-fold	Train	97.0	7	97.0	7	84.5	1	96.7	8	**97.6**	7	97.0	12
	Test	97.0		97.0		84.5		96.7		**97.6**		97.0	
	Total	97.0		97.0		84.5		96.7		**97.6**		97.0	
Fold 1	Train	97.4	7	97.4	6	87.4	1	**98.8**	8	96.3	7	97.7	12
	Test	**73.8**		**73.8**		58.4		53.8		53.8		69.2	
	Total	**92.8**		**92.8**		81.8		90.1		88.1		92.2	
Fol	Train	96.6	7	96.6	5	86.7	1	97.7	8	**98.5**	7	97.7	12

Percobaan		Tanpa Ekstraksi Fitur		Ekstraksi Fitur: GA Select		Ekstraksi Fitur: GE Global		Ekstraksi Fiture: GE Multi		Ekstraksi Fitur: GE Tatami		Ekstraksi Fitur: GE Gavrilis	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
d 2	Test	**81.5**		78.4		63.0		49.2		1.54		58.4	
	Total	**93.7**		93.1		82.1		88.3		79.7		90.1	
Fold 3	Train	97.7	7	97.7	7	89.3	1	**99.2**	8	97.0	7	98.5	26
	Test	70.7		70.7		53.8		**80.0**		69.2		53.8	
	Total	92.5		92.5		82.4		**95.5**		91.6		89.8	
Fold 4	Train	95.9	7	96.3	5	87.0	1	**98.5**	8	97.4	7	98.1	12
	Test	**73.8**		**73.8**		69.2		63.0		43.0		56.9	
	Total	**91.6**		91.9		83.6		**91.6**		86.9		90.1	
Fold 5	Train	97.0	7	97.0	4	86.7	1	98.1	8	96.3	7	**98.8**	18
	Test	73.8		**75.3**		61.5		38.4		38.4		67.6	
	Total	97.4		97.4		87.4		**98.8**		96.3		97.7	

4.6 Pengujian Terhadap Dataset Balanced-Scale

Dataset balanced-scale terdiri dari 625 data yang masing-masing terdiri dari 4 fitur(left_weight, left_distance, right_weight, right_distance) dan 3 kelas(B, R, L). Masing-masing atribut bertipe bilangan bulat positif dengan range antara 1-5. Dataset ini terdiri dari 625 record yang terdiri dari 49 B, 288 L, dan 288 R. Dataset ini dibuat untuk tujuan pengujian psikologi dan dapat didownload pada alamat (http://archive.ics.uci.edu/ml/datasets/Balance+Scale).

Hasil pengujian menunjukkan bahwa secara umum GE Multi lebih unggul dibandingkan semua metode lain.

Tabel 4.4 Pengujian Pada Dataset Balanced-Scale

Percobaan		Tanpa Ekstraksi Fitur		Ekstraksi Fitur: GA Select		Ekstraksi Fitur: GE Global		Ekstraksi Fiture: GE Multi		Ekstraksi Fitur: GE Tatami		Ekstraksi Fitur: GE Gavrilis	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
Un-fold	Train	70.8	4	70.8	3	84.8	1	**91.6**	1	**91.6**	2	82.5	9
	Test	70.8		70.8		84.8		**91.6**		**91.6**		82.5	
	Total	70.8		70.8		84.8		**91.6**		**91.6**		82.5	
Fold 1	Train	68.7	4	70.9	3	**100**	1	**100**	1	92.0	2	81.0	4
	Test	67.4		70.7		**91.8**		**91.8**		85.3		81.3	
	Total	68.4		70.8		**98.4**		**98.4**		90.7		81.1	
Fold 2	Train	70.1	4	71.9	3	85.4	1	**92.2**	1	**92.6**	2	83.8	9
	Test	61.7		66.6		69.9		**89.4**		82.9		78.8	
	Total	68.4		70.8		82.4		**91.6**		90.7		82.8	
Fold 3	Train	69.3	4	70.5	3	90.0	1	**99.0**	2	92.0	2	83.6	126
	Test	65.0		72.3		66.6		**85.3**		71.5		81.3	
	Total	68.4		70.8		85.4		**96.3**		88.0		83.2	
Fold 4	Train	72.5	4	72.5	3	86.6	1	**100**	1	91.8	2	82.2	2
	Test	68.2		68.2		78.0		73.9		**91.0**		77.2	
	Total	71.6		71.6		84.9		**94.8**		91.6		81.2	
Fold 5	Train	71.1	4	71.1	3	84.6	1	94.6	3	**100**	2	82.8	1
	Test	69.9		69.9		**82.9**		57.7		66.6		51.2	
	Total	70.8		70.8		84.3		87.3		**93.4**		76.6	

4.7 Rata-rata akurasi

Guna mendapatkan gambaran performa secara umum, maka dilakukan perhitungan terhadap rata-rata akurasi dari semua kasus.

Tabel 4.5 Rata-rata akurasi

Percobaan		**Tanpa Ekstraksi Fitur**		**Ekstraksi Fitur: GA Select**		**Ekstraksi Fitur: GE Global**		**Ekstraksi Fiture: GE Multi**		**Ekstraksi Fitur: GE Tatami**		**Ekstraksi Fitur: GE Gavrilis**	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
iris	Train	96.2	4	96.2	2	98.9	1	98.8	3	98.9	2	**99.0**	2
	Test	93.7		93.7		92.5		94.7		92.5		**97.5**	
	Total	95.7		95.7		97.6		98.0		97.6		**98.7**	
balance-scale	Train	70.4	4	71.3	3	88.6	1	**96.2**	2	93.3	2	82.7	25
	Test	67.2		69.8		79.0		**81.6**		81.5		75.4	
	Total	69.8		71.0		86.7		**93.3**		91.0		81.2	
ecoli	Train	96.9	7	97.0	6	86.9	1	**98.2**	8	97.2	7	98.0	15
	Test	**78.4**		78.2		65.1		63.5		50.6		67.2	
	Total	**93.4**		**93.4**		82.7		91.5		88.2		92.0	
synthesis_01	Train	73.9	4	73.9	3	78.8	1	99.8	3	**100**	2	87.9	20
	Test	71.8		71.8		42.42		**84.8**		82.2		80.7	
	Total	73.5		73.5		71.7		**96.9**		96.5		86.5	
synthesis_02	Train	78.3	4	78.3	4	71.4	1	99.8	4	**100**	3	89.4	12
	Test	74.4		74.4		39.7		73.0		**79.3**		78.0	
	Total	77.5		77.5		65.2		94.5		**95.9**		87.2	

Percobaan		Tanpa Ekstraksi Fitur		Ekstraksi Fitur: GA Select		Ekstraksi Fitur: GE Global		Ekstraksi Fiture: GE Multi		Ekstraksi Fitur: GE Tatami		Ekstraksi Fitur: GE Gavrilis	
		Akurasi (%)	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**	**Akurasi (%)**	**Jml. Fitur**
synthesis_03	Train	72.7	7	72.7	6	67.8	1	99.1	5	**100**	4	81.9	19
	Test	40.1		40.5		36.9		63.0		**77.6**		50.8	
	Total	66.2		66.3		61.7		92.0		**95.5**		75.8	
All Average	Train	81.4	5	81.6	4	82.1	1	**98.6**	4	98.2	3	89.8	16
	Test	70.9		71.4		59.3		76.8		**77.3**		74.9	
	Total	79.3		79.6		77.6		**94.4**		94.1		86.9	

Rata-rata akurasi yang diperoleh dari pengujian terhadap keenam dataset menunjukkan bahwa GE Multi memberikan hasil terbaik, sementara GE Tatami menempati peringkat kedua.

Adapun GE Tatami menunjukkan keunggulan mutlak pada data data sintesis 02 dan data sintesis 03. Kedua dataset tersebut dapat terpisah secara hirarkikal berdasarkan fitur-fitur yang di-*generate*. Keduanya juga memiliki jumlah kelas yang relatif cukup banyak (lebih dari tiga). Ini menunjukkan bahwa GE Tatami unggul dalam meningkatkan akurasi decission tree pada data-data yang dapat terpisah secara hirarkikal.

4.8 Analisis Karakteristik GE Tatami

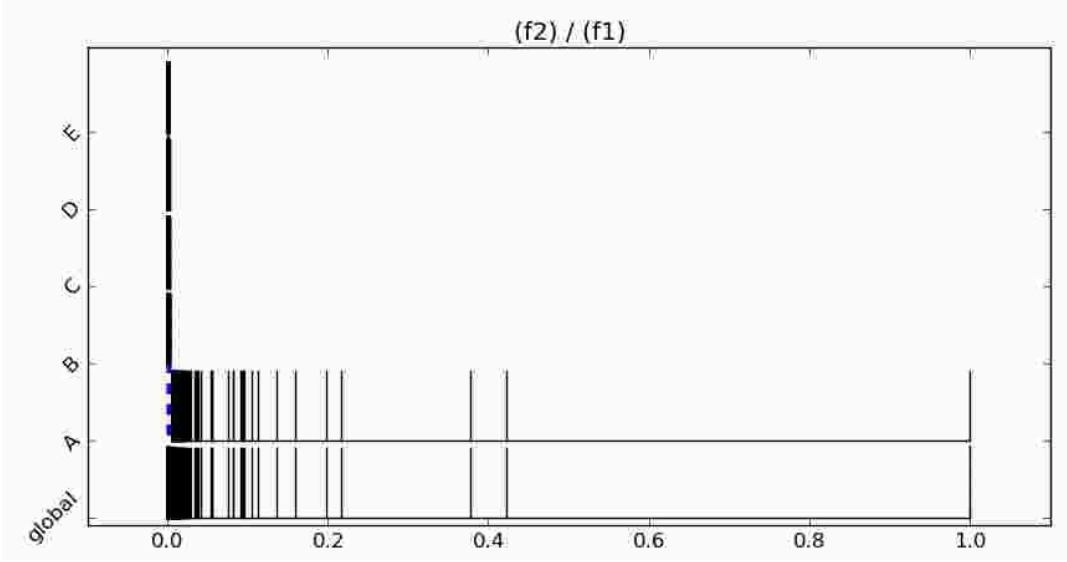
Dari hasil percobaan, tampak bahwa GE Tatami menunjukkan hasil yang cukup baik pada data-data sintesis dan data iris. Dalam hal ini proses *grammatical*

evolution berhasil menemukan fitur-fitur yang sanggup memisahkan data sesuai dengan hipotesis.

Pada dataset sintesis 03, tampak bahwa GE Tatami berhasil menemukan fitur-fitur yang sebanding dengan fitur asli (m1, m2, m3 dan m4) berdasarkan fitur-fitur tampak (f1, f2, f3, f4 dan f5). Hubungan antara fitur asli dan fitur tampak telah dijelaskan pada subbab 4.3. Adapun fitur-fitur yang berhasil di-*generate* oleh GE Tatami adalah sebagai berikut:

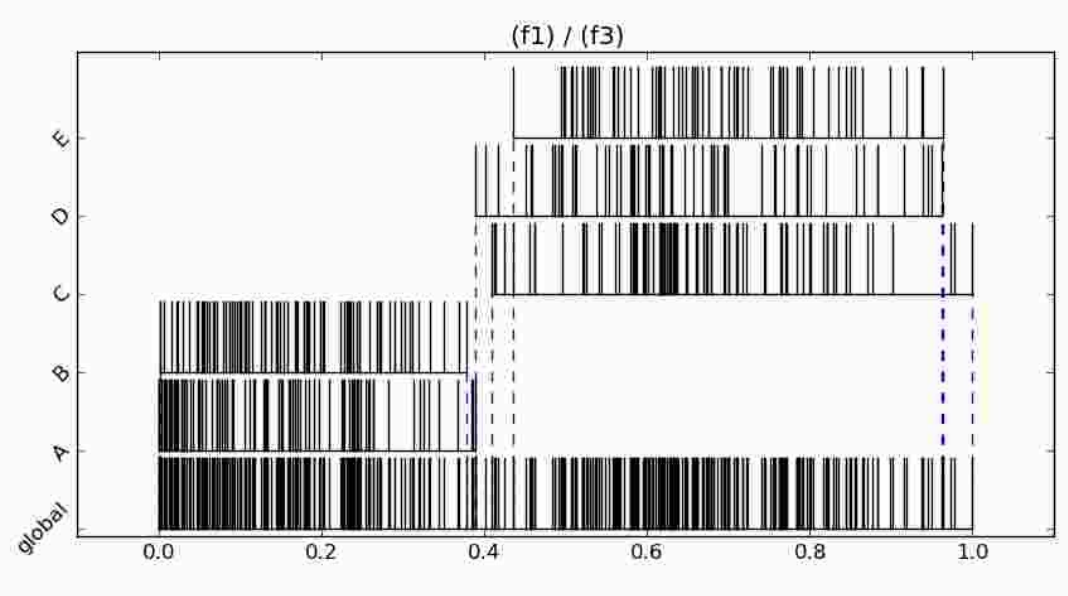
- (f2) / (f1)
- (f1) / (f3)
- sqrt(sqr(((f4) * (sqrt(sqr(f1+f1)/2))) - (n1)+n1)/2)
- (f5) / (f4)

Fitur (f2) / (f1) sanggup memisahkan kelas A dan keempat kelas lainnya. Hal ini tampak seperti pada gambar 4.2. Pada gambar 4.2, tampak bahwa kelas B, C, D dan E berimpit di sebelah kiri, terpisah secara linear dari kelas A. Panjang garis vertikal menunjukkan banyaknya data yang menempati nilai yang sama.



Gambar 4.2 Proyeksi Data terhadap fitur (f2) / (f1)

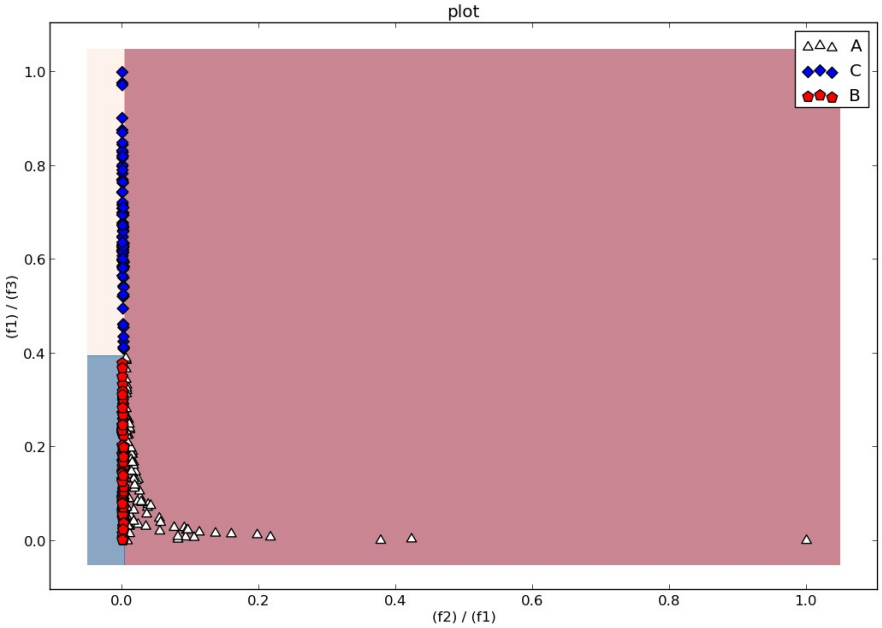
Setelah berhasil menemukan fitur (f2)/(f1) yang memisahkan kelas A dan keempat kelas lain (B, C, D, E), GE Tatami mencari fitur yang bisa memisahkan satu dari keempat kelas tersebut terhadap tiga kelas lainnya. Dalam proses ini ditemukan fitur (f1) / (f3). Proyeksi data ke dalam fitur (f1) / (f3) mengakibatkan kelas A tidak terpisah dengan kelas B. Namun hal ini tidak dipermasalahkan karena fitur sebelumnya, yakni (f2) / (f1) telah memisahkan kelas A dengan semua kelas lain.



Gambar 4.3 Proyeksi Data terhadap fitur (f1) / (f3)

Proses yang sama juga berlaku untuk langkah selanjutnya, sehingga ditemukan fitur sqrt(sqr(((f4) * (sqrt(sqr(f1+f1)/2))) – (n1)+n1)/2) yang memisahkan C dengan D dan E, serta fitur (f5) / (f4) yang memisahkan D dan E.

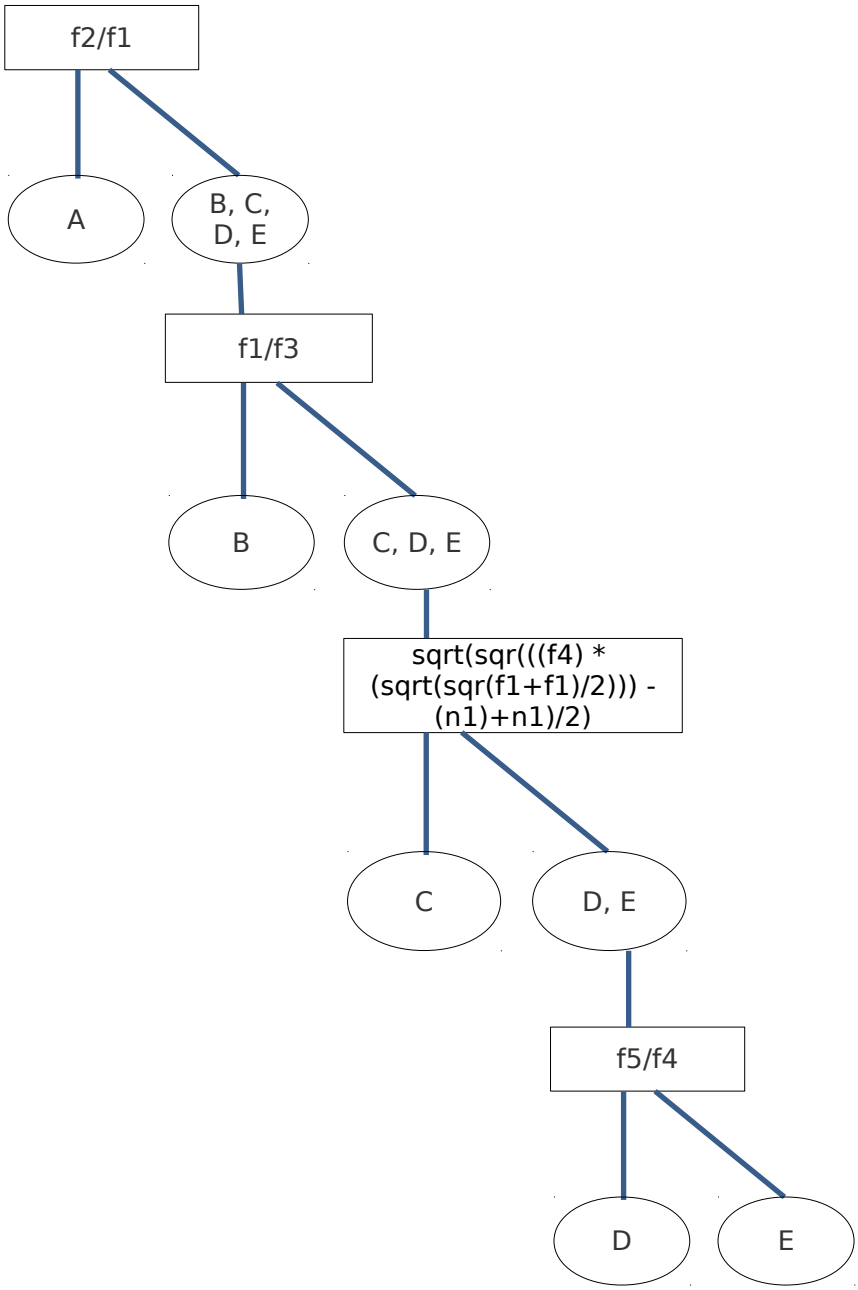
Skenario GE Tatami tampak berhasil menciptakan feature space yang ideal bagi *classifier Decision Tree*, seperti ditujukkan pada gambar 4.4. Fitur (f2) / (f1) dan (f3) / (f1), memberikan kemudahan bagi *decision tree* untuk membagi data sesuai dengan kelas yang ada.



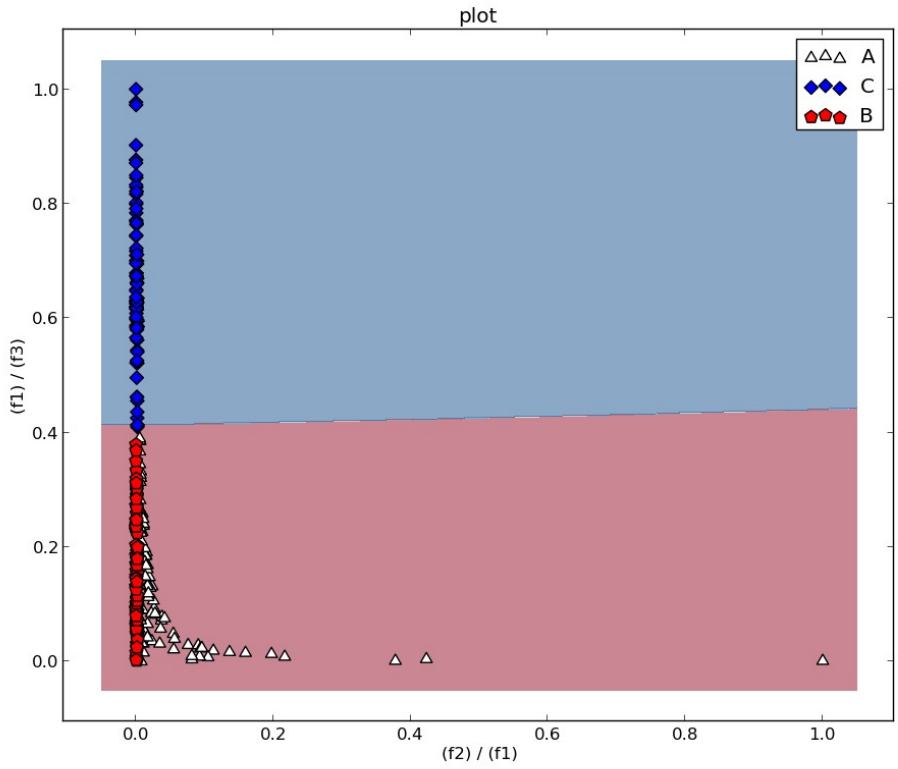
Gambar 4.4 Proyeksi Data terhadap Fitur (f2) / (f1) dan Fitur (f1) / (f3) dengan *Classifier Decission Tree*

Pendekatan GE Tatami ini sangat sesuai dengan karakteristik *Decision Tree* yang memisahkan data secara hirarkikal seperti digambarkan dalam gambar 4.5.

Sekalipun GE Tatami menunjukkan hasil yang baik jika dikombinasikan dengan *Decision Tree*, namun tampaknya penggunaan *classifier* lain semisal SVM. Proyeksi data yang sama jika menggunakan *classifier* SVM ditunjukkan pada gambar 4.6



Gambar 4.5 Decission Tree yang Terbentuk Berdasarkan Penggunaan Fitur-Fitur yang di-*generate* oleh GE Tatami.



Gambar 4.6 Proyeksi Data terhadap Fitur (f2) / (f1) dan Fitur (f1) / (f3) dengan *Classifier* SVM

Fenomena ini menunjukkan bahwa Feature Space yang di-*generate* oleh GE Tatami hanya akan membantu untuk *classifier* yang memanfaatkan keterpisahan data per fitur tanpa mempedulikan pola dan bentuk sebaran data.

4.9 Analisis Kelemahan GE Tatami

Pada semua kasus yang ada, GE Tatami menunjukkan hasil terbaik saat semua data digunakan sebagai training dan testing sekaligus. Namun pada

pengujian cross-validation didapati bahwa performa GE Tatami menurun. Terkadang (seperti dalam Fold-2 di dataset E-Coli), sekalipun akurasi yang diberikan dalam training sangat tinggi, akurasi testing nya sangat rendah. Ini menunjukkan bahwa GE Tatami cenderung membuat fitur-fitur yang overfit.

Kelemahan GE Tatami yang lain adalah jika terjadi kegagalan pada langkah pertama, maka langkah-langkah selanjutnya akan menjadi tidak efektif. Pada langkah pertama (*Predefined process* GE Multi pada flowchart di gambar 3.10), GE Tatami harus berhasil menciptakan fitur yang memisahkan satu kelas tertentu dengan semua kelas lainnya secara cukup baik. Jika hal ini gagal dilakukan, maka pada langkah selanjutnya kesalahan klasifikasi pada langkah awal akan sangat berpengaruh pada keseluruhan proses klasifikasi.

BAB 5
HASIL PENELITIAN DAN PEMBAHASAN

5.1 Kesimpulan

Hasil penelitian menunjukkan bahwa skenario GE Tatami berhasil membuat fitur-fitur yang membantu dalam proses klasifikasi untuk data sintesis dan iris. Namun metode tersebut gagal untuk menentukan fitur-fitur terbaik pada data ecoli. GE Tatami akan menghasilkan akurasi yang bagus jika ada proyeksi terhadap suatu fitur ter-*generate* yang menunjukkan keterpisahan menonjol antara satu kelas dengan semua kelas lainnya.

GE Tatami memberikan persyaratan yang lebih mudah dipenuhi daripada GE Global dan GE Multi. Pada GE Global, harus ter-*generate* sebuah fitur yang sanggup memisahkan setiap kelas. Pada GE Multil, harus ter-*generate* n buah fitur yang sanggup memisahkan n kelas dengan kelas-kelas lain. Sementara pada GE Tatami, jika ada satu kelas yang sudah berhasil dipisahkan dari kelas-kelas lain, maka untuk pencarian fitur berikutnya, kelas tersebut dapat diabaikan.

Walaupun GE Tatami memberikan persyaratan yang lebih mudah dipenuhi, namun kompleksitas yang diberikan lebih tinggi dari metode-metode lain. Hal ini dikarenakan GE Tatami perlu melakukan perhitungan ulang sebanyak jumlah kelas-1 kali.

GE Tatami juga menunjukkan kegagalan saat tidak berhasil di-*generate* fitur yang memisahkan satu kelas dengan kelas-kelas lain secara cukup menonjol. Selain itu, GE Tatami juga memiliki kecenderungan untuk membuat fitur space yang overfit.

5.2 Saran

Hasil pengujian menunjukkan bahwa GE Tatami dan GE Multi memberikan hasil yang cukup baik. Dalam kasus-kasus ideal, GE Tatami tampak sanggup memberikan hasil yang sangat baik, namun dalam kasus-kasus lain, tampak bahwa GE Tatami menunjukkan hasil yang kurang baik. Sesuai dengan analisa yang dilakukan, kelemahan ini terjadi karena di langkah pertama, GE Tatami gagal memisahkan satu kelas dengan semua kelas lain. Hal ini selanjutnya berdampak pada proses selanjutnya. Oleh sebab itu, disarankan untuk menggabungkan fitur-fitur yang di-*generate* oleh GE Multi dan fitur-fitur yang di-*generate* oleh GE Tatami.

Dalam penelitian, *decision tree classifier* masih dilibatkan dalam perhitungan *fitness value*. Penggunaan *fitness function* yang lebih sederhana tanpa melibatkan *classifier* diharapkan dapat meningkatkan performa GE Multi dan GE Tatami.

Untuk penggunaan GE Tatami dalam kasus nyata, sebaiknya digunakan sample yang cukup banyak dikarenakan kecenderungannya untuk membuat fitur-fitur yang *overfit*.

DAFTAR PUSTAKA

[1] Gunawan G. F., Gosaria S, Arifin A. Z. (2012). "*Grammatical Evolution For Feature Extraction In Local Thresholding Problem*", Jurnal Ilmu Komputer dan Informasi, Vol 5, No 2 (2012)

[2] Harper R., Blair A. (2006). "*Dynamically Define Functions in Grammatical Evolution*", IEEE Congress of Evolutionary Computation, July 16-21, 2006

[3] Gavrilis D., Tsoulous I. G., Georgoulas G., Glavas E. (2005). "*Classification of Fetal Heart Rate Using Grammatical Evolution*", IEEE Workshop on Signal Processing Systems Design and Implementation, 2005.

[4] Gavrilis D., Tsoulous I. G., Dermatas E. (2008). "*Selecting and Constructing Features Using Grammatical Evolution*", Journal Pattern Recognition Letters Volume 29 Issue 9, July, 2008 Pages 1358-1365 .

[5] Guo L., Rivero D., Dorado J., Munteanu C. R., Pazos A. (2011). "*Automatic feature extraction using genetic programming: An application to epileptic EEG classification* ", Expert Systems with Applications 38 Pages 10425-10436

[6] Li B., Zhang P.Y., Tian H., Mi S.S., Liu D.S., Ruo G.Q. (2011). "*A new feature extraction and selection scheme for hybrid fault diagnosis of gearbox*", Expert Systems with Applications 38 Pages 10000-10009

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V. , Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P. , Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011). "*Scikit-learn: Machine Learning in Python*", Journal of Machine Learning Research Vol. 12 Pages 2825-2830