TUGAS AKHIR – SM141501

IMPLEMENTASI SELF-ORGANIZING FUZZY MAPS PADA *INCOMPLETE DATA* UNTUK PENGELOMPOKAN GIZI BAHAN PANGAN

TARA AMILA MILATINA
NRP 1213 100 094

Dosen Pembimbing
Prof. Dr. Mohammad Isa Irawan, MT

JURUSAN MATEMATIKA
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember
Surabaya 2017

"Halaman ini sengaja dikosongkan"

FINAL PROJECT – SM141501

IMPLEMENTATION OF SELF-ORGANIZING FUZZY MAPS ON INCOMPLETE DATA FOR FOOD NUTRITION CLUSTERING

TARA AMILA MILATINA
NRP 1213 100 094

Supervisor
Prof. Dr. Mohammad Isa Irawan, MT

DEPARTMENT OF MATHEMATICS
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember
Surabaya 2017

“Halaman ini sengaja dikosongkan”

LEMBAR PENGESAHAN

IMPLEMENTASI *SELF-ORGANIZING FUZZY MAPS* PADA *INCOMPLETE DATA* UNTUK PENGELOMPOKAN GIZI BAHAN PANGAN
IMPLEMENTATION OF SELF-ORGANIZING MAPS ON INCOMPLETE DATA FOR FOOD NUTRITION CLUSTERING

Diajukan untuk memenuhi salah satu syarat
Untuk memperoleh gelar Sarjana Sains pada
Bidang studi Ilmu Komputer
Program Studi S-1 Jurusan Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Sepuluh Nopember Surabaya

Oleh :
TARA AMILA MILATINA
NRP. 1213 100 094

Menyetujui,
Dosen Pembimbing,

Prof. Dr. Mohammad Isa Irawan, MT
NIP. 19631225 198903 1 001

Mengetahui,
Ketua Jurusan Matematika,

Dr. Imam Mukhlash, S.Si, M.T
NIP. 19700831 199403 1 003

Surabaya, Januari 2017

"Halaman ini sengaja dikosongkan"

IMPLEMENTASI *SELF-ORGANIZING FUZZY MAPS* PADA *INCOMPLETE DATA* UNTUK PENGELOMPOKAN GIZI BAHAN PANGAN

Nama Mahasiswa	: Tara Amila Milatina
NRP	: 1213 100 094
Jurusan	: Matematika
Dosen Pembimbing	: Prof. Dr. M. Isa Irawan, MT

Abstrak

Incomplete data merupakan permasalahan yang dapat mempengaruhi hasil *clustering* pada kasus yang berkaitan dengan pengenalan pola maupun *clustering*. *Incomplete data* menjadi kelemahan dalam *clustering* dimana hampir semua metode *clustering* hanya dapat bekerja pada data yang lengkap. Hal ini yang mendorong perlunya dicari metode khusus untuk penanganan terhadap permasalahan yang berkaitan dengan *incomplete data*. Salah satu permasalahan dalam pengolahan data adalah pada pengelompokan data bahan pangan berdasarkan kandungan gizinya. Hal ini disebabkan karena data gizi bahan pangan merupakan *incomplete data*. Berdasarkan hal tersebut, pada penelitian ini akan digunakan *Self-Organizing Fuzzy Maps* untuk mengelompokkan data bahan pangan yang merupakan *incomplete data.* Setelah dilakukan implementasi dan analisis hasil, didapatkan bahwa *Self-Organizing Fuzzy Maps* dapat mengelompokkan data bahan pangan yang merupakan *incomplete data* berdasarkan kandungan gizinya. Berdasarkan hasil tersebut, dapat dibuktikan bahwa *Self-Organizing Fuzzy Maps* merupakan algoritma *clustering* yang dapat bekerja pada *incomplete data*.

Kata Kunci : Bahan Pangan, Clustering, Self-Organizing Fuzzy Maps.

"Halaman ini sengaja dikosongkan"

IMPLEMENTATION OF SELF-ORGANIZING MAPS ON INCOMPLETE DATA FOR FOOD NUTRITION CLUSTERING

Name of Student	: Tara Amila Milatina
NRP	: 1213 100 094
Department	: Mathematics
Supervisor	: Prof. Dr. M. Isa Irawan, MT

Abstract

Incomplete data is a problem that can affect the results of clustering on many cases related to the pattern recognition or clustering. Incomplete data become a weakness in clustering, where nearly all methods of clustering can only work on a complete data. This encourages the need to find specific methods for handling against problems associated with incomplete data. One of the problems in the processing of incomplete data is in the clustering of food based on the content of its nutrition value. This is because food nutrient data is incomplete data. Based on that, this research will use Self-Organizing Fuzzy Maps to cluster food nutrient data which is incomplete data. After the implementation and analysis of the results, obtained that Self-Organizing Fuzzy Maps can cluster food nutrient data which is incomplete data based on the content of its nutrition value. Based on these results, it can be proved that Self-Organizing Fuzzy Maps is a clustering algorithm that can be used on incomplete data.

Keywords : Food, Clustering, Self-Organizing Fuzzy Maps.

"Halaman ini sengaja dikosongkan"

KATA PENGANTAR

Segala puji syukur penulis panjatkan ke hadirat Allah SWT, karena dengan ridlo-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul

"IMPLEMENTASI *SELF-ORGANIZING FUZZY MAPS* PADA *INCOMPLETE DATA* UNTUK PENGELOMPOKAN GIZI BAHAN PANGAN"

yang merupakan salah satu persyaratan akademis dalam menyelesaikan Program Sarjana Jurusan Matematika, Fakultas Matematika dan Ilmu Pengethuan Alam, Institut Teknologi Sepuluh Nopember Surabaya.

Tugas Akhir ini dapat diselesaikan dengan baik berkat kerja sama, bantuan, dan dukungan dari banyak pihak. Sehubungan dengan hal tersebut, penulis ingin mengucapkan terima kasih kepada :

1. Dr. Imam Mukhlash, S.Si, MT selaku Ketua Jurusan Matematika ITS.
2. Dr. Budi Setiyono, S.Si, MT selaku Dosen Wali yang telah memberikan arahan akademik selama penulis menempuh pendidikan di Jurusan Matematika ITS.
3. Prof. Dr. Mohammad Isa Irawan, MT selaku Dosen Pembimbing yang telah memberikan bimbingan dan motivasi kepada penulis dalam mengerjakan Tugas Akhir ini sehingga dapat terselesaikan dengan baik.
4. Dr. Didik Khusnul Arif, S.Si, M.Si selaku Ketua Program Studi S1 Jurusan Matematika ITS.
5. Drs. Iis Herisman, M.Si selaku Sekretaris Program Studi S1 Jurusan Matematika ITS.
6. Seluruh jajaran dosen dan staf Jurusan Matematika ITS.
7. Keluarga tercinta yang senantiasa memberikan dukungan dan do'a yang tak terhingga.
8. Teman-teman angkatan 2013 yang saling mendukung dan memotivasi.

9. Semua pihak yang tak bisa penulis sebutkan satu-persatu, terima kasih telah membantu sampai terselesaikannya Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran dari pembaca. Akhir kata, semoga Tugas Akhir ini dapat bermanfaat bagi semua pihak yang berkepentingan.

Surabaya, Januari 2017

Penulis

DAFTAR ISI

DAFTAR GAMBAR

"Halaman ini sengaja dikosongkan"

DAFTAR TABEL

"Halaman ini sengaja dikosongkan"

BAB I
PENDAHULUAN

Pada bab ini dibahas mengenai latar belakang yang mendasari penulisan Tugas Akhir ini. Di dalamnya mencakup identifikasi permasalahan pada topik Tugas Akhir kemudian dirumuskan menjadi permasalahan yang diberikan batasan-batasan dalam pembahasan pada Tugas Akhir ini.

1.1 Latar Belakang

Perkembangan teknologi informasi mengakibatkan peningkatan dalam pertumbuhan data. Sejalan dengan hal tersebut, kebutuhan masyarakat dalam mendapatkan informasi yang berkualitas dari data juga semakin meningkat. Sesuai dengan kebutuhan tersebut, diperlukan data yang baik dan metode yang tepat dalam pengolahan data sehingga mendapatkan hasil yang berkualitas.

Data yang baik sangat dibutuhkan agar kesimpulan yang dihasilkan dari penelitian tidak jauh berbeda dengan keadaan yang sebenarnya. Data yang baik adalah data yang mampu memberikan gambaran mengenai keadaan yang diamati. Dalam kasus nyata, banyak sekali ditemukan bahwa data yang akan diolah adalah data yang memiliki *missing value* atau merupakan data yang tidak lengkap *(incomplete data)*. Pengumpulan data pengamatan tidak selalu berjalan dengan baik, sehingga mengakibatkan data menjadi tidak lengkap atau memuat beberapa nilai yang hilang. Hal ini akan mempersulit proses pengolahan data.

Dalam banyak kasus yang berkaitan dengan pengenalan pola maupun *clustering*, *incomplete data* merupakan permasalahan yang dapat mempengaruhi hasil *clustering*. *Missing data* menjadi kelemahan umum dalam *clustering* dimana hampir semua metode *clustering* hanya dapat bekerja pada data yang lengkap. Hal ini yang mendorong perlunya dicari metode khusus untuk penanganan terhadap

permasalahan yang berkaitan dengan *incomplete data*. Beberapa cara menangani masalah *incomplete data* yaitu dengan menghapus data ataupun dengan mengisi *missing value* dengan nilai dari rata-rata nilai dalam data. Namun cara tersebut dinilai tidak efektif karena akan mengurangi keakuratan data.

Salah satu metode dalam pengolahan data adalah *Self-Organizing Maps. Self-Organizing Maps* adalah sebuah algoritma jaringan syaraf yang diusulkan oleh Kohonen dan hingga sekarang telah dianalisis dan digunakan secara ekstensif. *Self-Organizing Maps* adalah jaringan syaraf tiruan yang *unsupervised* (tidak terawasi)[1]. *Self-Organizing Maps* dapat digunakan untuk mempelajari cluster dan mampu untuk memetakan data berdimensi tinggi kedalam bentuk pemetaan berdimensi rendah. Namun, *Self-Organizing Maps* tidak dapat memproses *incomplete data* atau data yang tidak lengkap.

Akan tetapi, Shouhong Wang pada tahun 2003 telah melakukan penelitian tentang penggunaan *Self-Organizing Maps* untuk dapat memetakan data yang tidak lengkap. Sebelum dipetakan menggunakan *Self-Organizing Maps*, data yang tidak lengkap akan ditransformasikan menjadi observasi fuzzy. Algoritma tersebut selanjutnya disebut sebagai *Self-Organizing Fuzzy Maps*. Dalam penelitian tersebut didapatkan hasil bahwa *Self-Organizing Fuzzy Maps* dapat memetakan data yang tidak lengkap[2].

Salah satu permasalahan dalam pengolahan data adalah pada pengelompokan data bahan pangan berdasarkan kandungan gizinya. Masyarakat Indonesia mengonsumsi tidak kurang dari 100 jenis kacang-kacangan, 450 jenis buah-buahan serta 250 jenis sayur-sayuran dan jamur. Begitu juga dengan sumber daya hayati laut, hewan serta miroba, sudah lama dimanfaatkan untuk menunjang kebutuhan hidup sehari-hari masyarakat Indonesia[3]. Banyaknya jenis bahan pangan tersebut menyebabkan sulit untuk mengetahui kandungan gizi tiap bahan pangan. Hal tersebut mengakibatkan data mengenai

kandungan gizi bahan pangan memiliki *missing value* atau merupakan *incomplete data*. Sehingga, belum terdapat pengelompokan bahan pangan berdasarkan kandungan gizinya.

Saat ini telah terdapat banyak aspek dalam pengelompokan bahan pangan. Badan Ketahanan Pangan 2010-2014 mengelompokkan komoditas pangan penting ke dalam dua kelompok yaitu pangan nabati dan pangan hewani[4]. Badan Pusat Statistik membagi bahan pangan kedalam sembilan kelompok yang meliputi : (1) padi-padian, (2) umbi-umbian, (3) pangan hewani, (4) minyak dan lemak, (5) buah/biji berminyak, (6) kacang-kacangan, (7) gula, (8) sayuran dan buah, (9) lain-lain[5]. Sedangkan Pemerintah Daerah Istimewa Yogyakarta mengeluarkan informasi mengenai kandungan gizi beberapa bahan pangan di Indonesia. Informasi tersebut mengolompokkan bahan pangan menjadi 8 golongan sesuai dengan penggunaannya yaitu (1) golongan serelia dan umbi, (2) golongan kacang-kacangan dan biji-bijian, (3) golongan daging, (4) golongan telur, (5) golongan ikan, kerang, dan udang, (6) golongan sayuran, (7) golongan lemak dan minyak, (8) golongan serba-serbi[6]. Namun, seperti yang telah dijelaskan sebelumnya bahwa belum terdapat pengelompokan bahan pangan berdasarkan kandungan gizinya. Pengelompokan ini akan memudahkan masyarakat mencari bahan makanan yang sesuai dengan kebutuhan tubuh.

Berdasarkan hal tersebut, penulis ingin mengetahui efektivitas *Self-Organizing Fuzzy Maps* untuk pengelompokan data bahan pangan yang merupakan *incomplete data.* Untuk itu, penulis mengangkat topik Tugas Akhir berjudul "Implementasi *Self-Organizing Fuzzy Maps* untuk *Incomplete Data* pada Pengelompokan Gizi Bahan Pangan".

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, dapat dirumuskan permasalahan dalam Tugas Akhir ini adalah sebagai berikut :

1. Bagaimana implementasi *Self-Organizing Fuzzy Maps* pada data gizi bahan pangan?
2. Apakah *Self-Organizing Fuzzy Maps* dapat digunakan dalam mengelompokkan bahan pangan yang merupakan *incomplete data*?

1.3 Batasan Masalah

Pada penelitian ini, penulis membuat batasan masalah sebagai berikut :

1. Objek yang digunakan pada penelitian ini adalah data gizi bahan pangan yang dikeluarkan oleh Badan Ketahanan Pangan dan Penyuluhan Daerah Istimewa Yogyakarta.
2. Tidak dilakukan proses pembandingan dengan metode lainnya.

1.4 Tujuan

Berdasarkan permasalahan yang telah dirumuskan sebelumnya, tujuan penelitian Tugas Akhir ini adalah :

1. Untuk mengetahui efektivitas *Self-Organizing Fuzzy Maps* pada data yang tidak lengkap (*incomplete data*).
2. Untuk mengetahui apakah *Self-Organizing Fuzzy Maps* dapat digunakan dalam mengelompokkan bahan pangan yang merupakan *incomplete data*.

1.5 Manfaat

Setelah diperoleh pengelompokan bahan pangan berdasarkan kandungan gizi, maka Tugas Akhir ini dapat memberikan manfaat sebagai berikut :

1. Penelitian ini dapat memberikan informasi mengenai penggunaan *Self-Organizing Fuzzy Maps* untuk pengelompokan bahan pangan sesuai kandungan gizi.

2. Sebagai salah satu referensi penggunaan *Self-Organizing Fuzzy Maps* untuk data yang tidak lengkap.

1.6 Sistematika Penulisan Tugas Akhir

Sistematika dari penulisan Tugas Akhir ini adalah sebagai berikut :

1. BAB I PENDAHULUAN
   Bab ini menjelaskan tentang gambaran umum dari penulisan Tugas Akhir ini yang meliputi latar belakang masalah, perumusan masalah, batasan masalah, tujuan, manfaat penelitian, dan sistematika penulisan.
2. BAB II TINJAUAN PUSTAKA
   Bab ini berisi tentang materi-materi yang mendukung Tugas Akhir ini, antara lain penelitian terdahulu, pengelompokan data, *incomplete data*, bahan pangan, *Principal Component Analysis*, *Kohonen Self-Organizing Maps*, himpunan fuzzy, aturan *fuzzy* dan *Self-Organizing Fuzzy Maps*
3. BAB III METODOLOGI PENELITIAN
   Pada bab ini dibahas tentang langkah – langkah dan metode yang digunakan untuk menyelesaikan Tugas Akhir ini.
4. BAB IV PERANCANGAN IMPLEMENTASI
   Pada bab ini akan menguraikan bagaimana tahapan tahapan dalam perancangan implementasi. Pembahasan perancangan implementasi dimulai dari perancangan pra-pemrosesan data.
5. BAB V IMPLEMENTASI
   Bab ini menjelaskan mengenai implementasi *Self-Organizing Fuzzy Maps* untuk mengelompokkan data kandungan gizi bahan pangan yang merupakan *incomplete* data. Setelah itu, dilakukan analisis terhadap hasil implementasi. Analisis ini bertujuan untuk melihat apakah *Self-Organizing Fuzzy Maps* dapat diterapkan

untuk mengelompokkan data kandungan gizi bahan pangan yang merupakan *incomplete data*.

6. BAB VI PENUTUP
   Bab ini berisi kesimpulan yang diperoleh dari pembahasan masalah sebelunya serta saran yang diberikan untuk pengembangan selanjutnya.

BAB II`
TINJAUAN PUSTAKA

Pada bab ini dibahas mengenai dasar teori yang digunakan dalam penyusunan Tugas Akhir ini. Dasar teori yang dijelaskan dibagi menjadi beberapa subbab yaitu gizi bahan pangan, *Self-Organizing Maps*, himpunan *fuzzy*, dan *Self-Organizing Fuzzy Maps*.

2.1 Penelitian Terdahulu

Pada tahun 2003 Shouhong Wang dalam penelitian yang berjudul "Application of Self-Organizing Maps for Data Mining with Incomplete Data Sets" meneliti tentang penggunaan *Self-Organizing Maps* untuk mengolah data yang tidak lengkap. *Self-Organizing Maps* dapat digunakan untuk memetakan data berdimensi tinggi kedalam bentuk pemetaan berdimensi rendah. Namun, *Self-Organizing Maps* tidak dapat memproses *incomplete data* atau data yang tidak lengkap. Pada penelitian yang dilakukan oleh Shouhong Wang ditemukan suatu metode untuk memetakan data yang tidak lengkap dengan *Self-Organizing Fuzzy Maps*[2].

Penelitian mengenai penggunaaan *Self-Organizing Maps* berbasis *Fuzzy* terus menerus dilakukan. Salah satunya yaitu penelitian mengenai pengelompokan dengan menggunakan *Self-Oraganizing Maps* dan logika *fuzzy* yang disesuaikan dengan data yang digunakan. Dalam penelitian ini digunakan pembandingan dengan beberapa metode dengan menggunakan beberapa dataset. Hasilnya menenjukkan bahwa akurasi dari pengelompokan menggunakan *Self-Organizing Maps* tinggi dan melebihi hasil yang diharapkan pada metode-metode lainnya[7].

Saat ini para peneliti juga berlomba-lomba dalam meneliti mengenai pengelompokan bahan pangan. Karl Presser, Hans Hinterberger, David Weber, Moira Norrie pada tahun 2016 melakukan penelitian yang diberi judul "A scope

classification of data quality requirements for food composition data". Penelitian tersebut memberikan persyaratan kualitas data yang dapat digunakan untuk sistem informasi yang mengelola data komposisi makanan, yang disebut FoodCASE. Food Composition And System Environment (FoodCASE) merupakan sistem manajemen data komposisi makanan yang dibuat oleh Swiss dan EuroFIR. Penelitian tersebut menemukan bahwa dalam pengelompokan komposisi makanan data yang digunakan harus berasal dari sampel yang beratnya atau banyaknya sama[8].

Sementara itu pada tahun 2014, Institute of Nutrition, Mahidol University bekerja sama dengan 6 negara anggota ASEAN yaitu Indonesia, Malaysia, Filipina, Singapura, Thailand, dan Brunei Darussalam mengeluarkan ASEAN Food Composition Database. ASEAN Food Composition Database merupakan data komposisi bahan pangan yang biasa dikonsumsi di Asia Tenggara. Data tersebut berisi 1740 jenis bahan pangan. Pembuatan data tersebut bertujuan agar dapat melayani kebutuhan pengguna di negara-negara anggota ASEANFOOD dan digunakan di seluruh dunia untuk penelitian pangan dan gizi masyarakat dan industri. Metode analisis yang digunakan untuk menetukan nilai-nilai gizi tersebut dengan menggunakan metode statistik. Median dan jangkauan interquartil nilai gizi dinormalisasi untuk setiap nutrisi yang diperoleh. Hal ini dilakukan karena variasi dalam komposisi kandungan dalam makanan di ASEAN lebih luas daripada yang ada di satu negara. Nilai mean dan deviasi standar dari kumpulan data yang diterima kemudian dihitung dan nilai rata-rata digunakan sebagai nilai akhir untuk setiap nutrisi bahan pangan[9].

2.2 Pengelompokan Data

Pengelompokkan data dalam data mining dibedakan menjadi 2, yaitu klasifikasi dan klasterisasi. Klasifikasi adalah pengelompokkan data yang membutuhkan data latih

(supervised). Sedangkan klasterisasi adalah pengelompokkan data tanpa membutuhkan data latih (unsupervised).

Klasterisasi adalah proses membagi data yang tidak berlabel menjadi kelompok-kelompok data yang memiliki kemiripan. Setiap kelompok data (klaster) terdiri dari obyek yang memiliki kemiripan satu sama lain dan setiap klaster memiliki ketidakmiripan dengan klaster lain. Klasterisasi lazim digunakan dalam analisis data multivariat[10].

2.3 *Incomplete Data*

Incomplete data adalah suatu keadaan dimana beberapa nilai atribut dalam suatu dataset kosong /tidak ada nilainya. Artinya terdapat isian yang tidak lengkap pada variabel tertentu pada suatu observasi.

Keacakan *incomplete data* dapat dibagi ke dalam tiga tipe [11] yaitu :

1. Missing Completely at Random (MCAR)
   yaitu terjadinya missing data tidak bergantung pada nilai seluruh variabel, baik variabel yang terisi (diketahui) maupun variabel yang mengandung missing values.
2. Missing at Random (MAR).
   yaitu terjadinya missing data bergantung pada variabel yang terisi (diketahui) namun tidak bergantung pada variabel yang mengandung missing values itu sendiri.
3. Not Missing at Random (NMAR)
   yaitu terjadinya missing data pada suatu variabel bergantung pada variabel itu sendiri sehingga tidak dapat diprediksi dari variabel yang lain.

Penanganan missing data dapat dilakukan dengan menghapus data yang tidak lengkap. Jika data yang tidak lengkap jumlahnya relatif kecil, dibandingkan dengan keseluruhan data, menghapus data yang tidak lengkap merupakan salah satu pendekatan yang masuk akal. Tetapi umumnya hal ini akan memberikan kesimpulan yang valid, hanya terjadi ketika missing data secara acak, dalam arti bahwa

probabilitas respon tidak tergantung pada nilai-nilai data yang diamati atau hilang. Dengan kata lain, penghapusan secara implisit mengasumsikan bahwa kasus yang dibuang seperti subsampel acak[12].

Penghapusan data hilang ini tentunya memberikan hasil pendugaan yang kurang baik dikarenakan beberapa alasan, yaitu :

- Jika melakukan penghapusan data hilang maka ukuran contoh yang terambil akan berkurang, sehingga akan menyebabkan berkurangnya ketepatan dalam pendugaan.
- Jika individu yang dikeluarkan ternyata hasilnya sangat berbeda dari data yang lain maka akan mengahasilkan penduga yang bias.

Sebagai solusi yang tepat jika terdapat missing data dalam survey adalah dengan melakukan imputasi. Imputasi, yaitu proses pengisian atau penggantian missing values pada dataset dengan nilai-nilai yang mungkin berdasarkan informasi yang didapatkan pada dataset tersebut.

2.4 Bahan Pangan

Pangan didefinisikan segala sesuatu yang berasal dari sumber hayati, produk pertanian, perkebunan, kehutanan, perikanan, peternakan, perairan, dan air, baik yang diolah maupun tidak diolah diperuntukkan sebagai makanan atau minuman bagi konsumsi manusia, termasuk bahan tambahan pangan, bahan baku pangan, dan bahan lainnya yang digunakan dalam proses penyimpanan, pengolahan, dan atau pembuatan makanan dan minuman[13].

Di alam terdapat berbagai jenis bahan pangan, baik yang berassal dari tanaman yang disebut sebagai bahan pangan nabati maupun yang berasal dari hewan yang dikenal sebagai bahan pangan hewani. Diantara beragam jenis bahan pangan tersebut ada yang kaya akan satu jenis zat gizi, begitu pula sebaliknya. Umumnya tidak ada satu bahan pangan yang

lengkap mengadung semua zat gizi dalam jumlah yang mencukupi keperluan tubuh, kecuali ASI untuk bayi. Oleh karena itu, manusia memerlukan berbagai macam bahan pangan untuk menjamin agar semua zat gizi yang diperlukan tubuh dapat terpenuhi dalam jumlah yang cukup[14].

Makanan merupakan salah satu sumber energi vital manusia agar ia dapat melaksanakan kegiatan sehari-hari dengan baik: untuk bekerja, berolahraga, belajar, bermain dan sebagainya. Oleh karena itu makanan harus disiapkan agar kebutuhan tersebut dapat terpenuhi. Salah gizi bisa timbul bukan karena kurangnya jumlah makanan yang dikonsumsi, melainkan karena susunannya yang tidak seimbang[15].

Pada umumnya bahan pangan tersusun oleh tiga pokok komponen yaitu karbohidrat, protein dan lemak serta turunannya, sedangkan sisanya yang hanya sebagian kecil terdiri dari bermacam-macam zat organik yaitu vitamin, enzim, zat penyebab asam, oksidan, antioksidan dan pigmen dan zat penyebab rasa dan bau serta air. Dalam setiap bahan makanan komponen tersebut sangat bervariasi jumlahnya sehingga akan membentuk struktur, tekstur, rasa, bau, warna serta kandungan gizi yang berlainan pula.

2.5 *Principal Component Analysis (PCA)*

Principal Component Analysis (PCA) adalah teknik statistik yang sudah digunakan secara luas baik dalam hal pengolahan data, pembelajaran mesin, maupun pengolahan citra atau pemrosesan signal. Metode *PCA* dibuat pertama kali oleh para ahli statistik dan ditemukan oleh Karl Pearson pada tahun 1901 yang memakainya pada bidang biologi. Pada tahun 1947 teori ini ditemukan kembali oleh Karhunen, dan kemudian dikembangkan oleh Loeve pada tahun l963, sehingga teori ini juga dinamakan Karhunen-Loeve transform pada bidang ilmu telekomunikasi.

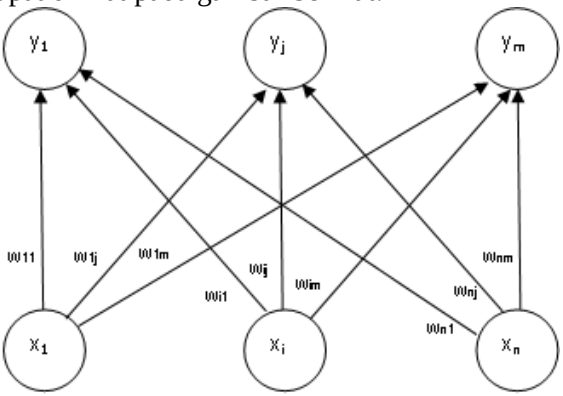
Principal Component Analysis (PCA) adalah teknik yang digunakan untuk menyederhanakan suatu data, dengan cara mentransformasi data secara linier sehingga terbentuk sistem koordinat baru dengan varians maksimum[16]. *PCA* dapat digunakan untuk mereduksi dimensi suatu data tanpa mengurangi karakteristik data tersebut secara signifikan[17].

2.6 *Kohonen Self-Organizing Maps*

Teuvo Kohonen memperkenalkan *Self-Organizing Maps* (SOMs) lebih dari tiga puluh tahun yang lalu dan sejak saat itu telah dianalisis dan digunakan secara ekstensif[1]. *Kohonen Self-Organizing Maps* adalah metode analisis data otomatis. Hal ini secara luas diterapkan untuk pengelompokan masalah dan eksplorasi data di industri, keuangan, ilmu pengetahuan alam, dan linguistik[18]. Jaringan *Kohonen Self-Organizing Maps* digunakan untuk mengelompokkan (*clustering*) data berdasarkan karakteristik/fitur-fitur data. *Kohonen Self-Organizing Maps* merupakan bentuk dari *Unsupervised Artificial Neural Network* (*Unsupervised ANN*) dimana dalam proses pelatihannya tidak memerlukan pengawasan atau target keluaran. Dalam *Kohonen Self-Organizing Maps* masukan berupa vektor yang terdiri dari n komponen (tuple) yang akan dikelompokan dalam maksimum m buah kelompok (disebut vektor contoh). Keluaran jaringan adalah kelompok yang paling dekat/ mirip dengan masukan yang diberikan.

Kohonen Self-Organizing Maps disusun oleh sebuah lapisan unit input yang dihubungkan seluruhnya ke lapisan unit output, yang kemudian unit-unit diatur di dalam topologi khusus seperti struktur jaringan [19]. Pada jaringan ini, suatu lapisan yang berisi *neuron-neuron* akan menyusun dirinya sendiri berdasarkan input nilai tertentu dalam suatu kelompok yang dikenal dengan istilah *cluster*. Selama proses penyusunan diri, *cluster* yang memiliki vektor bobot paling cocok dengan pola input (memiliki jarak paling dekat) akan terpilih sebagai pemenang. *Neuron* yang menjadi pemenang beserta *neuron-*

neuron tetangganya akan memperbaiki bobot – bobotnya. Secara umum arsitektur jaringan *Kohonen Self-Organizing Maps* dapat dilihat pada gambar berikut.



Gambar 2.1 *Kohonen Self-Organizing Maps*

Arsitektur *Kohonen Self-Organizing Maps* diatas terdiri dari dua lapisan (layer), yaitu lapisan input dan lapisan output. Setiap neuron dalam lapisan input terhubung dengan setiap neuron pada lapisan output. Setiap neuron dalam lapisan output merepresentasikan kelas dari input yang diberikan.

Kohonen Self-Organizing Maps menyediakan suatu teknik visualisasi data dan membantu memahami data yang memiliki dimensi yang kompleks dengan mengurangi dimensi data kedalam pemetaan. *Kohonen Self-Organizing Maps* juga merupakan konsep *clustering* dengan mengelompokkan data yang memiliki kemiripan tertentu. Penggunaan *Kohonen Self-Organizing Maps* dalam penganalisaan data dapat mengakomodir data yang beragam mulai dari data dengan *single-dimensional* mupun *multi-dimensional* atau data dengan *singlevariat* maupun data yang bersifat *multi-variat*. Dalam data mining *Kohonen Self-Organizing Maps* biasa digunakan untuk teknik data mining yang berkaitan dengan pengelompokan data (*clustering*).

2.6.1 Algoritma *Kohonen SOM* [20]:

Langkah 0. Misalkan y_j $(j = 1, \dots, m)$ dimana m adalah jumlah vektor keluaran. Misalkan x_i $(i = 1, \dots, n)$ dimana n adalah jumlah vektor masukan.
Inisialisasi bobot w_{ij} dengan nilai random.
Atur parameter topologi ketetanggaan (R).
Atur parameter *learning rate* (α).

Langkah 1. Misalkan ε adalah konstanta positif kecil yang telah ditentukan. Jika $\alpha < \varepsilon$ tidak terpenuhi, maka lakukan langkah 1.1-1.4.

Langkah 1.1. Untuk setiap masukan vektor x_i, lakukan langkah i-iii.

i. Untuk setiap j. Hitung jarak kuadratik eucllidian $D(j)$:

$$D(j) = \sum_i \left(w_{ij} - x_i\right)^2$$

ii. Temukan indeks j sedemikian hingga $D(j)$ minimum.

iii. Untuk semua unit j dalam *neighbourhood J* yang telah ditetapkan, dan untuk semua i :
$w_{ij}(\text{baru}) = w_{ij}(\text{lama}) + \alpha\left[x_i - w_{ij}(\text{lama})\right]$

Langkah 1.2. Perbarui *learning rate*.
$\alpha(\text{baru}) = \delta\left(\alpha(\text{lama})\right)$, dimana $0 < \delta < 1$

Langkah 1.3. Kurangi radius topologi *neighbourhood* pada waktu yang ditentukan.

Langkah 1.4. Uji kondisi berhenti.

2.6.2 *Kohonen SOM* untuk pengelompokan 4 vektor [20]

Misalkan vektor yang akan dikelompokkan adalah : (1, 1, 0, 0); (0, 0, 0, 1); (1, 0, 0, 0); (0, 0, 1, 1). Jumlah maksimum kelompok yang akan dibentuk adalah 2. *Lerning rate* $(\alpha(0))$ adalah 0,6 dengan $\alpha(t+1) = 0.5\ \alpha(t)$.

Langkah 0. Inisialisasi bobot awal (w_{ij})

$$\begin{bmatrix} 0,2 & 0,8 \\ 0,6 & 0,4 \\ 0,5 & 0,7 \\ 0,9 & 0,3 \end{bmatrix}$$

Radius awal : R=0

Learning Rate awal : $\alpha(0) = 0,6$

Langkah 1. Mulai proses *training*

Langkah 1.1. Untuk vector pertama, (1, 1, 0, 0) lakukan langkah i-iii

i. $D(1) = (0,2-1)^2 + (0,6-1)^2 + (0,5-0)^2 + (0,9-0)^2$
$= 1,86$
$D(2) = (0,8-1)^2 + (0,4-1)^2 + (0,7-0)^2 + (0,3-0)^2$
$= 0,98$

ii. Vektor masukan dekat dengan *node* keluaran 2, jadi J=2.

iii. Bobot yang menang diperbarui :
w_{i2}(baru)= w_{i2}(lama)+0,6$[x_i - w_{i2}$(lama)$]$
$= 0,4\ w_{ij}$(lama)+0,6$[x_i]$
Sehingga bobot menjadi :

$$\begin{bmatrix} 0,2 & 0,92 \\ 0,6 & 0,76 \\ 0,5 & 0,28 \\ 0,9 & 0,12 \end{bmatrix}$$

Langkah 1.1. Untuk vektor kedua, (0, 0, 0, 1) lakukan Langkah i-iii

i. $D(1) = (0,2-0)^2 + (0,6-0)^2 + (0,5-0)^2 + (0,9-1)^2$
$= 0,66$
$D(2) = (0,92-0)^2 + (0,76-0)^2 + (0,28-0)^2 + (0,12-1)^2$
$= 2,2768$

ii. Vektor masukan dekat dengan *node* keluaran 1, jadi J=1.

iii. Perbarui kolom pertama dari matriks bobot

$$\begin{bmatrix} 0{,}08 & 0{,}92 \\ 0{,}24 & 0{,}76 \\ 0{,}20 & 0{,}28 \\ 0{,}96 & 0{,}12 \end{bmatrix}$$

Langkah 1.1. Untuk vektor ketiga, (1, 0, 0, 0) lakukan Langkah i-iii

i. $D(1) = (0{,}08 - 1)^2 + (0{,}24 - 0)^2 + (0{,}2 - 0)^2 + (0{,}96 - 0)^2$
$= 1{,}8656$
$D(2) = (0{,}92 - 1)^2 + (0{,}76 - 0)^2 + (0{,}28 - 0)^2 + (0{,}12 - 0)^2$
$= 0{,}6768$

ii. Vektor masukan dekat dengan *node* keluaran 2, jadi J=2.

iii. Perbarui kolom pertama dari matriks bobot

$$\begin{bmatrix} 0{,}08 & 0{,}968 \\ 0{,}24 & 0{,}304 \\ 0{,}20 & 0{,}112 \\ 0{,}96 & 0{,}048 \end{bmatrix}$$

Langkah 1.1. Untuk vektor keempat, (0, 0, 1, 1) lakukan Langkah i-iii

i. $D(1) = (0{,}08 - 0)^2 + (0{,}24 - 0)^2 + (0{,}2 - 1)^2 + (0{,}96 - 1)^2$
$= 0{,}7056$
$D(2) = (0{,}968 - 0)^2 + (0{,}304 - 0)^2 + (0{,}112 - 1)^2 + (0{,}048 - 1)^2$
$= 0{,}6768$

ii. Vektor masukan dekat dengan *node* keluaran 1, jadi J=1.

iii. Perbarui kolom pertama dari matriks bobot

$$\begin{bmatrix} 0{,}032 & 0{,}968 \\ 0{,}096 & 0{,}304 \\ 0{,}680 & 0{,}112 \\ 0{,}984 & 0{,}048 \end{bmatrix}$$

Langkah 1.2. Kurangi nilai *learning rate* :

$$\alpha = 0{,}5(0{,}6) = 0{,}3$$

Rumus pembaharuan bobot menjadi :

$$w_{ij}(\text{baru}) = w_{ij}(\text{lama}) + 0{,}3\left[x_i - w_{ij}(\text{lama})\right]$$
$$= 0{,}7\ w_{ij}(\text{lama}) + 0{,}3[x_i]$$

Prosedur *training* tersebut dilakukan terus menerus hingga didapatkan nilai *learning rate* = 0,01

2.7 Himpunan *Fuzzy*

Fuzzy pertama kali diperkenalkan oleh profesor L.A Zadeh pada tahun 1965. Dalam bahasa manusia banyak hal yang mengandung ketidakpastian (ambiguity), seperti misalnya pengalaman seseorang, kemampuan seseorang dalam melakukan analisis sebelum akhirnya mengambil suatu kesimpulan. Dalam kasus-kasus seperti ini kehadiran komputer yang dirancang untuk dapat membantu pekerjaan manusia, kurang efektif karena komputer tidak mengerti tentang bahasa manusia dengan segala ketidakpastian yang terkandung di dalamnya. Maka dari itu diperlukan suatu cara atau teknik yang dapat mengartikan ketidakpastian dalam bahasa manusia menjadi bahasa yang dapat dimengerti oleh komputer, sehingga komputer akan dapat menjalankan fungsinya secara optimal. Fuzzy sangat tepat untuk mengekspresikan ketidakpastian. Di dalam teori himpunan klasik dinyatakan suatu objek adalah anggota (ditandai dengan “1”) atau bukan anggota (ditandai dengan “0”) dalam suatu himpunan dengan batas keanggotaan yang jelas/tegas (crips). Namun dalam teori fuzzy memungkinkan derajat keanggotaan (degree of numbership) suatu objek dalam himpunan untuk dinyatakan dalam interval antara “0”dan “1” atau ditulis [0 1].

Fungsi keanggotaan fuzzy (membership Function) atau derajat keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik input data kedalam nilai keanggotaannya (derajat keanggotaan) yang memiliki interval antara nol sampai satu. Salah satu cara yang dapat digunakan untuk mendapatkan

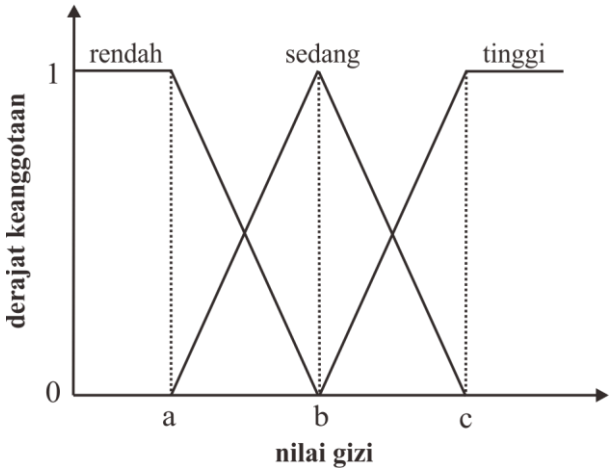
nilai kenaggotaan adalah dengan melalui pendekatan fungsi [21]. Ada beberapa fungsi keanggotaan yang digunakan dalam teori himpunan *fuzzy*. Fungsi keanggotaan yang akan digunakan dalam tugas akhir ini adalah representasi kurva bentuk bahu.

Representasi bentuk bahu adalah daerah yang terletak ditengah-tengah suatu variable yang direpresentasikan dalam bentuk segitiga, pada sisi kanan dan kirinya akan naik dan turun (misalkan: rendah bergerak ke sedang bergerak ke sedang dan bergerak ke tinggi). Tetapi terkadang salah satu sisi dari variabel tersebut tidak mengalami perubahan. Bahu kiri bergerak dari benar ke salah, demikian juga bahu kanan bergerak dari salah ke benar. Misalkan x adalah nilai gizi yang akan dicari derajat keanggotaannya. Maka, derajat keanggotaan dari x adalah :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq a \\ \dfrac{(b-x)}{b-a} & ,a < x < b \\ 0 & ,x \geq b \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq a \\ \dfrac{(x-a)}{b-a} & ,a < x \leq b \\ \dfrac{(c-x)}{c-b} & ,b < x < c \\ 0 & ,x \geq c \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq b \\ \dfrac{(x-b)}{c-b} & ,b < x < c \\ 1 & ,x \geq 10 \end{cases}$$



Gambar 2.2 Representasi Linear Bentuk Bahu

2.7.1 Definisi 1 [22]

Jika X adalah sebuah koleksi dari obyek –obyek yang dinyatakan dengan *x,* maka himpunan fuzzy (*fuzzy set*) $\tilde{A}$ dalam himpunan X adalah sebuah himpunan pasangan terurut :

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}$$

Dimana $\mu_{\tilde{A}}(x)$ disebut fungsi keanggotaan atau derajat keanggotaan (derajat kebenaran atau derajat kompatibilitas) dari *x* dalam $\tilde{A}$**,** yang memetakan *X* ke ruang keanggotaan *M*.

Sebuah himpunan fuzzy direpresentasikan dengan menyatakan fungsi keanggo-taannya saja [23].

$$\tilde{A} = \sum_{x_i \epsilon X} \mu_{\tilde{A}}(x_i)/x_i$$

disini tanda sigma mewakili gabungan dari derajat keanggotan. Simbol "/" adalah suatu penanda dan tidak mengimplikasikan pembagian.

2.7.2 Definisi 2 [22]

Sebuah proposisi fuzzy ialah sebuah kalimat deklaratif (pernyataan) yang memuat unsur kefuzzian dan mempunyai nilai kebenaran berada dalam selang (interval) [0,1]. Sebuah proposisi fuzzy primitif (tidak memuat penyambung sentensial/operator), dan biasanya direpresentasikan sebagai sebuah kalimat berpredikat fuzzy (variabel linguistik) yang mengadung unsur kefuzzyan.

Beberapa contoh proposisi fuzzy primitif :

- *Umur bapak Amir adalah tua*
- *Harga mobil balap adalah mahal*
- *Tinggi pohon cemara adalah tinggi*

disini *tua*, *mahal*, dan *tinggi* adalah predikat fuzzy (variabel linguistik) yang memuat unsur-unsur kefuzzian.

2.8 Aturan *Fuzzy*

Dalam tahun 1973, Lotfi Zadeh mempublikasikan paper keduanya yang paling berpengaruh. Garis besar isi dari paper Zadeh ini merupakan sebuah pendekatan baru terhadap analisis sistem kompleks, dimana Zadeh menyarankan pencakupan (*capturing*) pengetahuan manusia dalam aturan fuzzy.

Sebuah aturan fuzzy didefinisikan sebagai sebuah implikasi (*conditional statement*) dalam bentuk :

IF x is A
THEN y is B

dimana: x dan y adalah variabel linguistik; dan A dan B adalah nilai linguistik yang ditentukan oleh himpunan fuzzy pada himpunan semesta X dan Y

2.9 *Self-Organizing Fuzzy Maps*

Metode ini bertujuan untuk mengubah observasi yang memiliki *missing value* menjadi observasi *fuzzy* dan kemudian digunakan *Self Organizing Map* untuk membuat pemetaan fuzzy[2].

Langkah 1. Membangkitkan observasi fuzzy

(1.1) Misalkan S adalah jumlah observasi, S_1 adalah jumlah observasi tanpa *missing value*, dan S_2 adalah jumlah observasi dengan *missing value*. Untuk observasi ke-f $(f = 1 \ldots S)$, bangkitkan observasi *fuzzy* F_f. Untuk observasi dengan *missing value* akan dicari derajat keanggotaan dari *missing value* dengan aturan *fuzzy*. Derajat keanggotan dari observasi *fuzzy* F_f akan digunakan sebagai vektor masukan x_i.

Langkah 2. Inisialisasi prosedur *training Self-Organizing Maps*

(2.1) Atur waktu $t = 0$

(2.2) Atur *Self-Organizing Maps* dengan n *nodes* masukan dan m *nodes* keluaran. Misalkan y_j $(j = 1, \ldots, m)$ dimana m adalah jumlah vektor keluaran. Misalkan x_i $(i = 1, \ldots, n)$ dimana n adalah jumlah vektor masukan. Inisialisasi bobot awal w_{ij}, dari *node* masukan i $(i = 1, 2, \ldots, n)$ ke *node* keluaran j $(j = 1, 2, \ldots, m)$, ke angka acak kecil

(2.3) Atur learning rate awal $\alpha = p$, dimana $0 < p < 1$.

Langkah 3. *Training Self-Organizing Maps*

(3.1) Hitung jarak kuadratik euclidian

$$D_j = \sum_{i=1}^{n} \left(w_{ij} - x_i\right)^2$$

(3.2) Pilih keluaran node J yang menang sedemikian hingga $D_j = \min\{D_j\}$

(3.3) Perbarui bobot sebagai berikut :

$$w_{ij}(baru) = \left[w_{ij}(lama) + \alpha\left(x_i - w_{ij}\right)\right]$$

(3.4) Jika $\alpha < \varepsilon$, dimana ε adalah konstanta positif kecil yang telah ditentukan (misalnya 0,0001), maka ke langkah 4;

Jika sebaliknya, hitung :

$\alpha = \delta \ \eta(t)$

Dimana $0 < \delta < 1$, nilai δ diracang sebelumnya

Atur $t = t + 1$; dan kembali ke langkah (3.1)

2.9.1 Penggunaan *Self-Organizing Fuzzy Maps*

Misalkan vektor yang akan dikelompokkan adalah : (1, 1, 0, 0); (0, 0, 0, 1); (1, 0, 0, 0); (0, 0, 1, -). Vektor terakhir memiliki *missing value*. Misalkan setelah dilakukan proses fuzzifikasi didapatkan (1, 1, 0, 0); (0, 0, 0, 1); (1, 0, 0, 0);(0, 0, 1, 1). Setelah mendapatkan vektor yang akan dikelompokkan, tahap selanjutnya adalah melakukan proses training dengan langkah-langkah seperti pada subbab 2.6.2. Saat kondisi berhenti dicapai yaitu pada saat $\eta(t) < \varepsilon$.

BAB III
METODE PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang digunakan dalam penyusunan Tugas Akhir. Disamping itu, dijelaskan pula prosedur dan proses pelaksanaan tiap-tiap langkah yang dilakukan dalam menyelesaikan Tugas Akhir.

3.1 Tahapan Penelitian

Langkah-langkah yang digunakan dalam menyelesaikan Tugas Akhir ini yang diringkas dalam Gambar 3.1 adalah sebagai berikut :

1. **Studi Literatur**

   Studi Literatur ini dilakukan untuk identifikasi permasalahan dengan mencari referensi yang menunjang penelitian yang berupa Tugas Akhir, jurnal internasional, buku, maupun artikel yang berhubungan dengan topik Tugas Akhir ini.

2. **Pengumpulan Data**

   Pengumpulan Data merupakan tahap untuk mengumpulkan data yang diperlukan dalam pengerjaan Tugas Akhir ini yaitu data kandungan gizi bahan pangan. Data yang terkumpul akan digunakan sebagai masukan (*input*). Adapun variabel data yang akan dipakai berdasarkan data gizi bahan pangan yang dikeluarkan oleh Badan Ketahanan Pangan dan Penyuluhan Daerah Istimewa Yogyakarta adalah data bahan pangan dan hasil olahannya dengan nilai kandungan gizi yang meliputi : jumlah protein, jumlah lemak, jumlah karbohidrat, jumlah kalsium, jumlah zat fosfor, jumlah zat besi, nilai vitamin A, nilai vitamin B1, nilai vitamin C, serta kandungan air

3. **Implementasi**

   Pada tahap ini penulis membuat implementasi agar mudah dipahami oleh pembaca yang alurnya digambarkan pada Gambar 3.2. Implementasi tersebut dibuat dengan bahasa

pemrograman Java dan menggunakan aplikasi NETBEANS IDE 7.4 untuk proses *training* dan Matlab untuk menampilkan representasi hasil implementasi

Pada tahap ini dilakukan proses dimana persoalan diterjemahkan melalui suatu rangkaian aktivitas sesuai model proses, metode, dan alat bantu yang akan digunakan. Tahap-tahap yang dilakukan seperti yang digambarkan pada Gambar 3.3 adalah sebagai berikut :

a. Menentukan jumlah *missing value*

   Pada tahap ini akan dicari jumah data yang mengandung missing value dan jumlah data yang tidak mengandung *missing value*.

b. Menghasilkan observasi fuzzy

   Untuk mengubah data menjadi observasi fuzzy terdapat beberapa langkah. Langkah pertama yaitu menentukan derajat keanggotaan dari data-data yang mengandung *missing value*. Setelah ditentukan derajat keanggotaannya, akan dicari nilai dari *missing value* dengan menggunakan aturan *fuzzy*.

c. Melakukan inisialisasi prosedur *training Self-Organizing Maps*

   Inisialisasi merupakan tahap penentuan nilai-nilai awal yang akan digunakan dalam melakukan prosedur *training Self-Organizing Maps*. Nilai awal yang harus ditentukan antara lain waktu, *nodes* masukan, *nodes* keluaran, bobot, *neighbourhood*, dan *learning rate*

d. Melakukan prosedur *training Self-Organizing Maps*

   Setelah melakukan inisialisasi nilai-nilai awal maka langkah selanjutnya adalah melakukan prosedur *training* menggunakan algoritma *Self-Organizing Maps*.

e. Menampilkan representasi hasil implementasi

   Principal Component Analysis akan digunakan dengan bantuan Minitab untuk mereduksi data sehingga dapat direpresentasikan. Representasi hasil implementasi akan dibuat dengan Matlab. Representasi

akan ditampilkan dengan *Scatter Plot* pada koordinat 2 dimensi.

4. **Pengujian**

Setelah dibuat implementasi, akan dilakukan pengujian untuk memeriksa apakah hasil implementasi sudah sesuai atau terjadi eror.

5. **Kesimpulan dan Saran**

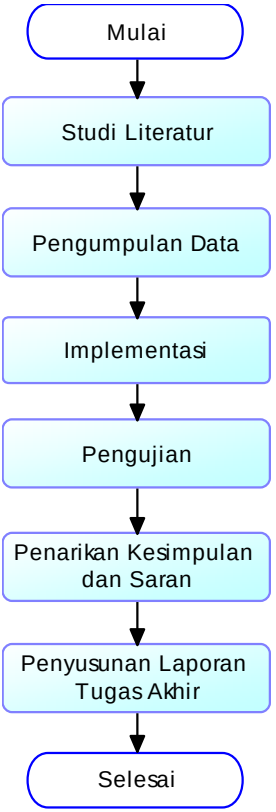
Setelah dilakukan analisis dan pembahasan maka dapat ditarik suatu kesimpulan dan saran sebagai masukan untuk pengembangan penelitian lebih lanjut.

6. **Penyusunan Laporan Tugas Akhir**

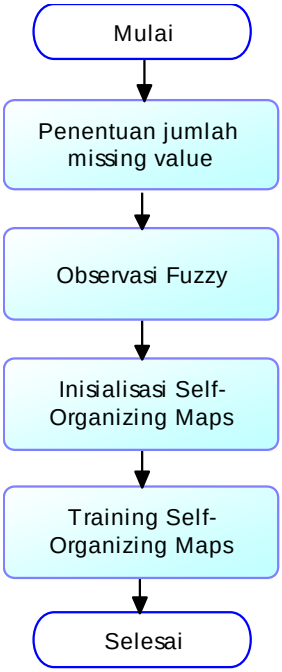
Setelah semua proses selesai dilakukan maka tahap terakhir adalah penyusuan laporan Tugas Akhir.

3.2 Diagram Alir Penelitian

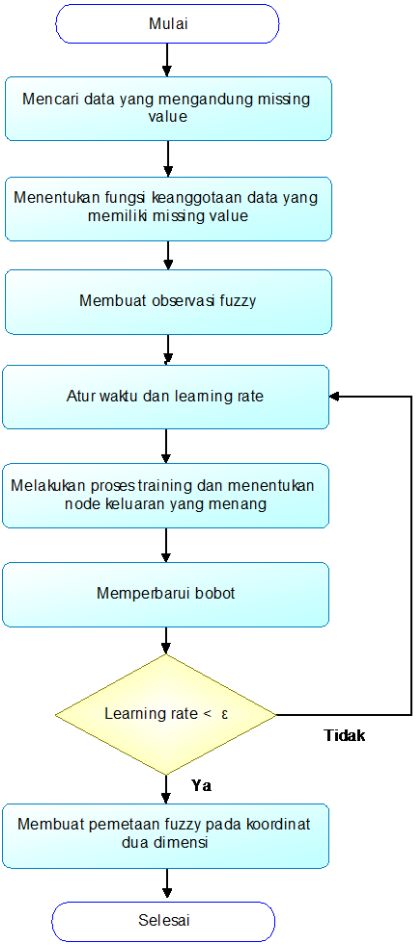
Berdasarkan uraian tersebut diatas, penelitian Tugas Akhir ini dapat dinyatakan dalam diagram alir sebagai berikut.



Gambar 3.1 Diagram Alir Metode Penelitian



Gambar 3.2 Diagram Alir Implementasi



Gambar 3.3 Diagram Alir Algoritma Self-Organizing Fuzzy Maps

BAB IV
PERANCANGAN IMPLEMENTASI

Bab ini menjelaskan rancangan implementasi yang digunakan sebagai acuan untuk implementasi. Perancangan implementasi menggambarkan proses rancang bangun secara terperinci dari awal tahap pengumpulan data hingga implementasi *Self-Organizing Fuzzy Maps*.

4.1 Perancangan Fuzzifikasi Data

Tahap ini bertujuan untuk menjelaskan tentang proses-proses yang dilakukan terhadap data awal yang diperoleh, sehingga data tersebut bisa digunakan. Data yang diperoleh penulis merupakan data kandungan gizi bahan pangan yang terdiri dari nama bahan pangan, kandungan protein, kandungan lemak, kandungan karbohidrat, kandungan kalsium, kandungan fosfor, kandungan besi, kandungan vitamin A, kandungan vitamin B1, kandungan Vitamin C, dan kandungan Air. Fungsi keanggotaan ini dibagi menjadi 3 kategori yaitu rendah, sedang, dan tinggi. Berdasarkan pedoman gizi seimbang dari Kementrian Kesehatan RI [24] dan Food and Agriculture Organization (FAO) [25], fungsi keanggotaan untuk gizi bahan pangan dapat dilihat sebagai berikut :

1. Fungsi keanggotaan untuk protein

Fungsi keanggotaan untuk nilai gizi protein adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 6{,}8 \\ \dfrac{(13-x)}{6{,}2} & ,6{,}8 < x < 13 \\ 0 & ,x \geq 13 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 6{,}8 \\ \dfrac{(x-6{,}8)}{6{,}2} & ,6{,}8 < x \leq 13 \\ \dfrac{(17-x)}{4} & ,13 < x < 17 \\ 0 & ,x \geq 17 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 13 \\ \dfrac{(17-x)}{4} & ,13 < x < 17 \\ 1 & ,x \geq 17 \end{cases}$$

2. Fungsi keanggotaan untuk lemak

Fungsi keanggotaan untuk nilai gizi lemak adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 5 \\ \dfrac{(16-x)}{11} & ,5 < x < 16 \\ 0 & ,x \geq 16 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 5 \\ \dfrac{(x-5)}{11} & ,5 < x \leq 16 \\ \dfrac{(30-x)}{14} & ,16 < x < 30 \\ 0 & ,x \geq 30 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 16 \\ \dfrac{(30-x)}{14} & ,16 < x < 30 \\ 1 & ,x \geq 30 \end{cases}$$

3. Fungsi keanggotaan untuk karbohidrat

Fungsi keanggotaan untuk nilai gizi karbohidrat adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 12 \\ \dfrac{(30-x)}{18} & ,12 < x < 30 \\ 0 & ,x \geq 30 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 12 \\ \dfrac{(x-12)}{18} & ,12 < x \leq 30 \\ \dfrac{(70-x)}{40} & ,30 < x < 70 \\ 0 & ,x \geq 70 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 30 \\ \dfrac{(x-30)}{40} & ,30 < x < 70 \\ 1 & ,x \geq 70 \end{cases}$$

4. Fungsi keanggotaan untuk kalsium

Fungsi keanggotaan untuk nilai gizi kalsium adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 5 \\ \dfrac{(20-x)}{15} & ,5 < x < 20 \\ 0 & ,x \geq 20 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 5 \\ \dfrac{(x-5)}{15} & ,5 < x \leq 20 \\ \dfrac{(37-x)}{17} & ,20 < x < 37 \\ 0 & ,x \geq 37 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 20 \\ \dfrac{(x-20)}{17} & ,20 < x < 37 \\ 1 & ,x \geq 37 \end{cases}$$

5. Fungsi keanggotaan untuk fosfor

Fungsi keanggotaan untuk nilai gizi fosfor adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 110 \\ \dfrac{(170-x)}{60} & ,110 < x < 170 \\ 0 & ,x \geq 170 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 110 \\ \dfrac{(x-110)}{60} & ,110 < x \leq 170 \\ \dfrac{(256-x)}{86} & ,170 < x < 256 \\ 0 & ,x \geq 256 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 170 \\ \dfrac{(x-170)}{86} & ,170 < x < 256 \\ 1 & ,x \geq 256 \end{cases}$$

6. Fungsi keanggotaan untuk besi

Fungsi keanggotaan untuk nilai gizi besi adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 2 \\ \dfrac{(5{,}7 - x)}{3{,}7} & ,2 < x < 5{,}7 \\ 0 & ,x \geq 5{,}7 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 2 \\ \dfrac{(x - 2)}{3{,}7} & ,2 < x \leq 5{,}7 \\ \dfrac{(20 - x)}{14{,}3} & ,5{,}7 < x < 20 \\ 0 & ,x \geq 20 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 5{,}7 \\ \dfrac{(x - 5{,}7)}{14.3} & ,5{,}7 < x < 20 \\ 1 & ,x \geq 20 \end{cases}$$

7. Fungsi keanggotaan untuk vitamin A

Fungsi keanggotaan untuk nilai gizi vitamin A adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 1000 \\ \dfrac{(3900 - x)}{2900} & ,1000 < x < 3900 \\ 0 & ,x \geq 3900 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 1000 \\ \dfrac{(x-1000)}{2900} & ,1000 < x \leq 3900 \\ \dfrac{(15000-x)}{11100} & ,3900 < x < 15000 \\ 0 & ,x \geq 15000 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 3900 \\ \dfrac{(x-3900)}{11100} & ,3900 < x < 15000 \\ 1 & ,x \geq 15000 \end{cases}$$

8. Fungsi keanggotaan untuk vitamin B1

Fungsi keanggotaan untuk nilai gizi vitamin B1 adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 0{,}1 \\ \dfrac{(0{,}28-x)}{0{,}18} & ,0{,}1 < x < 0{,}28 \\ 0 & ,x \geq 0{,}28 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 0{,}1 \\ \dfrac{(x-0{,}1)}{0{,}18} & ,0{,}1 < x \leq 0{,}28 \\ \dfrac{(0{,}45-x)}{0{,}17} & ,0{,}28 < x < 0{,}45 \\ 0 & ,x \geq 0{,}45 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 0{,}28 \\ \dfrac{(x-0{,}28)}{0{,}17} & ,0{,}28 < x < 0{,}45 \\ 1 & ,x \geq 0{,}45 \end{cases}$$

9. Fungsi keanggotaan untuk vitamin C

Fungsi keanggotaan untuk nilai gizi vitamin C adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 10 \\ \dfrac{(40-x)}{30} & ,10 < x < 40 \\ 0 & ,x \geq 40 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 10 \\ \dfrac{(x-10)}{30} & ,10 < x \leq 40 \\ \dfrac{(80-x)}{40} & ,40 < x < 80 \\ 0 & ,x \geq 80 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 40 \\ \dfrac{(x-40)}{40} & ,40 < x < 80 \\ 1 & ,x \geq 80 \end{cases}$$

10. Fungsi keanggotaan untuk air

Fungsi keanggotaan untuk nilai air adalah sebagai berikut :

$$\mu_{Rendah}(x) = \begin{cases} 1 & ,x \leq 25 \\ \dfrac{(55-x)}{30} & ,25 < x < 55 \\ 0 & ,x \geq 55 \end{cases}$$

$$\mu_{Sedang}(x) = \begin{cases} 0 & ,x \leq 25 \\ \dfrac{(x-25)}{30} & ,25 < x \leq 55 \\ \dfrac{(70-x)}{15} & ,55 < x < 70 \\ 0 & ,x \geq 70 \end{cases}$$

$$\mu_{Tinggi}(x) = \begin{cases} 0 & ,x \leq 55 \\ \dfrac{(x-55)}{15} & ,55 < x < 70 \\ 1 & ,x \geq 70 \end{cases}$$

Data yang memiliki *missing value* akan dicari derajat keanggotaannya, diubah menjadi variabel linguistik, dan selanjutnya akan digunakan aturan *fuzzy* untuk mengisi *missing value*. Berdasarkan Daftar Komposisi Zat Gizi Pangan Indonesia dari Kementrian Kesehatan RI dan Food and Agriculture Organization (FAO) [25], didapatkan beberapa aturan yang digunakan untuk mencari *missing value.* Aturan-aturan tersebut dapat dilihat pada Tabel 4.1.

Tabel 4.1 Aturan *Fuzzy*

No	Aturan
1	IF (protein=rendah) AND (lemak=rendah) AND (air=rendah) THEN (karbohidrat=tinggi)
2	IF (protein=rendah) AND (lemak=rendah) AND (air=tinggi) THEN (karbohidrat=tinggi)
3	IF (protein=rendah) AND (lemak=rendah) AND (air=sedang) THEN (karbohidrat=tinggi)
4	IF (protein=rendah) AND (lemak=sedang) AND (air=rendah) THEN (karbohidrat=tinggi)
5	IF (protein=rendah) AND (lemak=sedang) AND (air=sedang) THEN (karbohidrat=tinggi)

6	IF (protein=rendah) AND (lemak=sedang) AND (air=tinggi) THEN (karbohidrat=sedang)
7	IF (protein=rendah) AND (lemak=tinggi) AND (air=rendah) THEN (karbohidrat=sedang)
8	IF (protein=rendah) AND (lemak=tinggi) AND (air=tinggi) THEN (karbohidrat=rendah)
9	IF (protein=rendah) AND (lemak=tinggi) AND (air=sedang) THEN (karbohidrat=sedang)
10	IF (protein=sedang) AND (lemak=rendah) AND (air=rendah) THEN (karbohidrat=tinggi)
11	IF (protein=sedang) AND (lemak=rendah) AND (air=tinggi) THEN (karbohidrat=rendah)
12	IF (protein=sedang) AND (lemak=rendah) AND (air=sedang) THEN (karbohidrat=sedang)
13	IF (protein=sedang) AND (lemak=sedang) AND (air=rendah) THEN (karbohidrat=sedang)
14	IF (protein=sedang) AND (lemak=sedang) AND (air=tinggi) THEN (karbohidrat=rendah)
15	IF (protein=sedang) AND (lemak=sedang) AND (air=sedang) THEN (karbohidrat=rendah)
16	IF (protein=sedang) AND (lemak=tinggi) AND (air=rendah) THEN (karbohidrat=sedang)
17	IF (protein=sedang) AND (lemak=tinggi) AND (air=tinggi) THEN (karbohidrat=rendah)
18	IF (protein=sedang) AND (lemak=tinggi) AND (air=sedang) THEN (karbohidrat=sedang)
19	IF (protein=tinggi) AND (lemak=rendah) AND (air=rendah) THEN (karbohidrat=tinggi)
20	IF (protein=tinggi) AND (lemak=rendah) AND (air=tinggi) THEN (karbohidrat=rendah)
21	IF (protein=tinggi) AND (lemak=rendah) AND (air=sedang) THEN (karbohidrat=sedang)
22	IF (protein=tinggi) AND (lemak=sedang) AND (air=rendah) THEN (karbohidrat=sedang)

23	IF (protein=tinggi) AND (lemak=sedang) AND (air=tinggi) THEN (karbohidrat=rendah)
24	IF (protein=tinggi) AND (lemak=sedang) AND (air=sedang) THEN (karbohidrat=rendah)
25	IF (protein=tinggi) AND (lemak=tinggi) AND (air=rendah) THEN (karbohidrat=rendah)
26	IF (protein=tinggi) AND (lemak=tinggi) AND (air=tinggi) THEN (karbohidrat=rendah)
27	IF (protein=tinggi) AND (lemak=tinggi) AND (air=sedang) THEN (karbohidrat=rendah)
28	IF (protein=tinggi) THEN (fosfor=tinggi)
29	IF (protein=sedang) THEN (fosfor=sedang)
30	IF (protein=rendah) THEN (fosfor=rendah)
31	IF (protein=tinggi) THEN (vitamin b1=tinggi)
32	IF (protein=sedang) THEN (vitamin b1=sedang)
33	IF (protein=rendah) THEN (vitamin b1=rendah)
34	IF (kalsium=tinggi) THEN (besi=tinggi)
35	IF (kalsium=sedang) THEN (besi=sedang)
36	IF (kalsium=rendah) THEN (besi=rendah)

Untuk memudahkan dalam implementasi, dibangun basis data. Basis data digunakan sebagai penyimpanan data di MySQL.

4.2 Perancangan *Self-Organizing Fuzzy Maps*

Penerapan metode *Self-Organizing Fuzzy Maps* pada penelitian ini dapat digambarkan sebagai berikut :

a. Sumber data yang digunakan adalah data kandungan gizi pada bahan pangan yang dikeluarkan oleh Badan Ketahanan Pangan dan Penyuluhan Daerah Istimewa Yogyakarta. Terdapat 287 bahan pangan dalam data tersebut.
b. Pembangkitan observasi fuzzy :

1. Misalkan S adalah jumlah observasi yaitu 287 data bahan pangan.
2. Jumlah data bahan pangan tanpa *missing value* adalah 264. Jumlah data bahan pangan dengan *missing value* adalah 23.
3. Data bahan pangan yang memiliki *missing value* akan dibangkitkan menjadi observasi *fuzzy*..

c. Inisialisasi prosedur *training Self-Organizing Maps*
1. Atur waktu t = 0
2. Misalkan, jumlah nodes masukan adalah m dan jumlah nodes keluaran adalah N. Nodes masukan (m) adalah jumlah input data bahan pangan yaitu 287. Nodes keluaran (N) adalah jumlah cluster yang diinginkan. Berdasarkan beberapa kali percobaan, akan dipilih $N = 5$ karena untuk $N > 5$ akan terdapat *cluster* yang tidak memiliki anggota.
3. Misalkan i adalah indeks nodes masukan, dan j adalah indeks nodes keluaran. Bobot awal $w_{ij}(t = 0)$ akan secara acak dibuat oleh program.
4. Misalkan $\eta(t)$ adalah learning rate pada saat t dengan $0 < \eta(t) < 1$. Perubahan learning rate adalah $\eta(t) = \delta_1\ \eta(t)$ dengan $0 < \delta_1 < 1$. Berdasarkan beberapa kali percobaan, dipilih nilai $\eta(t = 0) = 0.3$ dan nilai $\delta_1 = 0.85$.
5. Nilai ε adalah nilai yang digunakan untuk *stopping condition.* Nilai ε harus memenuhi $0 < \varepsilon < 1$ dan $\varepsilon < \eta(t)$. Berdasarkan acuan penelitian yang digunakan, akan dipilih nilai $\varepsilon = 10^{-23}$ [2].

d. *Training Self-Organizing Maps*
1. Hitung :

$$D_j = \sum_{i=1}^{M} \left(w_{ij}(t) - x_i(t)\right)^2$$

Dimana $x_i(t)$ adalah masukan node i pada waktu t dan $w_{ij}(t)$ adalah bobot dari masukan node i ke keluaran node j pada waktu t

2. Pilih keluaran node J yang menang sedemikian hingga $D_j = \min\{D_j\}$
3. Perbarui bobot yang menang sebagai berikut :
   $w_{ij}(t+1) = \left[w_{ij}(t) + \eta(t)\left(x_i(t) - w_{ij}(t)\right)\right]$
   $w_{ij}(t+1)$ merupakan bobot baru
4. Jika $\eta(t) < \varepsilon$, maka iterasi dihentikan;
   Jika sebaliknya, hitung :
   $\eta(t) = \delta_1\ \eta(t)$

Atur $t = t + 1$; dan kembali ke langkah (1)

4.3 Perancangan Proses *Data Mining*

Dalam pengelompokan data bahan pangan, ada beberapa tahapan yang harus dilakukan. Tahapan-tahapan tersebut adalah sebagai berikut :

4.3.1 Pengambilan Data

Data yang telah diolah pada proses sebelumnya, selanjutnya sudah siap untuk diproses menggunakan algoritma *Self-Organizing Fuzzy Maps*. Data tersebut disimpan ke dalam DBMS MySQL, karena pembuatan program pada penelitian ini menggunakan Java, maka perlu dibuat koneksi antara Java dengan MySQL. Kode yang digunakan adalah sebagai berikut :

```
try {
            Class.forName("com.mysql.jdbc.Driver");
            connection                              =
DriverManager.getConnection
("jdbc:mysql://localhost:3306/tugasakhir?zeroDateTi
meBehavior=convertToNull", "root", "");
            statement                               =
connection.createStatement();
```

```
        }
        catch(Exception e){
            JOptionPane.showMessageDialog(null,
"Koneksi gagal");
```

Untuk memudahkan proses *training* dengan NETBEANS maka data dari *database* disimpan kedalam array yang meliputi :

a. Array untuk menyimpan nama bahan pangan. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(nama_training) FROM
training"
"SELECT nama_ training FROM training "
```

b. Array untuk menyimpan nilai protein. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(protein_ training) FROM
training"
"SELECT protein_ training FROM training"
```

c. Array untuk menyimpan nilai lemak. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(lemak_ training) FROM
training "
"SELECT lemak_ training FROM training"
```

d. Array untuk menyimpan nilai karbohidrat. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(karbohidrat_ training) FROM
training"
"SELECT karbohidrat_ training FROM
training"
```

e. Array untuk menyimpan nilai kalsium. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(kalsium_ training) FROM
training "
"SELECT kalsium_training FROM training "
```

f. Array untuk menyimpan nilai fosfor. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(fosfor_training) FROM
                training"
"SELECT fosfor_ training FROM training"
```

g. Array untuk menyimpan nilai besi. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(besi_ training) FROM
                training"
"SELECT besi_ training FROM training"
```

h. Array untuk menyimpan nilai vitamin A. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(a_ training) FROM training"
"SELECT a_ training FROM training"
```

i. Array untuk menyimpan nilai vitamin B1. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(b1_training) FROM training"
"SELECT b1_training FROM training"
```

j. Array untuk menyimpan nilai vitamin C. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(c_ training) FROM training"
"SELECT c_ training FROM training"
```

k. Array untuk menyimpan nilai air. Untuk mengambil nilai dari *database* maka digunakan *query* :

```
"SELECT count(air_ training) FROM
                training"
"SELECT air_ training FROM training"
```

4.3.2 Pengolompokan dengan Self-Organizing Fuzzy Maps

Proses selanjutnya adalah pengelompokan bahan pangan. Jumlah pengelompokan bergantung pada *user* dengan meng*input*kannya di awal program. Dalam hal ini cara yang digunakan adalah dengan membaca basis data untuk mengambil setiap *record* yang akan dikelompokkan. Kode yang digunakan dalam mencari bobot adalah :

```
for(int z=0; z<NAMA.length; z++){
   for(int i=0; i<cluster; i++){
     D[i]=(Math.pow((w0[i][0]-
PROTEIN[z]),2))+(Math.pow((w0[i][1]-LEMAK[z]),2))
```

```
                              +(Math.pow((w0[i][2]-
KARBOHIDRAT[z]),2))+(Math.pow((w0[i][3]-KALSIUM[z]),2))
                              +(Math.pow((w0[i][4]-
FOSFOR[z]),2))+(Math.pow((w0[i][5]-BESI[z]),2))
                              +(Math.pow((w0[i][6]-
VITA[z]),2))+(Math.pow((w0[i][7]-VITB1[z]),2))
                              +(Math.pow((w0[i][8]-
VITC[z]),2))+(Math.pow((w0[i][9]-AIR[z]),2));
                     System.out.println("Nilai  D["+i+"]  :
"+D[i]);
                }
    min = D[0];
    for(int b=0; b<cluster; b++){
       if (D[b]< min){
           min = D[b];
           w0[b][0]=w0[b][0]+(alpha*(PROTEIN[z]-
w0[b][0]));
           w0[b][1]=w0[b][1]+(alpha*(LEMAK[z]-w0[b][1]));
           w0[b][2]=w0[b][2]+(alpha*(KARBOHIDRAT[z]-
w0[b][2]));
           w0[b][3]=w0[b][3]+(alpha*(KALSIUM[z]-
w0[b][3]));
           w0[b][4]=w0[b][4]+(alpha*(FOSFOR[z]-
w0[b][4]));
           w0[b][5]=w0[b][5]+(alpha*(BESI[z]-w0[b][5]));
           w0[b][6]=w0[b][6]+(alpha*(VITA[z]-w0[b][6]));
           w0[b][7]=w0[b][7]+(alpha*(VITB1[z]-w0[b][7]));
           w0[b][8]=w0[b][8]+(alpha*(VITC[z]-w0[b][8]));
           w0[b][9]=w0[b][9]+(alpha*(AIR[z]-w0[b][9]));
                    }
           }
            System.out.println("Nilai D minimum : "+min);
            System.out.println("alpha "+y+" : "+alpha);

            for(int j=0; j<10; j++){
                System.out.println();
                for(int i=0; i<cluster; i++){
                    System.out.print("w["+i+"]["+j+"]:
"+w0[i][j]);
                  }}}
            alpha = 0.5*alpha;
         }
```

Sedangkan kode yang digunakan dalam menentukan hasil pengelompokan adalah :

```
for(int i=0; i<cluster; i++){
```

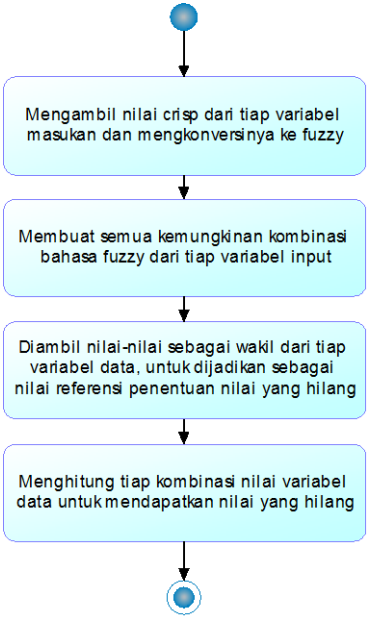
```
                C[i]=(Math.pow((w0[i][0]-
PROTEIN[z]),2))+(Math.pow((w0[i][1]-LEMAK[z]),2))
                            +(Math.pow((w0[i][2]-
KARBOHIDRAT[z]),2))+(Math.pow((w0[i][3]-KALSIUM[z]),2))
                            +(Math.pow((w0[i][4]-
FOSFOR[z]),2))+(Math.pow((w0[i][5]-BESI[z]),2))
                            +(Math.pow((w0[i][6]-
VITA[z]),2))+(Math.pow((w0[i][7]-VITB1[z]),2))
                            +(Math.pow((w0[i][8]-
VITC[z]),2))+(Math.pow((w0[i][9]-AIR[z]),2));
                    System.out.println("Nilai  C["+i+"]  :
"+C[i]);
            }

            kecil = C[0];
            for(t=0; t<cluster; t++){
                if (C[t]< kecil){
                        kecil = C[t];
                        index = t;
                }
            }
            Clust[z]=index;
            System.out.println("Nilai    D    minimum    :
"+kecil);
            System.out.println();
            System.out.println("Jadi,  "+NAMA[z]+"  masuk
kedalam cluster "+Clust[z]);
```

BAB V
IMPLEMENTASI

Pada bab ini dijelaskan tentang implementasi dalam bahasa pemrograman Java dan hasil uji coba program menggunakan data kandungan gizi pada 415 bahan pangan.

5.1 Fuzzifikasi Data

Pada tahap ini akan dijelaskan implementasi fuzzifikasi pada data gizi bahan pangan yang memiliki *missing value*: Alur pencarian *missing value* :



Gambar 5. 1 Alur pencarian *missing value*

Berikut merupakan salah satu contoh pencarian *missing value* pada data gizi bahan pangan :

a. Pada data gizi Beras Ketan Hitam, terdapat *missing value* pada bagian nilai gizi karbohidrat. Data nilai gizi karbohidrat digambarkan dalam vektor berikut :

   [7 0,7 – 10 148 0,8 0 0,2 0 13]

b. Dilakukan fuzzifikasi pada data gizi Beras Ketan Hitam
   i. Nilai kandungan protein pada Beras Ketan Hitam adalah 7. Berdasarkan derajat keanggotaan pada bab 4.1 didapatkan derajat keanggotaan $\mu_{p1} = 0{,}967$ untuk himpunan rendah dan derajat keanggotaan $\mu_{p2} = 0{,}032$ untuk himpunan sedang. Akan dipilih adalah derajat keanggotaan terkecil yaitu 0.032
   ii. Nilai kandungan lemak pada Beras Ketan Hitam adalah 0,7. Berdasarkan fungsi keanggotaan pada bab 4.1 didapatkan bahwa Beras Hitam masuk kedalam himpunan rendah dengan derajat keanggotaan $\mu_l = 1$.
   iii. Nilai kandungan air pada Beras Ketan Hitam adalah 13. Berdasarkan fungsi keanggotaan pada bab 4.1 didapatkan bahwa Beras Hitam masuk kedalam himpunan rendah dengan derajat keanggotaan $\mu_a = 1$.

c. Berdasarkan aturan *fuzzy* yang telah dibahas sebelumnya, untuk mencari nilai karbohidrat didapatkan:
   i. IF protein rendah AND lemak rendah AND air rendah THEN karbohidrat tinggi

      $\alpha = \mu_{p1} \wedge \mu_l \wedge \mu_a = \min(0{,}032, 1, 1) = 0{,}032$

d. Jadi, vektor Beras Ketan Hitam yang akan digunakan untuk *training Self-Organizing Maps* adalah :

   [0.032 1 0,032 0,33 0,37 0 0 0,44 0 1]

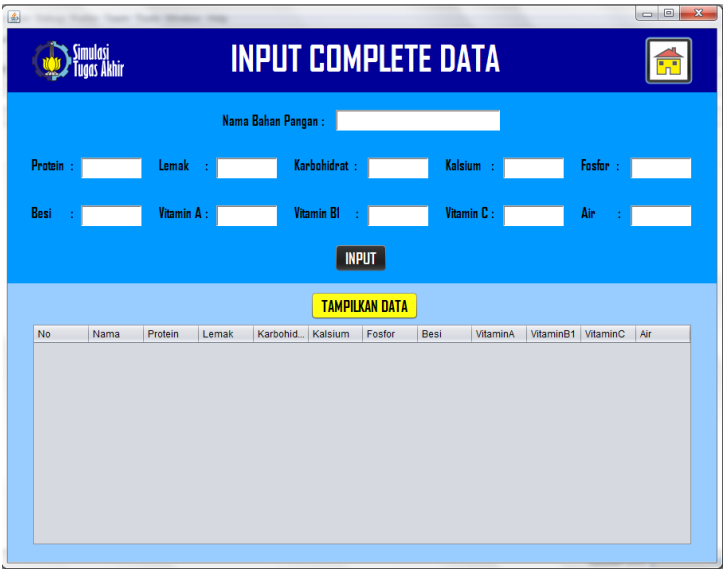
5.2 *Graphical User Interface (GUI)* Program

Perancangan program yang telah dibuat selanjutnya diimplementasikan dalam bahasa pemrograman Java dengan menggunakan *software* Netbeans IDE 7.4. Web server yang digunakan adalah XAMPP 3.2.1 dengan database MySQL. Gambar 5.2 merupakan tampilan menu utama.

Gambar 5.2 Tampilan Menu Utama

5.2.1 Form Input Complete Data

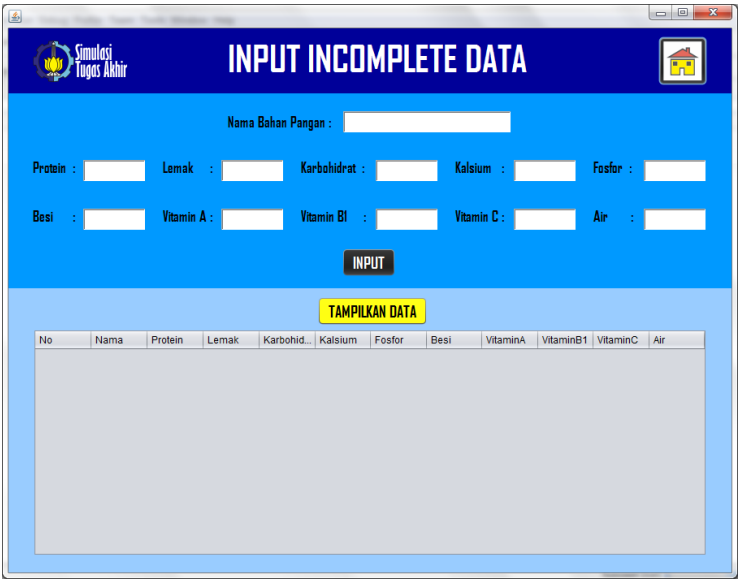
Gambar 5.3 merupakan form untuk menambahkan data bahan pangan yang nilai gizinya lengkap *(complete data)*. Jika ada salah satu bagian yang kosong maka data tidak dapat dimasukkan. Jika sebelumnya nama bahan pangan sudah dimasukkan kedalam basis data, maka tidak dapat dimasukkan lagi. Setelah memasukkan data, tabel akan otomatis menampilkan isi basis data.



Gambar 5.3 Form *Input Complete Data*

5.2.2 Form Input Incomplete Data

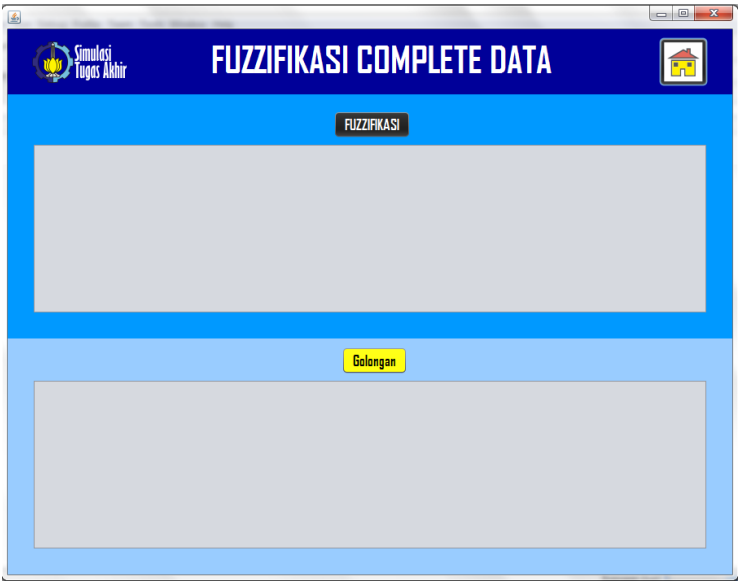
Gambar 5.3 merupakan form untuk menambahkan data bahan pangan yang nilai gizinya tidak lengkap *(incomplete data)*. Jika sebelumnya nama bahan pangan sudah dimasukkan kedalam basis data, maka tidak dapat dimasukkan lagi. Setelah memasukkan data, tabel akan otomatis menampilkan isi basis data. Gizi yang nilainya tidak diketahui, otomatis diisi dengan angka 999999999.



Gambar 5.4 Form *Input Incomplete Data*

5.2.3 Menu Fuzzifikasi Untuk *Complete Data*

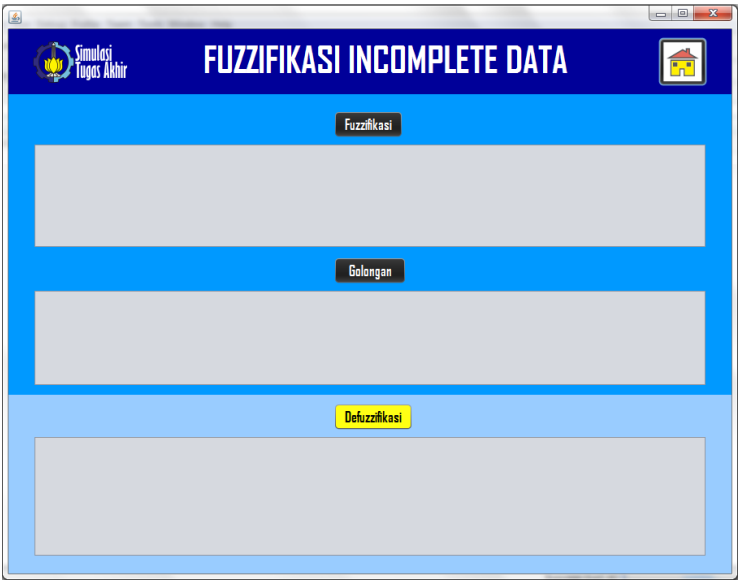
Gambar 5.5 merupakan menu untuk fuzzifikasi data bahan pangan yang merupakan *complete data* sebelum dilakukan *training*. Terdapat 2 tombol yaitu tombol fuzzifikasi dan tombol golongan untuk melihat gizi bahan pangan termasuk kedalam himpunan sedang, rendah, atau tinggi



Gambar 5. 5 Menu Fuzzifikasi Untuk Complete Data

5.2.4 Menu Fuzzifikasi Untuk *Incomplete Data*

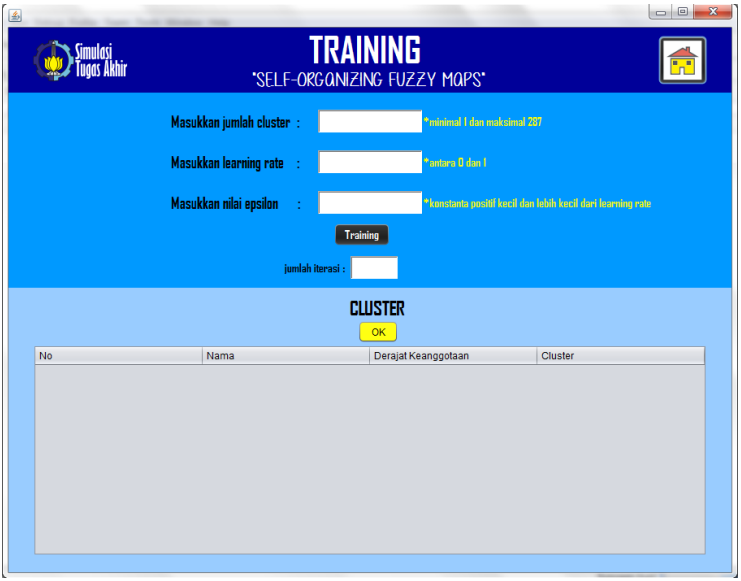
Gambar 5.6 merupakan menu untuk fuzzifikasi data bahan pangan yang merupakan In*complete data* sebelum dilakukan *training*. Terdapat 3 tombol yaitu tombol fuzzifikasi, golongan dan defuzzifikasi.



Gambar 5. 6 Menu Fuzzifikasi Untuk Incomplete Data

5.2.5 Menu Self-Organizing Fuzzy Maps

Menu ini digunakan untuk melakukan *training Self-Organizing Fuzzy Maps* untuk mendapatkan hasil pengelompokan bahan pangan sesuai kandungan gizinya. Sebelum melakukan *training*, harus dimasukkan jumlah cluster, learning rate, dan jumlah iterasi. Tampilan menu ini dapat dilihat pada Gambar 5.7.



Gambar 5.7 Menu *Training Self-Organizing Fuzzy Maps*

5.3 Implementasi

Pada sub bab ini akan ditunjukkkan hasil dari implementasi program. Pada penelitian ini digunakan 5 cluster, dengan learning rate 0,3.

5.3.1 Nilai Bobot Awal dan Nilai Bobot Akhir

a. Nilai bobot awal yang dibuat secara *random* oleh program pada saat implementasi ini adalah :

$$\begin{bmatrix} 0.409 & 0.345 & 0.213 & 0.987 & 0.504 \\ 0.698 & 0.346 & 0.220 & 0.774 & 0.614 \\ 0.270 & 0.693 & 0.905 & 0.803 & 0.053 \\ 0.699 & 0.381 & 0.108 & 0.913 & 0.813 \\ 0.758 & 0.335 & 0.002 & 0.453 & 0.645 \\ 0.653 & 0.857 & 0.432 & 0.084 & 0.232 \\ 0.819 & 0.121 & 0.339 & 0.463 & 0.961 \\ 0.478 & 0.932 & 0.732 & 0.021 & 0.513 \\ 0.853 & 0.938 & 0.418 & 0.561 & 0.782 \\ 0.217 & 0.998 & 0.464 & 0.733 & 0.660 \end{bmatrix}$$

b. Nilai bobot akhir yang diperoleh adalah :

$$\begin{bmatrix} 15.911 & 3.584 & 16.289 & 95.087 & 26.327 \\ 3.034 & 0.616 & 1.875 & 20.084 & 81.295 \\ 6.012 & 6.953 & 76.735 & 13.159 & 1.769 \\ 6.584 & 39.171 & 6.364 & 32.030 & 1.617 \\ 9.654 & 0.422 & 36.140 & 72.422 & 16.143 \\ 8.699 & 21.489 & 13.365 & 22.749 & 4.382 \\ 2.499 & 37.661 & 7.751 & 2.852 & 9.517 \\ 8.376 & 10.184 & 25.136 & 32.019 & 15.382 \\ 12.186 & 83.103 & 1.424 & 1.183 & 1.288 \\ 91.089 & 90.051 & 4.030 & 11.665 & 22.360 \end{bmatrix}$$

5.3.2 Hasil Pengelompokan

Hasil pengelompokan data bahan pangan yang merupakan *incomplete data* dapat dilihat pada Lampiran B. Berdasarkan hasil pengelompokan dapat dilihat bahwa bahan pangan dalam satu kelompok memiliki kemiripan pada kandungan gizi tertentu. Kemiripan kandungan gizi pada hasil pengelompokan dapat dilihat pada Tabel 5.1.

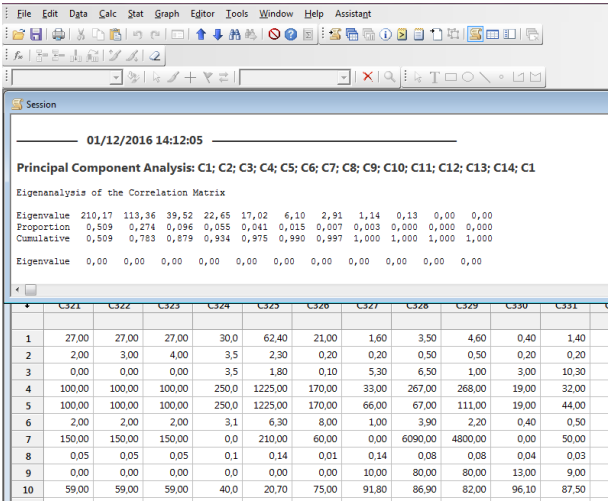
Tabel 5. 1 Kemiripan Kandungan Gizi Pada Hasil *Cluster*

Cluster	Kemiripan Kandungan Gizi
0	kandungan air tinggi,
	kandungan besi rendah
	kandungan lemak rendah

1	kandungan protein tinggi
	kandungan vitamin A rendah
2	kandungan vitamin C rendah
	kandungan fosfor tiggi
3	kandungan kalsium rendah
	kandungan vitamin B1 rendah
4	kandungan karbohidrat rendah

5.4 Reduksi Dimensi Data Untuk Representasi 2 Dimensi

Pada tahap ini akan digunakan tehnik *(PCA)* untuk reduksi dimensi data bahan pangan agar proses *training Self-Organizing Maps (SOM)* dapat direpresentasikan dalam koordinat 2 dimensi. Aplikasi yang digunakan untuk menerapkan tehnik *(PCA)* pada data bahan pangan adalah Minitab 17.



	C321	C322	C323	C324	C325	C326	C327	C328	C329	C330	C331
1	27,00	27,00	27,00	30,0	62,40	21,00	1,60	3,50	4,60	0,40	1,40
2	2,00	3,00	4,00	3,5	2,30	0,20	0,20	0,50	0,50	0,20	0,20
3	0,00	0,00	0,00	3,5	1,80	0,10	5,30	6,50	1,00	3,00	10,30
4	100,00	100,00	100,00	250,0	1225,00	170,00	33,00	267,00	268,00	19,00	32,00
5	100,00	100,00	100,00	250,0	1225,00	170,00	66,00	67,00	111,00	19,00	44,00
6	2,00	2,00	2,00	3,1	6,30	8,00	1,00	3,90	2,20	0,40	0,50
7	150,00	150,00	150,00	0,0	210,00	60,00	0,00	6090,00	4800,00	0,00	50,00
8	0,05	0,05	0,05	0,1	0,14	0,01	0,14	0,08	0,08	0,04	0,03
9	0,00	0,00	0,00	0,0	0,00	0,00	10,00	80,00	80,00	13,00	9,00
10	59,00	59,00	59,00	40,0	20,70	75,00	91,80	86,90	82,00	96,10	87,50

Gambar 5.8 *Principal Component Analysis* pada Minitab

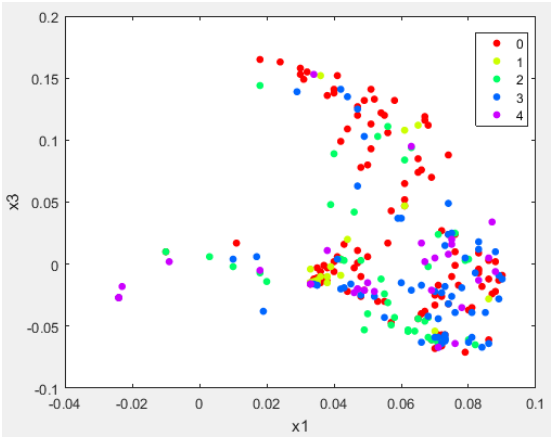
Berikut merupakan penerapan metode *Principal Component Analysis* untuk reduksi dimensi data:

i. *Principal component* yang dipilih adalah PC1 dan PC2. Pemilihan ini berdasarkan nilai eigen yang memiliki nilai lebih dari 1.
ii. Menghitung nilai $\bar{x}$ yang merupakan nilai rata-rata dari data awal.
iii. Menghitung nilai standar deviasi dari *principal component*.
iv. Menghitung $Z = \frac{x-\bar{x}}{sd}$, dengan x merupakan nilai-nilai *principal component*.
v. Nilai yang akan digunakan sebagai representasi adalah nilai xZ

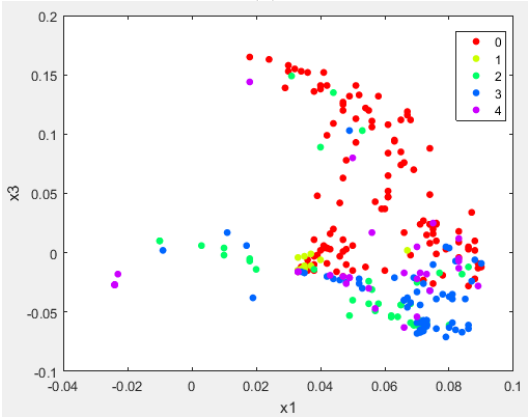
5.5 Representasi Hasil Implementasi

Setelah dilakukan implementasi penggunaan *Self-Organizing Fuzzy Maps* pada data kandungan gizi bahan pangan, selanjutnya akan dibuat representasi 2 dimensi dari tahap implementasi. Representasi dalam koordinat 2 dimensi ini menggunakan aplikasi Matlab. Data yang digunakan dalam membuat representasi ini adalah nilai yang telah didapatkan dari hasil *Principal Component Analysis (PCA).* Representasi akan ditampilkan dalam bentuk *Scatter Plot.*

Proses *training Self-Organizing Fuzzy Maps* menggunakan *learning rate* 0,3 dan perubahan learning rate adalah $\eta(t) = \delta_1 \ \eta(t)$ dengan $\delta_1 = 0{,}85$. Proses *training Self-Organizing Fuzzy Maps* dilakukan secara berulang hingga memenuhi $\eta(t) < \varepsilon$. Kondisi berhenti terpenuhi setelah dilakukan iterasi sebanyak 319 iterasi. Perbandingan representasi hasil cluster pada saat $\eta(t) = 0{,}0002$ dan representasi hasil cluster akhir untuk *principal component* 1 dan 3 dapat dilihat pada Gambar 5.9.



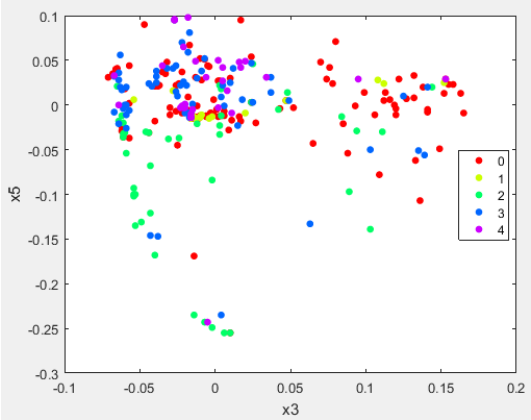
(a)



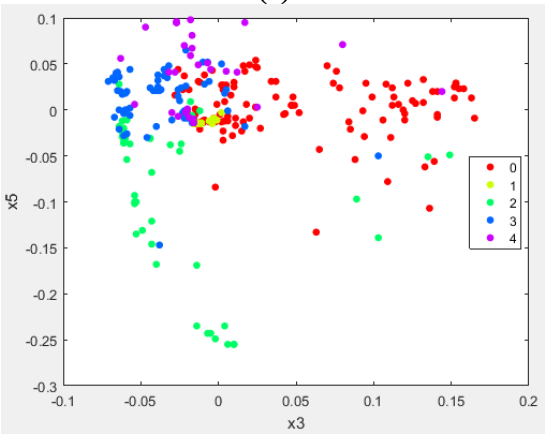
(b)

Gambar 5. 9. (a)Hasil *Cluster Principal Component* 1 dan 3 Untuk $\alpha = 0,0002$, (b)Hasil *Cluster* Akhir *Principal Component* 1 dan 3.

Perbandingan representasi hasil cluster pada saat $\eta(t) = 0{,}0002$ dan representasi hasil cluster akhir untuk *principal component* 3 dan 5 dapat dilihat pada Gambar 5.10.



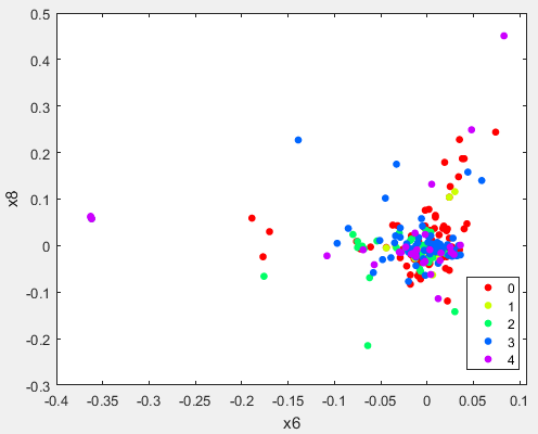
(a)



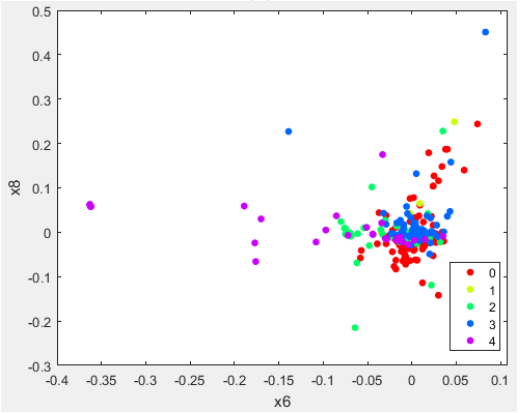
(b)

Gambar 5. 10. (a)Hasil *Cluster Principal Component* 3 dan 5 Untuk $\alpha = 0,0002$, (b)Hasil *Cluster* Akhir *Principal Component* 3 dan 5.

Perbandingan representasi hasil cluster pada saat $\eta(t) = 0{,}0002$ dan representasi hasil cluster akhir untuk *principal component* 6 dan 8 dapat dilihat pada Gambar 5.11.



(a)



(b)

Gambar 5. 11. (a)Hasil *Cluster Principal Component* 6 dan 8 Untuk $\alpha = 0,0002$, (b)Hasil *Cluster* Akhir *Principal Component* 6 dan 8.

BAB VI
PENUTUP

Pada bab ini berisi tentang beberapa kesimpulan yang dihasilkan berdasarkan penelitian yang telah dilaksanakan.dan saran yang dapat digunakan jika penelitian ini dikembangkan.

6.1 Kesimpulan

Berdasarkan analisis terhadap hasil pengujian program, maka dapat diambil kesimpulan sebagai berikut :

1. Algoritma *Self-Organizing Fuzzy Maps* telah berhasil diterapkan untuk mengelompokkan data kandungan gizi bahan pangan yang dikeluarkan oleh Badan Ketahanan Pangan dan Penyuluhan Daerah Istimewa Yogyakarta.
2. Bahan pangan yang memiliki *missing value* berhasil dikelompokkan menggunakan algoritma *Self-Organizing Fuzzy Maps.*
3. Bahan pangan dalam satu kelompok memiliki kemiripan kandungan gizi tertentu.
4. Algoritma *Self-Organizing Fuzzy Maps* telah terbukti sebagai algoritma *clustering* yang dapat berkerja pada data yang memiliki *missing value* atau *incomplete data*

6.2 Saran

Ada beberapa hal yang penulis sarankan untuk pengembangan penelitian selannjutnya :

1. Pada tugas akhir ini bahan pangan yang dapat dicari *missing value* nya adalah nilai gizi karbohidrat, lemak, protein, kalsium, fosfor, besi, vitamin B1 dan air. Akan lebih baik jika semua jenis gizi dapat dicari kandungan gizinya dengan melihat referensi hasil penelitian pangan yang lebih luas.

2. Data yang digunakan dalam tugas akhir ini berjumlah 288 data bahan pangan. Jumlah tersebut belum mencakup keseluruhan bahan pangan yang ada di Indonesia. Sehingga, lebih baik apabila data yang digunakan dapat lebih banyak.
3. Pada hasil representasi, cluster dapat dipisahkan jika dimensi representasinya bertambah. Untuk penelitian selanjutnya, dapat digunakan metode *Support Vector Machine* dengan fungsi kernel tidak linier.

DAFTAR PUSTAKA

[1] Oza, M. R. A, Martínez, M. J, Montero, E. P, Olivas, S. E. (2013), "Visual Data Mining with Self-Organising Maps for Ventricular Fibrillation Analysis", *Computer Method and Programs in Biomedicine*, Vol. 111, hal. 269-279.

[2] Wang, S. (2003), "Application of Self-Organizing Maps for Data Mining with Incomplete Data Sets", *Neural Comput & Applic*, Vol. 12, hal. 42-48.

[3] Walujo, E. (2011). *Keanekaragaman Hayati Untuk Pangan*, LIPI, Jakarta.

[4] Kementerian Pertanian RI. (2010), *Rencana strategis Badan Ketahanan Pangan 2010-2014*, Kementerian Pertanian Republik Indonesia, Jakarta.

[5] BPS. (2011), *Pengeluaran Untuk Konsumsi Penduduk Indonesia 2011*, Badan Pusat Statistik, Jakarta.

[6] Pemerintah Daerah DIY. (2016), *NBM sementara 2015*, Pemerintah Daerah DIY, Yogyakarta

[7] Meschino, J. G, Comas, S. D, Ballarin, L. V, Scandurra, G. A, Passoni, I, L. (2015), "Automatic design of interpretable fuzzy predicate systems for clustering using self-organizing maps". *Neurocomputing*, Vol. 147, hal. 47-59.

[8] Presser, K, Hinterberger, H, Weber, D, Norrie, M. (2016), "A scope classification of data quality requirements for food composition data". *Food Chemistry*, Vol. 193, hal. 166-172.

[9] Institute of Nutrition, Mahidol University. (2014), *ASEAN Food Composition Database*, ASEANFOODS Regional Centre, Thailand.

[10] Du, K.L. (2010), "Clustering A neural network approach". *Neural Network*, Vol. 23, hal.89-107.

[11] Longford, T.,Nicholas. (2005). *Missing Data and Small-Area*, Springer, New York.

[12] Little, R.J.A. and Rubin, D.B. 1987. Statistical Analysis with Missing Data. J. Wiley & Sons, New York.
[13] Undang-undang No. 18 Tahun 2012 tentang Pangan.
[14] Silva, A, Caro, C. J, Magaña-Lemus, D. (2016), "Household food security: Perceptions, behavior and nutritional quality of food purchases", *Journal of Economic Psychology*, Vol. 55, hal. 139-148.
[15] Muchtadi, D. (2009), *Pengantar Ilmu Gizi*, Alfabeta, Bandung.
[16] Miranda, A. A, Borgne, Y. A, Bontempi, G. (2008), "Routes from Minimal Approximation Error to Principal Components", *Neural Processing Letters*, Vol. 27
[17] Richard, A, Wichern, D. (1998), *Applied Multivariate Statistical Analysis*, Prentice-Hall, New Jersey.
[18] Kohonen, T. (2013), "Essentials of the self-organizing map", *Neural Networks*, Vol. 37, hal. 52-65.
[19] Jain, K. A, Dubes C. R. (1988), *Algorithms for clustering data*, Prentice Hall, New Jersey.
[20] Fausett, L. (1994), *Fundamentals of Neural Networks Architectures, Algorithms, and Applications*, Prentice-Hall, New Jersey.
[21] Wang, L. (1997), *A Course in Fuzzy Systems and Control*, Prentice-Hall, New Jersey.
[22] Zimmermann, H. (1992), *Fuzzy Set Theory and Its Applications*, Kluwer Academic Publihsers, Massachusetts.
[23] Negoita, C, Ralescu, A.D. (1975), "Representation Theorems for Fuzzy Concepts", *Kybernetes*, Vol. 4, hal. 169-174.
[24] Kementerian Kesehatan RI. (2014), *Pedoman Gizi Seimbang*, Kementerian Kesehatan Republik Indonesia, Jakarta.
[25] Food and Agriculture Organization. (2012), *FAO/INFOODS Guidelines for Food Matching*, Food and Agriculture Organization, Rome

LAMPIRAN A
Hasil *Cluster*

a. *Cluster* 0

Bahan pangan yang termasuk kedalam *cluster* 0 dapat dilihat pada Tabel A.1. Bahan pangan yang termasuk kedalam cluster ini memiliki kandungan air yang tinggi, kandungan besi yang rendah dan kandungan lemak yang rendah. Jumlah anggota pada *cluster* 0 adalah sebanyak 107 bahan pangan.

Tabel A.1 Daftar Bahan Pangan pada *Cluster* 0

Cluster 0	
Kacang Gude, Biji Muda	Daun Ketela Rambat (Ubi Jalar)
Daging Kuda	Daun Labu Siam
Ikan Mas	Daun Labu Waluh
Gadung	Daun Leunca
Ganyong	Daun Lobak
Gembili	Daun Lumpang Talas
Kentang	Daun Mangkokan
Ketela Pohon (Singkong)	Daun Melinjo
Mie Basah	Daun Oyong
Sente	Daun Pakis
Suweg	Daun Pepaya
Talas	Daun Singkong
Uwi	Daun Tales
Bongkrek (Tempe Bungkil Kelapa)	Jagung Muda Termasuk Tongkol
Jengkol	Jamur Kuping Segar
Kelapa Muda, Air	Jamur Pisang Segar
Kelapa Muda, Daging	Jotang

Kecap	Enceng
Koro Loke, Biji	Gambas (Oyong)
Lamtoro, Biji Muda	Genjer
Santan (Kelapa Diperas dengan Air)	Kangkung
Susu Kedelai	Kapri Muda
Tahu	Kacang Buncis, Buah
Taokoa	Kacang Kapri, Biji Segar
Ginjal Sapi	Kacang Panjang
Leverwors (Sosis Hati)	Katuk, Daun
Telur Ayam, Bagian Putih	Kelor, Daun
Telur Bebek, Bagian Putih	Kemangi
Telur Penyu	Kembang Turi
Beunteur	Ketimun
Keong	Kecipir
Kodok	Kelawi, Keluwih
Rebon (Udang Kecil Segar)	Kool Kembang
Teri Segar	Kool Merah, Kool Putih
Andewi	Koro Wedus, Buah Muda
Bayam	Krokot
Bayam Merah	Kucai
Baligo	Kucai Muda (Lokio)
Bawang Bombay	Labu Air
Bawang Merah	Labu Siam
Bawang Putih	Labu Waluh
Bengkoang	Leunca, Buah
Bit	Lobak
Boros Kunci	Melinjo

Boros Laja	Nangka Muda
Buncis	Pepaya Muda
Daun Bawang	Pare (Paria)
Daun Beluntas	Peterseli
Daun Jambu Mete Muda	Pe-cay
Daun Gandaria	Prei (Daun Bawang)
Daun Kedondong	Rebung
Daun Kemangi	Sawi
	Selada

b. *Cluster* 1

Bahan pangan yang termasuk kedalam *cluster* 1 dapat dilihat pada Tabel A.2. Bahan pangan yang termasuk kedalam cluster ini memiliki kandungan protein tinggi dan vitamin A rendah. Jumlah anggota pada *cluster* 1 adalah sebanyak 35 bahan pangan.

Tabel A.2 Daftar Bahan Pangan pada *Cluster* 1

Cluster 1	
Ikan Kembung	Bandeng
Cue Selar Kuning	Bawal
Ampas Tahu	Gabus Kering
Kemiri	Gabus Segar
Kecipir Biji	Ikan Hiu
Koro Benguk, Biji	Ikan Segar
Oncom	Ikan Kakap
Tempe Kedelai Murni	Kura-Kura
Ayam	Layang
Babat	Lemuru
Daging Domba	Petis Ikan

Daging Kerbau	Pindang Banjar
Dideh (Darah Sapi)	Pindang Selar Kecil
Hati Babi	Selar Kering
Krupuk Kulit Kerbau	Selar Segar
Sarang Burung	Sepat Kering
Telur Terubuk	Terasi Merah
Bader (Tawes)	

c. *Cluster* 2

Bahan pangan yang termasuk kedalam *cluster* 2 dapat dilihat pada Tabel A.3. Bahan pangan yang termasuk kedalam cluster ini memiliki kandungan vitamin C rendah dan fosfor tiggi. Jumlah anggota pada *cluster* 2 adalah sebanyak 59 bahan pangan.

Tabel A.3 Daftar Bahan Pangan pada *Cluster* 2

Cluster 2	
Beras Menir	Kenari
Kacang Endel, Biji	Ketumbar
Kacang Gude, Biji	Koro Wedus, Biji
Kacang Panjang, Biji	Kwaci
Kacang Tunggak (Kacang Tolo)	Pala, Biji
Ekor Kuning	Sari Dele, Bubuk
Beras Pecah Kulit	Tauco
Jagung Giling Kuning	Tepung Kacang Kedelai
Jagung Kuning Pipil Lama	Wijen
Jawawut	Corned Beef
Havemout	Daging Asap
Katul Beras	Daging Sapi

Katul Jagung	Dendeng Daging Sapi
Tepung Jagung Kuning	Hati Sapi
Tepung Jagung Putih	Otak
Cantel	Telur Ayam, Bagian Kuning
Biji Jambu Monyet	Telur Bebek, Bagian Kuning
Bungkil Biji Karet	Ikan Asin Kering
Bungkil Kacang Tanah	Kerang
Bungkil Kelapa	Kerupuk Udang dengan Pati
Emping (Krupuk Mlinjo)	Pada Banjar
Kacang Arab	Rebon Kering
Kacang Bogor	Sardiness dalam Kaleng
Kacang Ijo	Teri Bubuk
Kacang Kedelai Basah	Teri Kering
Kacang Kedelai Kering	Teri Kering Sekali, Tawar
Kacang Merah (Kacang Galing)	Teri Nasi, Kering
Kacang Tanah Terkelupas dengan Selaput	Udang Kering
Kacang Tanah Sangan dengan Selaput	Udang Segar
Keju Kacang Tanah	

d. *Cluster* 3

Bahan pangan yang termasuk kedalam *cluster* 3 dapat dilihat pada Tabel A.4. Bahan pangan yang termasuk kedalam cluster ini memiliki kandungan kalsium rendah, dan kandungan

vitamin B1 rendah. Jumlah anggota pada *cluster* 3 adalah sebanyak 72 bahan pangan.

Tabel A.4 Daftar Bahan Pangan pada *Cluster* 3

Cluster 3	
Beras Ketan Hitam	Tepung Terigu
Beras Paroiled	Ubi Jalar Merah
Jagung Giling Putih	Ubi Jalar Putih
Jagung Putih Pipil Baru	Kacang Tanah Rebus dengan Kulit
Jagung Putih Pipil Lama	Kelapa Tua, Daging
Jagung Muda Putih	Kluwak
Jagung Segar Putih	Koro Krupuk, Biji
Daging Anak Sapi	Lamtoro, Biji Tua
Bekasak	Nangka, Biji
Pindang Benggol	Pete Segar
Pindang Layang	Santan (Kelapa Diperas)
Beras Giling	Tempe Koro Benguk
Beras Ketan Putih	Tempe Lamtoro
Beras Setengah Giling	Tepung Hunkwe (Pati Kacang Hijau)
Bihun	Angsa
Biskuit	Bebek (Itik)
Jagung Kuning Pipil Baru	Lemak Babi (Becon)
Jagung Muda Kuning	Worst (Sosis Daging)
Jagung Segar Kuning	Kepiting
Jali	Kerupuk Ikan dengan Pati
Gaplek	Paling, Belut
Kentang Hitam	Petis Udang

Ketela Pohon Kuning	Tembang
Krupuk Aci	Daun Petai Cina
Makaroni	Daun Singkong Jenis Ambon
Maizena (Pati Jagung)	Jamur Kuping Kering
Mie Kering	Kacang Gude, Buah Muda
Pati Singkong (Tapioka)	Koro Krupuk, Buah
Roti Putih	Lemak Kerbau
Roti Warna Sawo Matang	Margarin
Tape Singkong	Minyak Hati Hiu (Eulamia)
Tepung Arrowroot (Tepung Garut)	Minyak Ikan
Tepung Beras	Minyak Kacang Tanah
Tepung Kentang	Minyak Kelapa
Tepung Gaplek	Minyak Kelapa Sawit
Tepung Sagu	Minyak Wijen

e. *Cluster* 4

Bahan pangan yang termasuk kedalam *cluster* 4 dapat dilihat pada Tabel A.5. Bahan pangan yang termasuk kedalam cluster ini memiliki kandungan karbohidrat rendah. Jumlah anggota pada *cluster* 4 adalah sebanyak 12 bahan pangan.

Tabel A.5 Daftar Bahan Pangan pada *Cluster* 4

Cluster 4	
Daging Babi Kurus	Telur Bebek (Telur Itik)
Kelapa Setengah Tua, Daging	Telur Bebek, Diasin
Daging Babi Gemuk	Pepetek
Ginjal Babi	Daun Kacang Panjang

Ginjal Domba Ham	Daun Kecipir Daun Koro

LAMPIRAN B
Source Code

1. Menu Home

```
package fuzzysom;
public class home extends javax.swing.JFrame {
    public home() {
        initComponents();
        setLocationRelativeTo(this);
    }
private void
jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        new incomplete().show();
        this.dispose();
    }

    private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        new complete().show();
        this.dispose();
    }

    private void
jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        new fuzzycomplete().show();
        this.dispose();
    }

    private void
jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        new fuzzyincomplete().show();
        this.dispose();
    }

    private void
jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        new som().show();
        this.dispose();
    }
```

2. *Input Complete Data*

```
package fuzzysom;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class complete extends javax.swing.JFrame {
    private Connection connection;
    private Statement statement;
    public complete() {
        initComponents();
        setLocationRelativeTo(this);
        try{
            Class.forName("com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection
("jdbc:mysql://localhost:3306/fuzzysom?zeroDateTimeBehavi
or=convertToNull", "root", "");
            statement = connection.createStatement();
        }
        catch(Exception e){
            JOptionPane.showMessageDialog(null, "Koneksi
gagal");
            NAMA.setEnabled(false);
            PROTEIN.setEnabled(false);
            LEMAK.setEnabled(false);
            KARBOHIDRAT.setEnabled(false);
            KALSIUM.setEnabled(false);
            FOSFOR.setEnabled(false);
            BESI.setEnabled(false);
            VITA.setEnabled(false);
            VITB1.setEnabled(false);
            VITC.setEnabled(false);
            AIR.setEnabled(false);
            INPUT.setEnabled(false);
            REFRESH.setEnabled(false);
        }
    }
public void RefreshData(){
        try{
            String Nama, Status;
            int No;
            Double Protein, Lemak, Karbohidrat, Kalsium,
Fosfor, Besi, VitaminA, VitaminB1, VitaminC, Air;
            Object []
rows={"No","Nama","Protein","Lemak","Karbohidrat","Kalsiu
m","Fosfor","Besi","VitaminA","VitaminB1",
                                "VitaminC","Air"};
            DefaultTableModel dtm=new
DefaultTableModel(null,rows);
            TABELPANGAN.setModel(dtm);
            TABELPANGAN.setBorder(null);
```

```
            try{
                String sql="select id_complete,
nama_complete, protein_complete, lemak_complete,
karbohidrat_complete, kalsium_complete, "
                        + "fosfor_complete,
besi_complete, a_complete, b1_complete, c_complete,
air_complete FROM  complete";
                Statement statement
=connection.createStatement();
                ResultSet rs
=statement.executeQuery(sql);
                while(rs.next()){
                    No=rs.getInt("id_complete");
                    Nama=rs.getString("nama_complete");

Protein=rs.getDouble("protein_complete");
                    Lemak=rs.getDouble("lemak_complete");

Karbohidrat=rs.getDouble("karbohidrat_complete");

Kalsium=rs.getDouble("kalsium_complete");

Fosfor=rs.getDouble("fosfor_complete");
                    Besi=rs.getDouble("besi_complete");
                    VitaminA=rs.getDouble("a_complete");

VitaminB1=rs.getDouble("b1_complete");
                    VitaminC=rs.getDouble("c_complete");
                    Air=rs.getDouble("air_complete");
                    String []
tampil={String.valueOf(No),Nama,
String.valueOf(Protein),String.valueOf(Lemak),

String.valueOf(Karbohidrat),String.valueOf(Kalsium),Strin
g.valueOf(Fosfor),

String.valueOf(Besi),String.valueOf(VitaminA),String.valu
eOf(VitaminB1),

String.valueOf(VitaminC),String.valueOf(Air)};
                    dtm.addRow(tampil);
                }
            }catch(SQLException e){
                e.printStackTrace();
                JOptionPane.showMessageDialog(null,"Query
Salah "+e);
            }
        }catch(Exception e){
            e.printStackTrace();
```

```
        }
    }
private void
INPUTActionPerformed(java.awt.event.ActionEvent evt) {
        if(
(NAMA.getText()).equals("")||(PROTEIN.getText()).equals("
")||(LEMAK.getText()).equals("")||

(KARBOHIDRAT.getText()).equals("")||(KALSIUM.getText()).e
quals("")||(FOSFOR.getText()).equals("")||

(BESI.getText()).equals("")||(VITA.getText()).equals("")|
|(VITB1.getText()).equals("")||

(VITC.getText()).equals("")||(AIR.getText()).equals("")){
            JOptionPane.showMessageDialog(null, "Data
yang diisikan belum lengkap");
        }
        else{
            try{
                String nama;
                PreparedStatement a =
connection.prepareStatement("INSERT INTO complete "
                        + "(nama_complete,
protein_complete, lemak_complete, karbohidrat_complete,
kalsium_complete, fosfor_complete, "
                        + "besi_complete, a_complete,
b1_complete, c_complete, air_complete) "
                        + "values
(?,?,?,?,?,?,?,?,?,?,?)");
                a.setString(1, NAMA.getText());
                a.setDouble(2,
Double.parseDouble(PROTEIN.getText()));
                a.setDouble(3,
Double.parseDouble(LEMAK.getText()));
                a.setDouble(4,
Double.parseDouble(KARBOHIDRAT.getText()));
                a.setDouble(5,
Double.parseDouble(KALSIUM.getText()));
                a.setDouble(6,
Double.parseDouble(FOSFOR.getText()));
                a.setDouble(7,
Double.parseDouble(BESI.getText()));
                a.setDouble(8,
Double.parseDouble(VITA.getText()));
                a.setDouble(9,
Double.parseDouble(VITB1.getText()));
                a.setDouble(10,
Double.parseDouble(VITC.getText()));
```

```
                a.setDouble(11,
Double.parseDouble(AIR.getText()));
                a.executeUpdate();
                statement = connection.createStatement();
                JOptionPane.showMessageDialog(null, "Data
telah dimasukkan");
                RefreshData();
                NAMA.setText("");
                PROTEIN.setText("");
                LEMAK.setText("");
                KARBOHIDRAT.setText("");
                KALSIUM.setText("");
                FOSFOR.setText("");
                BESI.setText("");
                VITA.setText("");
                VITB1.setText("");
                VITC.setText("");
                AIR.setText("");
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
            }
        }
    }
```

3. *Input Incomplete Data*

```
package fuzzysom;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class incomplete extends javax.swing.JFrame {
    private Connection connection;
    private Statement statement;
    public incomplete() {
        initComponents();
        setLocationRelativeTo(this);
        try{
            Class.forName("com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection
("jdbc:mysql://localhost:3306/fuzzysom?zeroDateTimeBehavi
or=convertToNull", "root", "");
            statement = connection.createStatement();
        }
```

```
        catch(Exception e){
            JOptionPane.showMessageDialog(null, "Koneksi
gagal");
            NAMA.setEnabled(false);
            PROTEIN.setEnabled(false);
            LEMAK.setEnabled(false);
            KARBOHIDRAT.setEnabled(false);
            KALSIUM.setEnabled(false);
            FOSFOR.setEnabled(false);
            BESI.setEnabled(false);
            VITA.setEnabled(false);
            VITB1.setEnabled(false);
            VITC.setEnabled(false);
            AIR.setEnabled(false);
            INPUT.setEnabled(false);
            REFRESH.setEnabled(false);
        }
    }
public void RefreshData(){
        try{
            String Nama, Status;
            int No;
            Double Protein, Lemak, Karbohidrat, Kalsium,
Fosfor, Besi, VitaminA, VitaminB1, VitaminC, Air;
            Object []
rows={"No","Nama","Protein","Lemak","Karbohidrat","Kalsiu
m","Fosfor","Besi","VitaminA","VitaminB1",
                              "VitaminC","Air"};
            DefaultTableModel dtm=new
DefaultTableModel(null,rows);
            TABELPANGAN.setModel(dtm);
            TABELPANGAN.setBorder(null);
            try{
                String sql="select id_incomplete,
nama_incomplete, protein_incomplete, lemak_incomplete,
karbohidrat_incomplete, "
                        + "kalsium_incomplete,
fosfor_incomplete, besi_incomplete, a_incomplete,
b1_incomplete, c_incomplete, "
                        + "air_incomplete FROM
incomplete";
                Statement statement
=connection.createStatement();
                ResultSet rs
=statement.executeQuery(sql);
                while(rs.next()){
                    No=rs.getInt("id_incomplete");
                    Nama=rs.getString("nama_incomplete");
```

```
Protein=rs.getDouble("protein_incomplete");

Lemak=rs.getDouble("lemak_incomplete");

Karbohidrat=rs.getDouble("karbohidrat_incomplete");

Kalsium=rs.getDouble("kalsium_incomplete");

Fosfor=rs.getDouble("fosfor_incomplete");
                    Besi=rs.getDouble("besi_incomplete");

VitaminA=rs.getDouble("a_incomplete");

VitaminB1=rs.getDouble("b1_incomplete");

VitaminC=rs.getDouble("c_incomplete");
                    Air=rs.getDouble("air_incomplete");
                    String []
tampil={String.valueOf(No),Nama,
String.valueOf(Protein),String.valueOf(Lemak),

String.valueOf(Karbohidrat),String.valueOf(Kalsium),Strin
g.valueOf(Fosfor),

String.valueOf(Besi),String.valueOf(VitaminA),String.valu
eOf(VitaminB1),

String.valueOf(VitaminC),String.valueOf(Air)};
                    dtm.addRow(tampil);
                }
            }catch(SQLException e){
                e.printStackTrace();
                JOptionPane.showMessageDialog(null,"Query
Salah "+e);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
private void
INPUTActionPerformed(java.awt.event.ActionEvent evt) {
        String protein, lemak, karbohidrat,kalsium,
fosfor, besi, vita, vitb1, vitc, air;
        protein=PROTEIN.getText();
        lemak=LEMAK.getText();
        karbohidrat=KARBOHIDRAT.getText();
        kalsium=KALSIUM.getText();
        fosfor=FOSFOR.getText();
```

```
        besi=BESI.getText();
        vita=VITA.getText();
        vitb1=VITB1.getText();
        vitc=VITC.getText();
        air=AIR.getText();
        if((NAMA.getText()).equals("")){
            JOptionPane.showMessageDialog(null, "Nama
Bahan Pangan Belum Diinputkan");
        }
        else{
            if((PROTEIN.getText()).equals("")){
                protein="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Protein Hilang");
            }
            else if((LEMAK.getText()).equals("")){
                lemak="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Lemak Hilang");
            }
            else if((KARBOHIDRAT.getText()).equals("")){
                karbohidrat="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Karbohidrat Hilang");
            }
            else if((KALSIUM.getText()).equals("")){
                kalsium="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Kalsium Hilang");
            }
            else if((FOSFOR.getText()).equals("")){
                fosfor="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Fosfor Hilang");
            }
            else if((BESI.getText()).equals("")){
                besi="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Besi Hilang");
            }
            else if((VITA.getText()).equals("")){
                vita="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Vitamin A Hilang");
            }
            else if((VITB1.getText()).equals("")){
                vitb1="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Vitamin B1 Hilang");
```

```
            }
            else if((VITC.getText()).equals("")){
                vitc="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Vitamin C Hilang");
            }
            else if((AIR.getText()).equals("")){
                air="999999999";
                JOptionPane.showMessageDialog(null,
"Nilai Kandungan Air Hilang");
            }
            try{
                String nama;
                PreparedStatement a =
connection.prepareStatement("INSERT INTO incomplete "
                    + "(nama_incomplete,
protein_incomplete, lemak_incomplete,
karbohidrat_incomplete, kalsium_incomplete, "
                    + "fosfor_incomplete,
besi_incomplete, a_incomplete, b1_incomplete,
c_incomplete, air_incomplete) "
                    + "values (?,?,?,?,?,?,?,?,?,?,?)");
                a.setString(1, NAMA.getText());
                a.setDouble(2,
Double.parseDouble(protein));
                a.setDouble(3,
Double.parseDouble(lemak));
                a.setDouble(4,
Double.parseDouble(karbohidrat));
                a.setDouble(5,
Double.parseDouble(kalsium));
                a.setDouble(6,
Double.parseDouble(fosfor));
                a.setDouble(7, Double.parseDouble(besi));
                a.setDouble(8, Double.parseDouble(vita));
                a.setDouble(9,
Double.parseDouble(vitb1));
                a.setDouble(10,
Double.parseDouble(vitc));
                a.setDouble(11, Double.parseDouble(air));
                a.executeUpdate();
                statement = connection.createStatement();
                JOptionPane.showMessageDialog(null, "Data
telah dimasukkan");
                RefreshData();
                NAMA.setText("");
                PROTEIN.setText("");
                LEMAK.setText("");
                KARBOHIDRAT.setText("");
```

```
                KALSIUM.setText("");
                FOSFOR.setText("");
                BESI.setText("");
                VITA.setText("");
                VITB1.setText("");
                VITC.setText("");
                AIR.setText("");
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
            }
        }
    }
```

4. Fuzzifikasi *Complete Data*

```
package fuzzysom;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class fuzzycomplete extends javax.swing.JFrame {
    private Connection connection;
    private Statement statement;
    public int[] ID;
    public String[] NAMA;
    public Double[] PROTEIN;
    public Double[] LEMAK;
    public Double[] KARBOHIDRAT;
    public Double[] KALSIUM;
    public Double[] FOSFOR;
    public Double[] BESI;
    public Double[] VITA;
    public Double[] VITB1;
    public Double[] VITC;
    public Double[] AIR;
    public Double[][] DerPROTEIN;
    public Double[][] DerLEMAK;
    public Double[][] DerKARBOHIDRAT;
    public Double[][] DerKALSIUM;
    public Double[][] DerFOSFOR;
    public Double[][] DerBESI;
    public Double[][] DerVITA;
    public Double[][] DerVITB1;
    public Double[][] DerVITC;
```

```
    public Double[][] DerAIR;
    public int[] GOLPROTEIN;
    public int[] GOLLEMAK;
    public int[] GOLKARBOHIDRAT;
    public int[] GOLKALSIUM;
    public int[] GOLFOSFOR;
    public int[] GOLBESI;
    public int[] GOLVITA;
    public int[] GOLVITB1;
    public int[] GOLVITC;
    public int[] GOLAIR;
    public fuzzycomplete() {
        initComponents();
        setLocationRelativeTo(this);
        try{
            Class.forName("com.mysql.jdbc.Driver");
            connection                                    =
DriverManager.getConnection("jdbc:mysql://localhost:3306/
fuzzysom?zeroDateTimeBehavior=convertToNull",     "root",
"");
            statement = connection.createStatement();
        }catch(Exception e){
            JOptionPane.showMessageDialog(null,   "Koneksi
gagal");
        }
    }
public void Get(){
        String nam = "";
        Double pro, lem, kar, kal, fos, bes, vita, vitb1,
vitc, air;
        String cntj1 = "";
        try{
            String sql = "SELECT count(nama_complete) FROM
complete";
            Statement             statement             =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            while(rs.next()){
                cntj1                                     =
rs.getString("count(nama_complete)");
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,    "Query
Salah " + e);
        }
        int index = Integer.parseInt(cntj1);
        ID = new int[index];
        NAMA = new String[index];
```

```
        PROTEIN = new Double[index];
        LEMAK = new Double[index];
        KARBOHIDRAT = new Double[index];
        KALSIUM = new Double[index];
        FOSFOR = new Double[index];
        BESI = new Double[index];
        VITA = new Double[index];
        VITB1 = new Double[index];
        VITC = new Double[index];
        AIR = new Double[index];
        try{
            String    sql    =    "SELECT    nama_complete,
protein_complete, lemak_complete, karbohidrat_complete, "
                    + "kalsium_complete, fosfor_complete,
besi_complete, a_complete, b1_complete, c_complete, "
                    + "air_complete FROM complete";
            Statement             statement             =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            int a = 0;
            while(rs.next()){
                nam = rs.getString("nama_complete");
                pro = rs.getDouble("protein_complete");
                lem = rs.getDouble("lemak_complete");
                kar                                       =
rs.getDouble("karbohidrat_complete");
                kal = rs.getDouble("kalsium_complete");
                fos = rs.getDouble("fosfor_complete");
                bes = rs.getDouble("besi_complete");
                vita = rs.getDouble("a_complete");
                vitb1 = rs.getDouble("b1_complete");
                vitc = rs.getDouble("c_complete");
                air = rs.getDouble("air_complete");
                ID[a] = a;
                NAMA[a] = nam;
                PROTEIN[a] = pro;
                LEMAK[a] = lem;
                KARBOHIDRAT[a] = kar;
                KALSIUM[a] = kal;
                FOSFOR[a] = fos;
                BESI[a] = bes;
                VITA[a] = vita;
                VITB1[a] = vitb1;
                VITC[a] = vitc;
                AIR[a] = air;
                a++;
            }
```

```
System.out.println("================================ NAMA
BAHAN PANGAN ================================");
            for (int i = 0; i < index; i++) {
                System.out.println(NAMA[i]);
            }
            System.out.println();

System.out.println("================================
PROTEIN ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("PROTEIN[" + i + "] =>
Protein " + NAMA[i] + " : " + PROTEIN[i]);
            }
            System.out.println();

System.out.println("================================
LEMAK ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("LEMAK[" + i + "] =>
Lemak " + NAMA[i] + " : " + LEMAK[i]);
            }
            System.out.println();

System.out.println("================================
KARBOHIDRAT ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("KARBOHIDRAT[" + i + "]
=> Karbohidrat " + NAMA[i] + " : " + KARBOHIDRAT[i]);
            }
            System.out.println();

System.out.println("================================
KALSIUM ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("KALSIUM[" + i + "] =>
Kalsium " + NAMA[i] + " : " + KALSIUM[i]);
            }
            System.out.println();

System.out.println("================================
FOSFOR ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("FOSFOR[" + i + "] =>
Fosfor " + NAMA[i] + " : " + FOSFOR[i]);
            }
            System.out.println();
```

```
System.out.println("================================ BESI
================================");
            for (int i = 0; i < index; i++) {
                System.out.println("BESI[" + i + "] =>
Besi " + NAMA[i] + " : " + BESI[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN A ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINA[" + i + "] =>
Vitamin A " + NAMA[i] + " : " + VITA[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN B1 ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINB1[" + i + "]
=> Vitamin B1 " + NAMA[i] + " : " + VITB1[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN C ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINC[" + i + "] =>
Vitamin C " + NAMA[i] + " : " + VITC[i]);
            }
            System.out.println();

System.out.println("================================  AIR
================================");
            for (int i = 0; i < index; i++) {
                System.out.println("AIR[" + i + "] => Air
" + NAMA[i] + " : " + AIR[i]);
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,    "Query
Salah " + e);
        }
    }
public void Fuzzifikasi(){
        GOLPROTEIN=new int[NAMA.length];
        GOLLEMAK=new int[NAMA.length];
        GOLKARBOHIDRAT=new int[NAMA.length];
```

```
        GOLKALSIUM=new int[NAMA.length];
        GOLFOSFOR=new int[NAMA.length];
        GOLBESI=new int[NAMA.length];
        GOLVITA=new int[NAMA.length];
        GOLVITB1=new int[NAMA.length];
        GOLVITC=new int[NAMA.length];
        GOLAIR=new int[NAMA.length];
        DerPROTEIN=new Double[3][NAMA.length];
        DerLEMAK=new Double[3][NAMA.length];
        DerKARBOHIDRAT=new Double[3][NAMA.length];
        DerKALSIUM=new Double[3][NAMA.length];
        DerFOSFOR=new Double[3][NAMA.length];
        DerBESI=new Double[3][NAMA.length];
        DerVITA=new Double[3][NAMA.length];
        DerVITB1=new Double[3][NAMA.length];
        DerVITC=new Double[3][NAMA.length];
        DerAIR=new Double[3][NAMA.length];
        //mulai fuzzyfikasi untuk tiap bahan pangan
        for (int i = 0; i < NAMA.length; i++) {
            System.out.println("-------------------------
--- " + NAMA[i] + " ---------------------------- ");
            //=======================================
Protein ========================================
            int st=0;
            Double min = 2.0;
            //rendah
            if (PROTEIN[i] <= 6.8){
                DerPROTEIN[0][i] = 1.0;
            }else if (PROTEIN[i] < 13 && PROTEIN[i] > 6.8){
                DerPROTEIN[0][i] = (13 - PROTEIN[i]) /
6.2;
            }else if (PROTEIN[i] >= 13){
                DerPROTEIN[0][i] = 0.0;}
            System.out.println("Protein  rendah  :  "  +
DerPROTEIN[0][i]);
            //sedang
            if (PROTEIN[i] <= 6.8) {
                DerPROTEIN[1][i] = 0.0;
            } else if (PROTEIN[i] <= 13 && PROTEIN[i] >
6.8) {
                DerPROTEIN[1][i] = (PROTEIN[i] - 6.8) /
6.2;
            } else if (PROTEIN[i] < 17 && PROTEIN[i] > 13)
{
                DerPROTEIN[1][i] = (17 - PROTEIN[i]) / 4;
            } else if (PROTEIN[i] >= 17) {
                DerPROTEIN[1][i] = 0.0;}
            System.out.println("Protein  sedang  :  "  +
DerPROTEIN[1][i]);
```

```
            //tinggi
            if (PROTEIN[i] <= 13){
                DerPROTEIN[2][i] = 0.0;
            }else if (PROTEIN[i] < 17 && PROTEIN[i] > 13){
                DerPROTEIN[2][i] = (PROTEIN[i] - 13) / 4;
            }else if (PROTEIN[i] >= 17){
                DerPROTEIN[2][i] = 1.0;}
            System.out.println("Protein  tinggi  :  "  +
DerPROTEIN[2][i]);
            //rumus gabungan(nilai min)
            if(DerPROTEIN[0][i] != 0.0){
                if(DerPROTEIN[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerPROTEIN[j][i] < min){
                            min=DerPROTEIN[j][i];
                            st=j;}}}
                else if(DerPROTEIN[0][i] == 1.0){
                    min=0.0;
                    st=0;}
             }
            else if(DerPROTEIN[2][i] != 0.0){
                if(DerPROTEIN[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerPROTEIN[j][i] < min){
                            min=DerPROTEIN[j][i];
                            st=j;}}}
                else if(DerPROTEIN[2][i] == 1.0){
                    min=1.0;
                    st=2;}
            }
            GOLPROTEIN[i]=st;
            PROTEIN[i] = min;
            System.out.println("Nilai  protein  =  "  +
PROTEIN[i]+" merupakan golongan "+GOLPROTEIN[i]);
            System.out.println("-------------------------
---");

            //=======================================
Lemak ========================================
            int st1=0;
            Double min1 = 2.0;
            //rendah
            if(LEMAK[i]<=5){
                DerLEMAK[0][i]=1.0;
            }else if(LEMAK[i]<16 && LEMAK[i]>5){
                DerLEMAK[0][i]=(16-LEMAK[i])/11;
            }else if(LEMAK[i]>=16){
                DerLEMAK[0][i]=0.0;}
```

```
            System.out.println("Lemak       rendah       :
"+DerLEMAK[0][i]);
            //sedang
            if(LEMAK[i]<=5){
                DerLEMAK[1][i]=0.0;
            }else if(LEMAK[i]<=16 && LEMAK[i]>5){
                DerLEMAK[1][i]=(LEMAK[i]-5)/11;
            }else if(LEMAK[i]<30 && LEMAK[i]>16){
                DerLEMAK[1][i]=(30-LEMAK[i])/14;
            }else if(LEMAK[i]>=30){
                DerLEMAK[1][i]=0.0;}
            System.out.println("Lemak       sedang       :
"+DerLEMAK[1][i]);
            //tinggi
            if(LEMAK[i]<=16){
                DerLEMAK[2][i]=0.0;
            }else if(LEMAK[i]<30 && LEMAK[i]>16){
                DerLEMAK[2][i]=(LEMAK[i]-16)/14;
            }else if(LEMAK[i]>=30){
                DerLEMAK[2][i]=1.0;}
            System.out.println("Lemak       tinggi       :
"+DerLEMAK[2][i]);
            //rumus gabungan(nilai min)
            if(DerLEMAK[0][i] != 0.0){
                if(DerLEMAK[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerLEMAK[j][i] < min1){
                            min1=DerLEMAK[j][i];
                            st1=j;}}}
                else if(DerLEMAK[0][i] == 1.0){
                    min1=0.0;
                    st1=0;}
             }
            else if(DerLEMAK[2][i] != 0.0){
                if(DerLEMAK[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerLEMAK[j][i] < min1){
                            min1=DerLEMAK[j][i];
                            st1=j;}}}
                else if(DerLEMAK[2][i] == 1.0){
                    min1=1.0;
                    st1=2;}
            }
            GOLLEMAK[i]=st1;
            LEMAK[i]=min1;
            System.out.println("Nilai lemak = "+LEMAK[i]+"
merupakan golongan "+GOLLEMAK[i]);
            System.out.println("-------------------------
---");
```

```
            //=========================================
Karbohidrat ==========================================
            int st2=0;
            Double min2 = 2.0;
            //rendah
            if(KARBOHIDRAT[i]<=12){
                DerKARBOHIDRAT[0][i]=1.0;
            }else          if(KARBOHIDRAT[i]<30          &&
KARBOHIDRAT[i]>12){
                DerKARBOHIDRAT[0][i]=(30-
KARBOHIDRAT[i])/18;
            }else if(KARBOHIDRAT[i]>=30){
                DerKARBOHIDRAT[0][i]=0.0;}
            System.out.println("Karbohidrat    rendah    :
"+DerKARBOHIDRAT[0][i]);
            //sedang
            if(KARBOHIDRAT[i]<=12){
                DerKARBOHIDRAT[1][i]=0.0;
            }else         if(KARBOHIDRAT[i]<=30         &&
KARBOHIDRAT[i]>12){
                DerKARBOHIDRAT[1][i]=(KARBOHIDRAT[i]-
12)/18;
            }else          if(KARBOHIDRAT[i]<70          &&
KARBOHIDRAT[i]>30){
                DerKARBOHIDRAT[1][i]=(70-
KARBOHIDRAT[i])/40;
            }else if(KARBOHIDRAT[i]>=70){
                DerKARBOHIDRAT[1][i]=0.0;}
            System.out.println("Karbohidrat    sedang    :
"+DerKARBOHIDRAT[1][i]);
            //tinggi
            if(KARBOHIDRAT[i]<=30){
                DerKARBOHIDRAT[2][i]=0.0;
            }else          if(KARBOHIDRAT[i]<70          &&
KARBOHIDRAT[i]>30){
                DerKARBOHIDRAT[2][i]=(KARBOHIDRAT[i]-
30)/40;
            }else if(KARBOHIDRAT[i]>=30){
                DerKARBOHIDRAT[2][i]=1.0;}
            System.out.println("Karbohidrat    tinggi    :
"+DerKARBOHIDRAT[2][i]);
            //rumus gabungan(nilai min)
            if(DerKARBOHIDRAT[0][i] != 0.0){
                if(DerKARBOHIDRAT[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerKARBOHIDRAT[j][i] < min2){
                            min2=DerKARBOHIDRAT[j][i];
                            st2=j;}}}
                else if(DerKARBOHIDRAT[0][i] == 1.0){
```

```
                    min2=0.0;
                    st2=0;}
             }
            else if(DerKARBOHIDRAT[2][i] != 0.0){
                if(DerKARBOHIDRAT[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerKARBOHIDRAT[j][i] < min2){
                            min2=DerKARBOHIDRAT[j][i];
                            st2=j;}}}
                else if(DerKARBOHIDRAT[2][i] == 1.0){
                    min2=1.0;
                    st2=2;}
            }
            GOLKARBOHIDRAT[i]=st2;
            KARBOHIDRAT[i]=min2;
            System.out.println("Nilai    karbohidrat    =
"+KARBOHIDRAT[i]+"           merupakan           golongan
"+GOLKARBOHIDRAT[i]);
            System.out.println("-------------------------
---");

            //=======================================
Kalsium ========================================
            int st3=0;
            Double min3 = 2.0;
            //rendah
            if(KALSIUM[i]<=50){
                DerKALSIUM[0][i]=1.0;
            }else if(KALSIUM[i]<120 && KALSIUM[i]>50){
                DerKALSIUM[0][i]=(120-KALSIUM[i])/70;
            }else if(KALSIUM[i]>=120){
                DerKALSIUM[0][i]=0.0;}
            System.out.println("Kalsium      rendah      :
"+DerKALSIUM[0][i]);
            //sedang
            if(KALSIUM[i]<=50){
                DerKALSIUM[1][i]=0.0;
            }else if(KALSIUM[i]<=120 && KALSIUM[i]>50){
                DerKALSIUM[1][i]=(KALSIUM[i]-50)/70;
            }else if(KALSIUM[i]<354 && KALSIUM[i]>120){
                DerKALSIUM[1][i]=(354-KALSIUM[i])/234;
            }else if(KALSIUM[i]>=354){
                DerKALSIUM[1][i]=0.0;}
            System.out.println("Kalsium      sedang      :
"+DerKALSIUM[1][i]);
            //tinggi
            if(KALSIUM[i]<=120){
                DerKALSIUM[2][i]=0.0;
            }else if(KALSIUM[i]<354 && KALSIUM[i]>120){
```

```
                DerKALSIUM[2][i]=(KALSIUM[i]-120)/234;
            }else if(KALSIUM[i]>=354){
                DerKALSIUM[2][i]=1.0;}
            System.out.println("Kalsium      tinggi      :
"+DerKALSIUM[2][i]);
            //rumus gabungan(nilai min)
            if(DerKALSIUM[0][i] != 0.0){
                if(DerKALSIUM[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerKALSIUM[j][i] < min3){
                            min3=DerKALSIUM[j][i];
                            st3=j;}}}
                else if(DerKALSIUM[0][i] == 1.0){
                    min3=0.0;
                    st3=0;}
             }
            else if(DerKALSIUM[2][i] != 0.0){
                if(DerKALSIUM[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerKALSIUM[j][i] < min3){
                            min3=DerKALSIUM[j][i];
                            st3=j;}}}
                else if(DerKALSIUM[2][i] == 1.0){
                    min3=1.0;
                    st3=2;}
            }
            GOLKALSIUM[i]=st3;
            KALSIUM[i]=min3;
            System.out.println("Nilai      kalsium      =
"+KALSIUM[i]+" merupakan golongan "+GOLKALSIUM[i]);
            System.out.println("-------------------------
---");

            //=========================================
Fosfor ==========================================
            int st4=0;
            Double min4 = 2.0;
            //rendah
            if(FOSFOR[i]<=110){
                DerFOSFOR[0][i]=1.0;
            }else if(FOSFOR[i]<170 && FOSFOR[i]>110){
                DerFOSFOR[0][i]=(170-FOSFOR[i])/60;
            }else if(FOSFOR[i]>=170){
                DerFOSFOR[0][i]=0.0;}
            System.out.println("Fosfor      rendah      :
"+DerFOSFOR[0][i]);
            //sedang
            if(FOSFOR[i]<=110){
                DerFOSFOR[1][i]=0.0;
```

```
            }else if(FOSFOR[i]<=170 && FOSFOR[i]>110){
                DerFOSFOR[1][i]=(FOSFOR[i]-110)/60;
            }else if(FOSFOR[i]<256 && FOSFOR[i]>170){
                DerFOSFOR[1][i]=(256-FOSFOR[i])/86;
            }else if(FOSFOR[i]>=256){
                DerFOSFOR[1][i]=0.0;}
            System.out.println("Fosfor      sedang      :
"+DerFOSFOR[1][i]);
            //tinggi
            if(FOSFOR[i]<=170){
                DerFOSFOR[2][i]=0.0;
            }else if(FOSFOR[i]<256 && FOSFOR[i]>170){
                DerFOSFOR[2][i]=(FOSFOR[i]-170)/86;
            }else if(FOSFOR[i]>=256){
                DerFOSFOR[2][i]=1.0;}
            System.out.println("Fosfor      tinggi      :
"+DerFOSFOR[2][i]);
            //rumus gabungan(nilai min)
            if(DerFOSFOR[0][i] != 0.0){
                if(DerFOSFOR[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerFOSFOR[j][i] < min4){
                            min4=DerFOSFOR[j][i];
                            st4=j;}}}
                else if(DerFOSFOR[0][i] == 1.0){
                    min4=0.0;
                    st4=0;}
             }
            else if(DerFOSFOR[2][i] != 0.0){
                if(DerFOSFOR[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerFOSFOR[j][i] < min4){
                            min4=DerFOSFOR[j][i];
                            st4=j;}}}
                else if(DerFOSFOR[2][i] == 1.0){
                    min4=1.0;
                    st4=2;}
            }
            GOLFOSFOR[i]=st4;
            FOSFOR[i]=min4;
            System.out.println("Nilai       fosfor       =
"+FOSFOR[i]+" merupakan golongan "+GOLFOSFOR[i]);
            System.out.println("-------------------------
---");

            //=========================================
Besi ========================================
            int st5=0;
            Double min5 = 2.0;
```

```
            //rendah
            if(BESI[i]<=2){
                DerBESI[0][i]=1.0;
            }else if(BESI[i]<5.7 && BESI[i]>2){
                DerBESI[0][i]=(5.7-BESI[i])/3.7;
            }else if(BESI[i]>=5.7){
                DerBESI[0][i]=0.0;}
            System.out.println("Besi        rendah        :
"+DerBESI[0][i]);
            //sedang
            if(BESI[i]<=2){
                DerBESI[1][i]=0.0;
            }else if(BESI[i]<=5.7 && BESI[i]>2){
                DerBESI[1][i]=(BESI[i]-2)/3.7;
            }else if(BESI[i]<20 && BESI[i]>5.7){
                DerBESI[1][i]=(20-BESI[i])/14.3;
            }else if(BESI[i]>=20){
                DerBESI[1][i]=0.0;}
            System.out.println("Besi        sedang        :
"+DerBESI[1][i]);
            //tinggi
            if(BESI[i]<=5.7){
                DerBESI[2][i]=0.0;
            }else if(BESI[i]<20 && BESI[i]>5.7){
                DerBESI[2][i]=(BESI[i]-5.7)/14.3;
            }else if(BESI[i]>=20){
                DerBESI[2][i]=1.0;}
            System.out.println("Besi        tinggi        :
"+DerBESI[2][i]);
            //rumus gabungan(nilai min)
            if(DerBESI[0][i] != 0.0){
                if(DerBESI[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerBESI[j][i] < min5){
                            min5=DerBESI[j][i];
                            st5=j;}}}
                else if(DerBESI[0][i] == 1.0){
                    min5=0.0;
                    st5=0;}
             }
            else if(DerBESI[2][i] != 0.0){
                if(DerBESI[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerBESI[j][i] < min5){
                            min5=DerBESI[j][i];
                            st5=j;}}}
                else if(DerBESI[2][i] == 1.0){
                    min5=1.0;
                    st5=2;}
```

```
            }
            GOLBESI[i]=st5;
            BESI[i]=min5;
            System.out.println("Nilai besi = "+BESI[i]+"
merupakan golongan "+GOLBESI[i]);
            System.out.println("-------------------------
---");

            //=========================================
Vitamin A ========================================
            int st6=0;
            Double min6 = 2.0;
            //rendah
            if(VITA[i]<=1000){
                DerVITA[0][i]=1.0;
            }else if(VITA[i]<3900 && VITA[i]>1000){
                DerVITA[0][i]=(3900-VITA[i])/2900;
            }else if(VITA[i]>=3900){
                DerVITA[0][i]=0.0;}
            System.out.println("Vitamin   A   rendah   :
"+DerVITA[0][i]);
            //sedang
            if(VITA[i]<=1000){
                DerVITA[1][i]=0.0;
            }else if(VITA[i]<=3900 && VITA[i]>1000){
                DerVITA[1][i]=(VITA[i]-1000)/2900;
            }else if(VITA[i]<15000 && VITA[i]>3900){
                DerVITA[1][i]=(15000-VITA[i])/11100;
            }else if(VITA[i]>=15000){
                DerVITA[1][i]=0.0;}
            System.out.println("Vitamin   A   sedang   :
"+DerVITA[1][i]);
            //tinggi
            if(VITA[i]<=3900){
                DerVITA[2][i]=0.0;
            }else if(VITA[i]<15000 && VITA[i]>3900){
                DerVITA[2][i]=(VITA[i]-3900)/11100;
            }else if(VITA[i]>=15000){
                DerVITA[2][i]=1.0;}
            System.out.println("Vitamin   A   tinggi   :
"+DerVITA[2][i]);
            //rumus gabungan(nilai min)
            if(DerVITA[0][i] != 0.0){
                if(DerVITA[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerVITA[j][i] < min6){
                            min6=DerVITA[j][i];
                            st6=j;}}}
                else if(DerVITA[0][i] == 1.0){
```

```
                    min6=0.0;
                    st6=0;}
                }
            else if(DerVITA[2][i] != 0.0){
                if(DerVITA[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerVITA[j][i] < min6){
                            min6=DerVITA[j][i];
                            st6=j;}}}
                else if(DerVITA[2][i] == 1.0){
                    min6=1.0;
                    st6=2;}
            }
            GOLVITA[i]=st6;
            VITA[i]=min6;
            System.out.println("Nilai    Vitamin    A    =
"+VITA[i]+" merupakan golongan "+GOLVITA[i]);
            System.out.println("-------------------------
---");

            //=======================================
Vitamin B1 =======================================
            int st7=0;
            Double min7 = 2.0;
            //rendah
            if(VITB1[i]<=0.1){
                DerVITB1[0][i]=1.0;
            }else if(VITB1[i]<0.28 && VITB1[i]>0.1){
                DerVITB1[0][i]=(0.28-VITB1[i])/0.18;
            }else if(VITB1[i]>=0.28){
                DerVITB1[0][i]=0.0;}
            System.out.println("Vitamin   B1   rendah   :
"+DerVITB1[0][i]);
            //sedang
            if(VITB1[i]<=0.1){
                DerVITB1[1][i]=0.0;
            }else if(VITB1[i]<=0.28 && VITB1[i]>0.1){
                DerVITB1[1][i]=(VITB1[i]-0.1)/0.18;
            }else if(VITB1[i]<0.45 && VITB1[i]>0.28){
                DerVITB1[1][i]=(0.45-VITB1[i])/0.17;
            }else if(VITB1[i]>=0.45){
                DerVITB1[1][i]=0.0;}
            System.out.println("Vitamin   B1   sedang   :
"+DerVITB1[1][i]);
            //tinggi
            if(VITB1[i]<=0.28){
                DerVITB1[2][i]=0.0;
            }else if(VITB1[i]<0.45 && VITB1[i]>0.28){
                DerVITB1[2][i]=(VITB1[i]-0.28)/0.17;
```

```
            }else if(VITB1[i]>=0.45){
                DerVITB1[2][i]=1.0;}
            System.out.println("Vitamin   B1   tinggi   :
"+DerVITB1[2][i]);
            //rumus gabungan(nilai min)
            if(DerVITB1[0][i] != 0.0){
                if(DerVITB1[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerVITB1[j][i] < min7){
                            min7=DerVITB1[j][i];
                            st7=j;}}}
                else if(DerVITB1[0][i] == 1.0){
                    min7=0.0;
                    st7=0;}
             }
            else if(DerVITB1[2][i] != 0.0){
                if(DerVITB1[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerVITB1[j][i] < min7){
                            min7=DerVITB1[j][i];
                            st7=j;}}}
                else if(DerVITB1[2][i] == 1.0){
                    min7=1.0;
                    st7=2;}
            }
            GOLVITB1[i]=st7;
            VITB1[i]=min7;
            System.out.println("Nilai   Vitamin   B1   =
"+VITB1[i]+" merupakan golongan "+GOLVITB1[i]);
            System.out.println("-------------------------
---");

            //========================================
Vitamin C ========================================
            int st8=0;
            Double min8 = 2.0;
            //rendah
            if(VITC[i]<=10){
                DerVITC[0][i]=1.0;
            }else if(VITC[i]<40 && VITC[i]>10){
                DerVITC[0][i]=(40-VITC[i])/30;
            }else if(VITC[i]>=40){
                DerVITC[0][i]=0.0;}
            System.out.println("Vitamin   C   rendah   :
"+DerVITC[0][i]);
            //sedang
            if(VITC[i]<=10){
                DerVITC[1][i]=0.0;
            }else if(VITC[i]<=40 && VITC[i]>10){
```

```
                DerVITC[1][i]=(VITC[i]-10)/30;
            }else if(VITC[i]<80 && VITC[i]>40){
                DerVITC[1][i]=(80-VITC[i])/40;
            }else if(VITC[i]>=80){
                DerVITC[1][i]=0.0;}
            System.out.println("Vitamin    C    sedang    :
"+DerVITC[1][i]);
            //tinggi
            if(VITC[i]<=40){
                DerVITC[2][i]=0.0;
            }else if(VITC[i]<80 && VITC[i]>40){
                DerVITC[2][i]=(VITC[i]-40)/40;
            }else if(VITC[i]>=80){
                DerVITC[2][i]=1.0;}
            System.out.println("Vitamin    C    tinggi    :
"+DerVITC[2][i]);
            //rumus gabungan(nilai min)
            if(DerVITC[0][i] != 0.0){
                if(DerVITC[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerVITC[j][i] < min8){
                            min8=DerVITC[j][i];
                            st8=j;}}}
                else if(DerVITC[0][i] == 1.0){
                    min8=0.0;
                    st8=0;}
             }
            else if(DerVITC[2][i] != 0.0){
                if(DerVITC[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerVITC[j][i] < min8){
                            min8=DerVITC[j][i];
                            st8=j;}}}
                else if(DerVITC[2][i] == 1.0){
                    min8=1.0;
                    st8=2;}
            }
            GOLVITC[i]=st8;
            VITC[i]=min8;
            System.out.println("Nilai    Vitamin    C    =
"+VITC[i]+" merupakan golongan "+GOLVITC[i]);
            System.out.println("-------------------------
---");

            //=========================================
Air ========================================
            int st9=0;
            Double min9 = 2.0;
            //rendah
```

```
            if(AIR[i]<=25){
                DerAIR[0][i]=1.0;
            }else if(AIR[i]<55 && AIR[i]>25){
                DerAIR[0][i]=(55-AIR[i])/30;
            }else if(AIR[i]>=55){
                DerAIR[0][i]=0.0;}
            System.out.println("Air        rendah        :
"+DerAIR[0][i]);
            //sedang
            if(AIR[i]<=25){
                DerAIR[1][i]=0.0;
            }else if(AIR[i]<=55 && AIR[i]>25){
                DerAIR[1][i]=(AIR[i]-25)/30;
            }else if(AIR[i]<70 && AIR[i]>55){
                DerAIR[1][i]=(70-AIR[i])/15;
            }else if(AIR[i]>=70){
                DerAIR[1][i]=0.0;}
            System.out.println("Air        sedang        :
"+DerAIR[1][i]);
            //tinggi
            if(AIR[i]<=55){
                DerAIR[2][i]=0.0;
            }else if(AIR[i]<70 && AIR[i]>55){
                DerAIR[2][i]=(AIR[i]-55)/15;
            }else if(AIR[i]>=70){
                DerAIR[2][i]=1.0;}
            System.out.println("Air        tinggi        :
"+DerAIR[2][i]);
            //rumus gabungan(nilai min)
            if(DerAIR[0][i] != 0.0){
                if(DerAIR[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerAIR[j][i] < min9){
                            min9=DerAIR[j][i];
                            st9=j;}}}
                else if(DerAIR[0][i] == 1.0){
                    min9=0.0;
                    st9=0;}
             }
            else if(DerAIR[2][i] != 0.0){
                if(DerAIR[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerAIR[j][i] < min9){
                            min9=DerAIR[j][i];
                            st9=j;}}}
                else if(DerAIR[2][i] == 1.0){
                    min9=1.0;
                    st9=2;}
            }
```

```
            GOLAIR[i]=st9;
            AIR[i]=min9;
            System.out.println("Nilai  Air  =  "+AIR[i]+"
merupakan golongan "+GOLAIR[i]);
        }
        for(int i=0;i<NAMA.length;i++){
            try{
                Class.forName("com.mysql.jdbc.Driver");
                PreparedStatement           b           =
connection.prepareStatement("INSERT INTO fuzzycomplete "

+   "(id_fcomplete,   nama_fcomplete,   protein_fcomplete,
lemak_fcomplete,"

+     "     karbohidrat_fcomplete,     kalsium_fcomplete,
fosfor_fcomplete,"

+ "besi_fcomplete, a_fcomplete, b1_fcomplete, c_fcomplete,
air_fcomplete) "

+ "values (?,?,?,?,?,?,?,?,?,?,?,?)");
                b.setInt(1, ID[i]);
                b.setString(2, NAMA[i]);
                b.setDouble(3, PROTEIN[i]*100);
                b.setDouble(4, LEMAK[i]*100);
                b.setDouble(5, KARBOHIDRAT[i]*100);
                b.setDouble(6, KALSIUM[i]*100);
                b.setDouble(7, FOSFOR[i]*100);
                b.setDouble(8, BESI[i]*100);
                b.setDouble(9, VITA[i]*100);
                b.setDouble(10, VITB1[i]*100);
                b.setDouble(11, VITC[i]*100);
                b.setDouble(12, AIR[i]*100);
                b.executeUpdate();
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
            }
        }
    }
private                                                 void
FuzzifikasiCompleteActionPerformed(java.awt.event.ActionE
vent evt) {
        Get();
        Fuzzifikasi();
        try{
            Object  []  rows={"No","Nama","MF  Protein","MF
Lemak","MF   Karbohidrat","MF   Kalsium","MF   Fosfor","MF
Besi","MF Vit.A","MF Vit.B1","MF Vit.C","MF Air"};
```

```
            DefaultTableModel                       dtm=new
DefaultTableModel(null,rows);
            TAMPIL1.setModel(dtm);
            TAMPIL1.setBorder(null);
            for(int z=0; z<NAMA.length; z++){
                String    []    tampil={String.valueOf(z),
NAMA[z],String.valueOf(PROTEIN[z]),String.valueOf(LEMAK[z
]),

String.valueOf(KARBOHIDRAT[z]),String.valueOf(KALSIUM[z])
,String.valueOf(FOSFOR[z]),

String.valueOf(BESI[z]),String.valueOf(VITA[z]),String.va
lueOf(VITB1[z]),String.valueOf(VITC[z]),
                    String.valueOf(AIR[z])};
                dtm.addRow(tampil);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
        int h=23;
        for(int i=0;i<NAMA.length;i++){
            try{
                Class.forName("com.mysql.jdbc.Driver");
                PreparedStatement           b           =
connection.prepareStatement("INSERT INTO hasilfuzzy "

+  "(id_hasilfuzzy,  nama_hasilfuzzy,  protein_hasilfuzzy,
lemak_hasilfuzzy,"

+     "    karbohidrat_hasilfuzzy,    kalsium_hasilfuzzy,
fosfor_hasilfuzzy,"

+    "besi_hasilfuzzy,    a_hasilfuzzy,    b1_hasilfuzzy,
c_hasilfuzzy, air_hasilfuzzy) "

+ "values (?,?,?,?,?,?,?,?,?,?,?,?)");
                b.setInt(1, h);
                b.setString(2, NAMA[i]);
                b.setDouble(3, PROTEIN[i]*1000);
                b.setDouble(4, LEMAK[i]*1000);
                b.setDouble(5, KARBOHIDRAT[i]*1000);
                b.setDouble(6, KALSIUM[i]*1000);
                b.setDouble(7, FOSFOR[i]*1000);
                b.setDouble(8, BESI[i]*1000);
                b.setDouble(9, VITA[i]*1000);
                b.setDouble(10, VITB1[i]*1000);
                b.setDouble(11, VITC[i]*1000);
                b.setDouble(12, AIR[i]*1000);
```

```
                b.executeUpdate();
                h++;
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
            }
        }
private                                                 void
GolonganActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Object                                        []
rows={"No","Nama","Protein","Lemak","Karbohidrat","Kalsiu
m","Fosfor","Besi","Vit.A","Vit.B1","Vit.C","Air"};
            DefaultTableModel                       dtm=new
DefaultTableModel(null,rows);
            TAMPIL2.setModel(dtm);
            TAMPIL2.setBorder(null);
            for(int z=0; z<NAMA.length; z++){
                String    []    tampil={String.valueOf(z),
NAMA[z],String.valueOf(GOLPROTEIN[z]),String.valueOf(GOLL
EMAK[z]),

String.valueOf(GOLKARBOHIDRAT[z]),String.valueOf(GOLKALSI
UM[z]),String.valueOf(GOLFOSFOR[z]),

String.valueOf(GOLBESI[z]),String.valueOf(GOLVITA[z]),Str
ing.valueOf(GOLVITB1[z]),String.valueOf(GOLVITC[z]),
                    String.valueOf(GOLAIR[z])};
                dtm.addRow(tampil);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
```

5. Fuzzifikasi & Defuzzifikasi *Incomplete Data*

```
package fuzzysom;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class fuzzyincomplete extends javax.swing.JFrame {
    private Connection connection;
    private Statement statement;
```

```
    public int[] ID;
    public String[] NAMA;
    public Double[] PROTEIN;
    public Double[] LEMAK;
    public Double[] KARBOHIDRAT;
    public Double[] KALSIUM;
    public Double[] FOSFOR;
    public Double[] BESI;
    public Double[] VITA;
    public Double[] VITB1;
    public Double[] VITC;
    public Double[] AIR;
    public Double[] RePROTEIN;
    public Double[] ReLEMAK;
    public Double[] ReKARBOHIDRAT;
    public Double[] ReKALSIUM;
    public Double[] ReFOSFOR;
    public Double[] ReBESI;
    public Double[] ReVITA;
    public Double[] ReVITB1;
    public Double[] ReVITC;
    public Double[] ReAIR;
    public Double[][] DerPROTEIN;
    public Double[][] DerLEMAK;
    public Double[][] DerKARBOHIDRAT;
    public Double[][] DerKALSIUM;
    public Double[][] DerFOSFOR;
    public Double[][] DerBESI;
    public Double[][] DerVITA;
    public Double[][] DerVITB1;
    public Double[][] DerVITC;
    public Double[][] DerAIR;
    public int[] GOLPROTEIN;
    public int[] GOLLEMAK;
    public int[] GOLKARBOHIDRAT;
    public int[] GOLKALSIUM;
    public int[] GOLFOSFOR;
    public int[] GOLBESI;
    public int[] GOLVITA;
    public int[] GOLVITB1;
    public int[] GOLVITC;
    public int[] GOLAIR;
    public fuzzyincomplete() {
        initComponents();
        setLocationRelativeTo(this);
        try{
            Class.forName("com.mysql.jdbc.Driver");
            connection                                    =
DriverManager.getConnection("jdbc:mysql://localhost:3306/
```

```
fuzzysom?zeroDateTimeBehavior=convertToNull",      "root",
"");
            statement = connection.createStatement();
        }catch(Exception e){
            JOptionPane.showMessageDialog(null,   "Koneksi
gagal");
        }
    }
public void Get(){
        String nam = "";
        Double pro, lem, kar, kal, fos, bes, vita, vitb1,
vitc, air;
        String cntj1 = "";
        try{
            String  sql  =  "SELECT  count(nama_incomplete)
FROM incomplete";
            Statement              statement              =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            while(rs.next()){
                cntj1                                      =
rs.getString("count(nama_incomplete)");
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,    "Query
Salah " + e);
        }
        int index = Integer.parseInt(cntj1);
        ID = new int[index];
        NAMA = new String[index];
        PROTEIN = new Double[index];
        LEMAK = new Double[index];
        KARBOHIDRAT = new Double[index];
        KALSIUM = new Double[index];
        FOSFOR = new Double[index];
        BESI = new Double[index];
        VITA = new Double[index];
        VITB1 = new Double[index];
        VITC = new Double[index];
        AIR = new Double[index];
        try{
            String   sql   =   "SELECT   nama_incomplete,
protein_incomplete,                      lemak_incomplete,
karbohidrat_incomplete, "
                    +               "kalsium_incomplete,
fosfor_incomplete,     besi_incomplete,     a_incomplete,
b1_incomplete, c_incomplete, "
                    + "air_incomplete FROM incomplete";
```

```
            Statement             statement             =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            int a = 0;
            while(rs.next()){
                nam = rs.getString("nama_incomplete");
                pro = rs.getDouble("protein_incomplete");
                lem = rs.getDouble("lemak_incomplete");
                kar                                    =
rs.getDouble("karbohidrat_incomplete");
                kal = rs.getDouble("kalsium_incomplete");
                fos = rs.getDouble("fosfor_incomplete");
                bes = rs.getDouble("besi_incomplete");
                vita = rs.getDouble("a_incomplete");
                vitb1 = rs.getDouble("b1_incomplete");
                vitc = rs.getDouble("c_incomplete");
                air = rs.getDouble("air_incomplete");
                ID[a] = a;
                NAMA[a] = nam;
                PROTEIN[a] = pro;
                LEMAK[a] = lem;
                KARBOHIDRAT[a] = kar;
                KALSIUM[a] = kal;
                FOSFOR[a] = fos;
                BESI[a] = bes;
                VITA[a] = vita;
                VITB1[a] = vitb1;
                VITC[a] = vitc;
                AIR[a] = air;
                a++;
            }

System.out.println("================================ NAMA
BAHAN PANGAN ================================");
            for (int i = 0; i < index; i++) {
                System.out.println(NAMA[i]);
            }
            System.out.println();

System.out.println("================================
PROTEIN ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("PROTEIN[" + i + "] =>
Protein " + NAMA[i] + " : " + PROTEIN[i]);
            }
            System.out.println();

System.out.println("================================
LEMAK ================================");
```

```
            for (int i = 0; i < index; i++) {
                System.out.println("LEMAK[" + i + "] =>
Lemak " + NAMA[i] + " : " + LEMAK[i]);
            }
            System.out.println();

System.out.println("================================
KARBOHIDRAT ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("KARBOHIDRAT[" + i + "]
=> Karbohidrat " + NAMA[i] + " : " + KARBOHIDRAT[i]);
            }
            System.out.println();

System.out.println("================================
KALSIUM ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("KALSIUM[" + i + "] =>
Kalsium " + NAMA[i] + " : " + KALSIUM[i]);
            }
            System.out.println();

System.out.println("================================
FOSFOR ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("FOSFOR[" + i + "] =>
Fosfor " + NAMA[i] + " : " + FOSFOR[i]);
            }
            System.out.println();

System.out.println("================================ BESI
================================");
            for (int i = 0; i < index; i++) {
                System.out.println("BESI[" + i + "] =>
Besi " + NAMA[i] + " : " + BESI[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN A ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINA[" + i + "] =>
Vitamin A " + NAMA[i] + " : " + VITA[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN B1 ================================");
            for (int i = 0; i < index; i++) {
```

```
                System.out.println("VITAMINB1[" + i + "]
=> Vitamin B1 " + NAMA[i] + " : " + VITB1[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN C ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINC[" + i + "] =>
Vitamin C " + NAMA[i] + " : " + VITC[i]);
            }
            System.out.println();

System.out.println("================================  AIR
================================");
            for (int i = 0; i < index; i++) {
                System.out.println("AIR[" + i + "] => Air
" + NAMA[i] + " : " + AIR[i]);
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,    "Query
Salah " + e);
        }
public void Fuzzifikasi(){
        GOLPROTEIN=new int[NAMA.length];
        GOLLEMAK=new int[NAMA.length];
        GOLKARBOHIDRAT=new int[NAMA.length];
        GOLKALSIUM=new int[NAMA.length];
        GOLFOSFOR=new int[NAMA.length];
        GOLBESI=new int[NAMA.length];
        GOLVITA=new int[NAMA.length];
        GOLVITB1=new int[NAMA.length];
        GOLVITC=new int[NAMA.length];
        GOLAIR=new int[NAMA.length];
        DerPROTEIN=new Double[3][NAMA.length];
        DerLEMAK=new Double[3][NAMA.length];
        DerKARBOHIDRAT=new Double[3][NAMA.length];
        DerKALSIUM=new Double[3][NAMA.length];
        DerFOSFOR=new Double[3][NAMA.length];
        DerBESI=new Double[3][NAMA.length];
        DerVITA=new Double[3][NAMA.length];
        DerVITB1=new Double[3][NAMA.length];
        DerVITC=new Double[3][NAMA.length];
        DerAIR=new Double[3][NAMA.length];
        //mulai fuzzyfikasi untuk tiap bahan pangan
        for (int i = 0; i < NAMA.length; i++) {
            System.out.println("-------------------------
--- " + NAMA[i] + " ---------------------------- ");
```

```
            //=========================================
Protein ========================================
            int st=0;
            Double min = 2.0;
            if(PROTEIN[i]==999999999){
                PROTEIN[i]=999999999.0;
            }else{
                //rendah
                if (PROTEIN[i] <= 6.8){
                    DerPROTEIN[0][i] = 1.0;
                }else if (PROTEIN[i] < 13 && PROTEIN[i] >
6.8){
                    DerPROTEIN[0][i] = (13 - PROTEIN[i]) /
6.2;
                }else if (PROTEIN[i] >= 13){
                    DerPROTEIN[0][i] = 0.0;}
                System.out.println("Protein rendah : " +
DerPROTEIN[0][i]);
                //sedang
                if (PROTEIN[i] <= 6.8) {
                    DerPROTEIN[1][i] = 0.0;
                } else if (PROTEIN[i] <= 13 && PROTEIN[i]
> 6.8) {
                    DerPROTEIN[1][i] = (PROTEIN[i] - 6.8)
/ 6.2;
                } else if (PROTEIN[i] < 17 && PROTEIN[i] >
13) {
                    DerPROTEIN[1][i] = (17 - PROTEIN[i]) /
4;
                } else if (PROTEIN[i] >= 17) {
                    DerPROTEIN[1][i] = 0.0;}
                System.out.println("Protein sedang : " +
DerPROTEIN[1][i]);
                //tinggi
                if (PROTEIN[i] <= 13){
                    DerPROTEIN[2][i] = 0.0;
                }else if (PROTEIN[i] < 17 && PROTEIN[i] >
13){
                    DerPROTEIN[2][i] = (PROTEIN[i] - 13) /
4;
                }else if (PROTEIN[i] >= 17){
                    DerPROTEIN[2][i] = 1.0;}
                System.out.println("Protein tinggi : " +
DerPROTEIN[2][i]);
                //rumus gabungan(nilai min)
                if(DerPROTEIN[0][i] != 0.0){
                if(DerPROTEIN[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerPROTEIN[j][i] < min){
```

```
                            min=DerPROTEIN[j][i];
                            st=j;}}}
                else if(DerPROTEIN[0][i] == 1.0){
                    min=0.0;
                    st=0;}
             }
            else if(DerPROTEIN[2][i] != 0.0){
                if(DerPROTEIN[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerPROTEIN[j][i] < min){
                            min=DerPROTEIN[j][i];
                            st=j;}}}
                else if(DerPROTEIN[2][i] == 1.0){
                    min=1.0;
                    st=2;}
            }
                GOLPROTEIN[i]=st;
                PROTEIN[i] = min;
            }
            System.out.println("Nilai   protein   =   "   +
PROTEIN[i]+" merupakan golongan "+GOLPROTEIN[i]);
            System.out.println("-------------------------
---");

            //=======================================
Lemak ========================================
            int st1=0;
            Double min1 = 2.0;
            if(LEMAK[i]==999999999){
                LEMAK[i]=999999999.0;
            }else{
                //rendah
                if(LEMAK[i]<=5){
                    DerLEMAK[0][i]=1.0;
                }else if(LEMAK[i]<16 && LEMAK[i]>5){
                    DerLEMAK[0][i]=(16-LEMAK[i])/11;
                }else if(LEMAK[i]>=16){
                    DerLEMAK[0][i]=0.0;}
                System.out.println("Lemak     rendah     :
"+DerLEMAK[0][i]);
                //sedang
                if(LEMAK[i]<=5){
                    DerLEMAK[1][i]=0.0;
                }else if(LEMAK[i]<=16 && LEMAK[i]>5){
                    DerLEMAK[1][i]=(LEMAK[i]-5)/11;
                }else if(LEMAK[i]<30 && LEMAK[i]>16){
                    DerLEMAK[1][i]=(30-LEMAK[i])/14;
                }else if(LEMAK[i]>=30){
                    DerLEMAK[1][i]=0.0;}
```

```
                System.out.println("Lemak     sedang     :
"+DerLEMAK[1][i]);
                //tinggi
                if(LEMAK[i]<=16){
                    DerLEMAK[2][i]=0.0;
                }else if(LEMAK[i]<30 && LEMAK[i]>16){
                    DerLEMAK[2][i]=(LEMAK[i]-16)/14;
                }else if(LEMAK[i]>=30){
                    DerLEMAK[2][i]=1.0;}
                System.out.println("Lemak     tinggi     :
"+DerLEMAK[2][i]);
                //rumus gabungan(nilai min)
                if(DerLEMAK[0][i] != 0.0){
                if(DerLEMAK[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerLEMAK[j][i] < min1){
                            min1=DerLEMAK[j][i];
                            st1=j;}}}
                else if(DerLEMAK[0][i] == 1.0){
                    min1=0.0;
                    st1=0;}
             }
            else if(DerLEMAK[2][i] != 0.0){
                if(DerLEMAK[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerLEMAK[j][i] < min1){
                            min1=DerLEMAK[j][i];
                            st1=j;}}}
                else if(DerLEMAK[2][i] == 1.0){
                    min1=1.0;
                    st1=2;}
            }
                GOLLEMAK[i]=st1;
                LEMAK[i]=min1;
            }
            System.out.println("Nilai lemak = "+LEMAK[i]+"
merupakan golongan "+GOLLEMAK[i]);
            System.out.println("-------------------------
---");

            //=======================================
Karbohidrat =======================================
            int st2=0;
            Double min2 = 2.0;
            if(KARBOHIDRAT[i]==999999999){
                KARBOHIDRAT[i]=999999999.0;
            }else{
                //rendah
                if(KARBOHIDRAT[i]<=12){
```

```
                    DerKARBOHIDRAT[0][i]=1.0;
                }else       if(KARBOHIDRAT[i]<30       &&
KARBOHIDRAT[i]>12){
                    DerKARBOHIDRAT[0][i]=(30-
KARBOHIDRAT[i])/18;
                }else if(KARBOHIDRAT[i]>=30){
                    DerKARBOHIDRAT[0][i]=0.0;}
                System.out.println("Karbohidrat  rendah  :
"+DerKARBOHIDRAT[0][i]);
                //sedang
                if(KARBOHIDRAT[i]<=12){
                    DerKARBOHIDRAT[1][i]=0.0;
                }else       if(KARBOHIDRAT[i]<=30       &&
KARBOHIDRAT[i]>12){
                    DerKARBOHIDRAT[1][i]=(KARBOHIDRAT[i]-
12)/18;
                }else       if(KARBOHIDRAT[i]<70       &&
KARBOHIDRAT[i]>30){
                    DerKARBOHIDRAT[1][i]=(70-
KARBOHIDRAT[i])/40;
                }else if(KARBOHIDRAT[i]>=70){
                    DerKARBOHIDRAT[1][i]=0.0;}
                System.out.println("Karbohidrat  sedang  :
"+DerKARBOHIDRAT[1][i]);
                //tinggi
                if(KARBOHIDRAT[i]<=30){
                    DerKARBOHIDRAT[2][i]=0.0;
                }else       if(KARBOHIDRAT[i]<70       &&
KARBOHIDRAT[i]>30){
                    DerKARBOHIDRAT[2][i]=(KARBOHIDRAT[i]-
30)/40;
                }else if(KARBOHIDRAT[i]>=30){
                    DerKARBOHIDRAT[2][i]=1.0;}
                System.out.println("Karbohidrat  tinggi  :
"+DerKARBOHIDRAT[2][i]);
                //rumus gabungan(nilai min)
                if(DerKARBOHIDRAT[0][i] != 0.0){
                if(DerKARBOHIDRAT[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerKARBOHIDRAT[j][i] < min2){
                            min2=DerKARBOHIDRAT[j][i];
                            st2=j;}}}
                else if(DerKARBOHIDRAT[0][i] == 1.0){
                    min2=0.0;
                    st2=0;}
             }
            else if(DerKARBOHIDRAT[2][i] != 0.0){
                if(DerKARBOHIDRAT[1][i] != 0.0){
                    for(int j=1; j<3; j++){
```

```
                        if(DerKARBOHIDRAT[j][i] < min2){
                            min2=DerKARBOHIDRAT[j][i];
                            st2=j;}}}
                else if(DerKARBOHIDRAT[2][i] == 1.0){
                    min2=1.0;
                    st2=2;}
            }
                GOLKARBOHIDRAT[i]=st2;
                KARBOHIDRAT[i]=min2;
            }
            System.out.println("Nilai    karbohidrat    =
"+KARBOHIDRAT[i]+"           merupakan           golongan
"+GOLKARBOHIDRAT[i]);
            System.out.println("-------------------------
---");

            //=========================================
Kalsium ==========================================
            int st3=0;
            Double min3 = 2.0;
            if(KALSIUM[i]==999999999){
                KALSIUM[i]=999999999.0;
            }else{
                //rendah
                if(KALSIUM[i]<=50){
                    DerKALSIUM[0][i]=1.0;
                }else if(KALSIUM[i]<120 && KALSIUM[i]>50){
                    DerKALSIUM[0][i]=(120-KALSIUM[i])/70;
                }else if(KALSIUM[i]>=120){
                    DerKALSIUM[0][i]=0.0;}
                System.out.println("Kalsium    rendah    :
"+DerKALSIUM[0][i]);
                //sedang
                if(KALSIUM[i]<=50){
                    DerKALSIUM[1][i]=0.0;
                }else        if(KALSIUM[i]<=120        &&
KALSIUM[i]>50){
                    DerKALSIUM[1][i]=(KALSIUM[i]-50)/70;
                }else         if(KALSIUM[i]<354         &&
KALSIUM[i]>120){
                    DerKALSIUM[1][i]=(354-
KALSIUM[i])/234;
                }else if(KALSIUM[i]>=354){
                    DerKALSIUM[1][i]=0.0;}
                System.out.println("Kalsium    sedang    :
"+DerKALSIUM[1][i]);
                //tinggi
                if(KALSIUM[i]<=120){
                    DerKALSIUM[2][i]=0.0;
```

```
                }else         if(KALSIUM[i]<354         &&
KALSIUM[i]>120){
                    DerKALSIUM[2][i]=(KALSIUM[i]-
120)/234;
                }else if(KALSIUM[i]>=354){
                    DerKALSIUM[2][i]=1.0;}
                System.out.println("Kalsium    tinggi    :
"+DerKALSIUM[2][i]);
                //rumus gabungan(nilai min)
                if(DerKALSIUM[0][i] != 0.0){
                if(DerKALSIUM[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerKALSIUM[j][i] < min3){
                            min3=DerKALSIUM[j][i];
                            st3=j;}}}
                else if(DerKALSIUM[0][i] == 1.0){
                    min3=0.0;
                    st3=0;}
             }
            else if(DerKALSIUM[2][i] != 0.0){
                if(DerKALSIUM[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerKALSIUM[j][i] < min3){
                            min3=DerKALSIUM[j][i];
                            st3=j;}}}
                else if(DerKALSIUM[2][i] == 1.0){
                    min3=1.0;
                    st3=2;}
            }
                GOLKALSIUM[i]=st3;
                KALSIUM[i]=min3;
            }
            System.out.println("Nilai      kalsium      =
"+KALSIUM[i]+" merupakan golongan "+GOLKALSIUM[i]);
            System.out.println("-------------------------
---");

            //=======================================
Fosfor ========================================
            int st4=0;
            Double min4 = 2.0;
            if(FOSFOR[i]==999999999){
                FOSFOR[i]=999999999.0;
            }else{
                //rendah
                if(FOSFOR[i]<=110){
                    DerFOSFOR[0][i]=1.0;
                }else if(FOSFOR[i]<170 && FOSFOR[i]>110){
                    DerFOSFOR[0][i]=(170-FOSFOR[i])/60;
```

```
                }else if(FOSFOR[i]>=170){
                    DerFOSFOR[0][i]=0.0;}
                System.out.println("Fosfor    rendah    :
"+DerFOSFOR[0][i]);
                //sedang
                if(FOSFOR[i]<=110){
                    DerFOSFOR[1][i]=0.0;
                }else if(FOSFOR[i]<=170 && FOSFOR[i]>110){
                    DerFOSFOR[1][i]=(FOSFOR[i]-110)/60;
                }else if(FOSFOR[i]<256 && FOSFOR[i]>170){
                    DerFOSFOR[1][i]=(256-FOSFOR[i])/86;
                }else if(FOSFOR[i]>=256){
                    DerFOSFOR[1][i]=0.0;}
                System.out.println("Fosfor    sedang    :
"+DerFOSFOR[1][i]);
                //tinggi
                if(FOSFOR[i]<=170){
                    DerFOSFOR[2][i]=0.0;
                }else if(FOSFOR[i]<256 && FOSFOR[i]>170){
                    DerFOSFOR[2][i]=(FOSFOR[i]-170)/86;
                }else if(FOSFOR[i]>=256){
                    DerFOSFOR[2][i]=1.0;}
                System.out.println("Fosfor    tinggi    :
"+DerFOSFOR[2][i]);
                //rumus gabungan(nilai min)
                if(DerFOSFOR[0][i] != 0.0){
                if(DerFOSFOR[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerFOSFOR[j][i] < min4){
                            min4=DerFOSFOR[j][i];
                            st4=j;}}}
                else if(DerFOSFOR[0][i] == 1.0){
                    min4=0.0;
                    st4=0;}
             }
            else if(DerFOSFOR[2][i] != 0.0){
                if(DerFOSFOR[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerFOSFOR[j][i] < min4){
                            min4=DerFOSFOR[j][i];
                            st4=j;}}}
                else if(DerFOSFOR[2][i] == 1.0){
                    min4=1.0;
                    st4=2;}
            }
                GOLFOSFOR[i]=st4;
                FOSFOR[i]=min4;
            }
```

```
            System.out.println("Nilai       fosfor       =
"+FOSFOR[i]+" merupakan golongan "+GOLFOSFOR[i]);
            System.out.println("-------------------------
---");

            //=========================================
Besi ========================================
            int st5=0;
            Double min5 = 2.0;
            if(BESI[i]==999999999){
                BESI[i]=999999999.0;
            }else{
                //rendah
                if(BESI[i]<=2){
                    DerBESI[0][i]=1.0;
                }else if(BESI[i]<5.7 && BESI[i]>2){
                    DerBESI[0][i]=(5.7-BESI[i])/3.7;
                }else if(BESI[i]>=5.7){
                    DerBESI[0][i]=0.0;}
                System.out.println("Besi     rendah     :
"+DerBESI[0][i]);
                //sedang
                if(BESI[i]<=2){
                    DerBESI[1][i]=0.0;
                }else if(BESI[i]<=5.7 && BESI[i]>2){
                    DerBESI[1][i]=(BESI[i]-2)/3.7;
                }else if(BESI[i]<20 && BESI[i]>5.7){
                    DerBESI[1][i]=(20-BESI[i])/14.3;
                }else if(BESI[i]>=20){
                    DerBESI[1][i]=0.0;}
                System.out.println("Besi     sedang     :
"+DerBESI[1][i]);
                //tinggi
                if(BESI[i]<=5.7){
                    DerBESI[2][i]=0.0;
                }else if(BESI[i]<20 && BESI[i]>5.7){
                    DerBESI[2][i]=(BESI[i]-5.7)/14.3;
                }else if(BESI[i]>=20){
                    DerBESI[2][i]=1.0;}
                System.out.println("Besi     tinggi     :
"+DerBESI[2][i]);
                //rumus gabungan(nilai min)
                if(DerBESI[0][i] != 0.0){
                if(DerBESI[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerBESI[j][i] < min5){
                            min5=DerBESI[j][i];
                            st5=j;}}}
                else if(DerBESI[0][i] == 1.0){
```

```
                    min5=0.0;
                    st5=0;}
             }
            else if(DerBESI[2][i] != 0.0){
                if(DerBESI[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerBESI[j][i] < min5){
                            min5=DerBESI[j][i];
                            st5=j;}}}
                else if(DerBESI[2][i] == 1.0){
                    min5=1.0;
                    st5=2;}
            }
                GOLBESI[i]=st5;
                BESI[i]=min5;
            }
            System.out.println("Nilai besi = "+BESI[i]+"
merupakan golongan "+GOLBESI[i]);
            System.out.println("-------------------------
---");

            //=======================================
Vitamin A ======================================
            int st6=0;
            Double min6 = 2.0;
            if(VITA[i]==999999999){
                VITA[i]=999999999.0;
            }else{
                //rendah
                if(VITA[i]<=1000){
                    DerVITA[0][i]=1.0;
                }else if(VITA[i]<3900 && VITA[i]>1000){
                    DerVITA[0][i]=(3900-VITA[i])/2900;
                }else if(VITA[i]>=3900){
                    DerVITA[0][i]=0.0;}
                System.out.println("Vitamin A rendah :
"+DerVITA[0][i]);
                //sedang
                if(VITA[i]<=1000){
                    DerVITA[1][i]=0.0;
                }else if(VITA[i]<=3900 && VITA[i]>1000){
                    DerVITA[1][i]=(VITA[i]-1000)/2900;
                }else if(VITA[i]<15000 && VITA[i]>3900){
                    DerVITA[1][i]=(15000-VITA[i])/11100;
                }else if(VITA[i]>=15000){
                    DerVITA[1][i]=0.0;}
                System.out.println("Vitamin A sedang :
"+DerVITA[1][i]);
                //tinggi
```

```
                if(VITA[i]<=3900){
                    DerVITA[2][i]=0.0;
                }else if(VITA[i]<15000 && VITA[i]>3900){
                    DerVITA[2][i]=(VITA[i]-3900)/11100;
                }else if(VITA[i]>=15000){
                    DerVITA[2][i]=1.0;}
                System.out.println("Vitamin  A  tinggi  :
"+DerVITA[2][i]);
                //rumus gabungan(nilai min)
                if(DerVITA[0][i] != 0.0){
                if(DerVITA[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerVITA[j][i] < min6){
                            min6=DerVITA[j][i];
                            st6=j;}}}
                else if(DerVITA[0][i] == 1.0){
                    min6=0.0;
                    st6=0;}
             }
            else if(DerVITA[2][i] != 0.0){
                if(DerVITA[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerVITA[j][i] < min6){
                            min6=DerVITA[j][i];
                            st6=j;}}}
                else if(DerVITA[2][i] == 1.0){
                    min6=1.0;
                    st6=2;}
            }
                GOLVITA[i]=st6;
                VITA[i]=min6;
            }
            System.out.println("Nilai    Vitamin    A    =
"+VITA[i]+" merupakan golongan "+GOLVITA[i]);
            System.out.println("-------------------------
---");

            //=========================================
Vitamin B1 =========================================
            int st7=0;
            Double min7 = 2.0;
            if(VITB1[i]==999999999){
                VITB1[i]=999999999.0;
            }else{
                //rendah
                if(VITB1[i]<=0.1){
                    DerVITB1[0][i]=1.0;
                }else if(VITB1[i]<0.28 && VITB1[i]>0.1){
                    DerVITB1[0][i]=(0.28-VITB1[i])/0.18;
```

```
                }else if(VITB1[i]>=0.28){
                    DerVITB1[0][i]=0.0;}
                System.out.println("Vitamin  B1  rendah  :
"+DerVITB1[0][i]);
                //sedang
                if(VITB1[i]<=0.1){
                    DerVITB1[1][i]=0.0;
                }else if(VITB1[i]<=0.28 && VITB1[i]>0.1){
                    DerVITB1[1][i]=(VITB1[i]-0.1)/0.18;
                }else if(VITB1[i]<0.45 && VITB1[i]>0.28){
                    DerVITB1[1][i]=(0.45-VITB1[i])/0.17;
                }else if(VITB1[i]>=0.45){
                    DerVITB1[1][i]=0.0;}
                System.out.println("Vitamin  B1  sedang  :
"+DerVITB1[1][i]);
                //tinggi
                if(VITB1[i]<=0.28){
                    DerVITB1[2][i]=0.0;
                }else if(VITB1[i]<0.45 && VITB1[i]>0.28){
                    DerVITB1[2][i]=(VITB1[i]-0.28)/0.17;
                }else if(VITB1[i]>=0.45){
                    DerVITB1[2][i]=1.0;}
                System.out.println("Vitamin  B1  tinggi  :
"+DerVITB1[2][i]);
                //rumus gabungan(nilai min)
                if(DerVITB1[0][i] != 0.0){
                if(DerVITB1[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerVITB1[j][i] < min7){
                            min7=DerVITB1[j][i];
                            st7=j;}}}
                else if(DerVITB1[0][i] == 1.0){
                    min7=0.0;
                    st7=0;}
             }
            else if(DerVITB1[2][i] != 0.0){
                if(DerVITB1[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerVITB1[j][i] < min7){
                            min7=DerVITB1[j][i];
                            st7=j;}}}
                else if(DerVITB1[2][i] == 1.0){
                    min7=1.0;
                    st7=2;}
            }
                GOLVITB1[i]=st7;
                VITB1[i]=min7;
            }
```

```
            System.out.println("Nilai   Vitamin   B1   =
"+VITB1[i]+" merupakan golongan "+GOLVITB1[i]);
            System.out.println("-------------------------
---");

            //=======================================
Vitamin C ======================================
            int st8=0;
            Double min8 = 2.0;
            if(VITC[i]==999999999){
                VITC[i]=999999999.0;
            }else{
                //rendah
                if(VITC[i]<=10){
                    DerVITC[0][i]=1.0;
                }else if(VITC[i]<40 && VITC[i]>10){
                    DerVITC[0][i]=(40-VITC[i])/30;
                }else if(VITC[i]>=40){
                    DerVITC[0][i]=0.0;}
                System.out.println("Vitamin  C  rendah  :
"+DerVITC[0][i]);
                //sedang
                if(VITC[i]<=10){
                    DerVITC[1][i]=0.0;
                }else if(VITC[i]<=40 && VITC[i]>10){
                    DerVITC[1][i]=(VITC[i]-10)/30;
                }else if(VITC[i]<80 && VITC[i]>40){
                    DerVITC[1][i]=(80-VITC[i])/40;
                }else if(VITC[i]>=80){
                    DerVITC[1][i]=0.0;}
                System.out.println("Vitamin  C  sedang  :
"+DerVITC[1][i]);
                //tinggi
                if(VITC[i]<=40){
                    DerVITC[2][i]=0.0;
                }else if(VITC[i]<80 && VITC[i]>40){
                    DerVITC[2][i]=(VITC[i]-40)/40;
                }else if(VITC[i]>=80){
                    DerVITC[2][i]=1.0;}
                System.out.println("Vitamin  C  tinggi  :
"+DerVITC[2][i]);
                //rumus gabungan(nilai min)
                if(DerVITC[0][i] != 0.0){
                if(DerVITC[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerVITC[j][i] < min8){
                            min8=DerVITC[j][i];
                            st8=j;}}}
                else if(DerVITC[0][i] == 1.0){
```

```
                    min8=0.0;
                    st8=0;}
                }
            else if(DerVITC[2][i] != 0.0){
                if(DerVITC[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerVITC[j][i] < min8){
                            min8=DerVITC[j][i];
                            st8=j;}}}
                else if(DerVITC[2][i] == 1.0){
                    min8=1.0;
                    st8=2;}
            }
                GOLVITC[i]=st8;
                VITC[i]=min8;
            }
            System.out.println("Nilai    Vitamin    C    =
"+VITC[i]+" merupakan golongan "+GOLVITC[i]);
            System.out.println("-------------------------
---");

            //=======================================
Air ========================================
            int st9=0;
            Double min9 = 2.0;
            if(AIR[i]==999999999){
                AIR[i]=999999999.0;
            }else{
                //rendah
                if(AIR[i]<=25){
                    DerAIR[0][i]=1.0;
                }else if(AIR[i]<55 && AIR[i]>25){
                    DerAIR[0][i]=(55-AIR[i])/30;
                }else if(AIR[i]>=55){
                    DerAIR[0][i]=0.0;}
                System.out.println("Air      rendah      :
"+DerAIR[0][i]);
                //sedang
                if(AIR[i]<=25){
                    DerAIR[1][i]=0.0;
                }else if(AIR[i]<=55 && AIR[i]>25){
                    DerAIR[1][i]=(AIR[i]-25)/30;
                }else if(AIR[i]<70 && AIR[i]>55){
                    DerAIR[1][i]=(70-AIR[i])/15;
                }else if(AIR[i]>=70){
                    DerAIR[1][i]=0.0;}
                System.out.println("Air      sedang      :
"+DerAIR[1][i]);
                //tinggi
```

```
                if(AIR[i]<=55){
                    DerAIR[2][i]=0.0;
                }else if(AIR[i]<70 && AIR[i]>55){
                    DerAIR[2][i]=(AIR[i]-55)/15;
                }else if(AIR[i]>=70){
                    DerAIR[2][i]=1.0;}
                System.out.println("Air      tinggi      :
"+DerAIR[2][i]);
                //rumus gabungan(nilai min)
                if(DerAIR[0][i] != 0.0){
                if(DerAIR[1][i] != 0.0){
                    for(int j=0; j<2; j++){
                        if(DerAIR[j][i] < min9){
                            min9=DerAIR[j][i];
                            st9=j;}}}
                else if(DerAIR[0][i] == 1.0){
                    min9=0.0;
                    st9=0;}
             }
            else if(DerAIR[2][i] != 0.0){
                if(DerAIR[1][i] != 0.0){
                    for(int j=1; j<3; j++){
                        if(DerAIR[j][i] < min9){
                            min9=DerAIR[j][i];
                            st9=j;}}}
                else if(DerAIR[2][i] == 1.0){
                    min9=1.0;
                    st9=2;}
            }
                GOLAIR[i]=st9;
                AIR[i]=min9;
            }
            System.out.println("Nilai  Air  =  "+AIR[i]+"
merupakan golongan "+GOLAIR[i]);
        }
    }
public void CariNilai(){
        for(int i=0;i<NAMA.length;i++){
            if(PROTEIN[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                    if(GOLKARBOHIDRAT[i]==0           &&
GOLLEMAK[i]==0 && GOLAIR[i]==0){
                    GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0      &&
GOLLEMAK[i]==0 && GOLAIR[i]==1){
                    GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0      &&
GOLLEMAK[i]==0 && GOLAIR[i]==2){
                    GOLPROTEIN[i]=0;}
```

```
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLLEMAK[i]==1 && GOLAIR[i]==0){
                     GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLLEMAK[i]==1 && GOLAIR[i]==1){
                     GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLLEMAK[i]==1 && GOLAIR[i]==2){
                     GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLLEMAK[i]==2 && GOLAIR[i]==0){
                     GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLLEMAK[i]==2 && GOLAIR[i]==1){
                     GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLLEMAK[i]==2 && GOLAIR[i]==2){
                     GOLPROTEIN[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==0 && GOLAIR[i]==0){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==0 && GOLAIR[i]==1){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==0 && GOLAIR[i]==2){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==1 && GOLAIR[i]==0){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==1 && GOLAIR[i]==1){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==1 && GOLAIR[i]==2){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==2 && GOLAIR[i]==0){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==2 && GOLAIR[i]==1){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLLEMAK[i]==2 && GOLAIR[i]==2){
                     GOLPROTEIN[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==0 && GOLAIR[i]==0){
                     GOLPROTEIN[i]=2;}
```

```
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==0 && GOLAIR[i]==1){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==0 && GOLAIR[i]==2){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==1 && GOLAIR[i]==0){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==1 && GOLAIR[i]==1){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==1 && GOLAIR[i]==2){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==2 && GOLAIR[i]==0){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==2 && GOLAIR[i]==1){
                    GOLPROTEIN[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLLEMAK[i]==2 && GOLAIR[i]==2){
                    GOLPROTEIN[i]=2;}
                //penentuan  derajat  keanggotaan  dengan
aturan min

if(KARBOHIDRAT[i]<=LEMAK[i]&&KARBOHIDRAT[i]<=AIR[i]){
                    PROTEIN[i]=KARBOHIDRAT[i];
                }
                else
if(LEMAK[i]<=KARBOHIDRAT[i]&&LEMAK[i]<=AIR[i]){
                    PROTEIN[i]=LEMAK[i];
                }
                else
if(AIR[i]<=KARBOHIDRAT[i]&&AIR[i]<=LEMAK[i]){
                    PROTEIN[i]=AIR[i];
                }
                System.out.println("Nilai  Karbohidrat  =
"+PROTEIN[i]+" merupakan golongan "+GOLPROTEIN[i]);
                System.out.println("----------------------
-------");
            }
            else if(LEMAK[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                if(GOLKARBOHIDRAT[i]==2                &&
GOLPROTEIN[i]==0 && GOLAIR[i]==0){
                    GOLLEMAK[i]=0;}
```

```
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==2){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==1){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==0){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==2){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==0){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==2){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==0 && GOLAIR[i]==1){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==0){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==2){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==1){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==0){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==2){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==1){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==0){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==1 && GOLAIR[i]==2){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==2       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==0){
                    GOLLEMAK[i]=1;}
```

```
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==2){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==1){
                    GOLLEMAK[i]=0;}
                else      if(GOLKARBOHIDRAT[i]==1       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==0){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==2){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==1){
                    GOLLEMAK[i]=1;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==0){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==2){
                    GOLLEMAK[i]=2;}
                else      if(GOLKARBOHIDRAT[i]==0       &&
GOLPROTEIN[i]==2 && GOLAIR[i]==1){
                    GOLLEMAK[i]=2;}
                //penentuan  derajat  keanggotaan  dengan
aturan min

if(KARBOHIDRAT[i]<=PROTEIN[i]&&KARBOHIDRAT[i]<=AIR[i]){
                    LEMAK[i]=KARBOHIDRAT[i];
                }
                else
if(PROTEIN[i]<=KARBOHIDRAT[i]&&PROTEIN[i]<=AIR[i]){
                    LEMAK[i]=PROTEIN[i];
                }
                else
if(AIR[i]<=KARBOHIDRAT[i]&&AIR[i]<=PROTEIN[i]){
                    LEMAK[i]=AIR[i];
                }
                System.out.println("Nilai  Karbohidrat  =
"+PROTEIN[i]+" merupakan golongan "+GOLPROTEIN[i]);
                System.out.println("----------------------
-------");
            }
            else if(KARBOHIDRAT[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                if(GOLPROTEIN[i]==0  &&  GOLLEMAK[i]==0  &&
GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=2;}
```

```
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==0
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=2;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==0
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=2;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==1
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=2;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==1
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=2;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==1
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==2
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==2
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==0 && GOLLEMAK[i]==2
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==0
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=2;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==0
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==0
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==1
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==1
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==1
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==2
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==2
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
```

```
                else if(GOLPROTEIN[i]==1 && GOLLEMAK[i]==2
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==1
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=2;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==0
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==0
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==1
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=1;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==1
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==1
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==2
&& GOLAIR[i]==0){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==2
&& GOLAIR[i]==2){
                    GOLKARBOHIDRAT[i]=0;}
                else if(GOLPROTEIN[i]==2 && GOLLEMAK[i]==2
&& GOLAIR[i]==1){
                    GOLKARBOHIDRAT[i]=0;}
                //penentuan  derajat  keanggotaan  dengan
aturan min

if(PROTEIN[i]<=LEMAK[i]&&PROTEIN[i]<=AIR[i]){
                    KARBOHIDRAT[i]=PROTEIN[i];
                }
                else
if(LEMAK[i]<=PROTEIN[i]&&LEMAK[i]<=AIR[i]){
                    KARBOHIDRAT[i]=LEMAK[i];
                }
                else
if(AIR[i]<=PROTEIN[i]&&AIR[i]<=LEMAK[i]){
                    KARBOHIDRAT[i]=AIR[i];
                }
                System.out.println("Nilai  Karbohidrat  =
"+KARBOHIDRAT[i]+"           merupakan           golongan
"+GOLKARBOHIDRAT[i]);
                System.out.println("--------------------
-------");
```

```
            }
            else if(KALSIUM[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                if(GOLBESI[i]==0){
                    GOLKALSIUM[i]=0;}
                else if(GOLBESI[i]==1){
                    GOLKALSIUM[i]=1;}
                else if(GOLBESI[i]==2){
                    GOLKALSIUM[i]=2;}
                //penentuan  derajat  keanggotaan  dengan
aturan min
                KALSIUM[i]=BESI[i];
                System.out.println("Nilai    kalsium    =
"+KALSIUM[i]+" merupakan golongan "+GOLKALSIUM[i]);
                System.out.println("----------------------
-------");
            }
            else if(FOSFOR[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                if(GOLPROTEIN[i]==0){
                    GOLFOSFOR[i]=0;}
                else if(GOLPROTEIN[i]==1){
                    GOLFOSFOR[i]=1;}
                else if(GOLPROTEIN[i]==2){
                    GOLFOSFOR[i]=2;}
                //penentuan  derajat  keanggotaan  dengan
aturan min
                FOSFOR[i]=PROTEIN[i];
                System.out.println("Nilai     fosfor     =
"+FOSFOR[i]+" merupakan golongan "+GOLFOSFOR[i]);
                System.out.println("----------------------
-------");
            }
            else if(BESI[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                if(GOLKALSIUM[i]==0){
                    GOLBESI[i]=0;}
                else if(GOLKALSIUM[i]==1){
                    GOLBESI[i]=1;}
                else if(GOLKALSIUM[i]==2){
                    GOLBESI[i]=2;}
                //penentuan  derajat  keanggotaan  dengan
aturan min
                BESI[i]=KALSIUM[i];
                System.out.println("Nilai      besi      =
"+BESI[i]+" merupakan golongan "+GOLBESI[i]);
                System.out.println("----------------------
-------");
            }
```

```
            else if(VITB1[i]==999999999.0){
                //penentuan berdasarkan aturan fuzzy
                if(GOLPROTEIN[i]==0){
                    GOLVITB1[i]=0;}
                else if(GOLPROTEIN[i]==1){
                    GOLVITB1[i]=1;}
                else if(GOLPROTEIN[i]==2){
                    GOLVITB1[i]=2;}
                //penentuan  derajat  keanggotaan  dengan
aturan min
                VITB1[i]=PROTEIN[i];
                System.out.println("Nilai  Vitamin  B1  =
"+VITB1[i]+" merupakan golongan "+GOLVITB1[i]);
                System.out.println("-------------------
-------");
            }
        }
        for(int i=0;i<NAMA.length;i++){
            try{
                Class.forName("com.mysql.jdbc.Driver");
                PreparedStatement           b           =
connection.prepareStatement("INSERT INTO fuzzyincomplete "

+ "(id_fincomplete, nama_fincomplete, protein_fincomplete,
lemak_fincomplete,"

+    "    karbohidrat_fincomplete,    kalsium_fincomplete,
fosfor_fincomplete,"

+   "besi_fincomplete,   a_fincomplete,   b1_fincomplete,
c_fincomplete, air_fincomplete) "

+ "values (?,?,?,?,?,?,?,?,?,?,?,?)");
                b.setInt(1, ID[i]);
                b.setString(2, NAMA[i]);
                b.setDouble(3, PROTEIN[i]*100);
                b.setDouble(4, LEMAK[i]*100);
                b.setDouble(5, KARBOHIDRAT[i]*100);
                b.setDouble(6, KALSIUM[i]*100);
                b.setDouble(7, FOSFOR[i]*100);
                b.setDouble(8, BESI[i]*100);
                b.setDouble(9, VITA[i]*100);
                b.setDouble(10, VITB1[i]*100);
                b.setDouble(11, VITC[i]*100);
                b.setDouble(12, AIR[i]*100);
                b.executeUpdate();
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
```

```
            }
        }
public void Defuzzifikasi(){
        String nam = "";
        Double pro, lem, kar, kal, fos, bes, vita, vitb1,
vitc, air;
        String cntj1 = "";
        try{
            String sql = "SELECT count(nama_incomplete)
FROM incomplete";
            Statement             statement             =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            while(rs.next()){
                cntj1                                    =
rs.getString("count(nama_incomplete)");
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,    "Query
Salah " + e);
        }
        int index = Integer.parseInt(cntj1);
        RePROTEIN = new Double[index];
        ReLEMAK = new Double[index];
        ReKARBOHIDRAT = new Double[index];
        ReKALSIUM = new Double[index];
        ReFOSFOR = new Double[index];
        ReBESI = new Double[index];
        ReVITA = new Double[index];
        ReVITB1 = new Double[index];
        ReVITC = new Double[index];
        ReAIR = new Double[index];
        try{
            String  sql  =  "SELECT  protein_incomplete,
lemak_incomplete, karbohidrat_incomplete, "
                    +                "kalsium_incomplete,
fosfor_incomplete,     besi_incomplete,     a_incomplete,
b1_incomplete, c_incomplete, "
                    + "air_incomplete FROM incomplete";
            Statement             statement             =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            int a = 0;
            while(rs.next()){
                pro = rs.getDouble("protein_incomplete");
                lem = rs.getDouble("lemak_incomplete");
                kar                                      =
rs.getDouble("karbohidrat_incomplete");
```

```
                kal = rs.getDouble("kalsium_incomplete");
                fos = rs.getDouble("fosfor_incomplete");
                bes = rs.getDouble("besi_incomplete");
                vita = rs.getDouble("a_incomplete");
                vitb1 = rs.getDouble("b1_incomplete");
                vitc = rs.getDouble("c_incomplete");
                air = rs.getDouble("air_incomplete");
                RePROTEIN[a] = pro;
                ReLEMAK[a] = lem;
                ReKARBOHIDRAT[a] = kar;
                ReKALSIUM[a] = kal;
                ReFOSFOR[a] = fos;
                ReBESI[a] = bes;
                ReVITA[a] = vita;
                ReVITB1[a] = vitb1;
                ReVITC[a] = vitc;
                ReAIR[a] = air;
                a++;
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null,     "Query
Salah " + e);
        }

        for(int i=0; i<NAMA.length;i++){
            //PROTEIN
            if(RePROTEIN[i]==999999999){
                //rendah
                if(GOLPROTEIN[i]==0){
                    if(PROTEIN[i]==1){
                        RePROTEIN[i]=6.8;}
                    else if(PROTEIN[i]==0){
                        RePROTEIN[i]=13.0;}
                    else{
                        RePROTEIN[i]=13-
(6.2*PROTEIN[i]);}
                }
                //sedang
                else if(GOLPROTEIN[i]==1){
                    if(PROTEIN[i]==0){
                        RePROTEIN[i]=6.8;}
                    else{

RePROTEIN[i]=(6.2*PROTEIN[i])+6.8;}
                }
                //tinggi
                else if(GOLPROTEIN[i]==2){
                    if(PROTEIN[i]==1){
```

```
                        RePROTEIN[i]=17.0;}
                    else if(RePROTEIN[i]==0){
                        RePROTEIN[i]=13.0;}
                    else{
                        RePROTEIN[i]=17-(4*PROTEIN[i]);}
                }
            }
            //LEMAK
            else if(ReLEMAK[i]==999999999){
                //rendah
                if(GOLLEMAK[i]==0){
                    if(LEMAK[i]==1){
                        ReLEMAK[i]=5.0;}
                    else if(LEMAK[i]==0){
                        ReLEMAK[i]=16.0;}
                    else{
                        ReLEMAK[i]=16-(11*PROTEIN[i]);}
                }
                //sedang
                else if(GOLLEMAK[i]==1){
                    if(LEMAK[i]==0){
                        ReLEMAK[i]=5.0;}
                    else{
                        ReLEMAK[i]=(11*PROTEIN[i])+5;}
                }
                //tinggi
                else if(GOLLEMAK[i]==2){
                    if(LEMAK[i]==1){
                        ReLEMAK[i]=30.0;}
                    else if(LEMAK[i]==0){
                        ReLEMAK[i]=16.0;}
                    else{
                        ReLEMAK[i]=30-(14*PROTEIN[i]);}
                }
            }
            //KARBOHIDRAT
            else if(ReKARBOHIDRAT[i]==999999999){
                //rendah
                if(GOLKARBOHIDRAT[i]==0){
                    if(KARBOHIDRAT[i]==1){
                        ReKARBOHIDRAT[i]=12.0;}
                    else if(KARBOHIDRAT[i]==0){
                        ReKARBOHIDRAT[i]=30.0;}
                    else{
                        ReKARBOHIDRAT[i]=30-
(18*KARBOHIDRAT[i]);}
                }
                //sedang
                else if(GOLKARBOHIDRAT[i]==1){
```

```
                    if(KARBOHIDRAT[i]==0){
                        ReKARBOHIDRAT[i]=12.0;}
                    else{

ReKARBOHIDRAT[i]=(18*KARBOHIDRAT[i])+12;}
                }
                //tinggi
                else if(GOLKARBOHIDRAT[i]==2){
                    if(KARBOHIDRAT[i]==1){
                        ReKARBOHIDRAT[i]=70.0;}
                    else if(KARBOHIDRAT[i]==0){
                        ReKARBOHIDRAT[i]=30.0;}
                    else{

ReKARBOHIDRAT[i]=(40*KARBOHIDRAT[i])+30;}
                }
            }
            //KALSIUM
            else if(ReKALSIUM[i]==999999999){
                //rendah
                if(GOLKALSIUM[i]==0){
                    if(KALSIUM[i]==1){
                        ReKALSIUM[i]=12.0;}
                    else if(KALSIUM[i]==0){
                        ReKALSIUM[i]=20.0;}
                    else{
                        ReKALSIUM[i]=20-(15*KALSIUM[i]);}
                }
                //sedang
                else if(GOLKALSIUM[i]==1){
                    if(KALSIUM[i]==0){
                        ReKALSIUM[i]=5.0;}
                    else{
                        ReKALSIUM[i]=(15*KALSIUM[i])+5;}
                }
                //tinggi
                else if(GOLKALSIUM[i]==2){
                    if(KALSIUM[i]==1){
                        ReKALSIUM[i]=37.0;}
                    else if(KALSIUM[i]==0){
                        ReKALSIUM[i]=20.0;}
                    else{
                        ReKALSIUM[i]=(17*KALSIUM[i])+20;}
                }
            }
            //FOSFOR
            else if(ReFOSFOR[i]==999999999){
                //rendah
                if(GOLFOSFOR[i]==0){
```

```
                    if(FOSFOR[i]==1){
                        ReFOSFOR[i]=110.0;}
                    else if(FOSFOR[i]==0){
                        ReFOSFOR[i]=170.0;}
                    else{
                        ReFOSFOR[i]=170-(60*FOSFOR[i]);}
                }
                //sedang
                else if(GOLFOSFOR[i]==1){
                    if(FOSFOR[i]==0){
                        ReFOSFOR[i]=110.0;}
                    else{
                        ReFOSFOR[i]=(60*FOSFOR[i])+110;}
                }
                //tinggi
                else if(GOLFOSFOR[i]==2){
                    if(FOSFOR[i]==1){
                        ReFOSFOR[i]=256.0;}
                    else if(FOSFOR[i]==0){
                        ReFOSFOR[i]=170.0;}
                    else{
                        ReFOSFOR[i]=(86*FOSFOR[i])+170;}
                }
            }
            //BESI
            else if(ReBESI[i]==999999999){
                //rendah
                if(GOLBESI[i]==0){
                    if(BESI[i]==1){
                        ReBESI[i]=2.0;}
                    else if(BESI[i]==0){
                        ReBESI[i]=5.7;}
                    else{
                        ReBESI[i]=5.7-(3.7*BESI[i]);}
                }
                //sedang
                else if(GOLBESI[i]==1){
                    if(BESI[i]==0){
                        ReBESI[i]=2.0;}
                    else{
                        ReBESI[i]=(3.7*BESI[i])+2;}
                }
                //tinggi
                else if(GOLBESI[i]==2){
                    if(BESI[i]==1){
                        ReBESI[i]=20.0;}
                    else if(BESI[i]==0){
                        ReBESI[i]=5.7;}
                    else{
```

```
                    ReBESI[i]=(14.3*BESI[i]+5.7);}
            }
        }
        //VITAMIN A
        else if(ReVITA[i]==999999999){
            //rendah
            if(GOLVITA[i]==0){
                if(VITA[i]==1){
                    ReVITA[i]=1000.0;}
                else if(VITA[i]==0){
                    ReVITA[i]=3900.0;}
                else{
                    ReVITA[i]=3900-(2900*VITA[i]);}
            }
            //sedang
            else if(GOLVITA[i]==1){
                if(VITA[i]==0){
                    ReVITA[i]=1000.0;}
                else{
                    ReVITA[i]=(2900*VITA[i])+1000;}
            }
            //tinggi
            else if(GOLVITA[i]==2){
                if(VITA[i]==1){
                    ReVITA[i]=15000.0;}
                else if(VITA[i]==0){
                    ReVITA[i]=3900.0;}
                else{
                    ReVITA[i]=(11100*VITA[i])+3900;}
            }
        }
        //VITAMIN B1
        else if(ReVITB1[i]==999999999){
            //rendah
            if(GOLVITB1[i]==0){
                if(VITB1[i]==1){
                    ReVITB1[i]=0.1;}
                else if(VITB1[i]==0){
                    ReVITB1[i]=0.28;}
                else{
                    ReVITB1[i]=0.28-(0.18*VITB1[i]);}
            }
            //sedang
            else if(GOLVITB1[i]==1){
                if(VITB1[i]==0){
                    ReVITB1[i]=0.1;}
                else{
                    ReVITB1[i]=(0.18*VITB1[i])+0.1;}
            }
```

```
                //tinggi
                else if(GOLVITB1[i]==2){
                    if(VITB1[i]==1){
                        ReVITB1[i]=0.45;}
                    else if(VITB1[i]==0){
                        ReVITB1[i]=0.28;}
                    else{
                        ReVITB1[i]=(0.17*VITB1[i])+0.28;}
                }
            }
            //VITAMIN C
            else if(ReVITC[i]==999999999){
                //rendah
                if(GOLVITC[i]==0){
                    if(VITC[i]==1){
                        ReVITC[i]=10.0;}
                    else if(VITC[i]==0){
                        ReVITC[i]=40.0;}
                    else{
                        ReVITC[i]=40-(30*VITC[i]);}
                }
                //sedang
                else if(GOLVITC[i]==1){
                    if(VITC[i]==0){
                        ReVITC[i]=10.0;}
                    else{
                        ReVITC[i]=(30*VITC[i])+10;}
                }
                //tinggi
                else if(GOLVITC[i]==2){
                    if(VITC[i]==1){
                        ReVITC[i]=80.0;}
                    else if(VITC[i]==0){
                        ReVITC[i]=40.0;}
                    else{
                        ReVITC[i]=(40*VITC[i])+40;}
                }
            }
            //AIR
            else if(ReAIR[i]==999999999){
                //rendah
                if(GOLAIR[i]==0){
                    if(AIR[i]==1){
                        ReAIR[i]=25.0;}
                    else if(AIR[i]==0){
                        ReAIR[i]=55.0;}
                    else{
                        ReAIR[i]=55-(30*AIR[i]);}
                }
```

```
                //sedang
                else if(GOLAIR[i]==1){
                    if(AIR[i]==0){
                        ReAIR[i]=25.0;}
                    else{
                        ReAIR[i]=(30*AIR[i])+25;}
                }
                //tinggi
                else if(GOLAIR[i]==2){
                    if(AIR[i]==1){
                        ReAIR[i]=70.0;}
                    else if(AIR[i]==0){
                        ReAIR[i]=55.0;}
                    else{
                        ReAIR[i]=(15*AIR[i])+55;}
                }
            }
        }
        int a=264;
        for(int i=0;i<NAMA.length;i++){
            try{
                Class.forName("com.mysql.jdbc.Driver");
                PreparedStatement          b          =
connection.prepareStatement("INSERT INTO hasilbiasa "

+  "(id_hasilbiasa,  nama_hasilbiasa,  protein_hasilbiasa,
lemak_hasilbiasa,"

+     "    karbohidrat_hasilbiasa,    kalsium_hasilbiasa,
fosfor_hasilbiasa,"

+    "besi_hasilbiasa,    a_hasilbiasa,    b1_hasilbiasa,
c_hasilbiasa, air_hasilbiasa) "

+ "values (?,?,?,?,?,?,?,?,?,?,?,?)");
                b.setInt(1, a);
                b.setString(2, NAMA[i]);
                b.setDouble(3, RePROTEIN[i]);
                b.setDouble(4, ReLEMAK[i]);
                b.setDouble(5, ReKARBOHIDRAT[i]);
                b.setDouble(6, ReKALSIUM[i]);
                b.setDouble(7, ReFOSFOR[i]);
                b.setDouble(8, ReBESI[i]);
                b.setDouble(9, ReVITA[i]);
                b.setDouble(10, ReVITB1[i]);
                b.setDouble(11, ReVITC[i]);
                b.setDouble(12, ReAIR[i]);
                b.executeUpdate();
                a++;
```

```
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
            }
        }
private                                                 void
FuzzifikasiCompleteActionPerformed(java.awt.event.ActionE
vent evt) {
        Get();
        Fuzzifikasi();
        CariNilai();
        for(int i=0;i<NAMA.length;i++){
            try{
                Class.forName("com.mysql.jdbc.Driver");
                PreparedStatement           b           =
connection.prepareStatement("INSERT INTO hasilfuzzy "

+  "(id_hasilfuzzy,  nama_hasilfuzzy,  protein_hasilfuzzy,
lemak_hasilfuzzy,"

+     "     karbohidrat_hasilfuzzy,     kalsium_hasilfuzzy,
fosfor_hasilfuzzy,"

+    "besi_hasilfuzzy,    a_hasilfuzzy,    b1_hasilfuzzy,
c_hasilfuzzy, air_hasilfuzzy) "

+ "values (?,?,?,?,?,?,?,?,?,?,?,?)");
                b.setInt(1, i);
                b.setString(2, NAMA[i]);
                b.setDouble(3, PROTEIN[i]*1000);
                b.setDouble(4, LEMAK[i]*1000);
                b.setDouble(5, KARBOHIDRAT[i]*1000);
                b.setDouble(6, KALSIUM[i]*1000);
                b.setDouble(7, FOSFOR[i]*1000);
                b.setDouble(8, BESI[i]*1000);
                b.setDouble(9, VITA[i]*1000);
                b.setDouble(10, VITB1[i]*1000);
                b.setDouble(11, VITC[i]*1000);
                b.setDouble(12, AIR[i]*1000);
                b.executeUpdate();
            }catch(Exception e){
                JOptionPane.showMessageDialog(null,
"Perintah Salah : "+e);
            }
        }
        try{
            Object  []  rows={"No","Nama","MF  Protein","MF
Lemak","MF   Karbohidrat","MF   Kalsium","MF   Fosfor","MF
Besi","MF Vit.A","MF Vit.B1","MF Vit.C","MF Air"};
```

```
            DefaultTableModel                       dtm=new
DefaultTableModel(null,rows);
            TAMPIL1.setModel(dtm);
            TAMPIL1.setBorder(null);
            for(int z=0; z<NAMA.length; z++){
                String    []    tampil={String.valueOf(z),
NAMA[z],String.valueOf(PROTEIN[z]),String.valueOf(LEMAK[z
]),

String.valueOf(KARBOHIDRAT[z]),String.valueOf(KALSIUM[z])
,String.valueOf(FOSFOR[z]),

String.valueOf(BESI[z]),String.valueOf(VITA[z]),String.va
lueOf(VITB1[z]),String.valueOf(VITC[z]),
                    String.valueOf(AIR[z])};
                dtm.addRow(tampil);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
private                                                 void
GolonganActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Object                                        []
rows={"No","Nama","Protein","Lemak","Karbohidrat","Kalsiu
m","Fosfor","Besi","Vit.A","Vit.B1","Vit.C","Air"};
            DefaultTableModel                       dtm=new
DefaultTableModel(null,rows);
            TAMPIL2.setModel(dtm);
            TAMPIL2.setBorder(null);
            for(int z=0; z<NAMA.length; z++){
                String    []    tampil={String.valueOf(z),
NAMA[z],String.valueOf(GOLPROTEIN[z]),String.valueOf(GOLL
EMAK[z]),

String.valueOf(GOLKARBOHIDRAT[z]),String.valueOf(GOLKALSI
UM[z]),String.valueOf(GOLFOSFOR[z]),

String.valueOf(GOLBESI[z]),String.valueOf(GOLVITA[z]),Str
ing.valueOf(GOLVITB1[z]),String.valueOf(GOLVITC[z]),
                    String.valueOf(GOLAIR[z])};
                dtm.addRow(tampil);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
```

```
private                                                    void
jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        Defuzzifikasi();
        try{
            Object                                          []
rows={"No","Nama","Protein","Lemak","Karbohidrat","Kalsiu
m","Fosfor","Besi","Vit.A","Vit.B1","Vit.C","Air"};
            DefaultTableModel                      dtm=new
DefaultTableModel(null,rows);
            TAMPIL3.setModel(dtm);
            TAMPIL3.setBorder(null);
            for(int z=0; z<NAMA.length; z++){
                String    []    tampil={String.valueOf(z),
NAMA[z],String.valueOf(RePROTEIN[z]),String.valueOf(ReLEM
AK[z]),

String.valueOf(ReKARBOHIDRAT[z]),String.valueOf(ReKALSIUM
[z]),String.valueOf(ReFOSFOR[z]),

String.valueOf(ReBESI[z]),String.valueOf(ReVITA[z]),Strin
g.valueOf(ReVITB1[z]),String.valueOf(ReVITC[z]),
                    String.valueOf(ReAIR[z])};
                dtm.addRow(tampil);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
```

6. *Training Self-Organizing Fuzzy Maps*

```
package fuzzysom;
import java.awt.Color;
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
public class som extends javax.swing.JFrame {
    private Connection connection;
    private Statement statement;
    int cluster;
    public int[] ID;
    public String[] NAMA;
    public Double[] PROTEIN;
    public Double[] LEMAK;
    public Double[] KARBOHIDRAT;
```

```
    public Double[] KALSIUM;
    public Double[] FOSFOR;
    public Double[] BESI;
    public Double[] VITA;
    public Double[] VITB1;
    public Double[] VITC;
    public Double[] AIR;
    public Double[][] w0;
    public Double[] D;
    public Double[] Dfinal;
    public Double[] C;
    public Double[] Der;
    public int[] Clust;
    public som() {
        initComponents();
        setLocationRelativeTo(this);
        try{
            Class.forName("com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection
("jdbc:mysql://localhost:3306/fuzzysom?zeroDateTimeBehavi
or=convertToNull", "root", "");
            statement = connection.createStatement();
        }
        catch(Exception e){
            JOptionPane.showMessageDialog(null, "Koneksi
gagal");
            INPUTCLUSTER.setEnabled(false);
            INPUTALPHA.setEnabled(false);
            TRAINING.setEnabled(false);
        }
    }
public void Get(){
        String nam = "";
        Double pro, lem, kar, kal, fos, bes, vita, vitb1,
vitc, air;
        String cntj1 = "";
        try{
            String sql = "SELECT count(nama_hasilfuzzy)
FROM hasilfuzzy";
            Statement statement =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            while(rs.next()){
                cntj1 =
rs.getString("count(nama_hasilfuzzy)");
            }
        }catch(SQLException e){
            e.printStackTrace();
```

```
            JOptionPane.showMessageDialog(null, "Query
Salah " + e);
        }
        int index = Integer.parseInt(cntj1);
        ID = new int[index];
        NAMA = new String[index];
        PROTEIN = new Double[index];
        LEMAK = new Double[index];
        KARBOHIDRAT = new Double[index];
        KALSIUM = new Double[index];
        FOSFOR = new Double[index];
        BESI = new Double[index];
        VITA = new Double[index];
        VITB1 = new Double[index];
        VITC = new Double[index];
        AIR = new Double[index];
        try{
            String sql = "SELECT nama_hasilfuzzy,
protein_hasilfuzzy, lemak_hasilfuzzy,
karbohidrat_hasilfuzzy, "
                    + "kalsium_hasilfuzzy,
fosfor_hasilfuzzy, besi_hasilfuzzy, a_hasilfuzzy,
b1_hasilfuzzy, c_hasilfuzzy, "
                    + "air_hasilfuzzy FROM hasilfuzzy";
            Statement statement =
connection.createStatement();
            ResultSet rs = statement.executeQuery(sql);
            int a = 0;
            while(rs.next()){
                nam = rs.getString("nama_hasilfuzzy");
                pro = rs.getDouble("protein_hasilfuzzy");
                lem = rs.getDouble("lemak_hasilfuzzy");
                kar =
rs.getDouble("karbohidrat_hasilfuzzy");
                kal = rs.getDouble("kalsium_hasilfuzzy");
                fos = rs.getDouble("fosfor_hasilfuzzy");
                bes = rs.getDouble("besi_hasilfuzzy");
                vita = rs.getDouble("a_hasilfuzzy");
                vitb1 = rs.getDouble("b1_hasilfuzzy");
                vitc = rs.getDouble("c_hasilfuzzy");
                air = rs.getDouble("air_hasilfuzzy");
                ID[a] = a;
                NAMA[a] = nam;
                PROTEIN[a] = pro;
                LEMAK[a] = lem;
                KARBOHIDRAT[a] = kar;
                KALSIUM[a] = kal;
                FOSFOR[a] = fos;
                BESI[a] = bes;
```

```
                VITA[a] = vita;
                VITB1[a] = vitb1;
                VITC[a] = vitc;
                AIR[a] = air;
                a++;
            }

System.out.println("================================ NAMA
BAHAN PANGAN ================================");
            for (int i = 0; i < index; i++) {
                System.out.println(NAMA[i]);
            }
            System.out.println();

System.out.println("================================
PROTEIN ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("PROTEIN[" + i + "] =>
Protein " + NAMA[i] + " : " + PROTEIN[i]);
            }
            System.out.println();

System.out.println("================================
LEMAK ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("LEMAK[" + i + "] =>
Lemak " + NAMA[i] + " : " + LEMAK[i]);
            }
            System.out.println();

System.out.println("================================
KARBOHIDRAT ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("KARBOHIDRAT[" + i +
"] => Karbohidrat " + NAMA[i] + " : " + KARBOHIDRAT[i]);
            }
            System.out.println();

System.out.println("================================
KALSIUM ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("KALSIUM[" + i + "] =>
Kalsium " + NAMA[i] + " : " + KALSIUM[i]);
            }
            System.out.println();

System.out.println("================================
FOSFOR ================================");
            for (int i = 0; i < index; i++) {
```

```
                System.out.println("FOSFOR[" + i + "] =>
Fosfor " + NAMA[i] + " : " + FOSFOR[i]);
            }
            System.out.println();

System.out.println("================================ BESI
================================");
            for (int i = 0; i < index; i++) {
                System.out.println("BESI[" + i + "] =>
Besi " + NAMA[i] + " : " + BESI[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN A ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINA[" + i + "]
=> Vitamin A " + NAMA[i] + " : " + VITA[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN B1 ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINB1[" + i + "]
=> Vitamin B1 " + NAMA[i] + " : " + VITB1[i]);
            }
            System.out.println();

System.out.println("================================
VITAMIN C ================================");
            for (int i = 0; i < index; i++) {
                System.out.println("VITAMINC[" + i + "]
=> Vitamin C " + NAMA[i] + " : " + VITC[i]);
            }
            System.out.println();

System.out.println("================================ AIR
================================");
            for (int i = 0; i < index; i++) {
                System.out.println("AIR[" + i + "] => Air
" + NAMA[i] + " : " + AIR[i]);
            }
        }catch(SQLException e){
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Query
Salah " + e);
        }
    }
```

```
public void SOM(){
        cluster =
Integer.parseInt(INPUTCLUSTER1.getText());
        Double alpha =
Double.parseDouble(INPUTALPHA.getText());
        Double epsilon =
Double.parseDouble(INPUTITERASI.getText());

        w0 = new Double[cluster][10];
        for(int i=0; i<cluster; i++){
            for(int j=0; j<10; j++){
                w0[i][j]= Math.random();
                System.out.println("nilai awal ===>
w["+i+"]["+j+"] : "+w0[i][j]);
            }
        }
        D = new Double[cluster];
        int x=0;
        while (alpha >= epsilon){
            System.out.println();
            System.out.println();

System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~ITERASI
"+x+"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~");
            x++;
            for(int z=0; z<NAMA.length; z++){
                System.out.println();
                System.out.println();

System.out.println("================================
Untuk vektor ke-"+z+"================================");
                System.out.println();
                for(int i=0; i<cluster; i++){
                    D[i]=(Math.pow((w0[i][0]-
PROTEIN[z]),2))+(Math.pow((w0[i][1]-LEMAK[z]),2))
                            +(Math.pow((w0[i][2]-
KARBOHIDRAT[z]),2))+(Math.pow((w0[i][3]-KALSIUM[z]),2))
                            +(Math.pow((w0[i][4]-
FOSFOR[z]),2))+(Math.pow((w0[i][5]-BESI[z]),2))
                            +(Math.pow((w0[i][6]-
VITA[z]),2))+(Math.pow((w0[i][7]-VITB1[z]),2))
                            +(Math.pow((w0[i][8]-
VITC[z]),2))+(Math.pow((w0[i][9]-AIR[z]),2));
                    System.out.println("Nilai D["+i+"] :
"+D[i]);
                }
                int ind=0;
```

```
                Double min = D[0];
                for(int b=0; b<cluster; b++){
                    if (D[b]< min){
                        min = D[b];
                        ind = b;
                    }
                }
                w0[ind][0]=w0[ind][0]+(alpha*(PROTEIN[z]-
w0[ind][0]));
                w0[ind][1]=w0[ind][1]+(alpha*(LEMAK[z]-
w0[ind][1]));

w0[ind][2]=w0[ind][2]+(alpha*(KARBOHIDRAT[z]-
w0[ind][2]));
                w0[ind][3]=w0[ind][3]+(alpha*(KALSIUM[z]-
w0[ind][3]));
                w0[ind][4]=w0[ind][4]+(alpha*(FOSFOR[z]-
w0[ind][4]));
                w0[ind][5]=w0[ind][5]+(alpha*(BESI[z]-
w0[ind][5]));
                w0[ind][6]=w0[ind][6]+(alpha*(VITA[z]-
w0[ind][6]));
                w0[ind][7]=w0[ind][7]+(alpha*(VITB1[z]-
w0[ind][7]));
                w0[ind][8]=w0[ind][8]+(alpha*(VITC[z]-
w0[ind][8]));
                w0[ind][9]=w0[ind][9]+(alpha*(AIR[z]-
w0[ind][9]));

            System.out.println("Input paling dekat dengan
node output : "+ind);
            System.out.println();
            System.out.println("alpha "+z+" : "+alpha);

            System.out.println();
            for(int j=0; j<10; j++){
                System.out.println();
                for(int i=0; i<cluster; i++){
                    System.out.print("w["+i+"]["+j+"]:
"+w0[i][j]);
                    System.out.print("         ");
                }
            }
            }
            alpha = 0.85*alpha;
          }
        ITERASI.setText(String.valueOf(x));
    }
public void Cluster(){
```

```
        Double kecil;
        C = new Double[cluster];
        Clust = new int[NAMA.length];
        Dfinal = new Double[cluster];
        for(int z=0; z<NAMA.length; z++){
            System.out.println();
                System.out.println();

System.out.println("================================
Untuk vektor ke-"+z+"================================");
                System.out.println();
            for(int i=0; i<cluster; i++){
                C[i]=(Math.pow((w0[i][0]-
PROTEIN[z]),2))+(Math.pow((w0[i][1]-LEMAK[z]),2))
                              +(Math.pow((w0[i][2]-
KARBOHIDRAT[z]),2))+(Math.pow((w0[i][3]-KALSIUM[z]),2))
                              +(Math.pow((w0[i][4]-
FOSFOR[z]),2))+(Math.pow((w0[i][5]-BESI[z]),2))
                              +(Math.pow((w0[i][6]-
VITA[z]),2))+(Math.pow((w0[i][7]-VITB1[z]),2))
                              +(Math.pow((w0[i][8]-
VITC[z]),2))+(Math.pow((w0[i][9]-AIR[z]),2));
                    System.out.println("Nilai C["+i+"] :
"+C[i]);
            }
            int index=0;
            kecil = C[0];
            for(int t=0; t<cluster; t++){
                if (C[t]< kecil){
                        kecil = C[t];
                        index = t;
                }
            }
            Clust[z]=index;
            System.out.println("Nilai D minimum :
"+kecil);
            System.out.println();
            System.out.println("Jadi, "+NAMA[z]+" masuk
kedalam cluster "+index);
            System.out.println();
        }
    }
private void
TRAININGActionPerformed(java.awt.event.ActionEvent evt) {
        int clus=0;
        Double alpha=0.0, ite=0.0;

if((INPUTCLUSTER1.getText()).equals("")||(INPUTALPHA.getT
ext()).equals("")||(INPUTITERASI.getText()).equals("")){
```

```
            JOptionPane.showMessageDialog(null, "Data
yang diisikan belum lengkap");
        }
        else{

clus=Integer.parseInt(INPUTCLUSTER1.getText());

alpha=Double.parseDouble(INPUTALPHA.getText());

ite=Double.parseDouble(INPUTITERASI.getText());

if(clus<1||clus>287||alpha<=0||alpha>=1||ite>=alpha||ite<
0){
                JOptionPane.showMessageDialog(null, "Data
yang diisikan salah");
            }
            else{
                Get();
                SOM();
                Cluster();
            }
        }
private void
CLUSTERActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Object [] rows={"No","Nama","Cluster"};
            DefaultTableModel dtm=new
DefaultTableModel(null,rows);
            TABEL.setModel(dtm);
            TABEL.setBorder(null);
            for(int z=0; z<NAMA.length; z++){
                String [] tampil={String.valueOf(z),
NAMA[z],String.valueOf(Clust[z])};
                dtm.addRow(tampil);
            }
        }catch(Exception e){
            e.printStackTrace();
        }
    }
```

BIODATA PENULIS

Penulis bernama lengkap **Tara Amila Milatina**, lahir di Gresik, 10 November 1995. Anak kedua dari pasangan Agus Setiyono dan Nanik Sri Suwarni, serta memiliki kakak perempuan Ellisa Kusuma Dewi dan adik laki-laki Hariz Athhar Yattaqi. Penulis mengikuti pendidikan dasar dari Sekolah Dasar hingga Sekolah Menegah Atas di Kota Gresik. Penulis menempuh pendidikan di SD Negeri Pongangan 2, SMP Negeri 1 Gresik, dan SMA Negeri 1 Gresik. Setelah Lulus dari SMAN 1 Gresik pada tahun 2013 yang lalu, penulis melanjutkan pendidikan tingginya di Institut Teknologi Sepeluh Nopember (ITS) Surabaya dengan mengambil Jurusan Matematika dengan bidang minat Ilmu Komputer. Selama mengikuti perkuliahan di ITS, penulis turut aktif dalam beberapa kegiatan kemahasiswaan sebagai Pemandu FMIPA ITS, staff Departemen Sains, Teknologi, dan Keprofesian Himatika ITS Periode 2014/2015,dan Head of Applied Science Department Himatika ITS Periode 2015/2016. Selain aktif dalam beberapa kegiatan kemahasiswaan, penulis juga mengikuti Kerja Praktek PT.Petrokimia Gresik dan ditempatkan di Departemen Tekinfo. Informasi lebih lanjut mengenai Tugas Akhir ini dapat ditujukan ke penulis melalui email: taraamila@gmail.com..