IMPLEMENTASI ALGORITMA *FUZZY C-MEANS CLUSTERING* UNTUK PEMBANGKITAN ATURAN *FUZZY* PADA DETEKSI DINI RISIKO PENYAKIT STROKE

SKRIPSI

LABORATORIUM
KOMPUTASI CERDAS DAN VISUALISASI

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

SHINTA AYU VALENSIA

NIM. 105090601111013

KEMENTERIAN RISET TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS BRAWIJAYA
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER
MALANG
2015

LEMBAR PERSETUJUAN

IMPLEMENTASI ALGORITMA *FUZZY C-MEANS CLUSTERING* UNTUK PEMBANGKITAN ATURAN *FUZZY* PADA DETEKSI DINI RISIKO PENYAKIT STROKE

SKRIPSI

KONSENTRASI KOMPUTASI CERDAS DAN VISUALISASI

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

SHINTA AYU VALENSIA

NIM. 105090601111013

Skripsi ini telah disetujui oleh dosen pembimbing pada tanggal 2 Januari 2015

Dosen Pembimbing I	**Dosen Pembimbing II**
Rekyan Regasari Mardi Putri, ST., MT.	**Novanto Yudistira, S.Kom., M.Sc**
NIK. 77041406120253	**NIP. 831110 16 1 1 0425**

LEMBAR PENGESAHAN

IMPLEMENTASI ALGORITMA *FUZZY C-MEANS CLUSTERING* UNTUK PEMBANGKITAN ATURAN *FUZZY* PADA DETEKSI DINI RISIKO PENYAKIT STROKE

SKRIPSI

LABORATORIUM KOMPUTASI CERDAS DAN VISUALISASI

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

SHINTA AYU VALENSIA

NIM. 105090601111013

Skripsi ini telah diuji dan dinyatakan lulus
pada tanggal 29 Januari 2015

Penguji I	**Penguji II**
Budi Darma Setiawan, S.Kom., M.Cs.	**Indriati, ST., M.Kom**
NIP. 198410152014041002	**NIK. 83101306120035**

Penguji III

Dr. Eng. Fitri Utaminingrum.,S.T,M.T
NIP. 19820710 200812 2 001

Mengetahui,
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Marji., M.T.
NIP. 19670801 199203 1 001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama	: Shinta Ayu Valensia
NIM	: 105090601111013
Program Studi	: Informatika / Ilmu Komputer
Jurusan	: Ilmu Komputer
Fakultas	: Program Teknologi Informasi dan Ilmu Komputer

Dengan ini menyatakan bahwa:

1. Isi dari skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam skripsi ini.
2. Apabila di kemudian hari ternyata skripsi yang saya tulis terbukti hasil jiplakan, maka saya bersedia menanggung segala resiko yang akan saya terima.

Demikian pernyataan ini dibuat dengan segala kesadaran dan penuh tanggung jawab dan digunakan sebagaimana mestinya.

Malang, 30 Januari 2015

Yang menyatakan,

Shinta Ayu Valensia

NIM.105090601111013

KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadirat Allah SWT, karena hanya dengan rahmat dan bimbingannya Penulis dapat menyelesaikan skripsi dengan judul "Implementasi Algoritma *Fuzzy C-Means Clustering* Untuk Pembangkitan Aturan *Fuzzy* Pada Deteksi Dini Risiko Penyakit Stroke" dengan baik.

Tidak dapat dipungkiri bahwa tidak mungkin penulis dapat menyelesaikan skripsi ini tanpa dukungan serta bantuan dari berbagai pihak. Penulis tidak lupa untuk menyampaikan penghargaan lewat rasa hormat dan terimakasih kepada semua pihak yang telah ikut andil dan berkontribusi, baik dalam bentuk bantuan tenaga, pikiran maupun dukungan moral selama penulisan skripsi sehingga dapat terselesaikannya skripsi ini. Pihak-pihak tersebut antara lain:

1. Rekyan Regasari Mardi Putri, ST., MT., dan Novanto Yudistira, S.Kom., M.Sc selaku dosen pembimbing I dan dosen pembimbing II yang telah banyak memberikan bimbingan, ilmu dan saran dalam penyusunan skripsi ini.
2. Orang tua Penulis, Drs. Moh. Syadeli, Ak., MM., dan Dra. Anis Sulistiani, serta kedua saudara Penulis, Moh. Fahrur Rizal dan Fachrial Zainuddin Ashari yang telah memberi semangat, motivasi, kasih sayang serta dukungan moril dan materil kepada Penulis.
3. Nurul Hidayat, S.Pd., MSc., selaku dosen pembimbing akademik yang telah memberikan bimbingan, ilmu, saran, motivasi selama Penulis nemempuh pendidikan.
4. Drs. Marji, M.T., dan Issa Arwani, ST., MT., selaku Ketua dan Sekretaris Program Studi Informatika serta segenap Bapak/Ibu Dosen, Staff Administrasi dan Perpustakaan Program Studi Teknik Informatika Universitas Brawijaya.
5. Ahmad Fashel Sholeh, S.Kom, Ely Ratna Sayekti, S.Kom, Fayruz Al-Baity, S.Kom, dan Dr. Farhad Bal'afif, Sp.BS sebagai mitra penulis dalam merancang skripsi.
6. Hady Rasikhun, Amelia Tri, Retania Astia, Sheila Silvalanda, Sholichah Isti, Anjar Dwi, Farah Bahtera, Dita Oktaria, Indah Kurnia, Monica Intan,

Meitasari dan semua teman-teman Ilkom 2010 terimakasih atas segala bantuan dan dukungannya

7. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Sebagai manusia biasa, penulis sadar bahwa begitu banyak kekurangan entah itu disengaja ataupun dari kebelummampuan penulis dalam menulis skripsi ini, oleh karena itu saran dan kritik dengan nuansa positif yang bertujuan membangunlah yang sangat dibutuhkan oleh penulis untuk memperbaiki skripsi ini agar kedepannya lebih baik. Penulis berharap skripsi ini dapat bermanfaat bagi penulis sendiri dan bagi semua pihak. Akhirnya penulis berharap agar skripsi ini dapat memberikan sumbangan dan manfaat bagi semua pihak yang berkepentingan.

Malang, Januari 2015

Penulis

ABSTRAK

Shinta Ayu Valensia. 2015. Implementasi Algoritma *Fuzzy C-Means Clustering* Untuk Pembangkitan Aturan *Fuzzy* Pada Deteksi Dini Risiko Stroke. Skripsi Program Studi Ilmu Komputer, Program Teknologi Informasi Dan Ilmu Komputer Universitas Brawijaya. Pembimbing : Rekyan Regasari Mardi Putri, ST., MT. dan Novanto Yudistira, S.Kom., M.Sc

Penyakit stroke merupakan penyebab kecacatan fisik pertama dan penyebab kematian ketiga (setelah jantung dan kanker) di dunia. Seseorang yang berpotensi terkena penyakit stroke dapat mengurangi potensi tersebut apabila mengetahui faktor risiko yang menyebabkan stroke, serta menanggulanginya sejak dini. Beberapa faktor risiko utama yang dapat mengakibatkan penyakit stroke yaitu tekanan darah, umur, jenis kelamin, kolesterol, serta riwayat diabetes. Bagi para ahli terkadang sulit untuk mendiagnosis penyakit stroke karena adanya beberapa faktor risiko yang mempengaruhi. Oleh karena itulah diperlukan suatu metode inferensi sebagai pendukung keputusan, apakah seseorang tersebut berisiko terkena penyakit stroke atau tidak. Aturan dari seorang pakar dapat digantikan dengan menggunakan algoritma *clustering*. Dalam penelitian ini diimplementasikan algoritma *fuzzy c-means clustering* sebagai media pembelajaran untuk membangkitkan aturan *fuzzy*.

Penelitian ini menggunakan data rekam medik dari rumah sakit XYZ yang berasal dari penelitian sebelumnya. Data rekam medik terdiri 109 data latih dan 30 data uji. Proses pelatihan diawali dengan proses clustering, dari hasil clustering kemudian dilakukan analisa varian. Hasil cluster dengan nilai varian terkecil akan dijadikan bahan untuk proses ekstraksi aturan *fuzzy*. Setelah aturan terbentuk barulah dilakukan proses pengujian menggunakan sistem inferensi *fuzzy* model sugeno orde-satu. Dari pengujian jumlah cluster dihasilkan jumlah cluster ideal yang terpilih, yaitu 3 cluster. Sedangkan untuk pengujian akurasi didapatkan akurasi sistem sebesar 87,33% dengan akurasi tertinggi sebesar 93,33% pada jumlah aturan 3.

Kata kunci : Deteksi Stroke, *Fuzzy C-Means, Clustering,* Aturan Fuzzy, *Fuzzy Inference System* Sugeno

ABSTRACT

Shinta Ayu Valensia. 2015. ***Implementation of Fuzzy C-Means Clustering Algorithm for Fuzzy Rules Extraction in Early Detection The Risk of Stroke*****. Skripsi Program Studi Ilmu Komputer, Program Teknologi Informasi Dan Ilmu Komputer Universitas Brawijaya.** ***Advisor*** **: Rekyan Regasari Mardi Putri, ST., MT. and Novanto Yudistira, S.Kom., M.Sc**

Stroke is a disease the number one cause of disability and the number three cause of death (after heart disease and cancer) in the world. People potentially affected by a stroke can be avoided if he is aware of it and can overcome the risk factor early. Some of the main risk factors that affect stroke are blood pressure, age, sex, colesterol, and history of diabetes.It is sometimes difficult for experts to diagnose the Stroke because of the so many risk factors. Therefore it is necessary to have the inference method to support the decision whether someone is affected stroke-risk or not. The rules from the experts can be replaced by using clustering algorithm. In this paper, fuzzy c-means clustering algorithm has been implemented as learning media to generate of fuzzy rules.

This paper used the medical records in XYZ hospital were derived from previous research. It is consist of 109 training data and 30 testing data. Generating rules begins with the process of clustering and then analyzed variance. The cluster results with the smallest value of the varianceused in the extraction of fuzzy rules. The smaller the value of the variance of a cluster, more ideal it is. The rules taken from subtractive clustering algorithm is combined with Takagi Sugeno Kang inference model first-order. The ideal number of clusters obtained is 3 clusters. Meanwhile accuracy test resulted in 87,33% of accuracy and the highest number of accuracy is 93,33% with 3 rules extracted.

Keywords *: Stroke Detection, Fuzzy C-Means, Clustering, Fuzzy Rule, Fuzzy Inference System Sugeno.*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR *SOURCE CODE*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Stroke adalah penyakit yang berhubungan dengan sistem pembuluh darah pada otak. Di dunia, penyakit stroke menjadi pembunuh ketiga setelah penyakit jantung dan kanker. Namun dalam segi penyebab kecacatan, penyakit stroke menjadi yang tertinggi di dunia [YAN-11].

Berdasarkan data di lapangan, angka kejadian stroke meningkat secara drastis seiring dengan pertambahan usia. Setiap penambahan usia 10 tahun sejak usia 35 tahun, risiko seseorang terserang penyakit stroke meningkat sebanyak dua kali lipat dan sekitar 5% orang yang berusia diatas 65 tahun pernah mengalami setidaknya satu kali stroke [YAS-11].

Saat ini Indonesia menduduki urutan pertama penderita stroke terbanyak di Asia. Berdasarkan data dari laporan *Institute of Health Metrics and Evaluation* 2013, penyakit stroke berada pada urutan pertama dari 10 penyakit tertinggi di Indonesia [IHM-13]. Hipertensi, yang menjadi faktor risiko utama stroke di Indonesia telah mengalami peningkatan sebesar 95%. Dari hal tersebut para ahli epidemiologi meramalkan bahwa dimasa mendatang akan ada sekitar 12 juta penduduk Indonesia yang berusia diatas 35 tahun akan mempunyai potensi terkena penyakit stroke [YAS-11].

Sebagian besar orang yang pernah mengalami stroke ringan, justru tidak menyadari bahwa dirinya terkena penyakit stroke. Hal ini dikarenakan gejala dari stroke ringan tersebut tidak mengakibatkan perubahan fungsi panca indera, keseimbangan tubuh, serta kemampuan berfikir. Padahal jika stroke ringan terjadi berulang kali, maka perubahan-berubahan tersebut dapat dialami oleh penderita [SHO-13]. Untuk pencegahan terjadinya stroke berat yang dapat mengakibatkan kelumpuhan, sebaiknya mengetahui faktor risiko dari penyakit stroke sejak dini. Beberapa faktor risiko utama yang dapat mengakibatkan penyakit stroke adalah tekanan darah, umur, jenis kelamin, kolesterol jahat (LDL), serta riwayat diabetes. Sebagian besar faktor risiko tersebut dapat dicegah dengan mengubah gaya hidup menjadi lebih baik, seperti tidak merokok, tidak mengkonsumsi alkohol, serta

menjaga pola makan dengan makan makanan yang sehat dan bergizi [YAS-11]. Bagi para ahli terkadang tidak mudah untuk mendiagnosis penyakit stroke karena adanya beberapa faktor risiko yang mempengaruhi. Oleh karena itulah diperlukan suatu metode inferensi sebagai pendukung keputusan, apakah seseorang tersebut berisiko terkena penyakit stroke atau tidak.

Fuzzy Inference System merupakan salah satu *alternative* yang dapat digunakan untuk menyelesaikan persoalan analisa risiko penyakit stroke.Logika *fuzzy* merupakan logika yang memiliki nilai kekaburan atau kesamaran yang digunakan untuk melakukan penalaran [KUP-10].

Penelitian tentang deteksi dini risiko penyakit stroke ini pernah dilakukan oleh Ahmad Fashel Sholeh pada tahun 2012 dengan menggunakan metode logika *fuzzy* mamdani. Hasil akurasi tertinggi yang didapat dari penelitian ini adalah 82,98% [SHO-13]. Metode mamdani ini lebih menyerupai pola pikir manusia. Berdasarkan kelemahan akan keakuratan aturan yang berasal dari pakar, maka diperlukan media pembelajaran dalam membangkitkan aturan. Aturan yang berasal dari pakar dapat digantikan dengan menggunakan algoritma *cluster*ing seperti FCM, *k-means, subtractive clustering* dan *nearest neighbourhood clustering* [AKH-10].

Suatu aturan dapat dibangkitkan secara otomatis melalui suatu algoritma tertentu berdasarkan data yang disediakan pakar [ARA-10]. Aturan *fuzzy* dapat diekstraksi dengan menggunakan beberapa teknik, seperti jaringan saraf tiruan, algoritma genetika, dan *clustering*. Salah satu algoritma *clustering* adalah *Fuzzy C-Means* (FCM). *Fuzzy c-means* merupakan adaptasi dari algoritma *k-means* dengan fungsi keanggotaan halus. *Fuzzy c-means* memungkinkan suatu titik data menjadi bagian untuk semua pusat [ABA-03]. Maka dari itulah, *fuzzy c-means* dapat digunakan untuk membangkitkan aturan *fuzzy* dari sekumpulan data. Aturan yang dihasilkan oleh *fuzzy c-means* kemudian digunakan untuk deteksi dini risiko penyakit stroke menggunakan sistem inferensi *fuzzy*. Sistem inferensi *fuzzy* yang digunakan adalah sugeno orde satu. Pemilihan metode sugeno orde satu dikarenakan konsekuen (*output*) berupa kumpulan konstanta [KUS-10:46]. Sedangkan untuk menghitung nilai derajat keanggotaan pada masing-masing

aturan dapat menggunakan fungsi *gauss* dengan bantuan nilai pusat *cluster* dan standar deviasi [KUS-10:152].

Selain itu juga penelitian mengenai pembangkitan aturan secara otomatis juga pernah dilakukan oleh Ely Ratna Sayekti pada tahun 2013. Pada penelitiannya, Ely Ratna Sayekti menggunakan metode *fuzzy c-means* sebagai media pembelajaran dalam membangkitkan aturan. Aturan yang dihasilkan dari *fuzzy c-means*, selanjutnya digunakan untuk melakukan pengelompokan tingkat risiko penyakit kanker payudara menggunakan sistem inferensi *fuzzy* sugeno orde-satu. Dari hasil penelitian didapatkan akurasi tertinggi sebesar 87% [SAY-13].

Berdasarkan latar belakang yang telah diuraikan, maka akan diimplementasikan algoritma fuzzy c-means *cluster*ing untuk pembangkitan aturan fuzzy pada deteksi dini resiko penyakit stroke dengan menggunakan sistem inferensi fuzzy metode sugeno orde-satu. Maka dari itu skripsi ini berjudul "**IMPLEMENTASI ALGORITMA *FUZZY C-MEANS CLUSTER*ING UNTUK PEMBANGKITAN ATURAN *FUZZY* PADA DETEKSI DINI RISIKO PENYAKIT STROKE**".

1.2 Rumusan Masalah

Adapun permasalahan yang dapat dirumuskan dari latar belakang adalah sebagai berikut :

1) Bagaimana menerapkan algoritma *fuzzy c-means* (FCM) untuk membangkitkan aturan *fuzzy* pada deteksi dini risiko penyakit stroke?
2) Berapa jumlah *cluster* ideal yang terbentuk berdasarkan perhitungan nilai batasan varian?
3) Bagaimanakah nilai akurasi dari hasil sistem inferensi *fuzzy* sugeno berdasarkan pembangkitan aturan *fuzzy c-means* (FCM) untuk pengujian data deteksi dini risiko penyakit stroke?

1.3 Batasan Masalah

Berdasarkan rumusan masalah, penulis perlu memberikan batasan masalah sebagai berikut :

1) Data yang digunakan pada skripsi ini berasal dari data penelitian yang dilakukan oleh Ahmad Fashel Sholeh pada tahun 2012.

2) Jenis data pada penelitian ini adalah rekam medik dari Rumah Sakit XYZ.
3) *Fuzzy Inference System* yang digunakan adalah model Sugeno orde satu dengan aturan yang dibangkitkan oleh algoritma *c-means clustering.*
4) Tingkat akurasi dan kinerja sistem yang akan diimplementasikan, yaitu kesesuaian hasil aturan *fuzzy* yang dibangkitkan oleh algoritma *c-means clustering* terhadap data uji yang berasal dari pakar.

1.4 Tujuan

Tujuan dari penelitian ini, antara lain:

1) Mengimplementasikan algoritma *fuzzy c-means* (FCM) untuk membangkitakan aturan *fuzzy* pada deteksi dini risiko penyakit stroke.
2) Mengetahui jumlah *cluster* ideal yang terbentuk berdasarkan perhitungan nilai batasan varian.
3) Mengetahui akurasi dari hasil sistem inferensi *fuzzy* sugeno berdasarkan pembangkitan aturan *fuzzy c-means* (FCM) untuk pengujian data deteksi dini risiko penyakit stroke.

1.5 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah untuk membantu pihak medis dalam menentukan tingkat risiko stroke pasien, sehingga dapat dilakukan penanganan yang sesuai untuk mengurangi tingkat risiko stroke tersebut. Dengan penanganan yang tepat dan cepat diharapkan dapat mengurangi angka pasien penderita penyakit stroke.

1.6 Sistematika Penulisan

Pembuatan hasil penelitian yang didokumentasikan dalam bentuk skripsi ini berdasarkan sistematika penulisan sebagai berikut:

1) **BAB I PENDAHULUAN**

   Bab pendahuluan ini menguraikan tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, dan sistematika penulisan dari penelitian ini.

2) **BAB II TINJAUAN PUSTAKA**

   Bab tinjauan pustaka berisi kajian pustaka dan dasar teori yang mendukung penelitian ini. Kajian pustaka membahas penelitian yang telah

ada dan yang diusulkan. Sedangkan dasar teori merupakan teori yang diperlukan dalam proses penyusunan penelitian yang diusulkan.

3) **BAB III METODOLOGI DAN PERANCANGAN SISTEM**

Pada bab metodologi dan perancangan sistem, dijelaskan metode dan langkah kerja yang dilakukan dalam proses pengerjaan penelitian serta proses perancangan sistem yang dibangun, seperti perancangan sistem, perhitungan manual, perancangan *database*, dan perancangan antarmuka.

4) **BAB IV IMPLEMENTASI**

Bab ini berisi spesifikasi perangkat keras dan perangkat lunak, implementasi algoritma *fuzzy c-means clustering* (FCM) dan implementasi antarmuka untuk pembangkitan aturan *fuzzy* pada deteksi dini risiko penyakit stroke.

5) **BAB V PENGUJIAN DAN ANALISIS**

Bab ini membahas tentang proses pengujian sistem untuk memastikan bahwa program sudah sesuai dengan tahap perancangan dan disertai analisisnya.

6) **BAB VI PENUTUP**

Bab penutup merupakan bab terakhir yang berisi kesimpulan dan saran. Kesimpulan didasarkan atas pengujian dan analisis yang dilakukan dalam proses penelitian. Saran berisi rekomendasi apa saja yang perlu dikembangkan kedepannya.

BAB II
TINJAUAN PUSTAKA

Bab tinjauan pustaka ini berisi kajian pustaka dan dasar teori yang berhubungan dengan penelitian mengenai implementasi algoritma *fuzzy c-means clustering* (FCM) untuk pembangkitan aturan *fuzzy* pada deteksi dini risiko penyakit stroke.

2.1 Kajian Pustaka

Kajian pustaka yang digunakan dalam penelitian ini membahas tentang penelitian sebelumnya mengenai implementasi algoritma *fuzzy c-means cluster*ing untuk pembangkitan aturan *fuzzy* pada pengelompokan tingkat risiko penyakit kanker payudara dan aplikasi pendukung keputusan untuk deteksi dini risiko penyakit stroke menggunakan logika *fuzzy* mamdani.

Penelitian yang dilakukan oleh Ely Ratna Sayekti dari Universitas Brawijaya menggunakan metode *fuzzy c-means* sebagai pembelajaran dalam pembangkitan aturan dan menggunakan sugeno orde-satu sebagai metode inferensi *fuzzy*. Objek penelitiannya adalah data mamografi (*Mammographic mass data set*). Gabungan dari kedua metode tersebut dapat digunakan untuk pengelompokan tingkat risiko penyakit kanker payudara. Akurasi tertinggi sebesar 84% dengan kombinasi kelas ganas 50% dan kelas jinak 50% [SAY-14].

Ahmad Fashel Sholeh dari Institut Teknologi Sepuluh Nopember (ITS) telah melakukan penelitian dengan menggunakan logika *fuzzy* mamdani untuk deteksi dini risiko penyakit stroke. Dalam penelitian ini tingkat risiko terserang penyakit stroke dibagi menjadi 3 kelas, yaitu tinggi, sedang, dan rendah. Tingkat akurasi yang dihasilkan dari penelitian ini sebesar 82,98% dengan potensi kesalahan tertinggi sebesar 23,08% pada kelas sedang. Prosentase tingkat kesalahan yang cukup tinggi ini disebabkan karena keberadaan rentan nilai variabel keluaran kelas sedang lebih pendek daripada rentan nilai variabel keluaran kelas lainnya [SHO-12].

2.2. Stroke

Penyakit stroke merupakan penyakit yang berhubungan dengan sistem pembuluh darah pada otak. Di dunia, penyakit stroke menjadi pembunuh ketiga setelah penyakit jantung dan kanker. Namun dalam segi penyebab kecatatan, penyakit stroke menjadi yang tertinggi di dunia [YAN-11].

Berdasarkan data dilapangan, angka terjadinya penyakit stroke meningkat secara drastis seiring usia. Setiap penambahan 10 tahun dari usia 35 tahun, terjadi peningkatan risiko terkena penyakit stroke sebanyak dua kali lipat. Sekitar 5% orang yang berusia diatas 65 tahun pernah mengalami stroke setidaknya sebanyak satu kali [YAS-11].

Penyakit stroke adalah serangan otak yang timbul secara mendadak dimana terjadi gangguan fungsi otak sebagian atau menyeluruh sebagai akibat dari adanya gangguan aliran darah karena sumbatan atau pecahnya pembuluh darah di otak, sehingga menyebabkan sel-sel otak kekurangan oksigen, darah, atau zat-zat makanan. Penyumbatan pembuluh darah otak diakibatkan adanya gumpalan darah yang menyumbat jalannya darah atau penyempitan pembuluh darah. Sedangkan pecahnya pembuluh darah disebabkan adanya tekanan darah yang sangat tinggi dan darah masuk kedalam jaringan otak. Pada akhirnya kedua gangguan aliran darah ini akan menyebabkan terjadinya kematian sel-sel otak dalam waktu relatif singkat [YAS-11].

Gejala awal dari penyakit stroke dapat dilihat dari suatu penilaian sederhana yang dikenal dengan singkatan *FAST* (*Face, Arms drive, Speech and Three of signs*), yaitu [YAS-11]:

- F = *Face* (Wajah)

  Wajah dari penderita penyakit stroke akan tampak mencong sebelah atau tidak simetris. Sebelah sudut mulut tertarik ke bawah dan lekukan antara hidung ke sudut mulut akan tampak mendatar.

- A = *Arms Drive* (Gerakan Lengan)

  Gejala penyakit stroke dapat dilihat dengan mengangkat tangan lurus sejajar kedepan dengan telapak tangan terbuka keatas selama 30 detik. Apabila terdapat kelumpuhan lengan yang ringan dan tidak disadari oleh penderita, maka lengan yang lumpuh tersebut akan turun (menjadi tidak sejajar lagi).

Pada kelumpuhan yang berat, lengan yang lumpuh sudah tidak dapat diangkat lagi bahkan sampai tidakdapat digerakkan sama sekali.

- S = *Speech* (Bicara)

  Penderita penyakit stroke akan mengalami ganguan artikulasi (menjadi pelo) atau tidak dapat berkata-kata (gagu) pada saat berbicara.

- T = *Three of Signs* (Tiga Tanda)

  Terdapat tiga gejala yang terjadi pada penderita stroke, yaitu perubahan bentuk wajah, terjadinya kelumpuhan sebagian atau seluruh bagian tubuh, dan gangguan pada saat berbicara.

Gejala lain yang terjadi pada penderita penyakit stroke adalah adanya gangguan pengelihatan secara tiba-tiba pada satu atau dua mata, merasa kesemutan separuh badan/rasa baal, pusing berputar, terkena gangguan daya ingat (amnesia), gangguan orientasi tempat, waktu dan orang, gangguang menelan cairan dan/atau makanan padat (disfagia) serta mendadak lemas seluruh badan dan terkulai tanpa hilang kesadaran (*drop attack*) atau disertai hilang kesadaran sejenak (sinkop) [YAS-11].

2.2.1 Jenis-Jenis Penyakit Stroke

Secara umum stroke dibagi menjadi dua, yaitu [DEW-12]:

1. Stroke Iskemik

Stroke iskemik terjadi apabila adanya penyumbatan pembuluh darah yang memasok darah ke otak. Stroke jenis ini merupakan jenis stroke yang paling umum dialami oleh seseorang. Kondisi yang mendasari terjadinya stroke istemik adalah adanya penumpukan lemak yang melapisi dinding pembuluh darah yang disebut dengan aterosklerosis. Kolesterol homocysteine dan zat lainnya dapat melekat pada dinding arteri dan membentuk zat lengket yang disebut dengan plak. Semakin lama plak tersebut akan menumpuk. Hal ini membuat darah sulit mengalir dengan baik dan menyebabkan pembekuan darah (trombus).

Stroke iskemik dibedakan menjadi dua berdasarkan penyebab sumbatan arteri, yaitu:

- Stroke trombotik adalah sumbatan yang disebabkan trombus yang berkembang didalam arteri otak yang sudah sangat sempit.
- Stroke embolik adalah sumbatan yang disebabkan thrombus, gelembung udara atau pecahan lemak (emboli) yang terbentuk dibagian tubuh lain seperti jantung dan pembuluh aorta di dada dan leher, yang terbawa aliran darah ke otak. Kelainan jantung yang disebut fibrilasi atrium dapat menciptakan kondisi dimana thrombus yang terbentuk di jantung terpompa dan beredar menuju otak.

2. Stroke hemoragik

Stroke hemoragik adalah jenis stroke yang disebabkan oleh pembuluh darah yang bocor atau pecah didalam atau di sekitar otak sehingga menghentikan suplai darah ke jaringan otak yang dituju. Selain itu, darah membanjiri dan memampatkan jaringan otak sekitarnya sehingga mengganggu atau mematikan fungsinya.

Stroke hemoragik dibagi menjadi dua, yaitu:

- Pendarahan intraserebral adalah pendarahan di dalam otak yang disebabkan oleh adanya trauma atau kelainan pembuluh darah (aneurisma atau angioma). Selain itu juga dapat disebabkan oleh tekanan darah tinggi kronis. Pendarahan intraserebral menjadi penyebab tertinggi kematian akibat stroke.
- Pendarahan subarachnoid adalah pendarahan dalam ruang subarachnoid, ruang diantara lapisan dalam (pia mater) dan lapisan tengah (arachnoid mater dan jaringan selaput otak (meninges). Penyebab paling umum adalah pecahnya tonjolan (aneurisma) dalam arteri. Perdarahan subarachnoid adalah kedaruratan medis serius yang dapat menyebabkan cacat permanen atau kematian. Stroke ini juga satu-satunya jenis stroke yang lebih sering terjadi pada wanita dibandingkan pada pria.

2.2.2 Faktor Risiko Penyakit Stroke

Faktor risiko stroke adalah suatu kondisi kesehatan atau penyakit yang ada pada seseorang yang berisiko terkena penyakit stroke. Apabila kondisi ini tidak

segera dikendalikan maka dapat memperburuk keadaan dan dapat mengakibatkan terjadinya penyempitan atau pecahnya pembuluh darah otak [YAS-11].

Terdapat dua macam faktor risiko penyakit stroke, yaitu faktor risiko yang dapat diubah/dikendalikan dan faktor risiko yang tidak dapat diubah/diendalikan. Faktor risiko yang tidak dapat diubah/dikendalikan meliputi:

1. Usia

Stroke dapat terjadi pada semua usia, bahkan anak-anak. Akan tetap stroke lebih sering terjadi pada orang yang telah lanjut usia (tua). Setiap penambahan 10 tahun setelah usia 55 tahun, terdapat peningkatan risiko penyakit stroke sebanyak dua kali lipat [ANN-13].

Berikut ini adalah klasifikasi usia [SHO-13]:

Tabel 2.1 Klasifikasi Usia

Range	Keterangan
<= 30	Muda
40 - 50	Paruh Baya
>= 60	Tua

2. Jenis Kelamin

Stroke lebih mungkin pada pria dibandingkan pada wanita. Namun, lebih dari separuh kematian stroke total yang terjadi pada wanita. Penggunaan pil KB dan kehamilan meningkatkan risiko stroke bagi perempuan [ZEN-12].

3. Ras

Kematian akibat penyakit stroke lebih banyak terjadi pada orang Afrika-Amerika daripada orang kulit putih. Hal ini dikarenakan mereka mempunyai risiko lebih tinggi menderita tekanan darah tinggi, diabetes, dan obesitas [ANN-13].

Sedangkan faktor risiko yang dapat diubah/dikendalikan meliputi:

1. Tekanan Darah

Tekanan darah adalah tekanan yang terjadi pada pembuluh darah arteri ketika darah dipompa oleh jantung untuk dialirkan ke seluruh tubuh. Ketika melakukan pengukuran tekanan darah, maka hasilnya akan dituliskan

misalnya, sebagai berikut 120/80 mHg. Nomor atas (120) disebut tekanan darah sistolik, sedangkan nomor bawah (80) disebut tekanan darah diastolik. Satuan tekanan darah adalah milimeer *hydrargyrum* atau air rakasa disingkat mmHg. Sehingga cara membaca tekanan darah tersebut yaitu seratus dua puluh per delapan puluh milimeter air raksa (Hg) [MUH-13].

Tekanan darah sistolik adalah tekanan darah yang terjadi pada saat otot jantung berkontrakasi (mengencang dan menekan). Tekanan sistolik disebut juga tekanan arterial maksimum saat terjadi kontraksi pada bolus ventrikular kiri jantung. Seperti telah disebutkan diatas, pada saat penulisan, tekanan sistolik ditulis sebagai angka pertama (pembilang). Contohnya tekanan darah 120/80 mmHg berarti bahwa tekanan darah sistoliknya adalah 120 mmHg [MUH-13].

Tekanan darah diastolik adalah tekanan darah yang terjadi pada saat otot jantung beristirahat atau tidak sedang berkontraksi (melonggar). Pada format penulisan, tekanan diastolik ditulis sebagai angka kedua (penyebut). Contohnya, tekanan darah 120/80 mmHg berarti bahwa tekanan darah diastoliknya adalah 80 mmHg [MUH-13].

Tekanan darah untuk setiap orang bervariasi secara alami. Tekanan darah pada bayi dan anak-anak secara normal jauh lebih rendah daripada orang dewasa. Dipengaruhi juga oleh aktivitas dan lebih rendah ketika beristirahat. Tekanan darah paling tinggi di pagi hari dan paling rendah pada saat tidur di malam hari [MUH-13].

Berikut ini merupakan klasifikasi tekanan darah menurut JNC 7 (*Seventh Report of the Joint National Committe on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure)* [MUH-13]:

Tabel 2.2 Klasifikasi Tekanan Darah

Range	Keterangan
< 120	Normal
120 - 139	Prahipertensi
140 - 159	Hipertensi Tahap I
> 160	Hipertensi Tahap II

2. Kadar Gula Darah

Gula darah adalah bahan bakar tubuh yang dibutuhkan untuk kerja otak, sistem saraf, dan jaringan tubuh yang lain. Gula darah yang terdapat di dalam tubuh dihasilkan oleh makanan yang mengandung karbohidrat, protein, dan lemak. Rata-rata, kadar gula darah normal adalah sebagai berikut [JEP-13]:

- Gula darah 8 jam sebelum makan atau setelah bangun pagi (70-110 mg/dl).
- Gula darah 2 jam setelah makan (100-150 mg/dl).
- Gula darah acak (70-125 mg/dl).

Kadar gula tinggi (hiperglikemia) maupun kadar gula rendah (hipoglikemia) tidak bagus untuk kesehatan. Kadar gula darah harus berada dalam posisi normal agar kinerja organ-organ tubuh tetap sehat dan normal [JEP-13].

Level gula darah atau glukosa diatur oleh kinerja pankreas. Jika kadar gula darah atau glukosa rendah, otomatis pankreas akan mengeluarkan glukagon, yang menargetkan sel-sel di hati untuk kemudian diubah menjadi glukosa dan setelah itu dilepaskan ke aliran darah hingga mencapai level tertentu [JEP-13].

Jika seseorang tetap mengalami kadar gula darah dalam level yang rendah, kemungkinan besar mengalami penurunan kerja pankreas atau kurangnya asupan gizi serta kurangnya istirahat [JEP-13].

Kadar gula darah yang sangat rendah dapat menyebabkan hilangnya kesadaran seseorang atau koma. Kondisi ini biasanya ditandai dengan gejala seperti berkeringat disertai gemetar, pusing, perasaaan linglung, dan jantung berdetak lebih kencang sampai kehilangan kesadaran [JEP-13].

Berikut ini adalah tabel klasifikasi kadar gula darah [SHO-13]:

Tabel 2.3 Klasifikasi Kadar Gula Darah

Range	Keterangan
< 60 mg/dl	Rendah
60 - 130 mg/dl	Normal
140 - 199 mg/dl	Intermediete
>= 200 mg/dl	Diabetes

3. Kadar Kolesterol Total

Kolesterol total merupakan kadar keseluruhan kolesterol yang beredar dalam tubuh manusia. Kolesterol adalah lipid amfipatik dan merupakan kompenen struktural esensial pada membran plasma. Senyawa kolesterol total ini di sintesis di banyak jaringan dari asetil-KoA dan merupakan prekusor utama semua steroid lain di dalam tubuh termasuk kortikosteroid, hormone seks, asam empedu, dan vitamin D. sebagai produk tipikal metabolisme hewan, kolesterol total terdapat dalam makanan yang berasal dari hewan misalnya kuning telur, daging, hati, dan otak [QAD-13].

Sekitar separuh kolesterol tubuh berasal dari proses sintesis di dalam tubuh (sekitar 700mg/hari) dan sisanya diperoleh dari makanan. Hati dan usus masing-masing menghasilkan sekitar 10% dari sintesis total pada manusia, dan sebagian besar jaringan yang mengandung sel berinti mampu membentuk kolesterol, yang berlangsung di reticulum endoplasma dan sitosol. Bahan baku pembuatan kolesterol di dalam tubuh adalah karbohidrat, protein, atau lemak [QAD-13].

Berikut ini merupakan klasifikasi kadar kolesterol total dalam tubuh [SHO-13]:

Tabel 2.4 Klasifikasi Kadar Kolesterol Total

Range	Keterangan
< 200 mg/dl	Normal
200 - 239 mg/dl	Tinggi
>= 240 mg/dl	Sangat Tinggi

4. *Low Density Lipoprotein* (LDL)

LDL pada umumnya dikenali dengan nama kolesterol jahat. Kolesterol LDL merupakan bentukan kendaraan yang membawa kolesterol dan ester kolesteril ke banyak jaringan. Kolesterol LDL disebut sebagai kolesterol jahat disebabkan peranannya membawa kolesterol total ke banyak jaringan di dalam tubuh. Sehingga memberikan peluang terjadinya penumpukan kolesterol di berbagai jaringan tubuh, termasuk diantaranya dalam pembuluh darah [QAD-13].

LDL mengandung lebih banyak lemak daripada HDL (*High Density Lipoprotein*) sehingga LDL akan mengambang di dalam darah. Protein utama yang membentuk LDL adalah Apo-B (apolipoprotein-B). LDL dianggap sebagai lemak yang "jahat" karena dapat menyebabkan penempelan kolesterol di dinding pembuluh darah.

Berikut ini adalah klasifikasi kadar *Low Density Lipoprotein* (LDL) dalam tubuh manusia [SHO-13]:

Tabel 2.5 Klasifikasi Kadar LDL

Range	Keterangan
< 100 mg/dl	Normal
130 - 189 mg/dl	Tinggi
>= 190 mg/dl	Sangat Tinggi

5. Asam Urat

Penyakit asam urat adalah penyakit yang timbul akibat kadar asam urat darah yang berlebihan. Yang menyebabkan kadar asam urat darah berlebihan adalah produksi asam urat di dalam tubuh lebih banyak dari pembuangannya. Organ yang bisa terserang adalah sendi, otot, jaringan di sekitar sendi, telinga, kelopak mata, jantung, ginjal, dan lain-lain. Jika kadar asam urat di dalam darah melebihi normal maka asam urat darah ini akan masuk ke organ-organ tersebut sehingga menimbulkan penyakit pada organ-organ tersebut. Penyakit asam urat lebih sering menyerang laki-laki. Jika penyakit ini menyerang wanita maka pada umumnya wanita yang menderita adalah sudah menopause [KER-09].

Berikut ini merupakan klasifikasi tingkat asam urat pada laki-laki [SHO-13]:

Tabel 2.6 Klasifikasi Tingkat Asam Urat Laki-laki

Range	Keterangan
<= 3,5 mg/dl	Rendah
3,5 - 7,0 mg/dl	Normal
>= 7,0 mg/dl	Asam Urat

Sedangkan berikut ini merupakan klasifikasi tingkat asam urat pada wanita [SHO-13]:

Tabel 2.7 Klasifikasi Tingkat Asam Urat Perempuan

Range	Keterangan
<= 2,6 mg/dl	Rendah
2,6 – 6,0 mg/dl	Normal
>= 6,0 mg/dl	Asam Urat

6. BUN dan Kreatinin

Blood Urea Nitrogen (BUN) dapat didefinisikan sebagai jumlah nitrogen urea yang hadir dalam darah. Urea adalah produk limbah yang dibentuk dalam tubuh selama proses pemecahan protein. Selama metabolisme protein, protein diubah menjadi asam amino yang juga menghasilkan amonia. Urea tidak lain adalah substansi yang dibentuk oleh beberapa molekul amonia. Metabolisme protein berlangsung dalam hati dan dengan demikian urea juga diproduksi oleh hati. Selanjutnya, urea ditransfer ke ginjal melalui aliran darah dan dikeluarkan dari tubuh dalam bentuk urin. Dengan demikian, setiap disfungsi ginjal akan menyebabkan kadar tinggi atau rendah BUN dalam darah [SHE-13].

Berikut ini adalah klasifikasi kadar BUN dalam tubuh [SHO-13]:

Tabel 2.8 Klasifikasi Kadar BUN

Range	Keterangan
< 6 mg/dl	Rendah
6 – 23 mg/dl	Normal
> 23 mg/dl	Tinggi

Kreatinin (*creatinine*) adalah produk penguraian dari kreatin fosfat dalam metabolisme otot dan dihasilkan dari kreatin (*creatine*). Kreatinin pada dasarnya merupakan limbah kimia yang selanjutnya diangkut ke ginjal melalui aliran darah untuk dikeluarkan melalui urin. Kadar kreatinin dapat diukur dalam urin serta darah. Tingkat kreatinin dalam darah umumnya tetap normal karena massa otot relatif konstan. Dengan demikian, ginjal yang berfungsi normal juga akan menunjukkan tingkat normal kreatinin dalam darah. Tapi ketika ginjal tidak berfungsi dengan baik, jumlah kreatinin dalam darah akan meningkat [SHE-13].

Berikut ini adalah klasifikasi kreatinin dalam tubuh [SHO-13]:

Tabel 2.9 Klasifikasi Kreatinin

Range	Keterangan
< 0,7 mg/dl	Rendah
0,7 – 1,2 mg/dl	Normal
> 1,2 mg/dl	Tinggi

2.2.3 Cara Pencegahan dan Penanganan Pertama Penyakit Stroke

Penyakit stroke dapat dicegah dengan merubah gaya hidup dan mengendalikan faktor-faktor risiko penyakit stroke. Terdapat dua macam cara pencegahan prenyakit stroke yang dapat dilakukan, yaitu [YAS-11]:

1. Pencegahan Primer

Pencegahan primer adalah upaya pencegahan yang dilakukan sebelum terkena penyakit stroke. Cara yang harus dilakukan adalah dengan mempertahankan tujuh gaya hidup sehat, yaitu:

- Hentikan kebiasaan merokok
- Pertahankan berat badan sesuai dengan berat badan ideal, dimana garis lingkar pinggang wanita < 80 cm dan garis lingkar pinggang pria <90 cm
- Makan makanan sehat yang rendah lemak jenuh dan kolesterol, menambah asupan kalium dan mengurangi natrium, serta perbanyak makan buah-buahan dan sayur-sayuran.
- Olahraga yang cukup dan teratur dengan melakukan aktivitas fisik yang punya nilai aerobic, seperti jalan cepat, bersepeda, berenang dan lain-lain secara teratur minimal 30 menit dan tiga kali per minggu
- Kadar lemak (kolesterol) dalam darah kurang dari 200 mg% (hasil laboraturium)
- Kadar gula puasa kurang dari 100 mg/dl (hasil laboratorium)
- Tekanan darah dipertahankan pada 120/80 mm Hg

2. Pencegahan Sekunder

Pencegahan sekunder adalah upaya pencegahan yang bertujuan agar tidak terkena stroke berulang. Adapun caranya adalah dengan:

- Mengendalikan faktor risiko yang telah ada seperti mengontrol darah tinggi, kadar kolesterol, gula darah, dan asam urat
- Mengubah gaya hidup menjadi lebih sehat
- Minum obat sesuai dengan anjuran dokter secara teratur
- Melakukan kontrol ke dokter secara rutin

Pertolongan pertama yang harus dilakukan apabila terjadi gejala awal stroke adalah dengan segera membawa kerumah sakit terdekat dan diutamakan yang mempunyai dokter spesialis saraf (neurology) dan fasilitas unit stroke dan sudut stroke [YAS-11].

Waktu adalah penyelamatan otak. Waktu pemulihan aliran darah ke bagian otak yang terganggu sangat pendek, yaitu hanya sekitar 3 jam dimulai sejak tanda awal stroke terjadi. Pertolongan yang akurat dan cepat harus segera dilakukan untuk menghindari kematian atau kecacatan yang menetap. Setiap menit keterlambatan pertolongan agar otak tidak kekurangan darah berarti 1,9 juta sel otak dan serabut otak sepanjang 10 kilometer akan mati [YAS-11].

2.3 *Cluster*ing

Clustering adalah teknik yang banyak digunakan untuk mengelompokkan data/objek kedalam kelompok data (kluster) sehingga tiap *cluster* mempunyai data yang hampir samadengan *cluster* lainnya. Jika terdapat suatu himpunan yang berjumlah terhingga, yaitu X, maka permasalahan *cluster*ing dalam X adalah mencari beberapa pusat *cluster* yang dapat memberikan ciri kepada masing-masing *cluster* dalam X. Distance-based *cluster*ing adalah pengelompokan objek berdasarkan jarak. Sedangkan conceptual *cluster*ing adalah pengelompokan objek berdasarkan kecocokannya menurut konsep deskriptif.

Dalam bukunya yang berjudul *An Introduction to Data Mining*, Dr. Daniel T. Larose (2005) mendefinisikan *clustering* sebagai upaya mengelompokkan *record*, observasi, atau mengelompokkan kedalam kelas yang memiliki kesamaan objek [LAR-05:147].

Pengklasteran berbeda dengan klasifikasi yaitu tidak adanya variabel target dalam pengklasteran. Pengklasteran tidak mencoba untuk melakukan klasifikasi, mengestimasi, atau memprediksi nilai dari variabel target. Akan tetapi, algoritma

pengklasteran mencoba untuk melakukan pembagian terhadap keseluruhan data menjadi kelompok-kelompok yang memiliki kemiripan (homogen), yang mana kemiripan *record* dalam suatu kelompok akan bernilai maksimal, sedangkan kemiripan dengan *record* dalam kelompok lain akan bernilai minimal. Prinsip dasar untuk mendapatkan homogen atau heterogen dapat menggunakan konsep ukuran jarak. Jarak yang dimaksud bisa berarti ukuran jarak kedekatan atau kemiripan (*similarity measures*), bisa juga jarak yang berjauhan atau ketidakmiripan (*disimilarity measures*).

2.4 *Fuzzy C-Means Clustering* (FCM)

Fuzzy c-means termasuk salah satu algoritma *fuzzy clustering*. Metode ini pertama kali diperkenalkan oleh Jim Bezdek pada tahun 1981. *Fuzzy c-means* merupakan suatu teknik peng-*cluster*-an data yang mana keberadaan tiap titik data dalam suatu *cluster* ditentukan oleh derajat keanggotaan [KUS-10].

Konsep dasar dari *fuzzy c-means* adalah dengan menentukan pusat *cluster* sebagai penanda lokasi rata-rata untuk tiap *cluster*. Dengan memperbaiki pusat *cluster* dan derajat keanggotaan setiap titik data secara berulang, maka akan dapat dilihat bahwa pusat *cluster* akan bergerak menuju lokasi yang tepat. Perulangan ini didasarkan pada minimasi fungsi obyektif yang menggambarkan jarak dari titik data yang diberikan menuju pusat *cluster* yang terbobot oleh derajat keanggotaan titik data tersebut [KUS-10].

Output dari fuzzy c-means bukanlah suatu *fuzzy inference system*, melaikan sebuah deretan pusat *cluster* dan beberapa derajat keanggotaan untuk tiap-tiap titik data. Informasi ini dapat digunakan untuk membangun suatu *fuzzy inference system* [KUS-10].

2.4.1 Algoritma Fuzzy C-Means *Cluster*ing (FCM)

Algoritma *Fuzzy C-Means* (FCM) adalah sebagai berikut [KUS-10]:

1. *Input* data yang akan di-*cluster* X berupa matriks berukuran n x m, dimana n adalah jumlah sampel data dan m adalah atribut setiap data. X_{ij}= data sampel ke-i (i = 1, 2, 3, …, n), atribut ke-j (j = 1, 2, 3, …, m).
2. Menentukan:

   Jumlah *cluster* = c;

Pangkat = w;

Maksimum iterasi = MaxIter;

Eror terkecil yang diharapkan = ξ;

Fungsi obyektif awal = P0 = 0;

Iterasi awal = t = 1;

3. Membangkitkan bilangan random μ_{ik}. i = 1, 2, …, n; k = 1, 2, …, c; sebagai elemen-elemen matriks partisi awal U dengan range antara 0 sampai dengan 1. Kemudian, menghitung jumlah setiap kolom. Jumlah setiap kolom bilangan random yang terbentuk dapat dihitung dengan persamaan (2-1).

$$Q_i = \sum_{k=1}^{c} \mu_{ik} \qquad \text{…… (2-1)}$$

dengan j= 1, 2, …, n.

Setelah menjumlahkan setiap kolom dari bilangan random yang terbentuk, maka kemudian menghitung matriks partisi awal U. Matriks Partisi awal U dapat dihitung dengan persamaan (2-2).

$$\mu_{ik} = \frac{\mu_{ik}}{Q_i} \qquad \text{…… (2-2)}$$

4. Menghitung pusat *cluster* ke-k (V_{kj}) berdasarkan persamaan (2-3), dengan k=1,2, …,c; dan j=1, 2, …, m

$$V_{kj} = \frac{\sum_{i=1}^{n}((\mu_{ik})^{w} * X_{ij})}{\sum_{i=1}^{n}(\mu_{ik})^{w}} \qquad \text{…… (2-3)}$$

5. Menghitung fungsi obyektif iterasi ke-t (Pt) berdasarkan persamaan (2-4).

$$P_t = \sum_{i=1}^{n}\sum_{k=1}^{c}\left(\left[\sum_{j=1}^{m}(X_{ij} - V_{ij})^2\right](\mu_{ik})^{w}\right) \qquad \text{…… (2-4)}$$

6. Menghitung perubahan matriks partisi berdasarkan persamaan (2-5).

$$\mu_{ik} = \frac{\left[\sum_{j=1}^{m}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{w-1}}}{\sum_{k=1}^{c}\left[\sum_{j=1}^{m}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{w-1}}} \qquad \text{…… (2-5)}$$

dengan i = 1, 2, …, n; dan k = 1, 2, …, c.

7. Memeriksa kondisi berhenti:

- Jika: Selisih nilai fungsi objektif pada iterasi ke-t (Pt) dengan fungsi objektif pada iterasi ke-t - 1 (Pt-1) atau iterasi kurang dari maksimum iterasi maka kondisi berhenti yang dapat ditunjukkan dengan persamaan (2-6) dan persamaan (2-7).

$$(|Pt - Pt\text{-}1| < \xi) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2\text{-}6)$$

atau

$$(t > MaxIter) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2\text{-}7)$$

- Jika tidak: iterasi dilanjutkan t = t+1 dan ulangi dari langkah ke-4.

2.5 Analisa *Cluster*

Cluster atau 'klaster' dapat diartikan 'kelompok', dengan demikian, pada dasarnya analisa klaster akan menghasilkan sejumlah klaster (kelompok). Analisis ini diawali dengan pemahaman bahwa sejumlah data tertentu sebenarnya mempunyai kemiripan di antara anggotanya, karena itu dimungkinkan untuk mengelompokkan anggota-anggota yang 'mirip' atau mempunyai karakteristik yang serupa tersebut dalam satu atau lebih dari satu klaster [SAN-10:111].

Seperti diketahui, analisis klaster akan membagi sejumlah data satu atau beberapa klaster tertentu. Pertanyaan yang kemudian timbul adalah 'apa yang menjadi batas bahwa sejumlah data dapat disebut sebagai satu klaster?' secara logika, sebuah klaster yang baik adalah klaster yang mempunyai [SAN-10:113]:

- Homogenitas (kesamaan) yang tinggi antara anggota dalam satu klaster (*within cluster*).
- Heterogenitas (perbedaan) yang tinggi antara klaster yang satu dengan klaster yang lainnya (*between cluster*).

Dari dua hal di atas dapat disimpulkan bahwa klaster yang baik adalah klaster yang mempunyai anggota-anggota yang semirip mungkin satu dengan yang lain, namun sangat tidak mirip dengan anggota-anggota klaster yang lain.

2.5.1 Analisa Varian

Ciri dari klaster yang baik adalah memiliki homogenitas yang tinggi antara anggota dalam satu klaster dan heterogenitas yang tinggi antara klaster yang satu

dengan klaster yang lainnya, maka bisa dilakukan pendekatan untuk mengetahui baik tidaknya suatu klaster berdasarakan nilai varian. Varian merupakan nilai penyebaran dari data. Sehingga nilai varian dapat digunakan untuk mengetahui baik atau tidaknya suatu klaster. Varian pada *clustering* ada dua macam, yaitu *variance within cluster* dan *variance between cluster*. Kepadatan suatu *cluster* bisa ditentukan dengan *variance within cluster* (Vw) dan *variance between cluster* (Vb) [ILH-11]. *Variance cluster* ditentukan dengan persamaan (2-8).

$$V_c^2 = \frac{1}{n_c - 1}\sum_{i=1}^{n_c}(d_i - \bar{d}_c)^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots \text{(2-8)}$$

dimana:

V_c^2= Varian pada *cluster* c

c = 1...k, dimana k = jumlah *cluster*

n_c= jumlah data pada *cluster c*

d_i= data ke-*i* pada suatu *cluster*

$\bar{d}_c$= rata-rata dari data pada suatu *cluster*

Berdasarkan nilai varian *cluster* yang diperoleh, maka nilai *variance within cluster* (V_w) dapat dihitung dengan persamaan (2-9).

$$V_w = \frac{1}{N - k}\sum_{i=1}^{k}(n_i - 1).V_i^2 \quad \ldots\ldots\ldots\ldots\ldots \text{(2-9)}$$

dimana :

V_w = *variance within cluster*

N = jumlah semua data,

k = jumlah *cluster*

n_i = jumlah data pada *cluster* ke-*i*

V_i^2 = varian pada *cluster* ke-*i*

Nilai *variance between cluster* (V_b) dapat dihitung dengan persamaan (2-10) berikut.

$$V_b = \frac{1}{k - 1}\sum_{i=1}^{k} n_i \left(\bar{d}_i - \bar{d}\right)^2 \quad \ldots\ldots\ldots\ldots \text{(2-10)}$$

dimana:

$\bar{d}_i$ = rata-rata data pada *cluster* ke-i

$\bar{d}$ = rata-rata dari d_i

Cluster yang ideal mempunyai V_w minimum yang merepresentasikan *internal homogenity* dan V_b maksimum yang menyatakan *external homogenity*. *Cluster* yang ideal dapat dilihat dari nilai varian yang kecil. Semakin kecil nilai varian, maka semakin ideal *cluster* tersrbut. Nilai batasan varian dinyatakan dalam persamaan (2-11).

$$V = \frac{V_w}{V_b} \qquad \text{..... (2-11)}$$

2.6 Logika *Fuzzy*

Konsep logika *fuzzy* dicetuskan oleh Lotfi Zadeh, seorang profesor *University of California* di Berkeley, dan dipresentasikan bukan sebagai metodologi kontrol, namun sebagai suatu cara pemrosesan data yang memperbolehkan anggota himpunan parsial daripada anggota himpunan kosong atau non-anggota. Pendekatan ini pada teori himpunan tidak diaplikasikan untuk mengontrol sistem sampai tahun 70-an karena kurangnya kemampuan komputer-mini pada saat itu. Profesor Zadeh beralasan bahwa masyarakat tidak butuh ketepatan, input informasi numeris, dan mereka belum sanggup dengan kontrol adaptif yang tinggi. Jika kembalian dari kontroler dapat diprogram untuk menerima *noisy*, input yang tidak teliti, mereka akan lebih efektif dan lebih mudah diimplementasikan [KUS-08:136].

Namun sekarang logika *fuzzy* telah menyentuh keberbagai pengembangan algoritma. Konsep dasar dari logika *fuzzy* yang berupa perluasan dari algoritma *boolean* (1 atau 0/ya atau tidak) dimana himpunan *fuzzy* berisi derajat keanggotaan sebagai penentu keberadaan elemen dalam suatu himpunan sangatlah penting dan membuat logika *fuzzy* bisa ditanam dalam algoritma lainnya.

Dalam banyak hal, logika *fuzzy* digunakan sebagai cara untuk memetakan permasalahan dari *input* menuju *output* yang diharapkan. Selain itu ada beberapa alasan yang membuat logika *fuzzy* banyak digunakan, diantaranya adalah logika *fuzzy* mudah dimengerti, sangat fleksibel, memiliki toleransi terhadap data yang tidak tepat, mampu memodelkan fungsi-fungsi nonlinear yang sangat kompleks, dapat membangun dan mengaplikasikan pengalaman pakar tanpa ada pelatihan

sebelumnya, dapat bekerjasama dengan teknik-teknik kendali secara konvensional, dan logika *fuzzy* didasarkan pada bahasa alami [KUS-10:1]

2.6.1 Himpunan *Fuzzy*

Pada himpunan tegas (*crisp*), nilai keanggotaan suatu x dalam suatu himpunan A, yang sering ditulis dengan $\mu_A(x)$, memiliki dua kemungkinan, yaitu 0 atau 1 [KUS-10:3]. Logika klasik tersebut memiliki kekurangan dari segi keadilan dalam memasukkan suatu nilai dalam keanggotaan berdasarkan range yang telah ditentukan sebelumnya. Misalnya, ditentukan range untuk usia dimana umur 0 sampai 35 tahun dikategorikan muda dan 35 sampai 55 tahun dikategorikan parobaya. Apabila seseorang berusia 15 tahun, maka dia dikatakan muda. Lalu, bagaimana jika ada orang yang berusia 35 tahun kurang 1 hari, maka dia dikatakan tetap muda. Padahal usia orang tersebut mendekati kategori parobaya. Dari sini bisa dikatakan bahwa pemakaian himpunan *crisp* untuk menyatakan umur sangat tidak adil, adanya perubahan kecil saja pada suatu nilai mengakibatkan perbedaan kategori yang cukup signifikan.

Berbeda dengan himpunan *crisp*, logika *fuzzy* mengelompokkan himpunan *fuzzy* dari semesta U untuk dikelompokkan oleh fungsi keanggotaan yang berada pada nilai antara [0,1]. Fungsi keanggotaan dari himpunan *fuzzy* merupakan kontinu dengan range 0 sampai 1[KUS-08:136].

Logika *fuzzy* digunakan untuk mengantisipasi ketidakadilan dari himpunan *crisp*. Lewat logika *fuzzy*, satu nilai dapat masuk dalam dua himpunan yang berbeda tergantung dari seberapa besar derajat keanggotaan nilai tersebut terhadap himpunan satu dengan lainnya. Derajat keanggotaan pada himpunan *fuzzy* dapat dihitung menggunakan fungsi keanggotaan. Himpunan *fuzzy* memiliki dua atribut, yaitu:

1. Linguistik, yaitu penamaan grup yang mewakili suatu keadaan dan kondisi tertentu dengan menggunakan bahasa alami, seperti: muda, parobaya, tua.
2. Numeris, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variabel seperti: 40, 25, 50, dsb.

2.6.2 Fungsi Keanggotaan

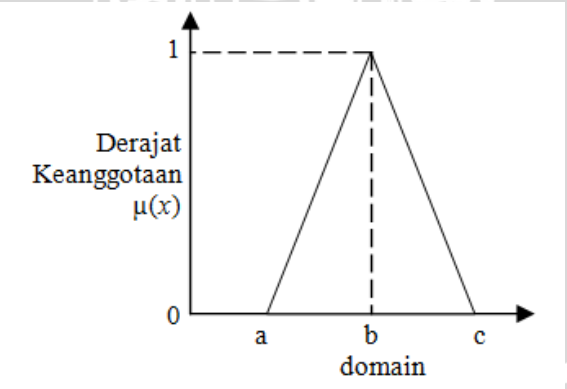
Fungsi keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaan yang memiliki nilai interval antara 0 dan 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Ada beberapa fungsi yang dapat digunakan diantaranya adalah representasi linear, Representasi Kurva Segitiga, Representasi Kurva Trapesium, Representasi Kurva Bentuk Bahu, Representasi Kurva-S, dan Representasi Kurva Bentuk Lonceng (*Bell Curve*) yang terbagi lagi menjadi Kurva PI, Kurva Beta, dan Kurva Gauss[KUS-10:8].

Menurut Jang dan Mirzutani (1997) fungsi keanggotaan *fuzzy* terparameterisasi satu dimensi yang umum digunakan, antara lain [JAN-97] :

1. Fungsi keanggotaan segitiga, disifati oleh parameter {*a*,*b*,*c*} yang didefinisikan seperti pada Persamaan (2-12).

$$segitiga(x;a,b,c) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ \dfrac{c-x}{c-b}, & b \le x \le c \\ 0, & c \le x \end{cases} \quad \ldots\ldots\ldots (2\text{-}12)$$

Parameter {*a*,*b*,*c*} (dengan $a<b<c$) yang menentukan koordinat *x* dari ketiga sudut segitiga tersebut, seperti terlihat pada gambar (2.1) berikut:

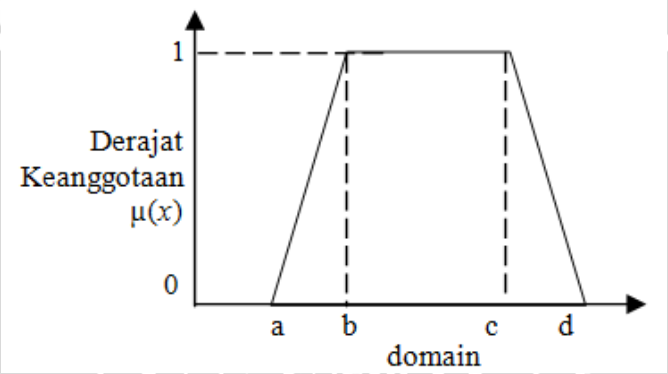


Gambar 2.1 Kurva Segitiga

2. Fungsi keanggotaan trapesium, disifati oleh parameter {*a*,*b*,*c*,*d*} yang didefinisikan pada Persamaan (2-13).

$$trapesium(x;a,b,c,d) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ 1, & b \le x \le c \\ \dfrac{d-x}{d-c}, & c \le x \le d \\ 0, & d \le x \end{cases} \quad \ldots\ldots\ldots (2\text{-}13)$$

Parameter {*a,b,c,d*} pada kurva trapesium dapat dilihat pada gambar (2.2) berikut:

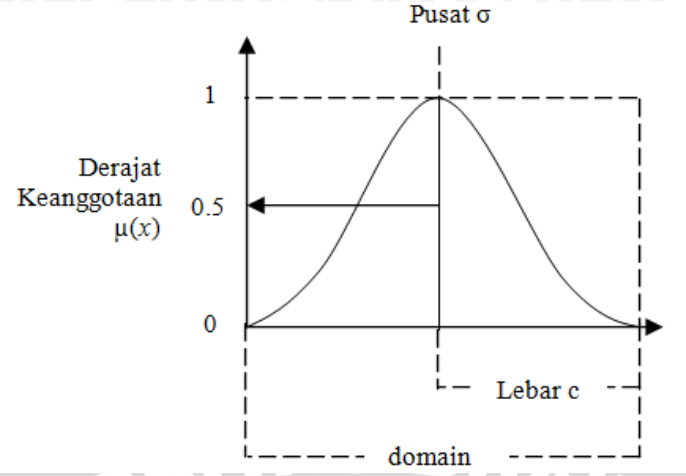


Gambar 2.2 Kurva Trapesium

3. Fungsi keanggotaan *gaussian*, disifati oleh parameter $\{c, \sigma\}$ yang didefinisikan pada Persamaan (2-14).

$$gaussian(x;c,\sigma) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2} \quad \ldots\ldots\ldots (2\text{-}14)$$

Karakteristik fungsi keanggotaan *gauss* ditentukan oleh parameter *c* dan σ seperti terlihat pada gambar (2.3) berikut:

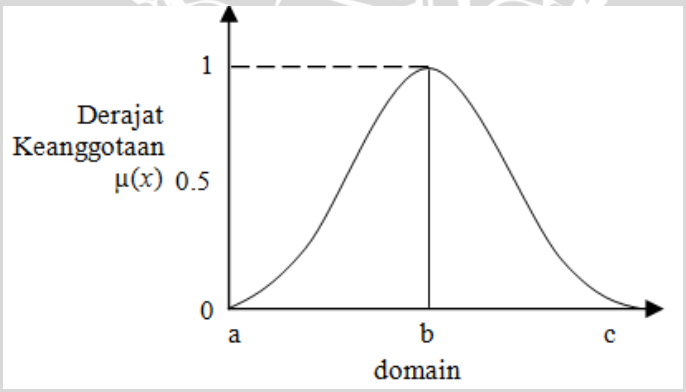


Gambar 2.3 Karakteristik Fungsi *Gauss*

4. Fungsi keanggotaan *generalized bell*, disifati oleh parameter {*a*,*b*,*c*} yang didefinisikan seperti pada Persamaan (2-15).

$$bell(x;a,b,c)=\frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}} \qquad \ldots\ldots\ldots (2\text{-}15)$$

Parameter *b* selalu positif, supaya kurva menghadap kebawah, seperti terlihat pada gambar (2.4) berikut:

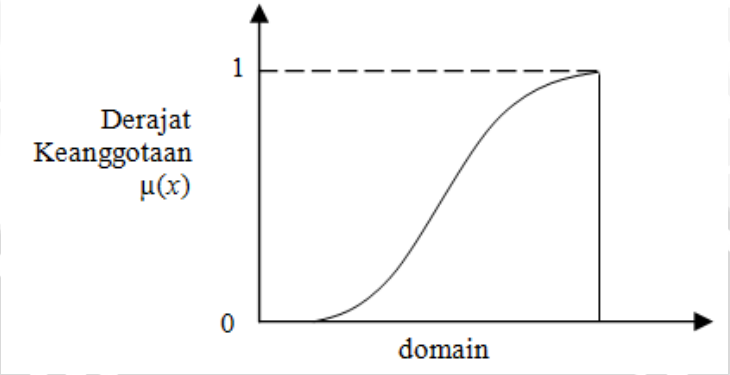


Gambar 2.4 Karakteristik Fungsi *Generalized Bell*

5. Fungsi keanggotaan *sigmoid*, disifati oleh parameter {*a*,*c*} yang didefinisikan seperti pada Persamaan (2-16).

$$sig(x;a,c)=\frac{1}{1+\exp\left[-a\left(x-c\right)\right]} \qquad \ldots\ldots\ldots (2\text{-}16)$$

Parameter *a* digunakan untuk menentukan kemiringan kurva pada saat *x* = *c*. Polaritas dari *a*akan menentukan kurva itu kanan atau kiri terbuka, seperti terlihat pada gambar (2.5) berikut:



Gambar 2.5 Kurva Sigmoid

2.6.3 Operator Dasar Zadeh

Seperti halnya himpunan konvensional, ada beberapa operasi yang didefinisikan secara khusus untuk mengkombinasikan dan memodifikasi himpunan *fuzzy*. Nilai keanggotaan sebagai hasil dari operasi 2 himpunan sering dikenal dengan nama *fire strength* atau α-predikat. Ada 3 operator dasar yang diciptakan oleh Zadeh, yaitu [KUS-10:23]:

a. Operator *AND*

Operator ini berhubungan dengan operasi interseksi pada himpunan α-predikat sebagai hasil operasi dengan operator *AND* diperoleh dengan mengambil nilai keanggotaan terkecil antar elemen pada himpunan – himpunan yang bersangkutan. Operator *AND* ditunjukkan pada persamaan (2-17).

$$\mu_{A\cap B} = \min\left(\mu_A(x), \mu_B(y)\right) \quad \text{..... (2-17)}$$

b. Operator *OR*

Operator ini berhubungan dengan operasi *union* pada himpunan α-predikat sebagai hasil operasi dengan operator *OR* diperoleh dengan mengambil nilai keanggotaan terbesar antarelemen pada himpunan – himpunan yang bersangkutan. Operator *OR* ditunjukkan pada persamaan (2-18).

$$\mu_{A\cup B} = \min\left(\mu_A(x), \mu_B(y)\right) \quad \text{..... (2-18)}$$

c. Operator *NOT*

Operator ini berhubungan denga operasi komplemen pada himpunan α-predikat sebagai hasil operasi dengan operator *NOT* diperoleh dengan mengurangkan nilai keanggotaan elemen pada himpunan yang bersangkutan dari 1. Operator *NOT* ditunjukkan pada persamaan (2-19).

$$\mu_{A'} = 1 - \mu_A(x) \quad \text{..... (2-19)}$$

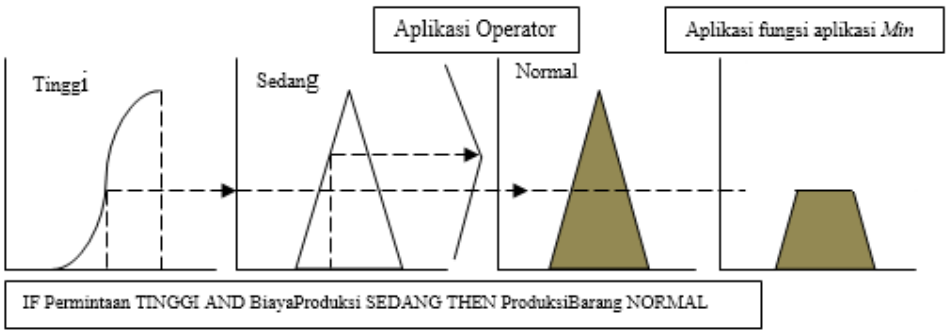
2.6.4 Fungsi Implikasi

Kaidah *fuzzy if-then* (dikenal juga sebagai kaidah *fuzzy*, implikasi *fuzzy* atau pernyataan kondisi *fuzzy*) diasumsikan pada persamaan (2-20).

Jika *x* adalah *A* maka *y* adalah *B* ..... (2-20)

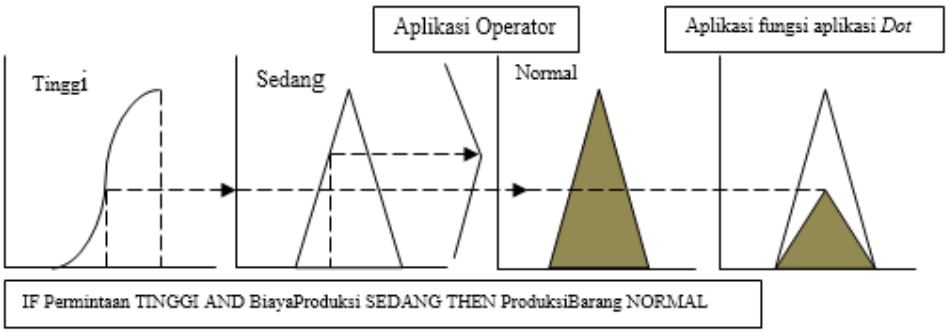
Dengan *A* dan *B* adalah nilai linguistik yang dinyatakan dengan himpunan *fuzzy* dalam semesta pembicaraan *X* dan *Y*. Seringkali "*x* adalah *A*" disebut sebagai *antecedent* atau *premise*, sedangkan "*y* adalah *B*" disebut *consequence* atau *conclusion* [JAN-97]. Secara umum, ada 2 fungsi implikasi yang dapat digunakan, yaitu [KUS-10:28]:

a. *Min* (*minimum*), fungsi ini akan memotong *output* himpunan *fuzzy*. Salah satu contoh penggunaan fungsi *min* ditunjukkan pada Gambar (2.6).



Gambar 2.6 Fungsi implikasi MIN

b. *Dot* (*product*), fungsi ini akan menskala *output* himpunan *fuzzy*. Contoh penggunaan fungsi *dot* ditunjukkan pada Gambar (2.7)



Gambar 2.7 Fungsi impliksai DOT

2.6.5 Fuzzy Inference System

Ada tiga metode dalam sistem inferensi *fuzzy* (*fuzzy inference system*) diantaranya adalah *fuzzy inference system* metode *tsukamoto*, *mamdani* dan *sugeno*. Metode *tsukamoto* merupakan perluasan dari penalaran monoton. Pada metode *tsukamoto*, setiap konsekuen pada aturan yang berbentuk *IF-THEN* harus direpresentasikan dengan suatu himpunan *fuzzy* dengan fungsi keanggotaan yang monoton. Sebagai hasilnya, *output* hasil inferensi dari tiap-tiap aturan diberikan secara tegas (*crisp*) berdasarkan *α-predikat* (*fire strength*). Hasil akhirnya diperoleh dengan menggunakan rata-rata terbobot [KUS-10:31].

Metode *mamdani* sering dikenal sebagai metode *Max-Min*. Metode ini diperkenalkan oleh Ebrahim Mamdani pada tahun 1975. Untuk mendapatkan *output*, diperlukan 4 tahapan yaitu pembentukan himpunan *fuzzy*, aplikasi fungsi implikasi, komposisi aturan dan penegasan (*defuzzy*) [KUS-10:37].

Metode *sugeno* hampir sama dengan penalaran *mamdani*, hanya saja *output* (konsekuen) sistem tidak berupa himpunan *fuzzy*, melainkan berupa konstanta atau persamaan linear. Metode ini diperkenalkan oleh Takagi-Sugeno Kang pada tahun 1985, sehingga metode ini sering juga dinamakan dengan metode TSK. Metode TSK terdiri atas 2 jenis, yaitu [KUS-10:46]:

a. Model *fuzzy* sugeno orde-nol

   Secara umum bentuk model *fuzzy* sugeno orde-nol ditunjukkan pada persamaan (2-21).

   IF (x_1 *is* A_1) ∘ (x_2 *is* A_2) ∘ (x_3 *is* A_3) ∘ ... ∘ (x_n *is* A_n) THEN $z = k$ ….(2-21)

dengan A_i adalah himpunan *fuzzy* ke-*i* sebagai anteseden dan *k* adalah suatu konstanta (tegas) sebagai konsekuen.

b. Model *fuzzy* sugeno orde-satu

Secara umum bentuk model *fuzzy* sugeno orde-satu ditunjukkan pada persamaan (2-22).

$$\text{IF } (x_1 \text{ is } A_1) \circ ... \circ (x_n \text{ is } A_n) \text{ THEN } z = p_1 * x_1 + ... + p_n * x_n + q \quad \ldots\ldots \text{ (2-22)}$$

dengan A_i adalah himpunan *fuzzy* ke-*i* sebagai anteseden dan p_i adalah suatu konstanta (tegas) ke-*i* dan *q* juga merupakan konstanta dalam konsekuen.

2.6.6 Ekstraksi Aturan *Fuzzy*

Untuk membentuk *fuzzy inference system* dari hasil *clustering*, kita dapat menggunakan metode inferensi *fuzzy* sugeno orde-satu. Sebelumnya, data yang ada dipisahkan terlebih dahulu antara data pada variable-variabel *input* dengan data pada variabel *output*. Misalkan jumlah variabel *input* adalah *m*, dan variabel *output* biasanya 1. Pada metode ini, akan diperoleh kumpulan aturan yang dinyatakan dengan persamaan (2-23) [KUS-10:151].

$$\begin{aligned}
&[R1] \quad \text{IF } (x_1 \text{ is } A_{11}) \circ (x_2 \text{ is } A_{12}) \circ ... \circ (x_n \text{ is } A_{1m}) \\
&\qquad \text{THEN } (z = k_{11}x_1 + ... + k_{1m}x_m + k_{10}); \\
&[R2] \quad \text{IF } (x_1 \text{ is } A_{21}) \circ (x_2 \text{ is } A_{22}) \circ ... \circ (x_n \text{ is } A_{2m}) \\
&\qquad \text{THEN } (z = k_{21}x_1 + ... + k_{2m}x_m + k_{20}); \\
&\qquad ... \\
&[Rr] \quad \text{IF } (x_1 \text{ is } A_{m1}) \circ (x_2 \text{ is } A_{m2}) \circ ... \circ (x_n \text{ is } A_{rm}) \\
&\qquad \text{THEN } (z = k_{r1}x_1 + ... + k_{rm}x_m + k_{r0}); \qquad \ldots\ldots\ldots\ldots \text{ (2-23)}
\end{aligned}$$

dengan:

- A_{ij} adalah himpunan *fuzzy* aturan ke-*i* variabel ke-*j* sebagai anteseden,
- $_{kij}$ adalah koefisien persamaan *output fuzzy* aturan ke-*i* variabel ke-*j* (i = 1,2,...,r; j = 1,2,...,m), dan k_{i0} adalah konstanta persamaan *output fuzzy* aturan ke-*i*,
- tanda ∘ menunjukkan operator yang digunakan

Hasil dari *clustering* ini adalah pusat *cluster* (*c*) dan nilai kecenderungan suatu data terhadap *cluster* tertentu. Setelah didapatkan hasil dari proses *cluster*ing

kemudian dilakukan perhitungan nilai standar deviasi. Standar Deviasi (σ) dinyatakan pada persamaan (2-24) [LUD-11].

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x_c})^2}{n_c - 1}} \qquad \text{.....(2-24)}$$

dimana:

σ = standar deviasi

x_i = data ke-i

$\bar{x_c}$= mean (rata-rata dari data pada suatu *cluster*)

n_c = jumlah data pada suatu *cluster*

Dari nilai *c* dan σ yang diperoleh pada perhitungan sebelumnya nantinya akan digunakan untuk mengetahui derajat keanggotaan setiap titik data. Derajat keanggotaan setiap titik data menggunakan fungsi *gauss* ditunjukkan pada persamaan (2-25).

$$\mu_{ki} = e^{-\sum_{j=1}^{m}\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} \qquad \text{.....(2-25)}$$

Kemudian derajat keanggotaan setiap data *i* dalam *cluster k* ini kita akan kalikan dengan setiap atribut *j* dari data *i*. Misalkan dinotasikan sebagai d_{ij}^k dengan persamaan (2-26).

$$d_{ij}^k = X_{ij} * \mu_{ki} \; dan \; d_{i(m+1)}^k = \mu_{ki} \qquad \text{.....(2-26)}$$

Proses normalisasi dilakukan dengan cara membagi d_{ij}^k dan $d_{i(m+1)}^k$ dengan jumlah derajat keanggotaan setiap titik data i pada *cluster* k menggunakan persamaan (2-27) untuk d_{ij}^k dan persamaan (2-28) untuk $d_{i(m+1)}^k$.

$$d_{ij}^k = \frac{d_{ij}^k}{\sum_{k=1}^{c}\mu_{ki}} \qquad \text{..... (2-27)}$$

$$d_{i(m+1)}^k = \frac{d_{i(m+1)}^k}{\sum_{k=1}^{c}\mu_{ki}} \qquad \text{..... (2-28)}$$

Langkah selanjutnya adalah membentuk matriks *U* yang berukuran *n* x (*r* * (*m* + 1)) yang menjadi persamaan (2-29).

$$u_{i1} = d_{i1}^1; \qquad u_{i(2m+1)} = d_{i(m+1)}^2;$$

$u_{i2} = d_{i2}^{1};$ $u_{i(r*(m+1)-m)} = d_{i1}^{r};$

$u_{im} = d_{im}^{1};$ $u_{i(r*(m+1)-m+1)} = d_{i2}^{r};$

$u_{i(m+1)} = d_{i(m+1)}^{1};$ $u_{i(r*(m+1)-1)} = d_{im}^{r};$

$u_{i(m+2)} = d_{i1}^{2};$ $u_{i(r*(m+1))} = d_{i(m+1)}^{r};$

$u_{i(m+3)} = d_{i2}^{2};$ ...

$u_{i(2m)} = d_{im}^{2};$ dan seterusnya .....(2-29)

Vektor *z* adalah vektor *output* yang disajikan dengan persamaan (2-30).

$$z = \begin{bmatrix} z1 & z2 & ... & zn \end{bmatrix}^T \quad \text{..... (2-30)}$$

Dari vektor *k*, matriks *U*, dan vektor *z* ini dapat dibentuk suatu sistem persamaan linier. Persamaan linier tersebut ditunjukkan dengan persamaan (2-31).

$$U * k = z \quad \text{..... (2-31)}$$

Untuk mencari nilai koefisien *output* tiap – tiap aturan pada setiap variabel $(k_{ij}, i = 1,2,...,r;$ dan $j = 1,2,...,m+1).$ Matriks *U* bukan matriks bujursangkar, sehingga untuk menyelesaikan persamaan ini digunakan metode kuadrat terkecil. Untuk membentuk anteseden, setiap variabel *input* juga akan terbagi menjadi *r* himpunan *fuzzy*, dengan setiap himpunan memiliki fungsi keanggotaan *gauss*, dengan derajat keanggotaan data X_i, variabel ke-*j*, himpunan ke-*k* yang dirumuskan pada persamaan (2-32) [KUS-10:154].

$$\mu_{Var-j;\,Himp-k}[X_i] = e^{-\frac{(Xij-Ckj)^2}{2\sigma_j^2}} \quad \text{..... (2-32)}$$

Dengan aturan-aturan yang ditunjukkan dengan persamaan (2-33).

[*R*1] : IF (X_{i1} is *V*1*H*1) ∘ (X_{i2} is *V*2*H*1) ∘...∘ (X_{im} is *VmH*1)
THEN $Y = Z_1$

[*R*2] : IF (X_{i1} is *V*1*H*2) ∘ (X_{i2} is *V*2*H*2) ∘...∘ (X_{im} is *VmH*2)
THEN $Y = Z_2$

[*R*3] : IF (X_{i1} is *V*1*H*3) ∘ (X_{i2} is *V*2*H*3) ∘...∘ (X_{im} is *VmH*3)
THEN $Y = Z_3$

....

[*R*r] : IF (X_{i1} is *V*1*H*r) ∘ (X_{i2} is *V*2*H*r) ∘...∘ (X_{im} is *VmH*r)

THEN $Y = Z_r$ ..... (2-33)

dengan *VpHq* adalah variabel ke-*p* himpunan ke-*q*.

2.7 Least Square Estimator (LSE)

Menurut Jang, Chiu, dan Mizutani (1997), salah satu metode yang dapat menentukan metode kuadrat terkecil adalah dengan menggunakan metode *least square estimator*. Namun sebelumnya diperlukan identifikasi struktur dalam langkah ini. Tujuannya agar dapat menerapkan pengetahuan tentang target sistem untuk dapat menentukan kelas yang paling cocok dari model yang dicari [JAN-97]. Jika parameter konsekuen *k* dinotasikan seperti pada persamaan (2-34)

$$k^t = \left[k_0^t, k_1^t, k_2^t, ..., k_n^t\right]^T = \begin{bmatrix} k_0^t \\ k_1^t \\ k_2^t \\ ... \\ k_n^t \end{bmatrix} \quad \text{..... (2-34)}$$

maka TS inferensi untuk 1 sampai *n* data latih dapat ditulis seperti pada persamaan (2.35)

$$\begin{bmatrix} Y_1 \\ Y_2 \\ ... \\ Y_N \end{bmatrix} = \begin{bmatrix} U_1 & U_1 & ... & U_1 \\ U_2 & U_2 & ... & U_2 \\ ... & ... & ... & ... \\ U_n & U_n & ... & U_n \end{bmatrix} \bullet \begin{bmatrix} k^1 \\ k^2 \\ ... \\ k^m \end{bmatrix} \quad \text{..... (2-35)}$$

dimana dimensi untuk masing-masing matriks tersebut adalah $[Y] = N$ x 1, $[U] = N$ x $((n+1).m)$, $[k] = ((n+1).m)$ x 1

Untuk menghitung koefisien *k* dapat digunakan metode *least square estimator* (dengan menggunakan *pseudo matrix*), sehingga persamaan TS inferensi menjadi seperti pada persamaan (2-36).

$$[U]^T \bullet [U][k] = [U]^T \bullet [Y] \quad \text{..... (2-36)}$$

sehingga nilai koefisien dapat ditunjukkan seperti pada persamaan (2-37).

$$[k] = \left([U]^T \bullet [U]\right)^{-1} \bullet [U]^T \bullet [Y] \quad \text{..... (2-37)}$$

$$[k] = \left[k_0^t, k_1^t, k_2^t, ..., k_n^t\right]^T$$

Dimensi dari matriks parameter *k* adalah $[k] = (m \cdot (n+1) \times N) \cdot (N \times 1) = (m \cdot (n+1)) \times 1$, dimana m adalah jumlah aturan, N adalah jumlah data latih dan n adalah jumlah *fuzzy* input [JAN-97].

2.8 Akurasi

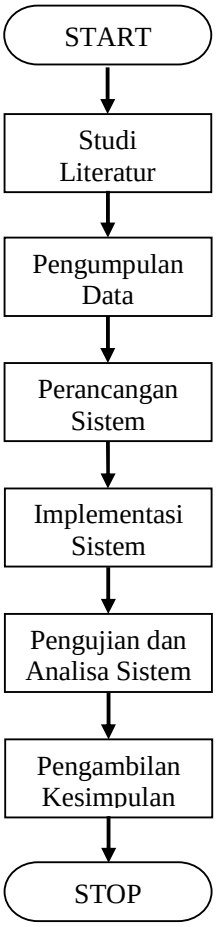
Salah satu cara untuk mengetahui hasil penelitian adalah melihat akurasi. Akurasi merupakan kedekatan suatu angka atau hasil pengujian terhadap angka ataupun data sebenarnya (*true value* atau *reference value*) [NUG-06]. Dalam penelitian ini perhitungan akurasi akan melibatkan hasil penelitian dan data nyata yang didapatkan dari sumber. Tingkat akurasi dan nilai akurasi dapat dihitung dengan persamaan (2-38) dan persamaan (2-39).

$$TingkatAkurasi = \frac{\sum dataUjiBenar}{\sum totalDataUji} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2\text{-}38)$$

$$Akurasi(\%) = \frac{\sum DataUji\ Benar}{Total\ DataUji} x100\% \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2\text{-}39)$$

BAB III
METODOLOGI DAN PERANCANGAN SISTEM

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam proses penelitian untuk mendapatkan aturan *fuzzy* pada deteksi dini risiko penyakit stroke yang dibangkitkan menggunakan algoritma *fuzzy c-means*. Beberapa proses yang dilakukan dalam penelitian ini meliputi studi literatur, pengumpulan data, perancangan sistem, implementasi sistem, pengujian dan analisa hasil dari program perangkat lunak yang akan dibuat, serta pengambilan kesimpulan sebagai catatan atas hasil penelitian. Bagan alur penelitian ditunjukkan oleh Gambar (3.1).



Gambar 3.1 Langkah-langkah Penelitian

3.1 Studi Literatur

Studi literatur merupakan sebuah proses mempelajari segala teori-teori dan dasar keilmuan yang mendukung penelitian serta meningkatkan pemahaman terhadap permasalahan yang diangkat dalam penelitian ini. Teori-teori yang dipelajari meliputi penyakit definisi stroke, jenis-jenis penyakit stroke, faktor risiko penyakit stroke, *clustering*, *fuzzy c-means* (FCM) dan *fuzzy inference system* (Sugeno orde-satu). Teori-teori tersebut berasal dari beberapa sumber, antara lain buku, jurnal, *e-book*, penelitian sebelumnya, situs-situs di internet, serta dari sumber pustaka lain yang dinilai dapat mendukung penelitian ini.

3.2 Data Penelitian

Data yang digunakan dalam penelitian ini merupakan data yang diambil dari data penelitian deteksi dini risiko penyakit stroke yang telah dilakukan oleh Ahmad Fashel Sholeh pada tahun 2012, yang diperoleh dari hasil data rekam medik pada Rumah Sakit XYZ. Dataset terdiri dari 10 atribut, antara lain tekanan darah, kadar gula darah, kolesterol total, *Low Density Lipoprotein* (LDL), usia, asam urat, jenis kelamin, *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko. Data yang digunakan pada penelitian ini berjumlah 190 data latih dan 30 data uji. Data latih digunakan sebagai bahan pembelajaran bagi algoritma *clustering* dalam membangkitkan aturan *fuzzy*. Sedangkan data uji merupakan data yang akan diuji coba pada sistem untuk mengetahui tingkat akurasi dari hasil penelitian.

3.3 Analisis dan Perancangan Sistem

Proses perancangan dilakukan sebagai dasar dalam proses implementasi. Adapun tahapan dalam perancangan ini adalah perancangan diagram alir algoritma, perhitungan manual, perancangan *database*, dan perancangan antarmuka.

3.3.1. Deskripsi Umum Sistem

Secara umum sistem yang akan dibangun merupakan pengujian dari algoritma *clustering* untuk pembangkitan aturan *fuzzy*. Aturan ini dibangkitkan dengan teknik *clustering* menggunakan metode *fuzzy c-means clustering* sebagai media pelatihan terhadap data latih. *Input* dari sistem ini adalah parameter

clustering dan data rekam medik dari Rumah Sakit XYZ. *Output* dari sistem ini merupakan nilai deteksi dini risiko penyakit stroke.

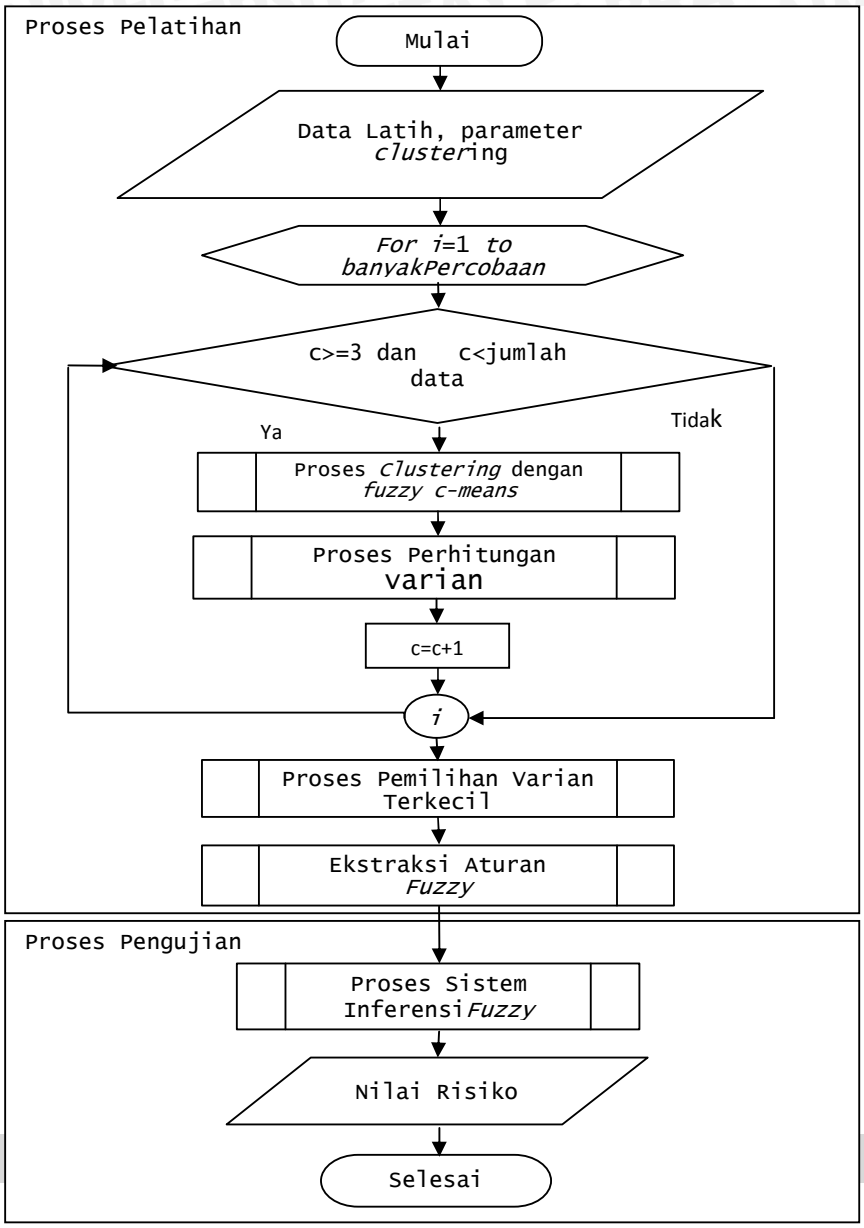
Proses pertama dari sistem ini adalah mengklaster data latih menggunakan algoritma *fizzy c-means clustering* dengan beberapa parameter yang ditentukan sebelumnya. Hasil dari *clustering* adalah pusat *cluster*, derajat keanggotaan, dan kelompok data yang akan digunakan pada proses ekstraksi aturan *fuzzy*.

Langkah berikutnya adalah mengekstraksi aturan *fuzzy*. Ekstraksi aturan *fuzzy* menggunakan pusat *cluster* dan sigma dari hasil *clustering*. Nantinya jumlah aturan sama dengan jumlah *cluster* yang terbentuk. Derajat keanggotaan data latih terhadap *cluster* dihitung menggunakan pusat *cluster* dan sigma. Derajat keanggotaan dan matriks *U* nantinya digunakan untuk mencari koefisien *output* menggunakan metode kuadrat terkecil. Metode kuadrat terkecil yang dipakai adalah LSE (*least square estimator*).

Aturan yang terbentuk diterapkan pada *fuzzy inference system* model sugeno orde-satu. Pemilihan model sugeno orde-satu dikarenakan konsekuennya berupa persamaan linier. Proses dari metode sugeno orde-satu dimulai dengan mencari derajat keanggotaan tiap parameter pada data uji menggunakan fungsi *gauss*. Kemudian menghitung *fire strength* dan nilai *Z* (proses *defuzzy*). Setelah nilai Z diketahui langkah terakhir adalah menghitung nilai derajat keanggotaan Z terhadap kelas tinggi, sedang, dan rendah untuk mlakukan deteksi dini risiko penyakit stroke. Hasil akhir dari sistem ini adalah tingkat risiko penyakit stroke dalam diri seseorang.

3.3.2. Perancangan Sistem

Secara garis besar, sistem terdiri dari dua proses utama, yaitu proses pelatihan yang terdiri dari proses *clustering*, perhitungan standar deviasi, sampai dihasilkan jumlah *cluster* ideal; dan proses pengujian akurasi aturan *fuzzy* dengan menggunakan *fuzzy* inferensi sistem sugeno orde-satu. Gambaran umum perancangan sistem ditunjukkan pada Gambar (3.2).



Tahapan pada proses perancangan sistem berdasarkan Gambar (3.2) adalah sebagai berikut:

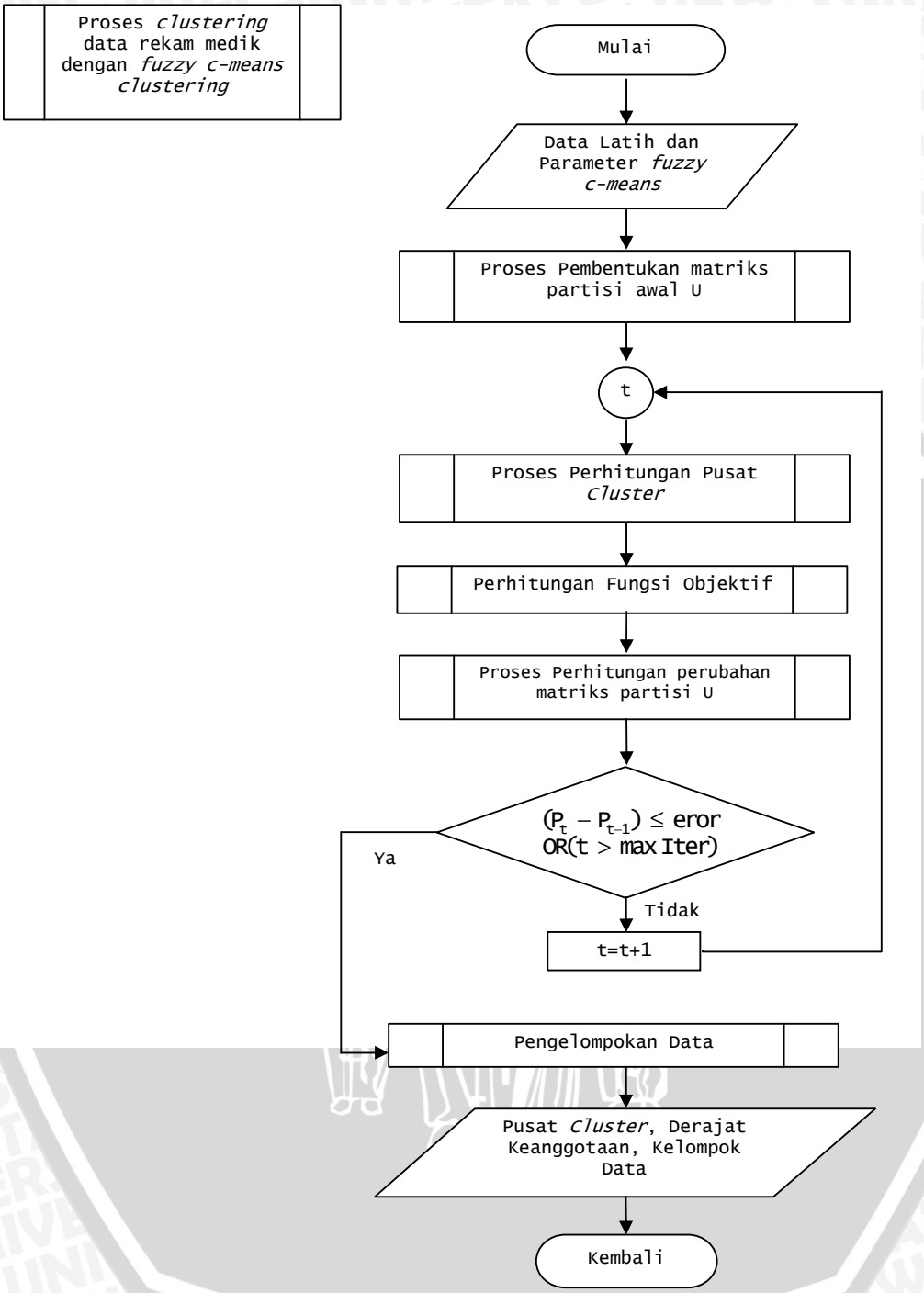
1. Proses pelatihan dari data rekam medik digunakan untuk membangkitkan aturan *fuzzy* secara otomatis dengan menggunakan metode *fuzzy c-means clustering.*
2. *Input* pada proses ini adalah data rekam medik sebagai data latih dan parameter *clustering*. Data rekam medik yang digunakan dalam proses ini terdiri dari tekanan darah, kadar gula, kolesterol total, *Low Density*

Lipoprotein (LDL), usia, asam urat, jenis kelamin, *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko. Sedangkan parameter *clustering* terdiri atas jumlah *cluster* (c), jumlah data (n), *error* terkecil yang diharapkan (ξ) dan iterasi maksimum. Hasil dari algoritma *fuzzy c-means clustering* adalah pusat *cluster* dan standar deviasi yang nantinya digunakan untuk menghitung derajat keanggotaan. Proses *clustering* ini akan dilakukan sebanyak jumlah percobaan dengan inisialisasi parameter *clustering* yang berbeda untuk mendapatkan jumlah *cluster* ideal.

3. Proses perhitungan varian adalah upaya untuk mengetahui nilai varian tiap *cluster*.
4. Proses pemilihan jumlah *cluster* dengan nilai varian terkecil digunakan untuk mengetahui berapa jumlah *cluster* yang tepat guna diterapkan pada proses selanjutnya.
5. Hasil *cluster* yang akan dipilih untuk digunakan pada proses selanjutnya merupakan hasil *cluster* dengan nilai varian terkecil.
6. Proses pembangkitan aturan *fuzzy* bisa memanfaatkan jumlah *cluster* yang didapatkan pada proses sebelumnya. Jumlah aturan yang terbentuk sama dengan jumlah *cluster*.
7. Tahap terakhir adalah proses pengujian, dimana pada proses ini menggunakan data rekam medik sebagai data uji yang terdiri dari tekanan darah, kadar gula, kolesterol total, *Low Density Lipoprotein* (LDL), usia, asam urat, jenis kelamin, *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko. Selain itu dalam proses pengujian ini juga menggunakan aturan *fuzzy* yang terbentuk dari proses sebelumnya. Pada proses ini nilai risiko akan dihitung menggunakan sistem inferensi *fuzzy* sugeno orde-satu.

3.3.2.1 Proses *Cluster*ing dengan Fuzzy C-Means *Cluster*ing

Proses *clustering* dengan *fuzzy c-means clustering* adalah proses pelatihan terhadap data latih untuk pengelompokan data yang hasilnya digunakan untuk pembangkitan aturan fuzzy menggunakan algoritma *fuzzy c-means clustering*. Alur proses *clustering* dengan *fuzzy c-means clustering* digambarkan oleh diagram alir yang ditunjukkan pada Gambar (3.3).



Gambar 3.3 Alur Proses *Fuzzy C-Means clustering*

Alur proses *clustering* data rekam medik terdiri atas 5 subproses, yaitu pembentukan matriks partisi awal U, perhitungan pusat *cluster*, perhitungan

fungsi objektif, perhitungan perubahan matriks partisi U, dan pengelompokan data.

Input proses *fuzzy c-means clustering* berupa data latih dan parameter *cluster*ing. Sedangkan *output* proses ini berupa matriks pusat *cluster*, matriks partisi U, dan kelompok data. Matriks pusat *cluster* berisi pusat data atribut pada setiap *cluster* (V_{kj}), matriks partisi U berisi kecenderungan data latih pada semua *cluster* (μ_{ik}) dan kelompok data berisi hasil pengelompokan data latih berdasarkan kecenderungan data terhadap suatu *cluster.*

1. Proses pembentukan matriks partisi awal

Proses pembentukan matriks partisi awal U menggunakan masukan parameter jumlah data latih dan jumlah *cluster* yang akan dibentuk. *Output* proses ini adalah matriks derajat keanggotaan data terhadap *cluster* (μ_{ik}) dengan dimensi jumlah data x jumlah *cluster*. Rincian alur proses pembentukan matriks partisi awal U adalah sebagai berikut:

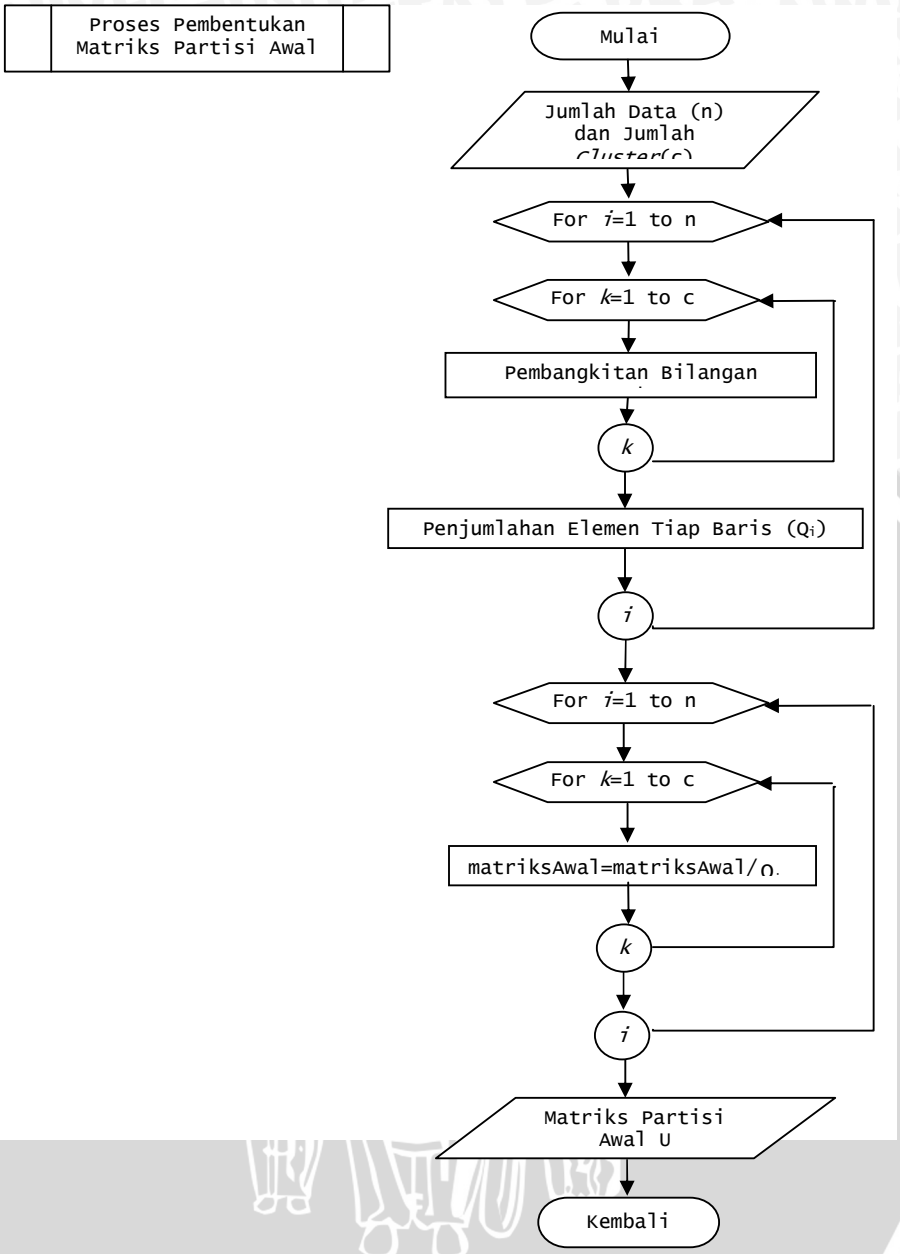
1. Pembangkitkan bilangan *random*

   Bilangan *random* yang dibangkitkan berguna untuk merepresentasikan derajat keanggotaan suatu data ke-i ke dalam *cluster* ke-k (μ_{ik}). Rentang bilangan random yang ditentukan yaitu bilangan antara 0 sampai 1.
2. Penjumlahan elemen setiap kolom

   Proses ini menjumlah bilangan random μik setiap iterasi ke-i ($Q_i = \mu_{i1} + \mu_{i2} + \mu_{i3} + \ldots + \mu_{ik}$) berdasarkan Persamaan (2-1) sehingga didapatkan Q_i satu dimensi berukuran n.
3. Perhitungan nilai elemen matriks

   Setelah perhitungan nilai Q_i, dilakukan perhitungan dengan Persamaan (2-2) untuk matriks partisi awal U sehingga menghasilkan derajat keanggotaan (μ_{ik}). Jumlah μ_{ik} pada setiap iterasi ke-i harus sama dengan 1 ($\mu_{i1} + \mu_{i2} + \mu_{i3} + \ldots + \mu_{ik} \neq 1$) apabila tidak bernilai 1 maka terjadi kesalahan selama perhitungan derajat keanggotaan μik.

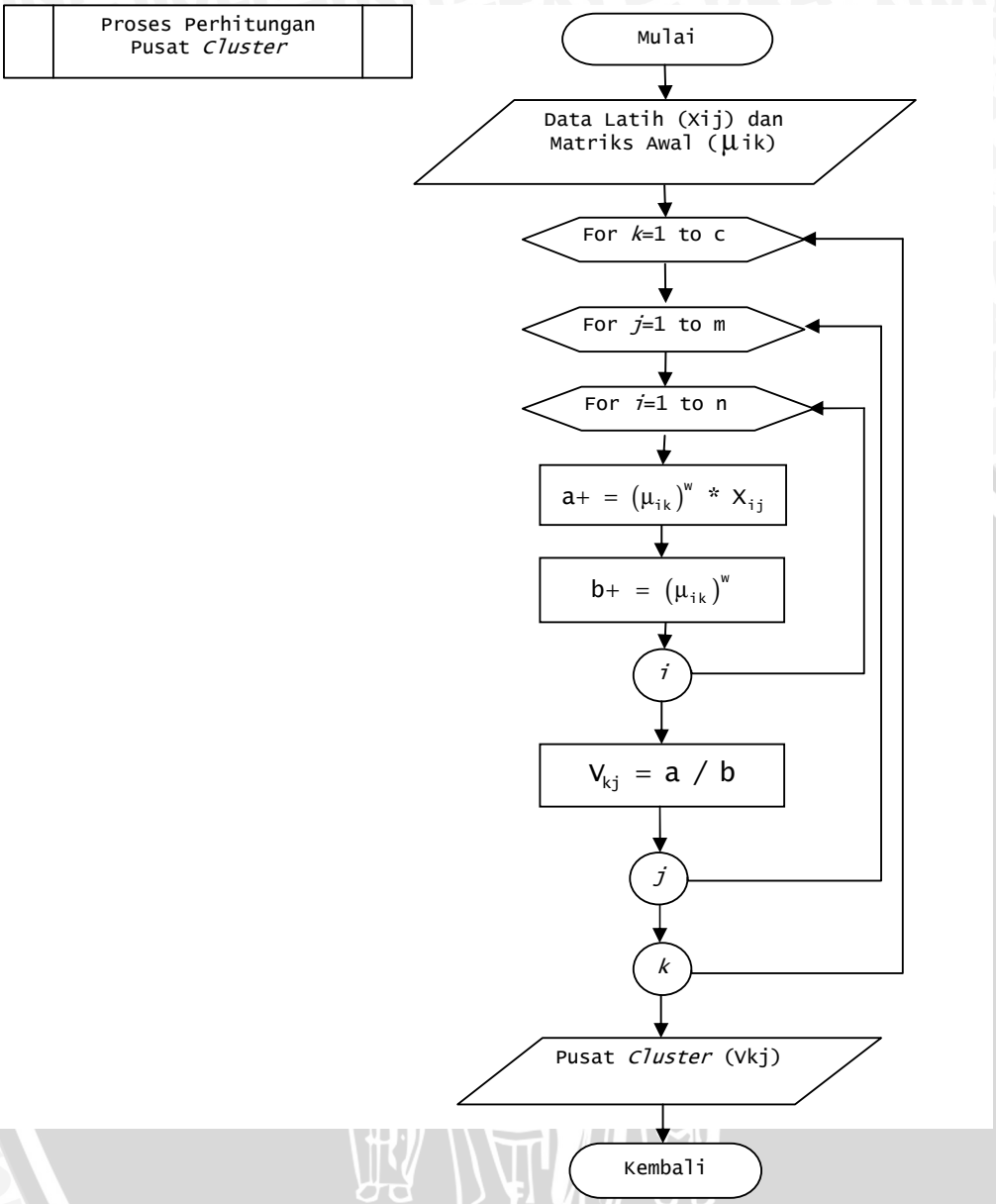
Alur proses pembentukan matriks partisi awal U ditunjukkan oleh Gambar (3.4) dibawah ini.



Gambar 3.4 Alur proses pembentukan matriks partisi awal U

2. Proses perhitungan pusat *cluster*

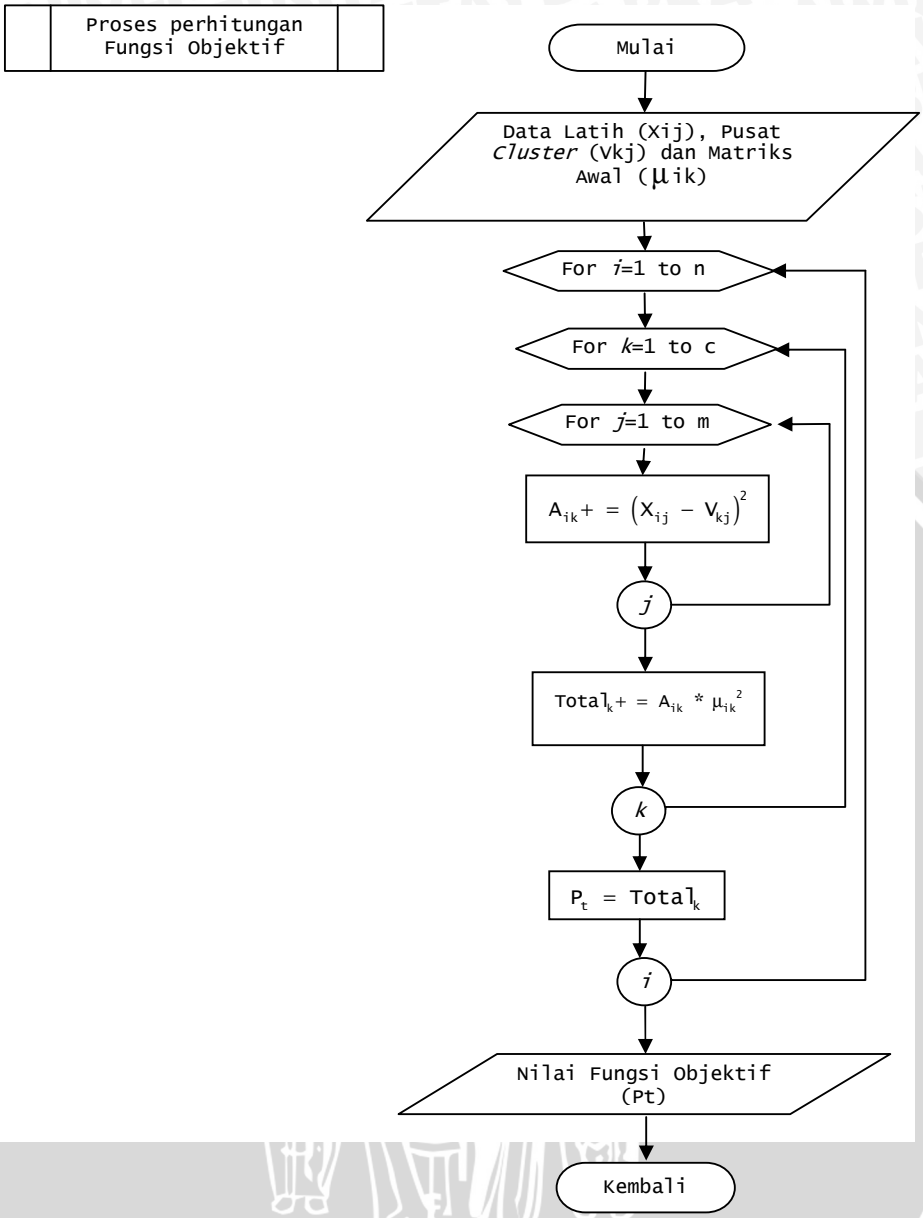
Input dari proses perhitungan pusat *cluster* adalah data latih (X_{ij}) dan matriks partisi awal U (μ_{ik}). Perhitungan pusat *cluster* dilakukan menurut persamaan (2-3). *Output* dari proses perhitungan pusat *cluster* adalah pusat *cluster* (V_{kj}). Alur proses perhitungan pusat *cluster* ditunjukkan oleh Gambar (3.5).



Gambar 3.5 Alur Proses Perhitungan Pusat *Cluster*

3. Proses perhitungan fungsi objektif

Proses perhitungan fungsi objektif menggunakan *input* data latih (X_{ij}), pusat *cluster* (V_{kj}) dan matriks awal (μ_{ik}). Perhitungan fungsi objektif dilakukan berdasarkan persamaan (2-4). Alur proses perhitungan fungsi objektif ditunjukkan pada Gambar (3.6).



Gambar 3.6 Alur Proses Perhitungan Fungsi Objektif

Rincian alur proses perhitungan fungsi objektif adalah sebagai berikut:

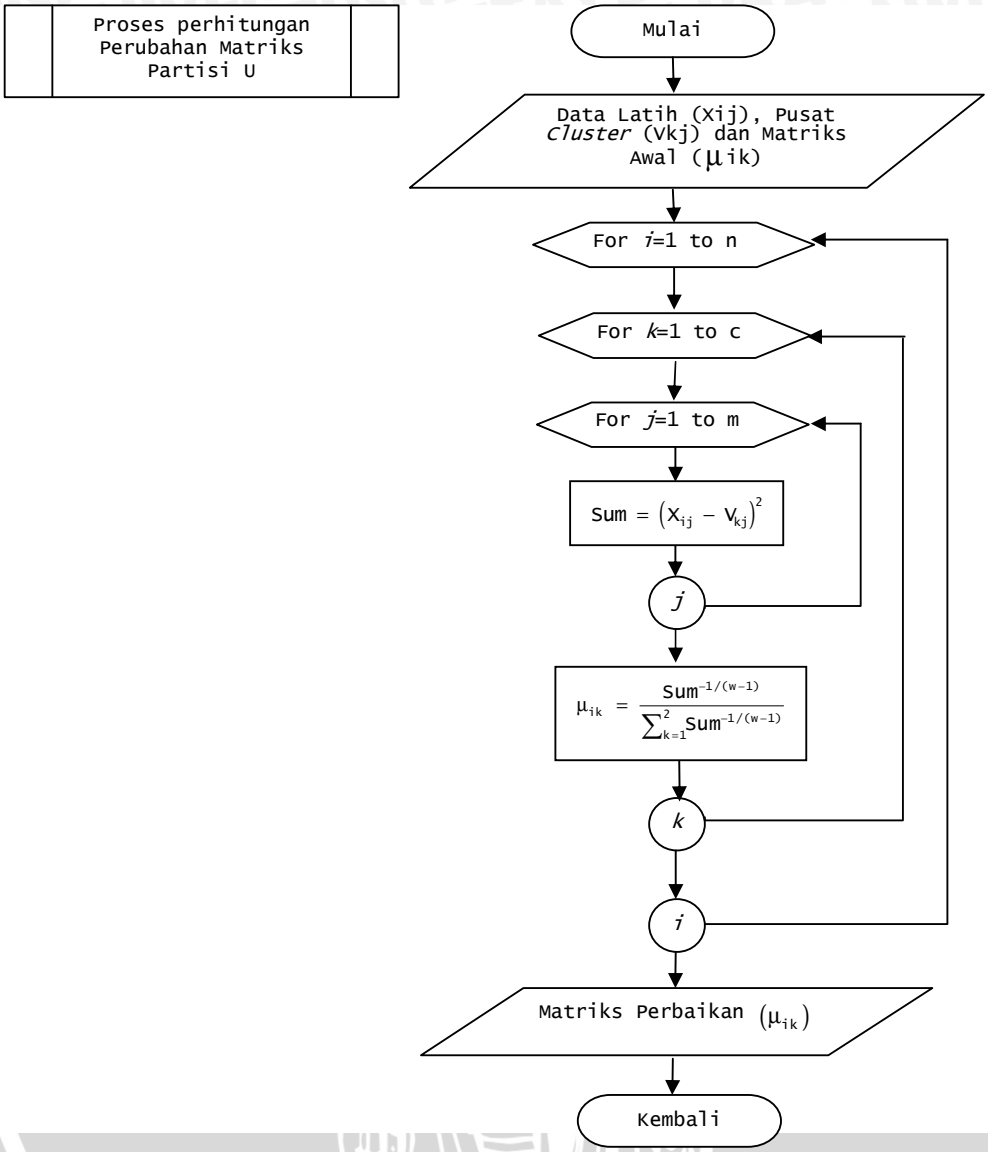
- Pada setiap iterasi ke-i dilakukan iterasi k sebanyak c (jumlah *cluster* yang akan dibentuk) untuk melakukan perhitungan total jarak atribut (A_{ik}). Total jarak (A_{ik}) didapatkan dari penjumlahan jarak semua atribut ($A_{ik} = (X_{i1} – V_{k1})^2 + (X_{i2} – V_{k2})^2 + (X_{i3} – V_{k3})^2 + … + (X_{ij} – V_{kj})^2$)

- Nilai A_{ik} dikalikan dengan derajat keanggotaan (μ_{ik}) kemudian ditotal untuk setiap iterasi i. Pada akhirnya, semua total perhitungan akan dijumlah sehingga menghasilkan satu nilai fungsi objektif (P_t).
- Selisih Nilai P_t dengan nilai P_{t-1} dihitung kemudian dibandingkan dengan nilai kesalahan minimum (ξ) yang telah ditentukan untuk pemeriksaan kondisi berhenti pada proses *clustering* data. Jika $|P_t - (P_{t-1})| < \xi$ atau t > maxIter maka perulangan pada proses *clustering* dihentikan.

4. Proses perubahan matriks partisi U

Proses perubahan matriks partisi U dilakukan untuk memperbaiki nilai derajat keanggotaan (μ_{ik}) berdasarkan nilai pusat *cluster*. Seperti pada perhitungan fungsi objektif, proses perubahan matriks partisi U memiliki *input* berupa data latih rekam medik (X_{ij}), pusat *cluster* (V_{kj}), dan matriks awal (μ_{ik}). *Output* proses ini adalah perbaikan derajat keanggotaan μik yang merupakan hasil perubahan matriks partisi U. Perhitungan perubahan matriks partisi U dapat dihitung dengan menggunakan persamaan (2-5).

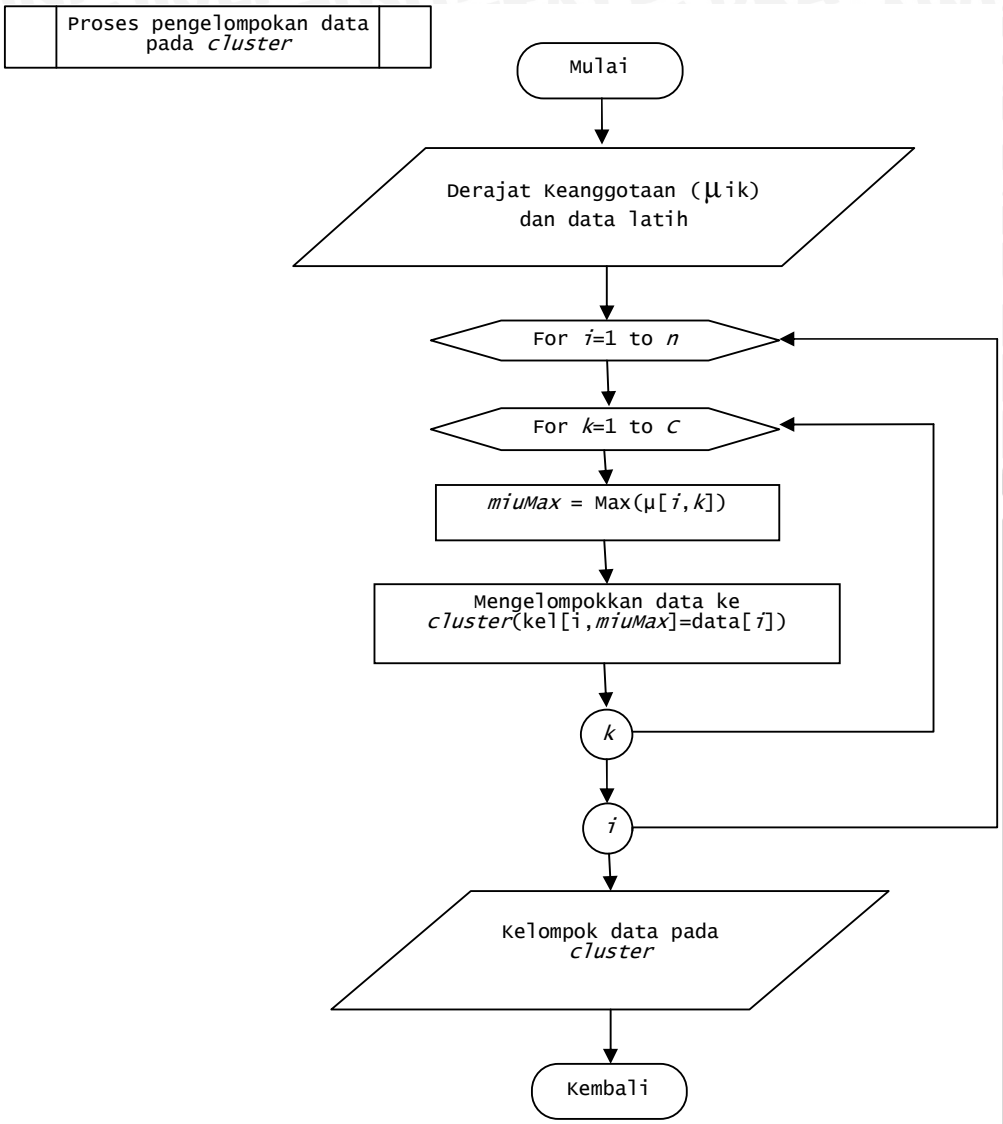
Sama seperti perhitungan nilai elemen matriks pada matriks partisi awal, jumlah μik pada setiap iterasi ke-i adalah 1 ($\mu_{i1} + \mu_{i2} + \mu_{i3} + \ldots + \mu_{ik} = 1$). Apabila jumlah μik pada setiap iterasi ke-i tidak sama dengan 1 ($\mu_{i1} + \mu_{i2} + \mu_{i3} + \ldots + \mu_{ik} \neq 1$), maka terjadi kesalahan selama proses *clustering* data. Alur proses perubahan matriks partisi U ditunjukkan oleh Gambar (3.7).



Gambar 3.7 Alur proses perubahan matriks partisi U

5. Proses pengelompokan data

Proses pengelompokan data pada *cluster* merupakan proses untuk mengelompokkan data latih ke dalam *cluster* yang terbentuk berdasarkan derajat keanggotaan masing-masing titik data terhadap pusat *cluster*. Alur proses pengelompokan data pada *cluster* ditunjukkan pada Gambar (3.8).



Gambar 3.8 Alur proses pengelompokan data

Penjelasan langkah-langkah pengelompokan data seperti pada Gambar (3.8) adalah sebagai berikut:

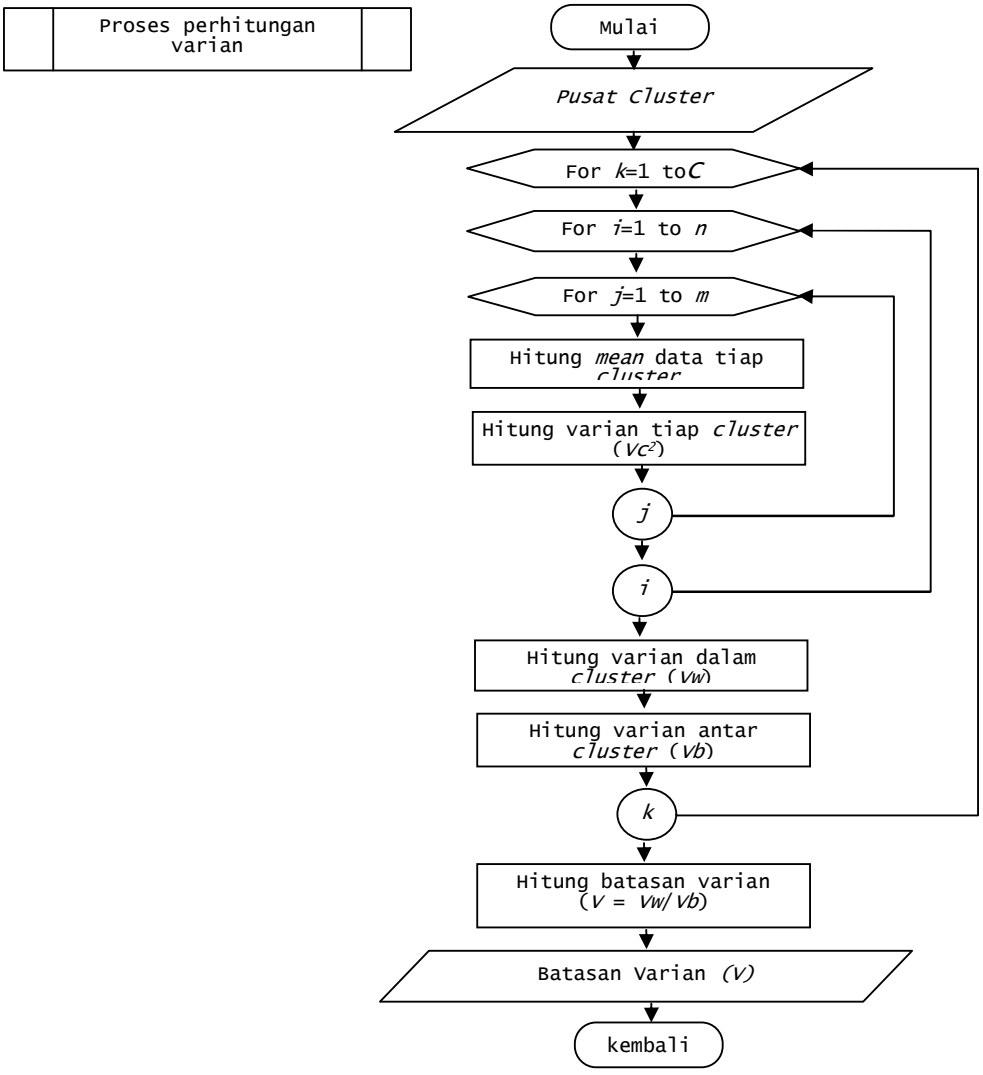
1. Masukan pada proses ini adalah derajat keanggotaan dan data latih.
2. Iterasi dari *i* = 1 sampai *n*, dilakukan langkah berikut:

   Iterasi dari *j* = 1 sampai *C*, dilakukan pemilihan nilai derajat keanggotaan tertinggi pada setiap titik data, sehingga dapat disimpulkan bahwa titik data menempati *cluster* dengan derajat keanggotaan tertinggi.

3. Hasil akhir dari proses ini adalah kelompok data yang masuk sesuai dengan *cluster*-nya masing-masing.

3.3.2.2 Proses Perhitungan Varian

Pada proses ini akan dihitung nilai varian pada tiap hasil *cluster* yang terbentuk sebagai langkah untuk menganalisa *cluster*. Analisa *cluster* digunakan untuk mengetahui hasil *cluster* yang ideal untuk proses pembangkitan aturan *fuzzy*. Alur proses perhitungan varian dapat dilihat pada diagram alir yang ditunjukkan pada Gambar (3.9).



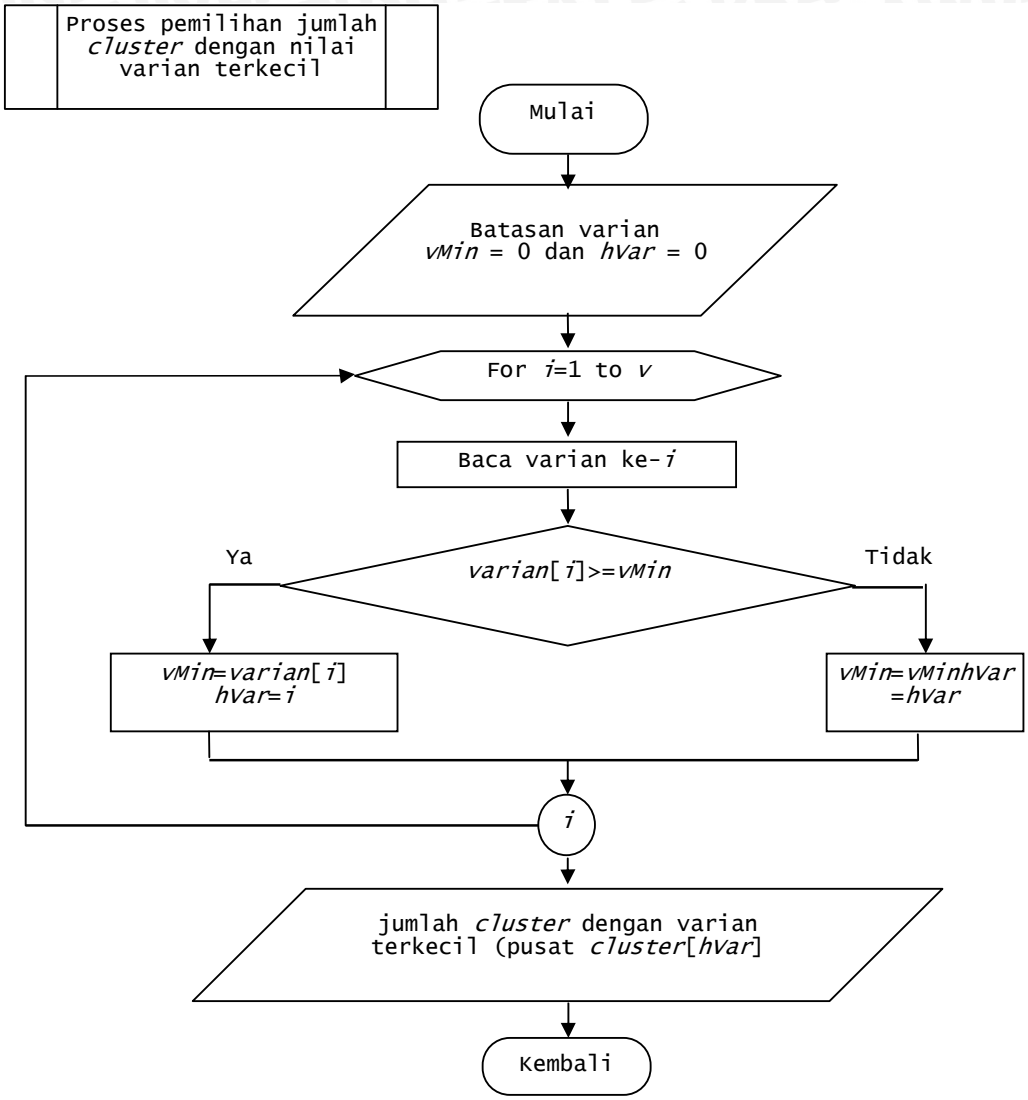
Gambar 3.9 Alur Proses Perhitungan Varian

Berikut langkah-langkah perhitungan varian pada proses analisa *cluster* sesuai dengan Gambar (3.9):

1. Mengelompokkan data latih berdasarkan hasil *cluster* yang terbentuk.
2. Menghitung nilai varian pada tiap *cluster* seperti pada persamaan (2-8). Nilai dari varian ini akan digunakan untuk menghitung nilai *variance within cluster*.
3. Menghitung nilai *variance within cluster* seperti pada persamaan (2-9) untuk mengetahui sebaran data dalam sebuah *cluster*.
2. Menghitung nilai *variance between cluster* seperti pada persamaan (2-10) untuk mengetahui sebaran data antar *cluster*.
3. Menghitung batasan varian dengan persamaan (2-11). Hasil dari perhitungan inilah yang nantinya dijadikan bahan pertimbangan untuk dapat menentukan jumlah *cluster* mana yang akan diambil untuk dijadikan bahan pada proses pembangkitan aturan *fuzzy*.

3.3.2.3 Proses Pemilihan Jumlah *Cluster* dengan Varian Terkecil

Proses pemilihan jumlah *cluster* dengan varian terkecil adalah proses untuk mendapatkan jumlah *cluster* yang dijadikan bahan pada proses pembangkitan aturan *fuzzy*. Berdasarkan literatur pada bab dua, *cluster* yang baik adalah *cluster* yang memiliki varian kecil. Hal ini dapat diasumsikan bahwa sebaran data pada *cluster* tidak memiliki variasi yang tinggi. Langkah-langkah pemilihan jumlah *cluster* dengan varian terkecil ditunjukkan oleh Gambar 3.10:



Gambar 3.10 Alur Proses Pemilihan Jumlah *Cluster* Dengan Varian Terkecil

Tahapan dari proses pemilihan jumlah *cluster* dengan varian terkecil adalah sebagai berikut :

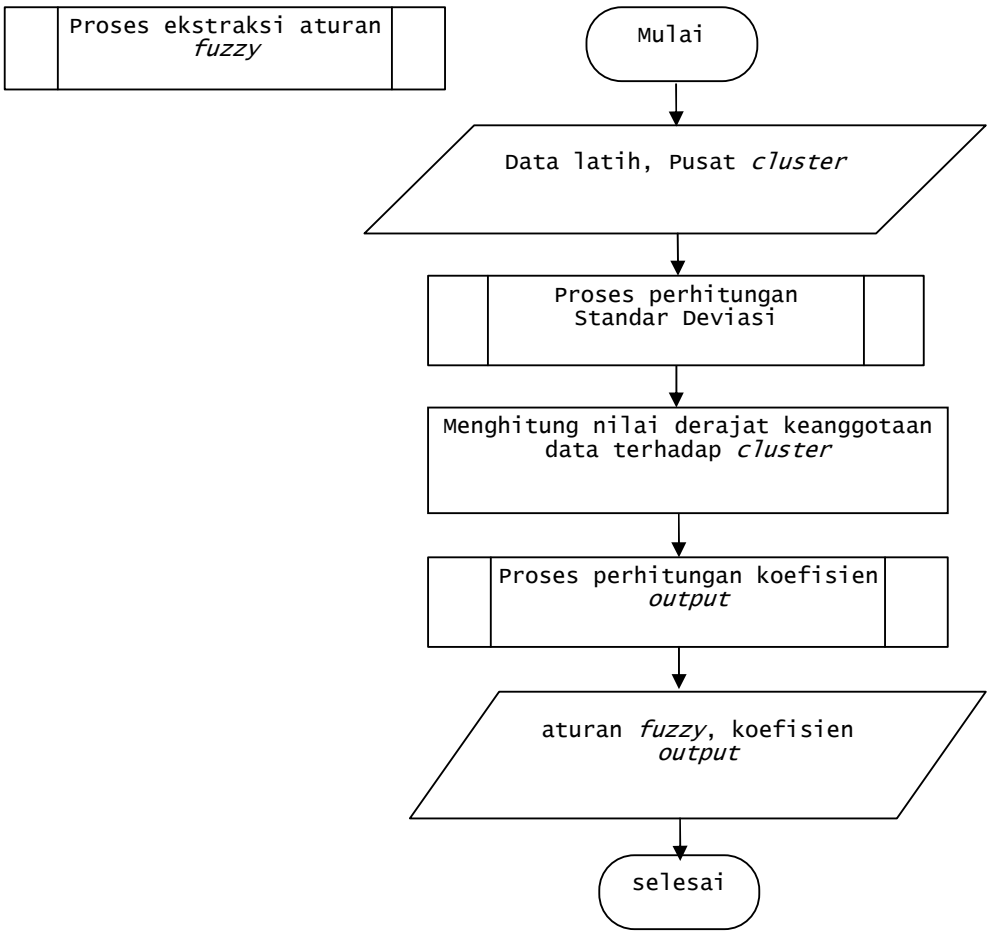
1. Masukan pada proses ini adalah nilai batasan varian (*varian*) pada beberapa jumlah *cluster* yang diujikan. Inisialisasi awal untuk variabel *v* sebagai jumlah varian, *vMin* = 0 dimana *vMin* adalah variabel untuk menampung nilai varian terkecil, dan nilai *hVar* sebagai variabel untuk menampung indeks posisi pusat *cluster* yang nilai variannya terkecil.
2. Iterasi *i*=1 sampai *v*, dilakukan langkah berikut:

a. Lakukan pengecekan: jika *varian* ≥ *vMin*, maka nilai *vMin* = *varian*[*i*] dan nilai *hVar* = *i*.
b. Lakukan pengecekan: jika *varian*[*i*] < *vMin*, maka nilai *vMin* = *vMin* dan nilai *hVar* = *hVar*.

3. Hasil akhir dari proses ini adalah jumlah *cluster* yang memiliki nilai varian terkecil. Jumlah *Cluster* dengan varian terkecil memiliki varian terkecil, dimana *hVar* adalah indeks posisi dari pusat *cluster*.

3.3.2.4 Proses Ekstraksi Aturan Fuzzy dari *Cluster*

Proses ekstraksi aturan *fuzzy* merupakan proses untuk mengubah *cluster* yang telah terbentuk menjadi kumpulan aturan yang nantinya akan diterapkan pada sistem inferensi *fuzzy* model sugeno orde-satu. Alur proses ekstraksi aturan *fuzzy* dari *cluster* ditunjukkan oleh Gambar (3.11) dibawah ini.



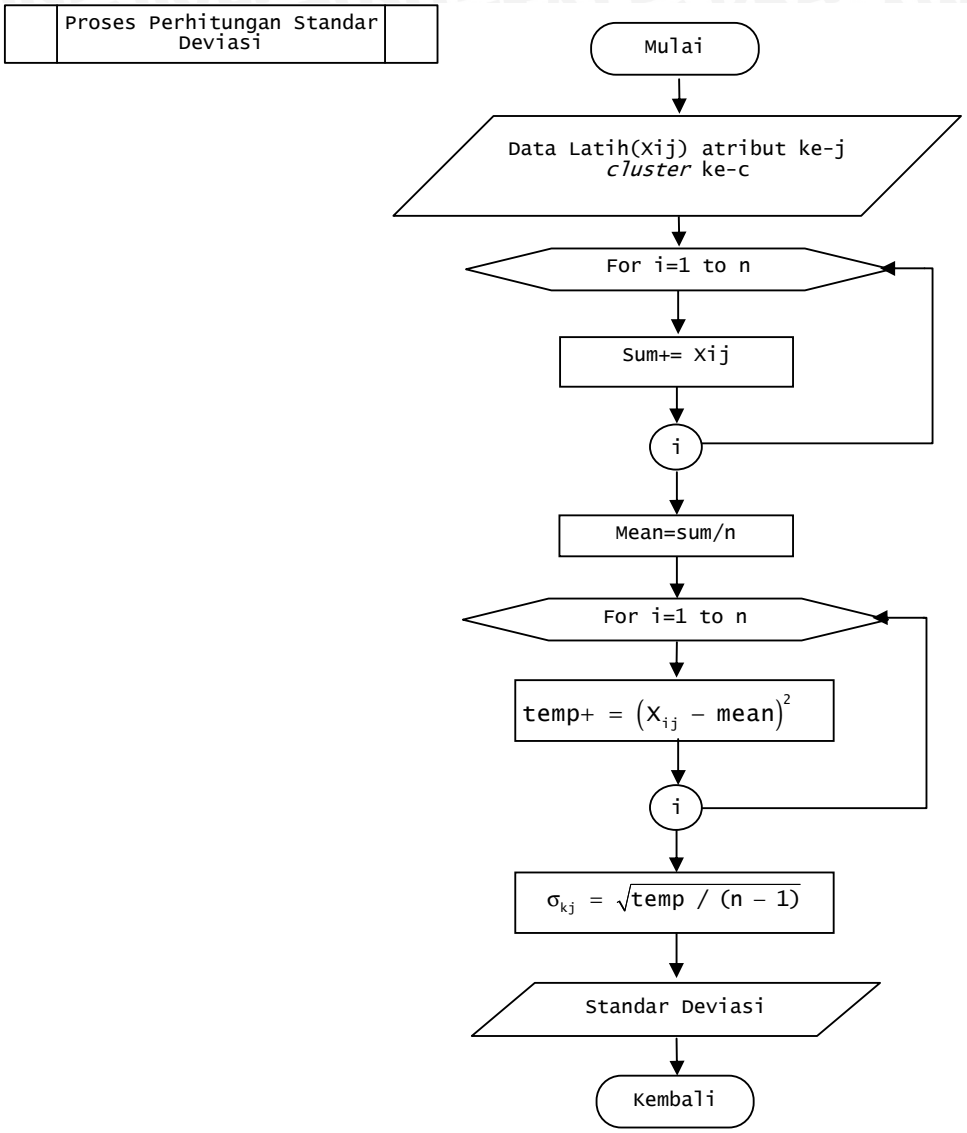
Gambar 3.11 Alur proses ekstraksi aturan fuzzy

Tahapan dari proses ekstraksi aturan *fuzzy* dapat dilihat pada penjelasan sebagai berikut:

1. Menyiapakan data latih (X_{ij}) dan pusat *cluster* (V_{kj})
2. Menghitung standar deviasi berdasarkan Persamaan (2-24)
3. Menghitung nilai derajat keanggotaan data menggunakan fungsi gauss terhadap masing-masing *cluster* untuk mengetahui kelompok data pada suatu *cluster* dengan menggunakan Persamaan (2-29).
4. Menghitung nilai koefisien *output*.
5. Hasil akhir dari proses ini adalah aturan *fuzzy* dan koefisien *output*.

1. Proses perhitungan standar deviasi

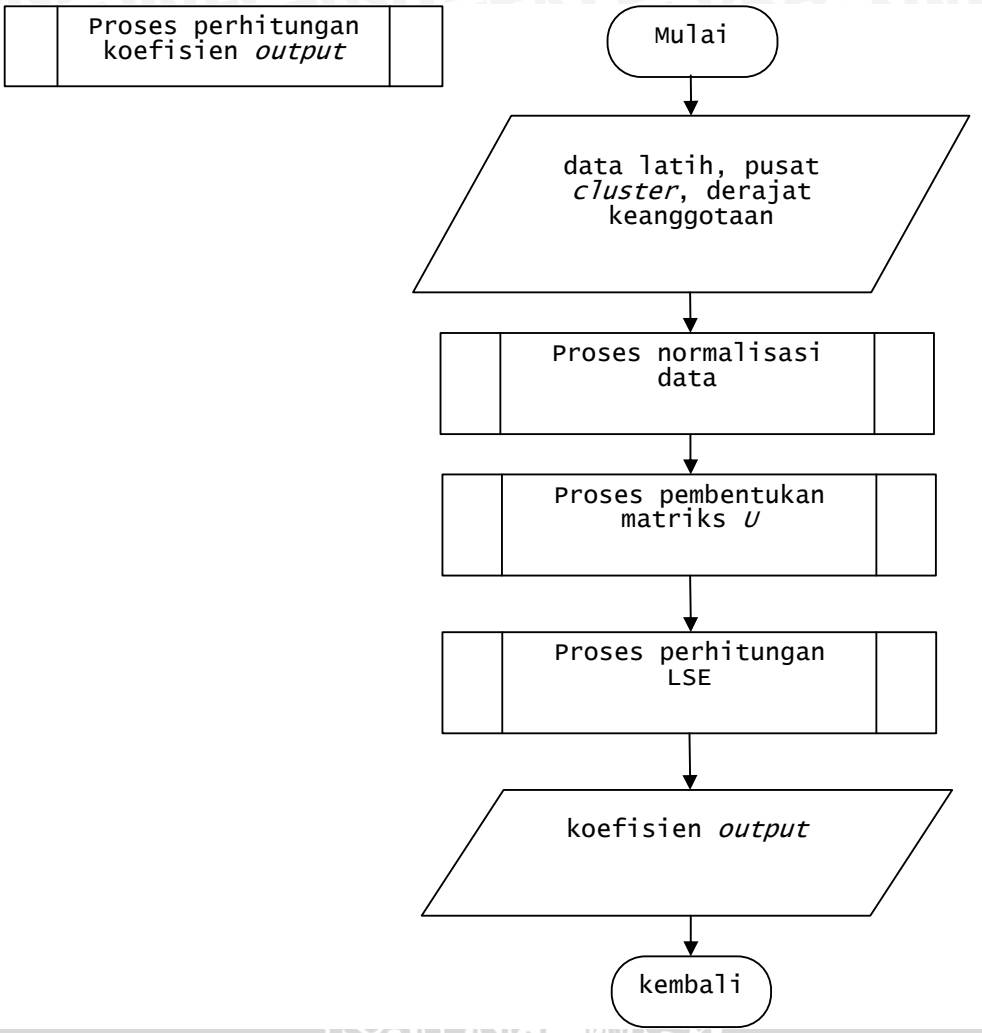
Standar deviasi ini dihitung dengan menggunakan persamaan (2-24). Bersama dengan pusat *cluster,* standar deviasi berguna untuk membentuk derajat keanggotaan data terhadap *cluster* menggunakan fungsi *Gauss*. Derajat keanggotaan (μ_{ik}) yang dihasilkan memiliki kombinasi nilai pusat *cluster* (V_{kj}) dan standar deviasi (σ_{kj}) yang berbeda untuk setiap atribut pada setiap aturan. Alur proses perhitungan standar deviasi ditunjukkan oleh Gambar (3.12).



Gambar 3.12 Alur proses perhitungan standar deviasi

2. Proses Perhitungan Koefisien *Output*

Proses perhitungan koefisien *output* adalah proses untuk mendapatkan nilai koefisien *output* pada sistem inferensi *fuzzy*. Koefisien *output* adalah suatu konstanta yang mempengaruhi variabel dalam menentukan target *output* dari sistem inferensi *fuzzy*. Alur proses perhitungan koefisien *output* secara umum ditunjukkan pada Gambar (3.13).



Gambar 3.13 Alur proses perhitungan koefisien output

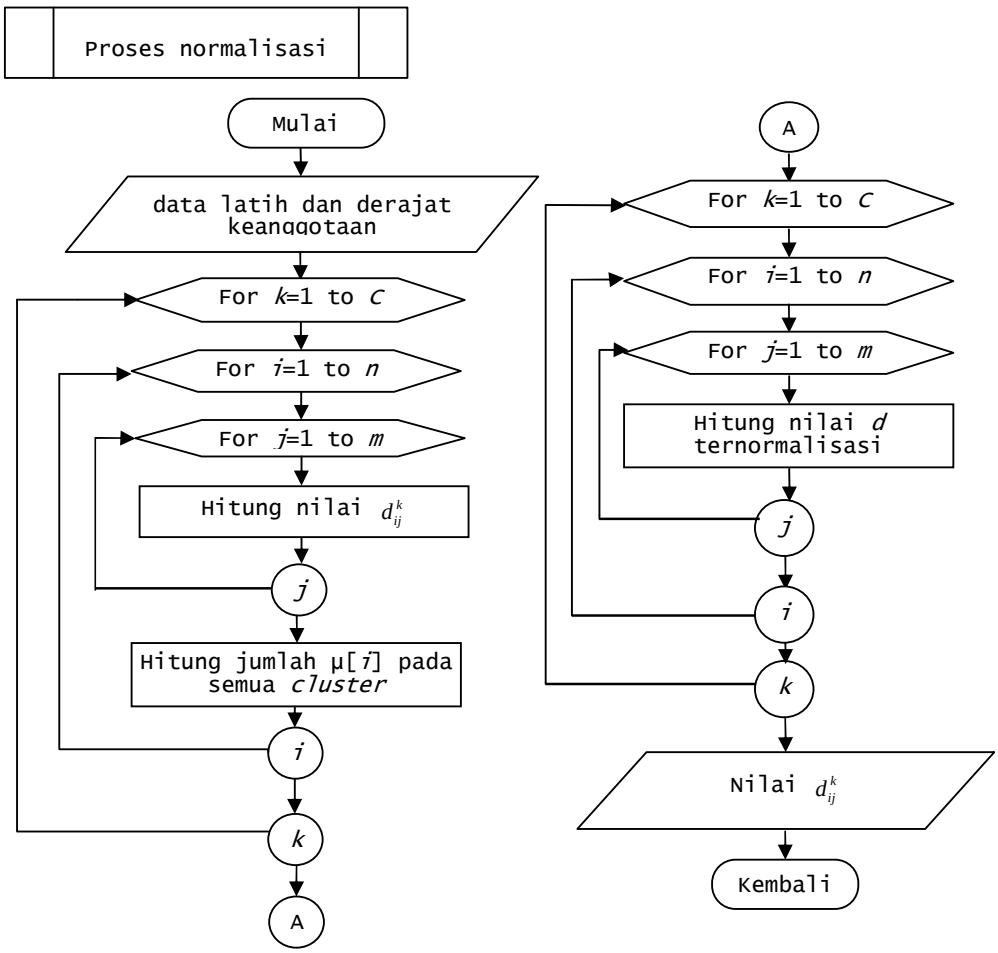
- **Proses Normalisasi**

Proses normalisasi adalah proses untuk mendapatkan nilai d_{ij}^{k} yang nantinya digunakan untuk pembentukan matriks *U*. Tahapan proses normalisasi adalah sebagai berikut:

1. Masukan untuk proses normalisasi adalah data latih dan derajat keanggotaan masing-masing titik data (μ).
2. Menghitung nilai d_{ij}^{k} dengan menggunakan persamaan (2-30).
3. Menjumlahkan derajat keanggotaan (μ_{ik}) semua *cluster*.

4. Menghitung nilai d_{ij}^{k} ternormalisasi dengan cara membaginya dengan $d_{i(m+1)}^{k}$.
5. Hasil akhir dari proses ini adalah nilai d_{ij}^{k} yang nantinya digunakan pada proses pembentukan matriks *U*.

Alur proses normalisasi ditunjukkan pada Gambar (3.14) dibawah ini.

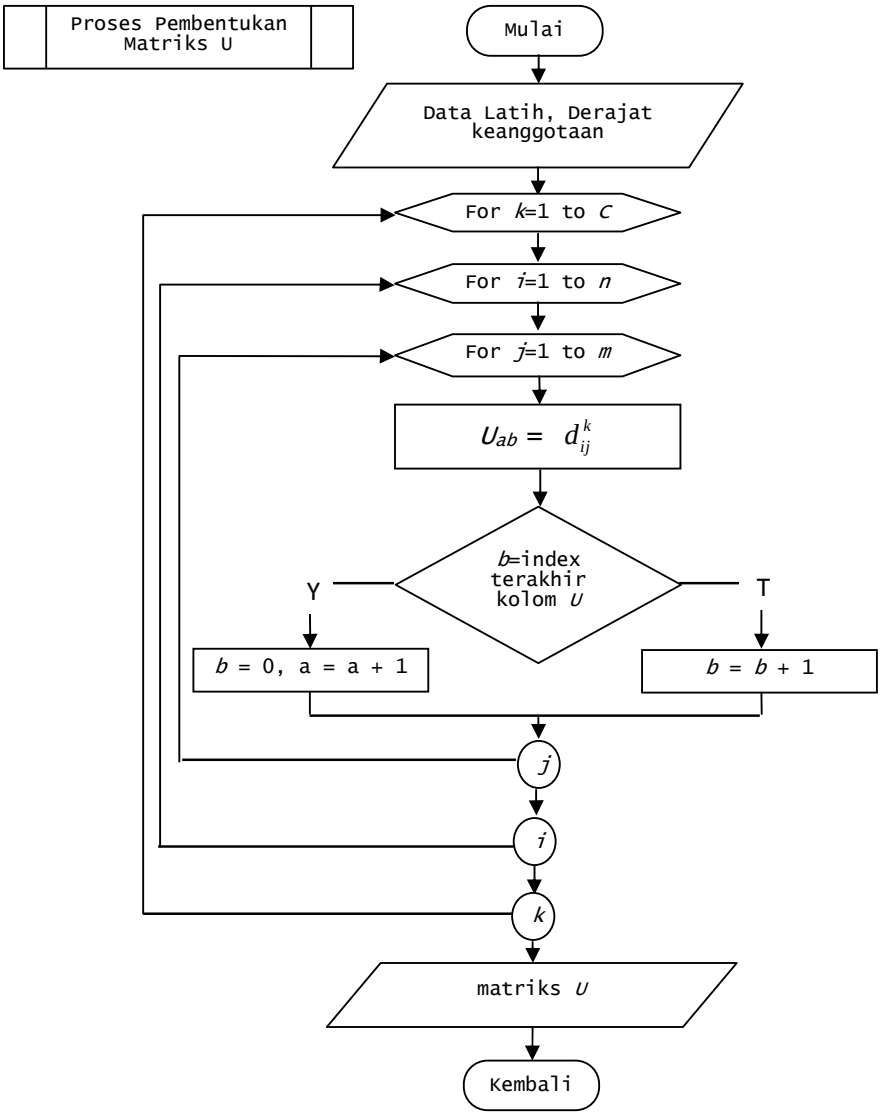


Gambar 3.14 Alur proses normalisasi

- **Proses Pembentukan Matriks *U***

Proses pembentukan matriks *U* adalah proses untuk mendapatkan sebuah matriks yang berisi normalisasi derajat keanggotaan data dikalikan dengan data latih pada tiap *cluster*. Matrik *U* nantinya berperan pada proses perhitungan LSE untuk mendapatkan nilai koefisien *output*. Masukan dari matriks *U* adalah matriks

d^k_{ij} dan menghasilkan keluaran berupa matriks *U* yang berdimensi jumlah data (*n*)×(jumlah *cluster* * (jumlah atribut + 1)). Alur proses pembentukan matriks *U* ditunjukkan pada Gambar (3.15).

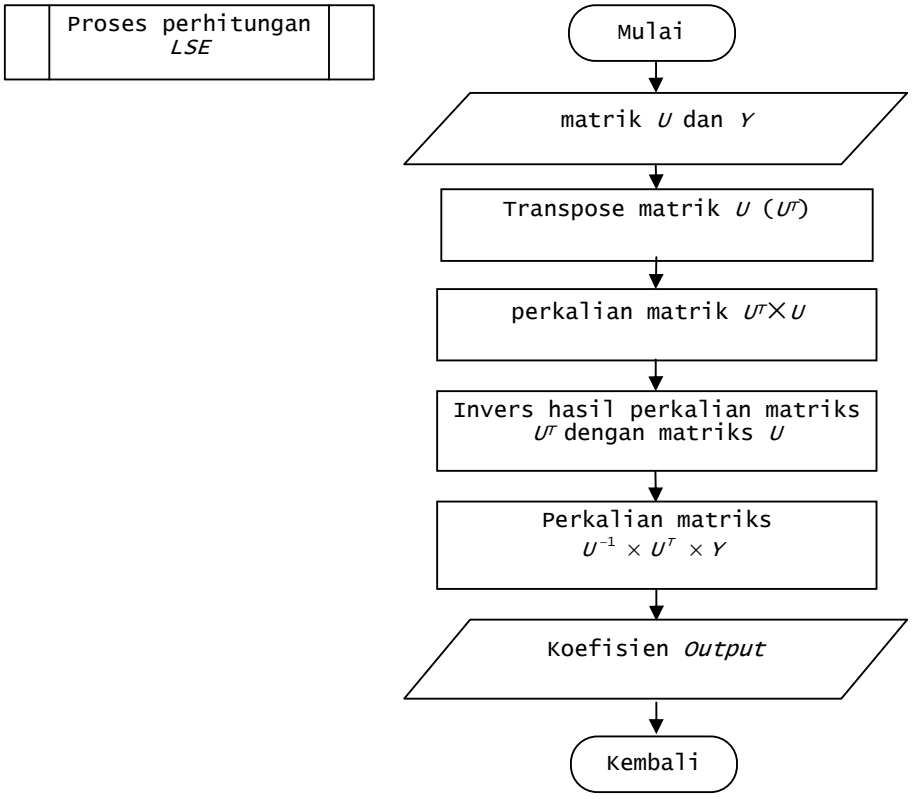


Gambar 3.15 Alur proses pembentukan matriks U

- **Proses Perhitungan *LSE (Least Square Estimator)***

Proses perhitungan *LSE* adalah proses untuk mendapatkan koefisien *output* dengan metode kuadrat terkecil karena matriks *U* sebagai variabel pembentuk koefisien *output* berbentuk bukan matriks bujur sangkar. Alur proses perhitungan *LSE* ditunjukkan pada Gambar (3.16) dan dijelaskan sebagai berikut:

1. Masukan dari proses ini adalah matriks *U* dan nilai kelayakan (*Y*).
2. Melakukan proses transpose matriks *U* (U^T).
3. Melakukan perkalian matriks $U^T \times U$.
4. Melakukan proses invers matriks hasil pekalian (U^{-1}).
5. Melakukan proses perkalian matriks $U^{-1} \times U^T \times Y$, dimana *Y* adalah matriks nilai kelayakan dari data latih.
6. Hasil akhir dari proses ini adalah koefisien *output*.



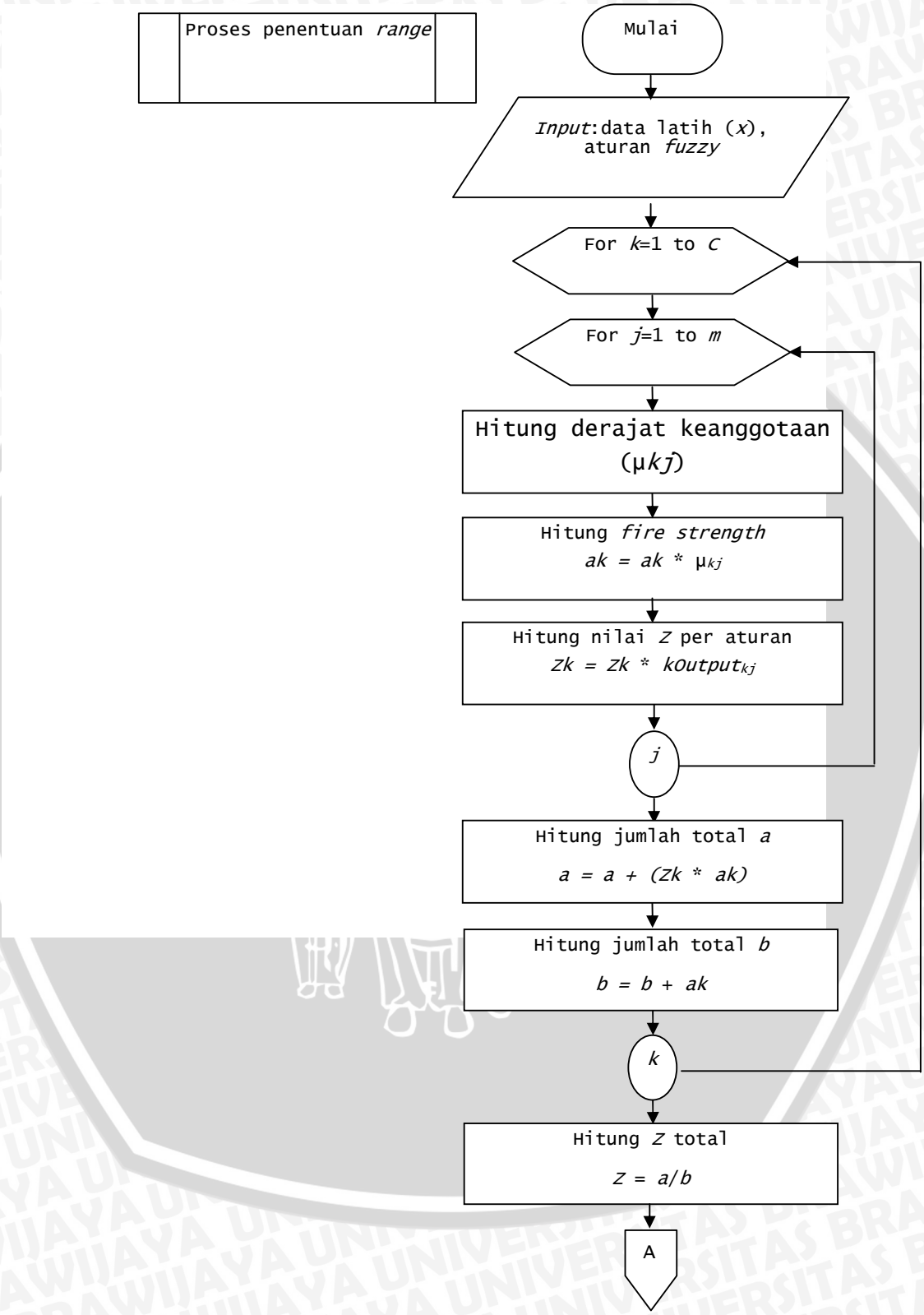
.

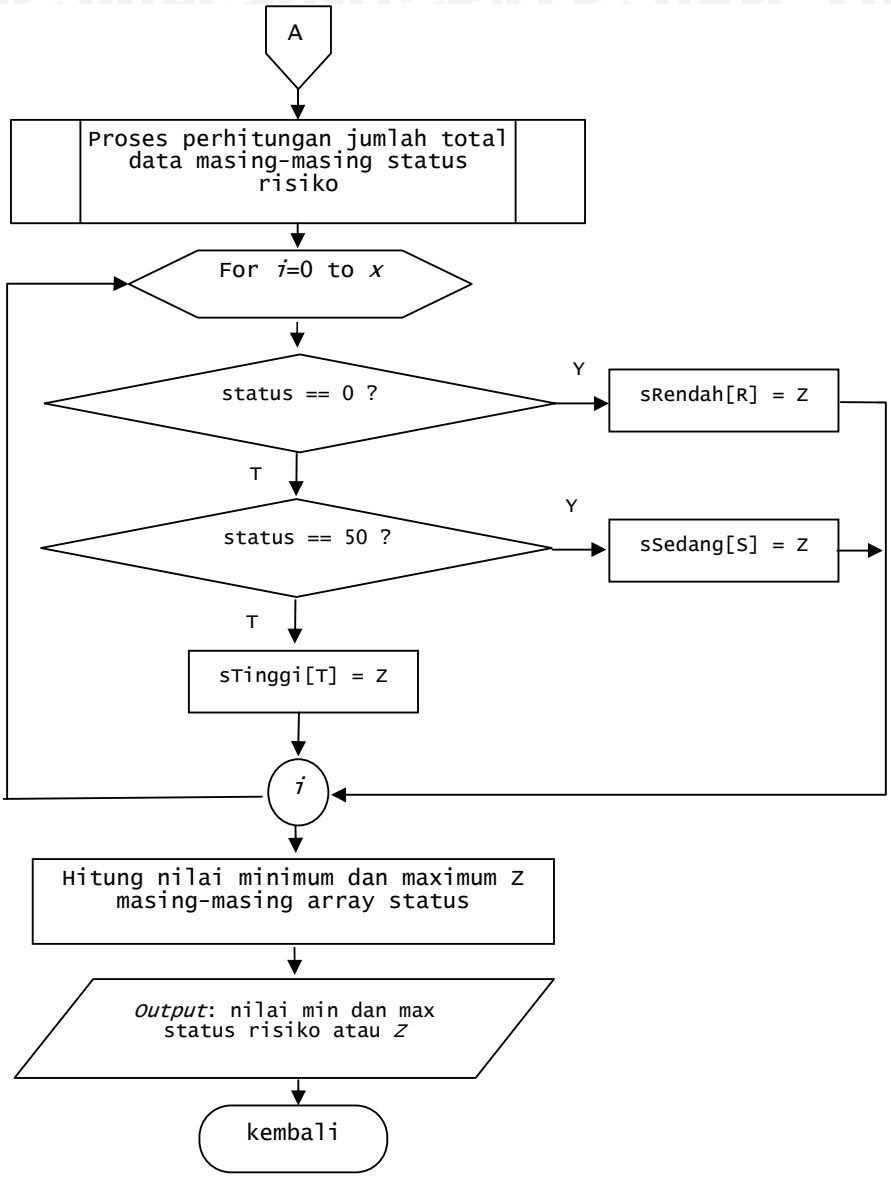
Gambar 3.16 Alur proses perhitungan LSE

3.3.2.5 Proses Sistem Inferensi Fuzzy

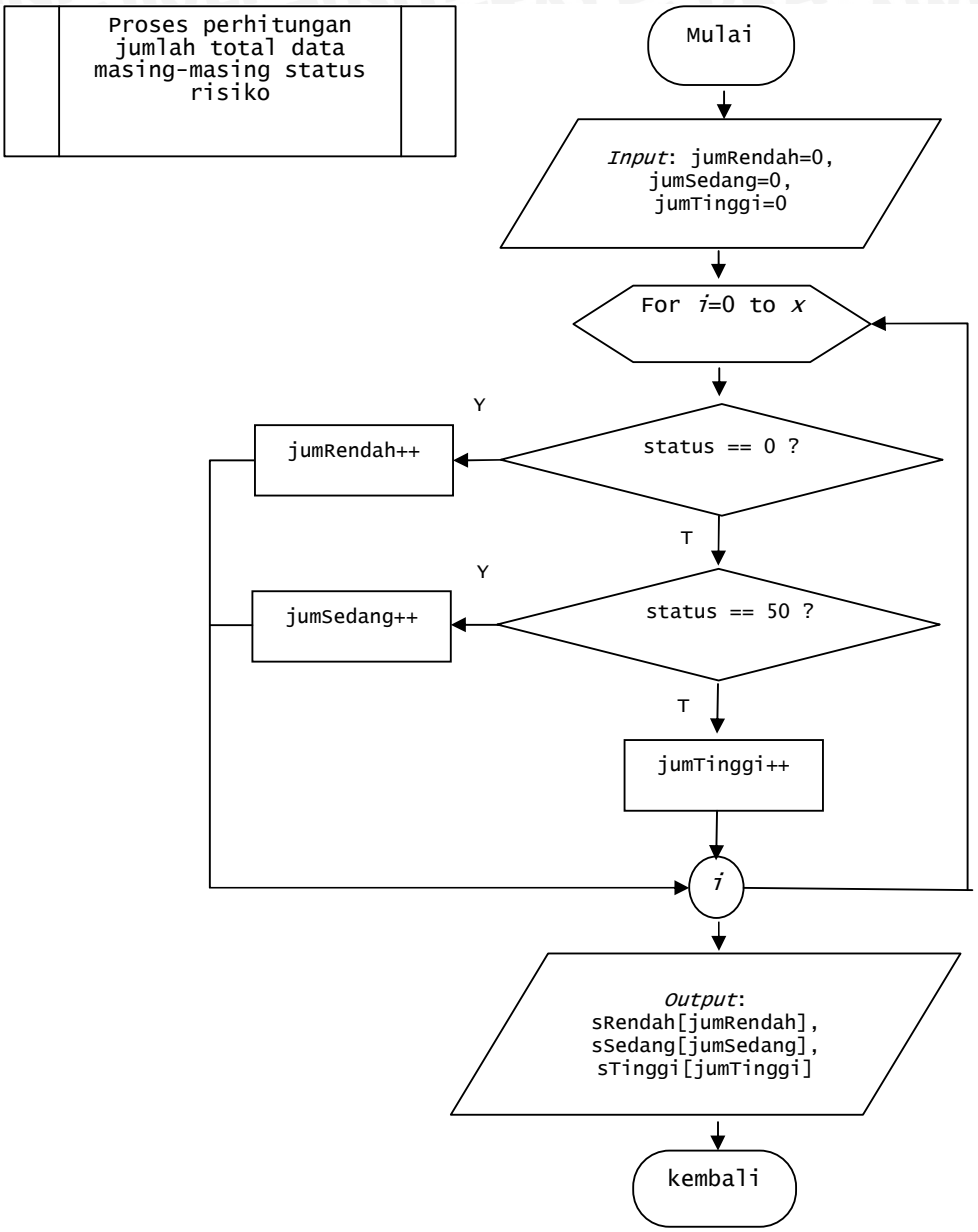
Proses sistem inferensi *fuzzy* merupakan proses untuk mendapatkan nilai risiko. Pada proses ini akan dilakukan pengujian terhadap data uji untuk diketahui nilai risikonya. Sebelum dilakukan pengujian tersebut, dilakukan penentuan *range* untuk mendapat rentang nilai masing-masing status risiko dari data latih. Langkah-langkah proses penentuan *range* ini digambarkan oleh bagan alir pada Gambar (3.17) dan dijelaskan sebagai berikut :

1. Masukan dari proses ini adalah data latih (x) dan aturan *fuzzy*.
2. Iterasi k=1 sampai C, dilakukan langkah berikut:
   a. Iterasi j=1 sampai m, dilakukan langkah berikut:
      - Menghitung derajat keanggotaan menggunakan fungsi *Gauss*.
      - Menghitung *fire strength* masing-masing aturan (a_k).
      - Menghitung nilai Z masing-masing aturan (Z_k).
   b. Hitung nilai a, dimana a adalah penjumlahan dari tiap hasil perkalian Z_k dengan a_k.
   c. Hitung nilai b, dimana b adalah penjumlahan dari tiap a_k pada *cluster*.
3. Hitung Z dengan cara membagi antara a dengan b dimana hasil akhir dari pembagian tersebut dinamakan dengan nilai risiko.
4. Hitung jumlah total data dari masing-masing status risiko yaitu jumRendah, jumSedang dan jumTinggi. Proses perhitungan ini digambarkan oleh bagan alir pada Gambar (3.18).
5. Iterasi i=0 sampai x , dilakukan sebagai berikut :
   a. Setelah perhitungan jumlah total data dari masing-masing status risiko selesai, dilakukan pengecekan : jika status risiko pada data latih sama dengan 0, maka sRendah[R] = zi. Namun jika tidak sama dengan 0, maka lakukan pengecekan kembali.
   b. Jika status risiko pada data latih sama dengan 50, maka sSedang[S] = zi. Jika nilai status risiko. Namun jika nilai status risiko tidak sama dengan 50, maka sTinggi[T] = zi.
6. Hitung nilai minimum dan maksimum dari masing-masing array sRendah, sSedang dan sTinggi.
7. Hasil akhir dari proses ini adalah nilai minimum dan maksimum atau range dari masing-masing kelas status risiko.


Proses penentuan *range*
Mulai
Input:data latih (*x*), aturan *fuzzy*
For *k*=1 to *C*
For *j*=1 to *m*
Hitung derajat keanggotaan (μ*kj*)
Hitung *fire strength*
ak = ak * μ*kj*
Hitung nilai *Z* per aturan
*Zk = Zk * kOutput*$_{kj}$
j
Hitung jumlah total *a*
*a = a + (Zk * ak)*
Hitung jumlah total *b*
b = b + ak
k
Hitung *Z* total
Z = a/b
A



Gambar 3.17 Alur Proses Penentuan *Range*



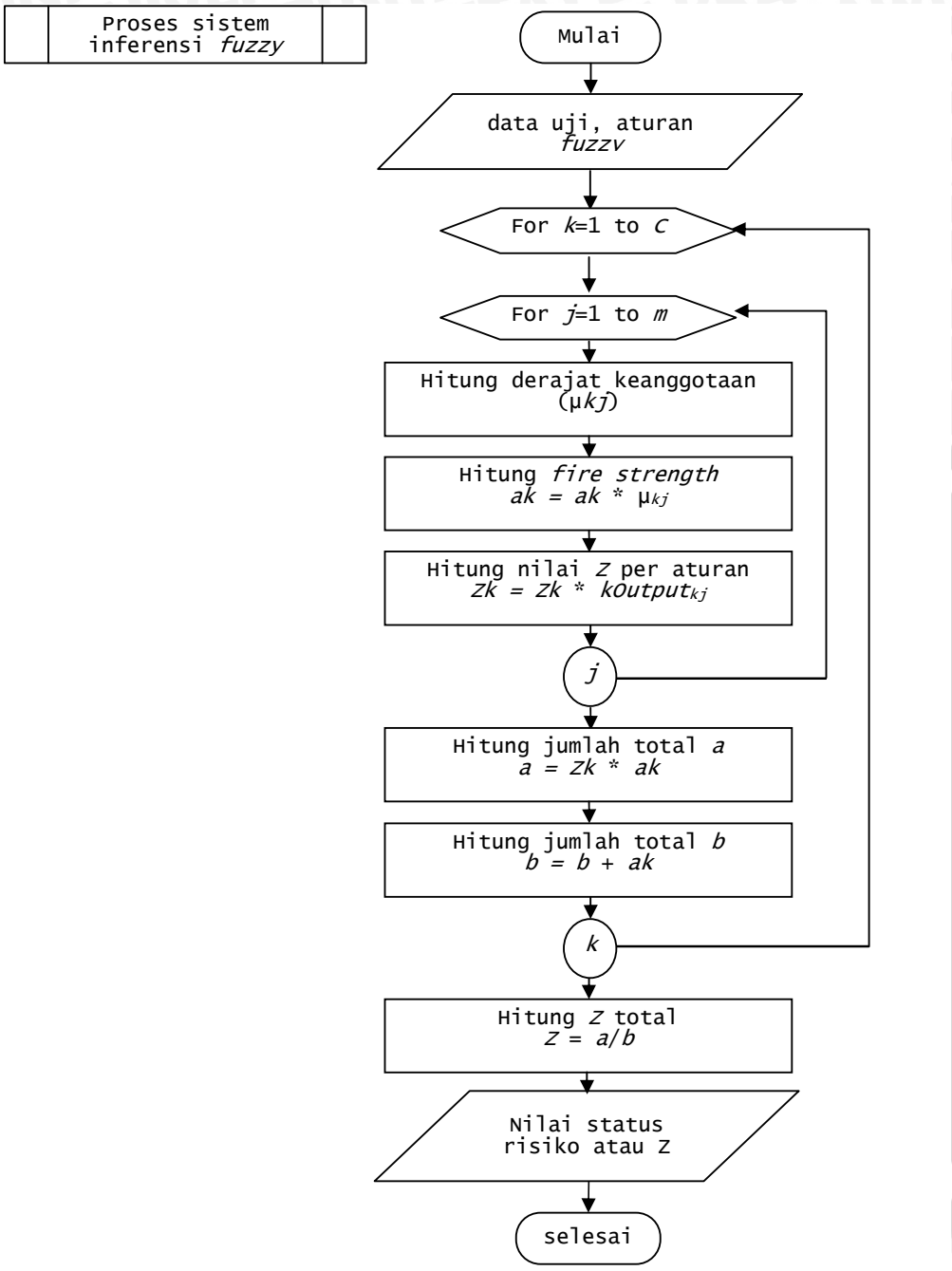
Gambar 3.18 Alur Proses Perhitungan Jumlah Total Data Dari Masing-masing Status Risiko

Setelah didapatkan nilai range dari masing-masing status risiko, maka dilakukan proses pengujian data uji dengan menggunakan sistem inferensi *fuzzy* sugeno orde satu. Berikut langkah-langkahnya digambarkan pada Gambar (3.19). Penjelasan proses system inferensi *fuzzy* adalah sebagai berikut :

1. Masukan dari proses ini adalah data uji (x) dan aturan *fuzzy*.
2. Iterasi k=1 sampai C, dilakukan langkah berikut:

a. Iterasi *j*=1 sampai *m*, dilakukan langkah berikut:
   i. Menghitung derajat keanggotaan menggunakan fungsi *Gauss*.
   ii. Menghitung *fire strength* masing-masing aturan (*ak*).
   iii. Menghitung nilai *Z* masing-masing aturan (*Zk*).
b. Hitung nilai *a*, dimana *a* adalah perkalian *Zk* dengan *ak*.
c. Hitung nilai *b*, dimana *b* adalah penjumlahan dari tiap *ak*pada *cluster*.

3. Hitung *Z* dengan cara membagi antara *a* dengan *b* dimana hasil akhir dari pembagian tersebut dinamakan dengan status risiko.
4. Hasil akhir dari proses ini adalah nilai *Z* atau nilai status risiko.

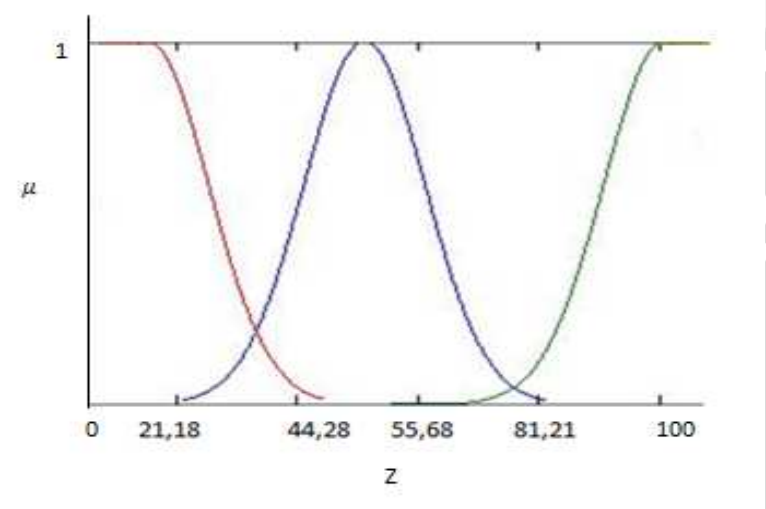


Gambar 3.19 Alur proses sistem inferensi *fuzzy*

3.3.3. Perancangan Penentuan Kelas Tingkat Risiko Penyakit Stroke

Dalam penelitian ini, *output* dari sistem berupa tingkat status risiko penyakit stroke yang terbagi menjadi tiga kelas, yaitu rendah dengan nilai status 0, sedang dengan nilai status 50, dan tinggi dengan nilai status 100. Proses

penentuan *range* dilakukan terhadap 109 data latih. Derajat keanggotaan kelas rendah dan kelas tinggi digambarkan dengan grafik representasi linear, sedangkan kelas sedang digambarkan dengan grafik representasi kurva gauss. Nilai rentang (*range*) untuk masing-masing kelas ditentukan dari perhitungan penentuan *range* pada sub subbab 3.3.2.5. Contoh grafik derajat keanggotaan tingkat status risiko ditunjukkan pada Gambar (3.20).



Gambar 3.20 Grafik Derajat Keanggotaan Tingkat Status Risiko

Fungsi keanggotaan kelas rendah :

$$\mu(x) = e^{\frac{(x-c)^2}{2(\sigma)^2}} = e^{\frac{(x-28,89)^2}{2(15,39)^2}}$$

Fungsi keanggotaan kelas sedang :

$$\mu(x) = e^{\frac{(x-c)^2}{2(\sigma)^2}} = e^{\frac{(x-30,015)^2}{2(51,19)^2}}$$

Fungsi keanggotaan kelas tinggi :

$$\mu(x) = e^{\frac{(x-c)^2}{2(\sigma)^2}} = e^{\frac{(x-45,075)^2}{2(100,75)^2}}$$

3.3.4. Perancangan *Database*

Sistem ini menggunakan *Database Management System* (DBMS) *yaitu* MySQL. Pada penelitian ini, membutuhkan dua tabel yang digunakan untuk menyimpan informasi yang berkaitan dengan proses pembangkitan aturan *fuzzy* dan proses pengujian. Kedua tabel tersebut adalah tabel data latih dan tabel data uji. Tabel pada *database* ini ditampilkan dengan *Physical data model. Physical data model* merupakan desain nyata (struktur fisik) dari *database*. *Physical data model* yang digunakan pada penelitian ini ditunjukkan pada Gambar (3.21).

Gambar 3.21 Physical data model

Penjelasan mengenai tabel pada Gambar (3.21) yang digunakan dalam penelitian ini adalah sebagai berikut :

1. Tabel datalatih digunakan untuk menyimpan data latih yang digunakan untuk proses *clustering* dan pembangkitan aturan *fuzzy*. Atribut dari tabel ini adalah tekanan darah, kadar gula darah, kolesterol total, *Low Density Lipoprotein* (LDL), usia, asam urat, jenis kelamin, *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko.
2. Tabel datauji digunakan untuk menyimpan data uji yang digunakan saat proses pengujian. Atribut dari tabel ini adalah tekanan darah, kadar gula darah, kolesterol total, *Low Density Lipoprotein* (LDL), usia, asam urat, jenis kelamin, *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko.

3.3.5. Perancangan Antarmuka

Pada subbab perancangan sistem telah dijelaskan bagaimana proses berjalannya sistem. Mulai dari pelatihan terhadap data latih, pemilihan hasil

cluster dengan varian terkecil, proses ekstraksi aturan *fuzzy* dari *cluster* terpilih, dan pengujian terhadap data uji menggunakan sistem inferensi *fuzzy*. Berdasarakan alur proses dari sistem yang akan dibangun, maka perancangan antarmuka juga disesuaikan dengan kebutuhan. Sehingga nantinya terdapat 4 tampilan utama antarmuka, diantaranya adalah antarmuka untuk menampilkan data latih, menampilkan data uji, melakukan proses pelatihan terhadap data latih dengan pemilihan jumlah *cluster* ideal, dan antarmuka untuk proses pengujian data uji menggunakan aturan yang terpilih.

3.3.5.1 Antarmuka Lihat Data Latih

Antarmuka lihat data latih digunakan untuk menampilkan data latih yang akan digunakan saat proses pelatihan pada sistem deteksi dinirisiko stroke ini. Perancangan antarmuka data latih ditunjukkan pada Gambar (3.22).

Gambar 3.22 Rancangan Antarmuka Lihat Data Latih

Keterangan Gambar (3.22):

1. Tombol Load digunakan untuk mengambil data latih dari database dan menampilkannya pada table data latih
2. Tabel data latih terdiri dari tekanan darah, kadar gula, kolesterol total, LDL, usia, jenis kelamin, asam urat, BUN, kreatinin, dan status risiko

3.3.5.2 Antarmuka Lihat Data Uji

Antarmuka lihat data uji digunakan untuk menampilkan data uji yang akan digunakan saat proses pengujian pada sistem deteksi dinirisiko stroke ini. Perancangan antarmuka data latih ditunjukkan pada Gambar (3.23).

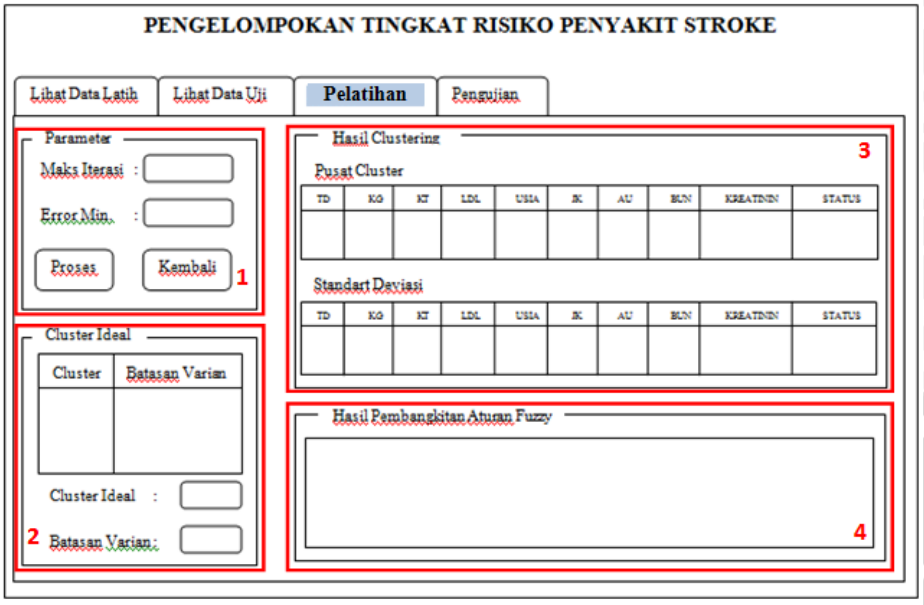
Gambar 3.23 Rancangan Antarmuka Lihat Data Latih

Keterangan Gambar (3.23):

1. Tombol Load digunakan untuk mengambil data uji dari database dan menampilkannya pada table data uji
2. Tabel data uji terdiri dari tekanan darah, kadar gula, kolesterol total, LDL, usia, jenis kelamin, asam urat, BUN, kreatinin, dan status risiko

3.3.5.3 Antarmuka Pelatihan

Antarmuka pelatihan merupakan antarmuka untuk menampilkan nilai pusat *cluster*, standar deviasi, dan aturan *fuzzy* dari proses pelatihan yang dihasilkan dari penentuan jumlah *cluster* ideal yang terpilih. Rancangan antarmuka pelatihan ditunjukkan oleh Gambar (3.24).



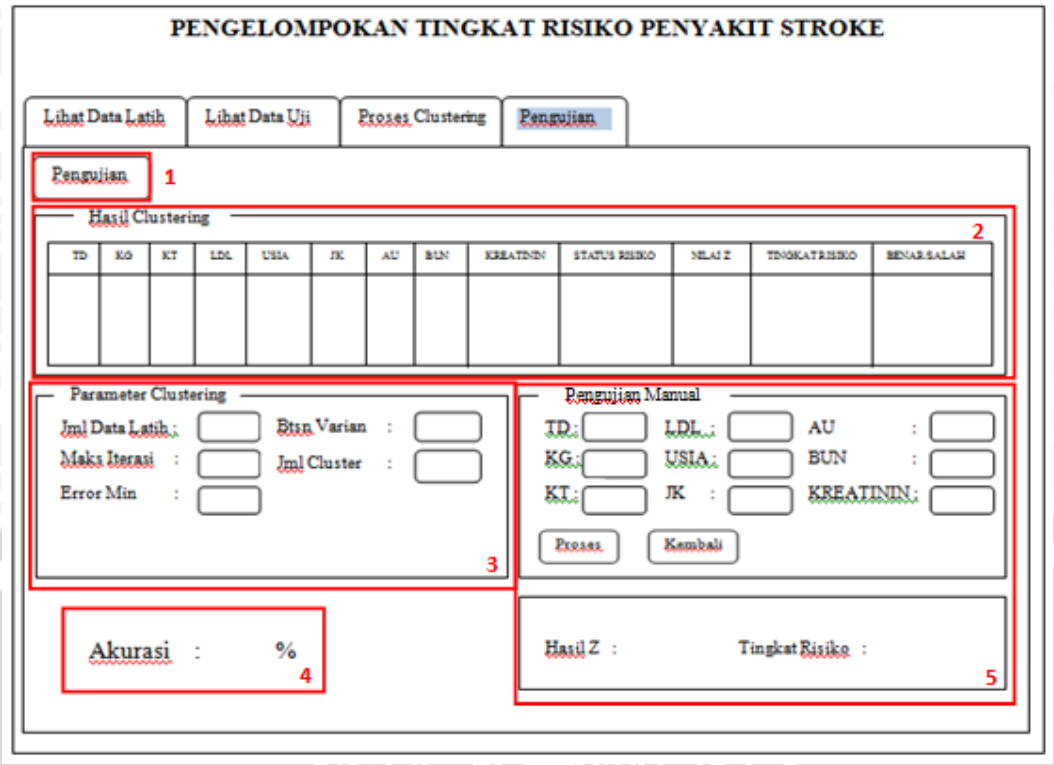
Gambar 3.24 Rancangan Antarmuka Pelatihan

Keterangan Gambar (3.24):

1. Parameter terdiri dari inputan text untuk memasukkan nilai maksimum iterasi dan nilai error minimum. Tombol proses digunakan untuk melakukan proses *clustering*, perhitungan batasan varian untuk penentuan *cluster* yang terpilih, dan pembentukan aturan *fuzzy*. Tombol Kembali digunakan untuk mengulangi proses pelatihan dengan mengosongkan semua *fields* dan nilai variabel pada program.
2. *Cluster* Ideal berisi hasil perhitungan nilai batasan varian dari tiap *cluster*. Hasil *cluster* ideal yang terpilih serta nilai batasan variannya akan ditampilkan pada *fields cluster* ideal dan *fields* batasan varian.
3. Hasil *Clustering* berisi tampilan nilai pusat *cluster* dan standar deviasi.
4. Hasil Pembangkitan Aturan *Fuzzy* ditampilkan aturan *fuzzy* yang terbentuk.

3.3.5.4 Antarmuka Pengujian

Antarmuka pengujian merupakan antarmuka untuk menampilkan hasil pengujian aturan *fuzzy* yang terbentuk terhadap data uji. Hasil pengujian berupa nilai akurasi. Rancangan antarmuka pengujian ditunjukkan pada Gambar (3.25).



Gambar 3.25 Rancangan Antarmuka Pengujian

Keterangan Gambar (3.25):

1. Tombol Pengujian digunakan untuk untuk melakukan proses pengujian aturan *fuzzy* terhadap data uji yang ada pada form pengujian
2. Hasil *Clustering* berisi tabel yang meliputi data uji, nilai Z, tingkat risiko yang dihasilkan berdasarkan proses pengujian yang dilakukan oleh sistem, dan benar/salah untuk mengetahui kesesuaian nilai status risiko dari pakar dengan nilai tingkat risiko yang dihasilkan oleh sistem.
3. Parameter *clustering* menampilkan parameter FCM *clustering* yang digunakan pada proses sebelumnya beserta jumlah data latih, jumlah *cluster*, dan nilai batasan varian yang dihasilkan.
4. Akurasi untuk menampilkan presentase nilai kebenaran dari seluruh data yang diujikan pada proses pengujian.
5. Pengujian manual digunakan untuk melakukan pengujian data uji yang diinputkan oleh *user* berdasarkan aturan *fuzzy* yang terbentuk sebelumnya. Tombol proses digunakan untuk melakukan pengujian manual. Sedangkan tombol kembali digunakan untuk mengosongkan *field* dan melakukan proses perhitungan manual kembali. Hasil Z akan menampilkan nilai

risiko stroke dan tingkat risiko akan menampilkan tingkat risiko penyakit stroke berdasarkan data uji yang diinputkan pada proses pengujian manual.

3.4 Perhitungan Manual

Perhitungan manual dilakukan untuk mengimplementasikan sistem secara matematis lewat perhitungan langkah demi langkah pada data latih dan data uji. Proses perhitungan manual dibedakan menjadi dua bagian, yaitu proses perhitungan manual proses pelatihan dan perhitungan manual proses pengujian.

3.4.1. Proses Pelatihan

Proses pelatihan dari data rekam medik sebagai upaya untuk membangkitkan aturan *fuzzy* secara otomatis dengan menggunakan metode *fuzzy c-means clustering*.

1. Proses *Fuzzy C-Means Clustering*

Langkah pertama pada proses *fuzzy c-means clustering* adalah menyiapkan data latih yang akan diklaster dan parameter *fuzzy c-means*. Jumlah atribut yang digunakan sebanyak sepuluh, antara lain tekanan darah (TD), kadar gula darah (GD), kolesterol total (KT), *Low Density Lipoprotein* (LDL), usia (U), asam urat (AU), jenis kelamin (JK), *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko. Data rekam medik (X) yang digunakan sebagai data latih pada proses *clustering* disajikan pada Tabel 3.1.

Tabel 3.1 Data Latih

No	TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin	Status Risiko
1	192	231	183	134	76	1	9.8	17.9	1.2	100
2	100	100	195	127	28	1	6.2	16.2	0.6	0
3	120	106	195	127	55	1	6.2	9.6	1	0
4	140	155	195	127	46	0	6.2	20	1.5	50
5	200	138	267	188	63	0	6.4	13	0.9	100
6	160	192	285	183	63	0	7.5	10.7	0.8	100
7	190	120	197	117	50	0	7.8	13.3	0.9	50
8	140	188	195	127	60	1	6.2	10.5	1	50
9	110	86	153	105	72	1	6.2	27.4	1.7	0
10	150	155	195	127	66	0	6.2	30.6	1	50
11	210	204	229	162	49	0	7.2	23.1	0.7	100
12	200	253	249	187	49	0	5.2	14.1	1	100
13	190	119	195	127	63	1	6.2	13.2	0.7	50

14	100	155	173	133	56	1	7.2	31.6	1.9	0
15	110	85	97	53	35	0	6.2	48.9	2.2	0

Berdasarkan tabel data latih, terdapat data sejumlah 15 (n=15) dengan atribut sejumlah 10 (m=10). Pada contoh perhitungan manual ini data akan dikelompokkan ke dalam 3 kelompok dengan menggunakan FCM *clustering*.

Nilai awal parameter pada algoritma *fuzzy c-means* (FCM) dapat diinisialisasi sebagai berikut:

Jumlah *cluster* = 3;

Pangkat = w = 2;

Maksimum iterasi = MaxIter = 50;

Eror terkecil yang diharapkan = ξ = 0.00001;

Fungsi obyektif awal = P_0 = 0;

Iterasi awal = t = 1;

Setelah inisialisasi parameter, langkah selanjutnya adalah pembentukan matriks partisi awal U. Proses pembentukan matriks partisi awal U didasarkan pada diagram alir proses pembentukan matriks partisi awal U pada Gambar (3.4).

Langkah awal pembentukan matriks partisi awal U yaitu membangkitkan bilangan random μik dalam matrik berukuran jumlah data x jumlah *cluster* (15x3). Pembangkitan bilangan random dilakukan dengan fungsi RAND() pada Microsoft Excel. Hasil pembangkitan bilangan random μi1, μi2 dan μi3 ditunjukkan pada Tabel 3.2.

Tabel 3.2 Bilangan Random

μi1	μi2	μi3
0.471167311	0.988384033	0.591278427
0.733800463	0.283854057	0.50538354
0.870649508	0.246835563	0.500872322
0.645815802	0.173568793	0.75608488
0.251260508	0.52823026	0.297235078
0.368794529	0.821755528	0.40433074
0.557108837	0.414022855	0.818950978
0.653946224	0.698731574	0.624405269
0.737475475	0.29153976	0.581871236
0.325439517	0.129750119	0.772489409
0.38749964	1.232305047	0.495519427

0.24861829	0.536885729	0.278409358
0.65833327	0.503490921	0.963163473
0.6691469	0.224863527	0.407349587
0.557899165	0.319511761	0.525078938

Setelah proses pembangkitan bilangan random langkah selanjutnya dilakukan penjumlahan kolom setiap baris bilangan random yang terbentuk untuk masing-masing elemen I (Qi) berdasarkan persamaan (2-1). Berikut ini contoh perhitungan nilai Qi untuk elemen ke satu :

$$Q_1 = 0{,}471167311 + 0{,}988384033 + 0{,}591278427 = 2{,}050829772$$

Hasil penjumlahan setiap baris bilangan random (Q_i) ditunjukkan pada Tabel 3.3.

Tabel 3.3 Nilai Qi

Q_i
2.050829772
1.52303806
1.618357394
1.575469475
1.076725846
1.594880797
1.79008267
1.977083067
1.61088647
1.227679045
2.115324114
1.063913377
2.124987664
1.301360014
1.402489864

Setiap bilangan random yang telah terbentuk kemudian dibagi dengan Q_i berdasarkan persamaan (2-2). Berikut ini contoh perhitungan nilai matriks partisi awal U untuk μ_{11}:

$$\mu_{11} = \frac{\mu_{11}}{Q_1} = \frac{0{,}471167311}{2{,}050829772} = 0{,}22974472$$

Matriks partisi awal U

0.22974472	0.48194347	0.28831182
0.48180047	0.18637358	0.33182594
0.53798345	0.15252228	0.30949425
0.40991959	0.11016956	0.47991084
0.23335606	0.49058937	0.27605455
0.23123642	0.51524573	0.25351784
0.31121964	0.23128703	0.45749334
0.33076315	0.35341538	0.31582146
0.45780721	0.18098094	0.36121181
0.26508517	0.10568732	0.62922749
0.18318688	0.58256086	0.23425224
0.23368281	0.50463294	0.26168423
0.30980569	0.23693829	0.45325603
0.51419045	0.17279116	0.31301836
0.39779194	0.22781759	0.37439054

Setelah matriks partisi awal U terbentuk, langkah selanjutnya adalah melakukan perhitungan pusat *cluster*. Perhitungan nilai pusat *cluster* sesuai dengan diagram alir proses perhitungan pusat *cluster* pada Gambar (3.5). Input dari proses perhitungan pusat *cluster* adalah data latih dan matriks partisi awal U. Output proses ini menghasilkan matriks pusat *cluster* berukuran jumlah *cluster* x jumlah atribut.

Perhitungan nilai Pusat *Cluster* didasarkan pada persamaan (2-3) sehingga dapat dihitung 3 pusat *cluster* Vkj pada iterasi 1 dengan k=1,2,3; dan j=1,2,3,4,5,6,7,8,9,10 sebagai berikut :

$$V_{kj} = \frac{\sum_{i=1}^{n}\left((\mu_{ik})^{w} \times X_{ij}\right)}{\sum_{i=1}^{n}(\mu_{ik})^{w}}$$

$$V_{11} = \frac{\left((0{,}22974472)^2 \times 192\right) + \left((0{,}481800477)^2 \times 100\right) + \ldots + \left((0{,}397791941)^2 \times 110\right)}{(0{,}22974472)^2 + (0{,}481800477)^2 + \ldots + (0{,}397791941)^2}$$

$$V_{11} = 132{,}98434$$

Pusat *cluster* dapat ditunjukkan pada Tabel 3.4.

Tabel 3.4 Pusat *Cluster* Iterasi 1

Cluster **(k)**	**ATRIBUT(j)**									
	1	2	3	4	5	6	7	8	9	10
1	132.98434	132.53067	185.86891	124.65013	52.90507	0.64518	6.54414	21.28413	1.26327	26.71611

2	180.20212	186.91173	227.17191	157.30073	57.71914	0.30288	7.01098	17.05841	0.97138	82.90052
3	151.86497	143.26777	192.51914	127.05380	55.99271	0.39892	6.59632	21.76931	1.15321	44.48249

Pada *cluster* ke-1 sampai dengan *cluster* ke-3 terdapat 10 pusat *cluster*. Secara berturut-turut dari kiri ke kanan adalah pusat *cluster* tekanan darah (TD), kadar gula darah (GD), kolesterol total (KT), *Low Density Lipoprotein* (LDL), usia (U), asam urat (AU), jenis kelamin (JK), *Blood Urea Nitrogen* (BUN), kreatinin, dan status risiko.

Langkah selanjutnya setelah perhitungan pusat *cluster* adalah perhitungan nilai fungsi objektif. Perhitungan nilai fungsi objektif sesuai dengan diagram alir perhitungan fungsi objektif pada Gambar (3.6).

Proses perhitungan fungsi objektif menggunakan input data latih (X_{ij}), pusat *cluster* (V_{kj}) dan matriks awal (μ_{ik}). Proses perhitungan fungsi objektif akan menghasilkan *output* berupa nilai fungsi objektif iterasi ke-t (P_t). Perhitungan fungsi objektif pada iterasi pertama (P_1) dapat dihitung dengan menggunakan persamaan (2-4) sebagai berikut :

$$L_1 = \left(\left[\sum_{j=1}^{10} (X_{ij} - V_{ij})^2 \right] (\mu_{i1})^2 \right)$$

$$L_1 = \left(\left((192 - 132{,}98434)^2 + (231 - 132{,}53067)^2 + \ldots + (100 - 26{,}71611)^2 \right) \times (0{,}22974472)^2 \right)$$

$$L_1 = 1013{,}46864$$

$$L_2 = \left(\left[\sum_{j=2}^{10} (X_{ij} - V_{ij})^2 \right] (\mu_{i2})^2 \right)$$

$$L_2 = \left(\left((192 - 180{,}20212)^2 + (231 - 186{,}91173)^2 + \ldots + (100 - 82{,}90052)^2 \right) \times (0{,}481943478)^2 \right)$$

$$L_2 = 1210{,}74062$$

$$L_3 = \left(\left[\sum_{j=3}^{10} (X_{ij} - V_{ij})^2 \right] (\mu_{i3})^2 \right)$$

$$L_3 = \left(\left((192 - 151{,}86497)^2 + (231 - 143{,}26777)^2 + \ldots + (100 - 44{,}48249)^2 \right) \times (0{,}288311802)^2 \right)$$

$$L_3 = 1076{,}84367$$

$$P_t = \sum_{i=1}^{15} \sum_{k=1}^{3} \left(\left[\sum_{j=1}^{10} (X_{ij} - V_{ij})^2 \right] (\mu_{ik})^2 \right) = 3301{,}05293$$

Detail perhitungan fungsi objektif dapat ditunjukkan pada Tabel 3.5.

Tabel 3.5 Fungsi Objektif

L_1	L_2	L_3	$L_1+L_2+L_3$
1013.46864	1210.74062	1076.84367	3301.05293
834.66548	823.11575	810.65776	2468.43899
525.69628	443.41302	434.75007	1403.85937
207.53020	70.60529	96.37375	374.50924
1124.93815	1359.89202	1128.99580	3613.82598
1234.31507	1274.19605	1116.94664	3625.45776
407.44908	442.10093	472.30342	1321.85344
429.30821	587.33677	231.59852	1248.24351
1105.80057	1000.09465	1219.65146	3325.54668
118.35663	58.29339	141.02554	317.67556
660.57210	546.55505	699.85175	1906.97890
1764.73855	1654.91424	1666.34699	5085.99979
406.25512	437.19412	452.59994	1296.04919
702.98470	539.33763	521.31421	1763.63654
2787.72744	2675.09850	3211.61275	8674.43868
		P_t	39727.56656

Setelah perhitungan fungsi objketif langkah selanjutnya adalah melakukan perubahan matriks partisi U. Perhitungan nilai perubahan matriks partisi U sesuai dengan diagram alir proses perubahan matriks partisi U pada Gambar (3.6). Perubahan matriks partisi U dapat dihitung dengan menggunakan persamaan (2-5) sehingga dapat memperbaiki nilai derajat keanggotaan setiap elemen matriks partisi U (μ_{ik}). Contoh perhitungan perubahan matriks partisi U adalah sebagai berikut:

$$L_1 = \left[\sum_{j=1}^{m} \left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{w-1}}$$

$$L_1 = \left[\sum_{j=1}^{10} \left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{2-1}}$$

$$L_1 = \left[\left((192 - 132{,}98434)^2 + (231 - 132{,}53067)^2 + \ldots + (100 - 26{,}71611)^2\right)\right]^{-1}$$

$$L_1 = 0{,}000052081$$

$$L_2 = \left[\sum_{j=1}^{m}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{w-1}}$$

$$L_2 = \left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{2-1}}$$

$$L_2 = \left[\left((192-180{,}20212)^2 + (231-186{,}91173)^2 + \ldots + (100-44{,}48249)^2\right)\right]^{-1}$$

$$L_2 = 0{,}000191841$$

$$L_3 = \left[\sum_{j=1}^{m}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{w-1}}$$

$$L_3 = \left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{2-1}}$$

$$L_3 = \left[\left((192-151{,}86497)^2 + (231-143{,}26777)^2 + \ldots + (100-44{,}48249)^2\right)\right]^{-1}$$

$$L_3 = 0{,}000077192$$

$$L_T = \sum_{k=1}^{3}\left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{2-1}} = L1 + L2 + L3 = 0{,}000052081 + 0{,}000191841 + 0{,}000077192 = 0{,}000321114$$

$$\mu_{i1} = \frac{\left[\sum_{j=1}^{10}\left(X_{1j} - V_{1j}\right)^2\right]^{\frac{-1}{2-1}}}{\sum_{k=1}^{3}\left[\sum_{j=1}^{10}\left(X_{1j} - V_{1j}\right)^2\right]^{\frac{-1}{2-1}}} = \frac{L_1}{L_T} = \frac{0{,}000052081}{0{,}000321114} = 0{,}162189042$$

Hasil Proses perhitungan matriks partisi U ditunjukkan pada Tabel 3.6.

Tabel 3.6 Perhitungan matriks partisi U

$\left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{1}{2}}$	$\left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{1}{2}}$	$\left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{1}{2}}$	$\sum_{k=1}^{3}\left[\sum_{j=1}^{10}\left(X_{ij} - V_{ij}\right)^2\right]^{\frac{-1}{2-1}}$	**μ i1**	**μ i2**	**μ i3**
L_1	**L_2**	**L_3**	**$L_1+L_2+L_3$**	**L_1/L_T**	**L_2/L_T**	**L_3/L_T**
0.000052081	0.000191841	0.000077192	0.000321114	0.162189042	0.597422879	0.240388079
0.000278113	0.000042200	0.000135826	0.000456139	0.609711977	0.092514651	0.297773372
0.000550558	0.000052464	0.000220326	0.000823347	0.668682357	0.063719898	0.267597745
0.000809685	0.000171904	0.002389804	0.003371393	0.240163286	0.050989017	0.708847697
0.000048407	0.000176983	0.000067499	0.000292889	0.165274568	0.604266254	0.230459177
0.000043320	0.000208350	0.000057542	0.000309211	0.140097727	0.673809599	0.186092674
0.000237717	0.000120999	0.000443148	0.000801864	0.296455840	0.150896951	0.552647209
0.000254839	0.000212659	0.000430673	0.000898170	0.283730706	0.236769090	0.479500204
0.000189535	0.000032751	0.000106976	0.000329262	0.575634504	0.099467898	0.324897598
0.000593715	0.000191614	0.002807486	0.003592815	0.165250744	0.053332461	0.781416795
0.000050801	0.000620939	0.000078408	0.000750147	0.067720775	0.827755506	0.104523719
0.000030944	0.000153878	0.000041095	0.000225917	0.136969924	0.681126324	0.181903751
0.000236254	0.000128409	0.000453913	0.000818577	0.288616103	0.156868873	0.554515025
0.000376099	0.000055358	0.000187949	0.000619406	0.607192727	0.089373097	0.303434176

0.000056763	0.000019401	0.000043644	0.000119808	0.473778307	0.161937705	0.364283988

Matriks Perbaikan

$$\begin{bmatrix} 0.16219 & 0.59742 & 0.24039 \\ 0.60971 & 0.09251 & 0.29777 \\ 0.66868 & 0.06372 & 0.26760 \\ 0.24016 & 0.05099 & 0.70885 \\ 0.16527 & 0.60427 & 0.23046 \\ 0.14010 & 0.67381 & 0.18609 \\ 0.29646 & 0.15090 & 0.55265 \\ 0.28373 & 0.23677 & 0.47950 \\ 0.57563 & 0.09947 & 0.32490 \\ 0.16525 & 0.05333 & 0.78142 \\ 0.06772 & 0.82776 & 0.10452 \\ 0.13697 & 0.68113 & 0.18190 \\ 0.28862 & 0.15687 & 0.55452 \\ 0.60719 & 0.08937 & 0.30343 \\ 0.47378 & 0.16194 & 0.36428 \end{bmatrix}$$

Setelah dihitung perubahan matriks partisi kemudian dilakukan pemeriksaan kondisi berhenti. Kondisi berhenti diperiksa dengan menghitung selisih fungsi objektif iterasi ke satu dengan fungsi objektif iterasi sebelumnya. $|P_1-P_0|$ = |39727,5665576 - 0| = 39727,5665576 >> ξ = 0,00001 dan iterasi = 1 < maxIter (=50). Pada iterasi pertama ini syarat kondisi berhenti belum terpenuhi yaitu nilai fungsi objektif masih lebih besar dari nilai eror terkecil, dan iterasi masih kurang dari maksimum iterasi sehingga proses harus dilanjutkan ke iterasi 2 (t=2) dan seterusnya sampai didapatkan syarat kondisi berhenti.

Setelah iterasi ke-20, salah satu syarat kondisi berhenti terpenuhi, yaitu $|P_{20} - P_{19}|$ = |29262,9008406 – 29262,9008452| = 0,0000046 < ξ = 0,00001 dengan pusat *cluster* (V_{kj}) akhir yang ditunjukkan pada Tabel 3.7 dan fungsi objektif iterasi ke-20 ditunjukkan pada tabel 3.8.

Tabel 3.7 Pusat *Cluster* Iterasi ke-20 (Pusat *Cluster* Akhir)

Cluster (k)	ATRIBUT(j)									
	1	2	3	4	5	6	7	8	9	10
1	110.864	103.799	166.668	111.097	51.136	0.837	6.355	24.593	1.393	2.043
2	192.637	206.136	244.817	172.142	56.833	0.121	6.927	16.287	0.886	98.715
3	157.678	149.933	194.741	126.317	57.120	0.362	6.568	19.556	1.094	49.852

Tabel 3.8 Fungsi Objektif Iterasi ke-20 (P_{20})

L1	L1	L3	L1+L2+L3
373.78999	2014.69901	1167.26502	3555.75402
1155.39373	59.47035	226.72023	1441.58430
846.47145	39.66905	197.79949	1083.94000
25.55845	14.15675	379.42871	419.14392
353.32286	2249.95679	911.62492	3514.90457
147.93591	1894.38930	368.79310	2411.11831
248.22881	162.04465	1158.28351	1568.55697
164.09435	189.17905	1024.68650	1377.95990
803.21684	17.93321	70.14058	891.29063
9.36508	6.26204	254.05856	269.68569
13.75138	643.15393	46.19254	703.09784
91.69998	1879.41193	222.15844	2193.27036
234.69746	163.33948	1143.13108	1541.16802
1251.54668	149.87506	642.29278	2043.71452
4135.02860	567.07683	1545.60638	6247.71180
		Pt	29262.90084

Nilai matriks U akhir pada iterasi ke-20 berisi informasi mengenai kecenderungan suatu data untuk masuk ke dalam kelompok kelas tingkat risiko penyakit stroke. Derajat keanggotaan terbesar menunjukkan kecenderungan tertinggi untuk menjadi anggota kelompok. Pengelompokan data ini sesuai dengan diagram alir proses pengelompokan data pada Gambar 3.7. Derajat keanggotaan pada data ke-1 adalah 0,10512 pada *cluster* 1; 0,56660 pada *cluster* 2; dan 0,32828 pada *cluster* 3. Derajat keanggotaan terbesar pada data ke-1 adalah 0,56660 pada *cluster* 2, sehingga data ke-1 menjadi anggota *cluster* 2.

Detail derajat keanggotaan setiap data pada setiap *cluster* beserta dengan kecenderungan tertinggi untuk masuk ke suatu kelompok tingkat risiko penyakit stroke ditunjukkan pada Tabel 3.9.

Tabel 3.9 Kecenderungan data terhadap *cluster*

No.	Ui1	Ui2	Ui3	*Cluster* 1	*Cluster* 2	*Cluster* 3
1	0.105122394	0.566601069	0.328276537		v	
2	0.801467291	0.041254954	0.157277755	v		
3	0.780910464	0.036598766	0.182490771	v		
4	0.060973984	0.033773961	0.905252055			v
5	0.100520082	0.640122086	0.259357832		v	
6	0.061354723	0.78569076	0.152954516		v	
7	0.158255081	0.103310423	0.738434496			v
8	0.119081281	0.137286761	0.743631959			v
9	0.901189224	0.020119419	0.078691357	v		
10	0.034724098	0.023218908	0.942056994			v
11	0.019558299	0.914742644	0.065699057		v	
12	0.041809685	0.85689847	0.101291845		v	
13	0.152287465	0.105986643	0.741725893			v
14	0.612379181	0.07333561	0.314285209	v		
15	0.661852391	0.090764037	0.247383571	v		

Berdasarkan Tabel 3.9 pengelompokan data latih untuk anggota *cluster* 1 ditunjukkan pada Tabel 3.10, data latih anggota *cluster* 2 ditunjukkan pada Tabel 3.11, dan data latih anggota *cluster* 3 ditunjukkan pada Tabel 3.12.

Tabel 3.10 Data Anggota *Cluster* 1

No/ Atribut	TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin	Status Risiko
1	100	100	195	127	28	1	6,2	16,2	0,6	0
2	120	106	195	127	55	1	6,2	9,6	1	0
3	110	86	153	105	72	1	6,2	27,4	1,7	0
4	100	155	173	133	56	1	7,2	31,6	1,9	0
5	110	85	97	53	35	0	6,2	48,9	2,2	0

Tabel 3.11 Data Anggota *Cluster* 2

No/ Atribut	TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin	Status Risiko
1	192	231	183	134	76	1	9,8	17,9	1,2	100
2	200	138	267	188	63	0	6,4	13	0,9	100
3	160	192	285	183	63	0	7,5	10,7	0,8	100
4	210	204	229	162	49	0	7,2	23,1	0,7	100
5	200	253	249	187	49	0	5,2	14,1	1	100

Tabel 3.12 Data Anggota *Cluster* 3

No/ Atribut	TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin	Status Risiko
1	140	155	195	127	46	0	6,2	20	1,5	50
2	190	120	197	117	50	0	7,8	13,3	0,9	50
3	140	188	195	127	60	1	6,2	10,5	1	50
4	150	155	195	127	66	0	6,2	30,6	1	50
5	190	119	195	127	63	1	6,2	13,2	0,7	50

Setelah proses *fuzzy c-means clustering* selesai, langkah selanjutnya adalah melakukan analisa varian. Pada proses analisa varian akan dihitung nilai varian pada tiap hasil *cluster* yang terbentuk sebagai langkah untuk menganalisa *cluster*. Analisa *cluster* digunakan untuk mengetahui hasil *cluster* yang ideal untuk proses pembangkitan aturan *fuzzy*.

2. Proses Perhitungan Varian

Proses perhitungan varian digunakan untuk mengetahui baik tidaknya hasil proses *clustering* berdasarkan kepadatan sebaran datanya. Perhitungan nilai batasan varian sesuai dengan diagram alir proses analisa varian pada Gambar (3.9).

Langkah pertama adalah mencari rata-rata data dari tiap *cluster* ($\bar{d}_i$). Contoh perhitungan nilai rata-rata nilai tekanan darah (TD) untuk *cluster* 1 adalah sebagai berikut:

$$\bar{d}_1 = \frac{100+120+110+100+110}{5} = 108$$

Dari rata-rata data setiap *cluster* $\bar{d}_i$ dapat dihitung nilai rata-rata data *cluster* ($\bar{\bar{d}}$) untuk setiap atribut. Contoh perhitungan nilai ($\bar{\bar{d}}$) untuk atribut tekanan darah (TD) adalah sebagai berikut:

$$\bar{\bar{d}} = \frac{108+192{,}4+162}{3} = 154{,}133$$

Rata-rata (d_i) dan ($\bar{d}$) setiap atribut dapat disajikan pada Tabel 3.13.

Tabel 3.13 Rata-rata

Cluster	Rata-rata(d_i)									
	TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin	Status Risiko
1	108	106,4	162,6	109	49,2	0,8	6,4	26,74	1,48	0
2	192,4	203,6	242,6	170,8	60	0,2	7,22	15,76	0,92	100
3	162	147,4	195,4	125	57	0,4	6,52	17,52	1,02	50
rata-rata	**154,133**	**152,467**	**200,2**	**134,933**	**55,4**	**0,467**	**6,713**	**20,007**	**1,14**	**50**

Setelah rata-rata ($\bar{d}_i$) dan ($\bar{d}$) diketahui, langkah selanjutnya yaitu menghitung varian setiap *cluster*. Varian tiap *cluster* (V_c) dapat dihitung dengan menggunakan persamaan (2-8). Berikut ini perhitungan varian tiap *cluster* (V_c) :

$$V_1^2 = \frac{1}{n_c - 1}\sum_{i=1}^{n_c}\left(d_1 - \bar{d}\right)^2$$

$$V_1^2 = \frac{1}{5-1}\begin{pmatrix}(100-108)^2+\ldots+(110-108)^2+(100-106{,}4)^2+\ldots+(85-104{,}6)^2\\+(195-162{,}6)^2+\ldots+(97-162{,}6)^2+(127-109)^2+\ldots+(53-109)^2+(28-49{,}2)^2\\+\ldots+(35-49{,}2)^2+(1-0{,}8)^2+\ldots+(0-0{,}8)^2+(6{,}2-6{,}4)^2+\ldots+(6{,}2-6{,}4)^2+(16{,}2-26{,}74)^2\\+\ldots+(48{,}9-26{,}74)^2+(0{,}6-1{,}48)^2+\ldots+(2{,}2-1{,}48)^2+(0-0)^2+\ldots+(0-0)^2\end{pmatrix}$$

$$V_1^2 = 4177{,}635$$

Dengan langkah perhitungan yang sama pada *cluster* kedua dan *cluster* ketiga, maka didapatkan hasil nilai varian pada kedua *cluster* adalah sebagai berikut:

$$V_1^2 = 4177{,}635$$
$$V_2^2 = 4509{,}317$$
$$V_3^2 = 1661{,}746$$

Setelah nilai varian setiap *cluster* diketahui, maka selanjutnya dapat dihitung *variance within cluster* (V_w) dan *variance between cluster* (V_b). Nilai V_w didapatkan dengan menggunakan persamaan (2-9) dengan N = jumlah data, k = jumlah *cluster*, n_i = jumlah anggota tiap *cluster* dan Vi = varian tiap *cluster*. Hasil perhitungan *variance within cluster* (V_w) adalah sebagai berikut :

$$V_w = \frac{1}{N-k}\sum_{i=1}^{k}(n_i - 1)\times V_i^2$$

$$V_w = \frac{1}{15-3}(((5-1)\times 4177{,}635)+((5-1)\times 4509{,}317)+((5-1)\times 1661{,}746))$$

$$V_w = 3449{,}566$$

Nilai *variance between cluster* (V_b) dapat dihitung dengan menggunakan Persamaan (2-10). Hasil perhitungan V_b adalah sebagai berikut :

$$V_b = \frac{1}{k-1}\sum_{i=1}^{k} n_i\left(d_i - \bar{d}\right)^2$$

$$V_b = \frac{1}{3-1}\left(\begin{array}{l}\left(5\times(108-154{,}133)^2\right)+\ldots+\left(5\times(0-50)^2\right)+\left(5\times(192{,}4-154{,}133)^2\right)+\ldots+\left(5\times(100-50)^2\right)\\ +\left(5\times(162-154{,}133)^2\right)+\ldots+\left(5\times(50-50)^2\right)\end{array}\right)$$

$$V_b = 47103{,}982$$

Setelah nilai *variance within cluster* (V_w) dan *variance between cluster* (V_b) diketahui, maka batasan varian *(V)* dapat dihitung dengan Persamaan (2-11) sebagai berikut:

$$V = \frac{V_w}{V_b} = \frac{3449{,}566}{47103{,}982} = 0{,}0732330$$

Suatu *cluster* dikatakan layak apabila nilai V_w lebih kecil daripada V_b. Semakin kecil V_w dan semakin besar V_b maka *cluster* semakin baik. Apabila nilai V_w lebih besar dari V_b berarti ada kesalahan dalam proses *clustering*. Setelah dilakukan perhitungan varian terhadap *cluster*, maka jumlah *cluster* yang paling ideal dapat diketahui.

3. Proses Ekstraksi Aturan *Fuzzy*

Proses ekstraksi aturan *fuzzy* merupakan proses untuk membentuk aturan beserta koefisien *output*nya yang nantinya akan digunakan pada *fuzzy inference system* model sugeno orde-satu.

Proses ekstraksi aturan *fuzzy* sesuai dengan diagram alir proses ekstraksi aturan fuzzy pada Gambar (3.11). Langkah pertama adalah mencari derajat keanggotaan data pada masing-masing *cluster*. Derajat keanggotaan ini dapat dibentuk dengan menggunakan fungsi *Gauss*. Nilai pusat *cluster* dan standar deviasi perlu diketahui agar dapat membentuk fungsi keanggotaan *Gauss*.

Nilai pusat *cluster* didapatkan dari hasil perhitungan pusat *cluster*. Standar deviasi diperoleh dari perhitungan peramaan (2-24) dan sesuai dengan diagram alir proses perhitungan standar deviasi pada Gambar (3.12). Berikut ini adalah contoh perhitungan standar deviasi:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}\left(x_i - \overline{x_c}\right)^2}{n_c - 1}} = \sqrt{\frac{\sum_{i=1}^{4}\left(x_i - \overline{x_c}\right)^2}{4-1}}$$

$$\sigma = \sqrt{\frac{\left((192-108)^2 + (200-108)^2 + (160-108)^2 + ... + (200-108)^2\right)}{5-1}}$$

$$\sigma = 8{,}367$$

Hasil perhitungan standar deviasi ditunjukkan pada Tabel 3.14.

Tabel 3.14 Standar Deviasi

Sigma (kj)	1	2	3	4	5	6	7	8	9	10
1	8.367	28.623	40.630	33.076	17.683	0.447	0.447	15.166	0.661	0.000
2	19.204	43.673	39.278	23.124	11.358	0.447	1.695	4.859	0.192	0.000
3	25.884	28.815	0.894	4.472	8.602	0.548	0.716	8.108	0.295	0.000

Setelah standar deviasi diketahui selanjutnya adalah menghitung derajat keanggotaan dengan menggunakan fungsi gauss berdasarkan Persamaan (2-29).

$$\mu_{ki} = e^{-\sum_{j=1}^{m}\frac{\left(X_{ij} - C_{kj}\right)^2}{2\sigma_j^2}}$$

$$\mu_{11} = e^{-\sum_{j=1}^{10}\frac{\left(X_{1j} - C_{1j}\right)^2}{2\sigma_j^2}}$$

$$\mu_{11} = e^{-\left(\left(\frac{(192-192{,}4)^2}{2\times(19{,}204)^2}\right)+\left(\frac{(231-203{,}6)^2}{2\times(43{,}673)^2}\right)+\left(\frac{(183-242{,}6)^2}{2\times(39{,}278)^2}\right)+\left(\frac{(134-170{,}8)^2}{2\times(23{,}124)^2}\right)+\left(\frac{(76-60)^2}{2\times(11{,}358)^2}\right)\right.}$$
$$\left.+\left(\frac{(1-0{,}2)^2}{2\times(0{,}447)^2}\right)+\left(\frac{(9{,}8-7{,}22)^2}{2\times(1{,}695)^2}\right)+\left(\frac{(17{,}9-15{,}76)^2}{2\times(4{,}859)^2}\right)+\left(\frac{(1{,}2-0{,}92)^2}{2\times(0{,}192)^2}\right)+\left(\frac{(1-1)^2}{2\times(0)^2}\right)\right)$$

$$\mu_{11} = 4{,}3356E-21$$

Hasil perhitungan derajat keanggotaan ditunjukkan pada Tabel 3.15.

Tabel 3.15 Derajat Keanggotaan

μ_{1i}	μ_{2i}	μ_{3i}
4.33564E-21	0.007391703	1.8568E-17
0.004628542	0.002615535	2.61248E-08
0.723273541	0.00434132	8.09215E-08
0.001580111	0.02412709	1.61843E-09
0.003923865	0.483187529	0.482567757
3.65606E-05	0.189647109	0.189647109
4.52842E-11	0.041593681	0.003792942
0.351035183	0.103328428	2.08288E-09
0.003656042	2.70339E-08	1.6542E-294
1.96645E-12	0.005676815	0.108896276
1.0565E-17	0.201578131	0.199251564
9.90198E-15	0.323448728	1.25E-08
1.81108E-06	0.261174496	1.44095E-08
0.000896922	3.22929E-08	1.00136E-57
3.66201E-10	1.28378E-21	4.93051E-09

Langkah selanjutnya untuk ekstraksi aturan fuzzy adalah mencari nilai d_{ij}^{k} yang terbentuk dari derajat keanggotaan data pada masing-masing *cluster* menggunakan persamaan (2-30). Berikut ini adalah contoh perhitungan d_{ij}^{k} untuk aturan pertama.

$$d_{11}^{1} = x_{11} \times \mu_{11} = 192 \times 4{,}3356\ E-21 = 8{,}3244\ E-19$$
$$d_{12}^{1} = x_{12} \times \mu_{12} = 231 \times 4{,}3356\ E-21 = 1{,}00153\ E-18$$
$$d_{13}^{1} = x_{13} \times \mu_{13} = 183 \times 4{,}3356\ E-21 = 7{,}934\ E-19$$
$$d_{14}^{1} = x_{14} \times \mu_{14} = 134 \times 4{,}3356\ E-21 = 5{,}81\ E-19$$
$$d_{15}^{1} = x_{15} \times \mu_{15} = 76 \times 4{,}3356\ E-21 = 3{,}295\ E-19$$
$$d_{16}^{1} = x_{16} \times \mu_{16} = 1 \times 4{,}3356\ E-21 = 4{,}3356\ E-21$$
$$d_{17}^{1} = x_{17} \times \mu_{17} = 9{,}8 \times 4{,}3356\ E-21 = 4{,}249\ E-20$$
$$d_{18}^{1} = x_{18} \times \mu_{18} = 17{,}9 \times 4{,}3356\ E-21 = 7{,}761\ E-20$$
$$d_{19}^{1} = x_{19} \times \mu_{19} = 1{,}2 \times 4{,}3356\ E-21 = 5{,}2028\ E-21$$
$$d_{110}^{1} = x_{110} \times \mu_{110} = 100 \times 4{,}3356\ E-21 = 4{,}3356\ E-21$$

Hasil perhitungan d_{ij}^k adalah sebagai berikut:

$$\begin{bmatrix} 8.3244E-19 & 1.00153E-18 & 7.934E-19 & \cdots & 1.8568E-17 \\ 0.41010629 & 0.410106287 & 0.7997073 & \cdots & 3.13933E-08 \\ 86.9645334 & 76.81867117 & 141.31737 & \cdots & 5.42502E-08 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 5.8264E-07 & 4.5022E-07 & 5.138E-07 & \cdots & 0 \end{bmatrix}$$

Setelah d_{ij}^k diketahui, kemudian dilakukan proses normalisasi. Proses normalisasi adalah proses untuk mendapatkan nilai d_{ij}^k yang nantinya digunakan untuk pembentukan matriks *U*. Normalisasi dilakukan dengan menggunakan persamaan (2-27) dan persamaan (2-28). Berikut ini adalah contoh perhitungan normalisasi matriks d_{ij}^k.

$$d_{11}^1 = \frac{d_{11}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{8{,}3244\,E-19}{0{,}07391703} = 1{,}1262\,E-16$$

$$d_{12}^1 = \frac{d_{12}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{1{,}00153\,E-18}{0{,}07391703} = 1{,}35494\,E-16$$

$$d_{13}^1 = \frac{d_{13}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{7{,}934\,E-19}{0{,}07391703} = 1{,}073\,E-16$$

$$d_{14}^1 = \frac{d_{14}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{5{,}81E-19}{0{,}07391703} = 4{,}4578\,E-17$$

$$d_{15}^1 = \frac{d_{15}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{3{,}2951\,E-19}{0{,}07391703} = 4{,}4578\,E-17$$

$$d_{16}^1 = \frac{d_{16}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{4{,}3356\;E-21}{0{,}07391703} = 5{,}8655\;E-19$$

$$d_{17}^1 = \frac{d_{17}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{4{,}249\,E-20}{0{,}07391703} = 5{,}748\,E-18$$

$$d_{18}^1 = \frac{d_{18}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{7{,}761E-20}{0{,}07391703} = 1{,}05E-17$$

$$d_{19}^1 = \frac{d_{19}^1}{\sum_{k=1}^{3} \mu_{k1}} = \frac{5{,}2028\;E-21}{0{,}07391703} = 7{,}0387\;E-19$$

$$d^{1}_{110} = \frac{d^{1}_{110}}{\sum_{k=1}^{3} \mu_{k1}} = \frac{4{,}3356\ E-21}{0{,}07391703} = 5{,}8655\ E-19$$

Setelah proses normalisasi selanjutnya dilakukan pembentukan matriks U. Masukan dari matriks *U* adalah matriks d^{k}_{ij} dan menghasilkan keluaran berupa matriks *U* yang berdimensi jumlah data (*n*) × (jumlah *cluster* * (jumlah atribut + 1)). Pembentukan matriks U didasarkan pada persamaan (2-29). Hasil Pembentukan matriks U adalah sebagai berikut:

$$\begin{bmatrix} 1.1262E-16 & 1.35494E-16 & 1.073E-16 & \cdots & 2.51201E-15 \\ 56.6124369 & 56.61243691 & 110.39425 & \cdots & 4.33364E-06 \\ 119.519994 & 105.5759946 & 194.21999 & \cdots & 7.45589E-08 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 110 & 85 & 97 & \cdots & 0 \end{bmatrix}$$

Setelah pembentukan matriks U, langkah selanjutnya adalah perhitungan LSE. Matrik *U* nantinya berperan pada proses perhitungan LSE untuk mendapatkan nilai koefisien *output*. Perhitungan LSE dilakukan sesuai dengan diagram alir proses perhitungan LSE pada Gambar (4.16). Inputan proses perhitungan LSE adalah matriks U dan matriks Y. Setelah pembentukan matriks U selesai, maka langkah selanjutnya adalah membentuk matriks U Transpose ($\boldsymbol{U^T}$). Matriks U Transpose dihitung menggunakan Microsoft excel dengan fungsi TRANSPOSE(). Hasil Pembentukan matriks $\boldsymbol{U^T}$ adalah sebagai berikut:

$$\begin{bmatrix} 1.1262E-16 & 56.6124369 & 119.519994 & \cdots & 110 \\ 1.35494E-16 & 56.6124369 & 105.5759946 & \cdots & 85 \\ 1.073E-16 & 110.39425 & 194.21999 & \cdots & 97 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 2.51201E-15 & 4.33364E-06 & 7.45589E-08 & \cdots & 0 \end{bmatrix}$$

Setelah dilakukan perhitungan matriks U Transpose kemudian dilanjutkan operasi perkalian matriks antara matriks $\boldsymbol{U^T}$ dengan matriks *U* ($\boldsymbol{U^T * U}$) sehingga didapatkan matriks hasil pekalian seperti berikut ini:

$$\begin{bmatrix} 104652.7325 & 110788.405 & 146236.1296 & \cdots & 0.000385319 \\ 110788.405 & 123302.346 & 157600.23 & \cdots & 0.00039376 \\ 146236.1296 & 157600.23 & 210568.6632 & \cdots & 0.000669612 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.000385319 & 0.00039376 & 0.000669612 & \cdots & 1.92358E-11 \end{bmatrix}$$

Dari hasil perkalian matriks $\mathbf{U^T * U}$ kemudian dilakukan operasi invers pada matriks tersebut. Matriks $\mathbf{U^T * U}$ diinverse menggunakan bantuan Microsoft Excel dengan fungsi MINVERSE(). Hasil inverse matriks $\mathbf{(U^T * U)^{-1}}$ adalah sebagai berikut:

$$\begin{bmatrix} 0.023578443 & -0.0047986 & -0.035099666 & \cdots & 9.54319E-07 \\ -0.004798584 & 0.00128009 & 0.007083948 & \cdots & -1.2722E-07 \\ -0.035099666 & 0.00708395 & 0.053290132 & \cdots & -1.38578E-06 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 9.54319E-07 & -1.272E-07 & -1.38578E-06 & \cdots & 1.18679E-10 \end{bmatrix}$$

Matriks $\mathbf{(U^T x U)^{-1})}$ kemudian dikalikan dengan matriks $\boldsymbol{U^T}$ sehingga menghasilkan matriks $\mathbf{(U^T x U)^{-1} x U^T}$ berukuran 30 x 15 seperti berikut ini:

$$\begin{bmatrix} -0.000373854 & -0.0199636 & -0.052702578 & \cdots & -0.016719207 \\ -0.000311121 & 0.00400169 & 0.000488386 & \cdots & 0.004038754 \\ 0.000799078 & 0.03604597 & 0.086598491 & \cdots & 0.034199268 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1.78053E-07 & 4.4937E-06 & -6.43561E-06 & \cdots & -1.38806E-06 \end{bmatrix}$$

Hasil perkalian matriks invers dan matriks transpose $\mathbf{(U^T x U)^{-1} x U^T}$ tersebut kemudian dikalikan lagi dengan matriks data output Y (15x1) yaitu $\mathbf{((U^T x U)^{-1} x U^T) * Y}$ sehingga menghasilkan matriks K berukuran (15x1). Matriks K ini berisi koefisien output. Hasil perhitungan pada matriks K yaitu:

Matriks data output y

100
0
0
50
100
100
50
50
0
50
100
100
50
0
0

Dengan demikian didapatkan nilai koefisien output sebagai berikut :

Matriks K

3.508655524
-0.186052416
-4.792126521
5.029370287
-1.001744722
-4.628640495
-2.985150081
-2.132653133
-5.71725684
-1.424305892
-0.404859613
0.020808918
-0.588558601
1.31768843
0.610516898
-15.31870143
4.313983979
1.932941481
-0.716101759
-3.035795935
-0.005953171
-0.005985442

-0.011682822
-0.007608812
-0.001553706
-6.00705E-05
-0.000371454
-0.000923802
-3.78378E-05
-5.99119E-05

Dari hasil perhitungan matriks K, barulah dibentuk menjadi matriks yang berukuran $r \times m$, dimana r merupakan jumlah aturan dan m panjang atribut. Sebagai hasil akhir untuk koefisien *output* (K_{ij}) yang siap digunakan pada proses inferensi *fuzzy*. Nilai koefisien *output* ditunjukkan pada Tabel 3.16 dan Tabel 3.17.

Tabel 3.16 Koefisien *Output* (1)

Koefisien Output					
R	j=1	j=2	j=3	j=4	j=5
R1	3.508655524	-0.186052416	-4.79212652	5.029370287	-1.001744722
R2	-0.404859613	0.020808918	-0.5885586	1.31768843	0.610516898
R3	-0.005953171	-0.005985442	-0.01168282	-0.007608812	-0.001553706

Tabel 3.17 Koefisien *Output* (2)

Koefisien Output					
R	j=6	j=7	j=8	j=9	j=10
R1	-4.628640495	-2.985150081	-2.132653133	-5.71725684	-1.424305892
R2	-15.31870143	4.313983979	1.932941481	-0.716101759	-3.035795935
R3	-6.00705E-05	-0.000371454	-0.000923802	-3.78378E-05	-5.99119E-05

Dari matriks koefisien *output* ini diperoleh parameter-parameter *output* untuk setiap aturan sebagai berikut:

$$Z_1 = (k_{11} * X_{11})+(k_{12} * X_{12})+(k_{13} * X_{13})+(k_{14} * X_{14})+(k_{15} * X_{15}) +(k_{16} * X_{16})+(k_{17} * X_{17})+(k_{18} * X_{18})+(k_{19} * X_{19})+k_{110}$$

$$= (3.508655524 * X_{11})+(-0.186052416 * X_{12})+(-4.79212652 * X_{13}) +(5.029370287 * X_{14})+(-1.001744722 * X_{15})+(-4.628640495 * X_{16}) +(-2.985150081 * X_{17})+(-2.132653133 * X_{18})+(-5.71725684 * X_{19}) +(-1.424305892)$$

$$Z_2 = (k_{21} * X_{21})+(k_{22} * X_{22})+(k_{23} * X_{23})+(k_{24} * X_{24})+(k_{25} * X_{25}) +(k_{26} * X_{26})+(k_{27} * X_{27})+(k_{28} * X_{28})+(k_{29} * X_{29})+k_{2.10}$$

$= (-0.404859613 * X_{21})+(0.020808918 * X_{22})+(-0.5885586 * X_{23})$
$+(1.31768843 * X_{24})+(0.610516898 * X_{25})+(-15.31870143 * X_{26})$
$+(4.313983979 * X_{27})+(1.932941481 * X_{28})+(-0.716101759 * X_{29})$
$+(-3.035795935)$

$Z_3 = (k_{31} * X_{31})+(k_{32} * X_{32})+(k_{33} * X_{33})+(k_{34} * X_{34})+(k_{35} * X_{35})$
$+(k_{36} * X_{36})+(k_{37} * X_{37})+(k_{38} * X_{38})+(k_{39} * X_{39})+k_{3.10}$
$= (-0.005953171 * X_{31})+(-0.005985442 * X_{32})+(-0.01168282 * X_{33})$
$+(-0.007608812 * X_{34})+(-0.001553706 * X_{35})+(-6.00705E-05 * X_{36})$
$+(-0.000371454 * X_{37})+(-0.000923802 * X_{38})+(-3.78378E-05 * X_{39})$
$+(-5.99119E-05)$

Setiap variabel *input* juga akan terbagi menjadi 3 himpunan *fuzzy*. Aturan-aturan yang terbentuk adalah sebagai berikut:

[R1]: IF (TEKANAN DARAH is $center_{11}$) and (KADAR GULA is $center_{12}$) and (KOLESTEROL TOTAL is $center_{13}$) and (LDL is $center_{14}$) and (Usia is $center_{15}$) and (JENIS KELAMIN is $center_{16}$) and (ASAM URAT is $center_{17}$) and (BUN is $center_{18}$) and (KREATININ is $center_{19}$) THEN Status Risiko = Z_1

[R2]: IF (TEKANAN DARAH is $center_{21}$) and (KADAR GULA is $center_{22}$) and (KOLESTEROL TOTAL is $center_{23}$) and (LDL is $center_{24}$) and (Usia is $center_{25}$) and (JENIS KELAMIN is $center_{26}$) and (ASAM URAT is $center_{27}$) and (BUN is $center_{28}$) and (KREATININ is $center_{29}$) THEN Status Risiko = Z_2

[R3]: IF (TEKANAN DARAH is $center_{31}$) and (KADAR GULA is $center_{32}$) and (KOLESTEROL TOTAL is $center_{33}$) and (LDL is $center_{34}$) and (Usia is $center_{35}$) and (JENIS KELAMIN is $center_{36}$) and (ASAM URAT is $center_{37}$) and (BUN is $center_{38}$) and (KREATININ is $center_{39}$) THEN Status Risiko = Z_3

3.4.2. Proses Pengujian

Sebelum dilakukan pengujian terhadap data uji, terlebih dahulu dilakukan penentuan *range* untuk menentukan *range* dari tiap kelas status risiko (rendah, sedang, dan tinggi). Pada proses penentuan *range* yang dibutuhkan yaitu data latih, pusat *cluster* (*center*), nilai *sigma cluster*, serta parameter *output* pada tiap aturan. Data-data tersebut kemudian digunakan untuk mencari nilai Z dari tiap data latih, Setelah didapat nilai Z dari tiap data latih, kemudian dihitung berapakah jumlah total data dari masing-masing status risiko (kelas rendah, kelas sedang, dan kelas tinggi). Setelah itu dilakukan perulangan untuk pengecekan status risiko, lalu nilai Z dimasukkan kedalam array sRendah[S], sSedang[R], dan

sTinggi[T]. Contoh array sRendah[S], sSedang[R], dan sTinggi[T] adalah sebagai berikut:

sRendah[R] = {41,530; 25,161; 20,471; ...; -6,269}

sSedang[S] = {47,056; 64,934; 45,645; ...; 59,714}

sTinggi[T] = {86,214; 66,153; 89,319; ...; 55,679}

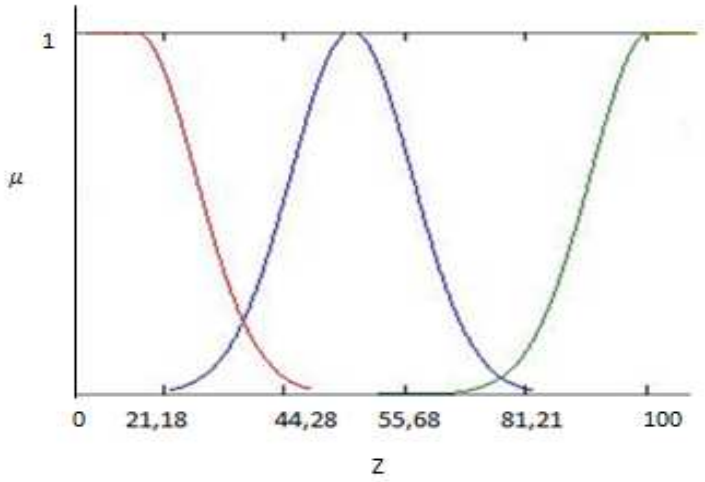
Dari ketiga array tersebut, masing-masing dicari nilai minimum dan maksimumnya. Nilai minimum dan maksimum ini nantinya digunakan sebagai *range* dari masing-masing status risiko (kelas rendah, kelas sedang, dan kelas tinggi). Berikut hasil nilai minimum dan maksimum dari masing-masing status risiko :

Rmin = -13,501 ; Rmax = 44,278

Smin = 21,184 ; Smax = 81,208

Tmin = 55,679 ; Tmax = 145,834

Sehingga dari nilai minimum dan maksimum itulah didapatkan grafik *range* status risiko adalah seperti berikut :



Gambar 3.26 Grafik *range* status risiko

Setelah dilakukan penentuan range dari tiap status risiko, maka dilakukan proses pengujian. Proses pengujian dilakukan terhadap data uji dengan menggunakan aturan yang telah terbentuk pada proses pembangkitan aturan *fuzzy*. Data yang dibutuhkan dalam proses ini, atara lain data uji, pusat *cluster* (*center*), nilai *sigma cluster*, serta parameter *output* pada tiap aturan. Proses inferensi *fuzzy*

dilakukan sesuai dengan diagram alir proses sistem inferensi *fuzzy* pada Gambar (3.19). Data uji yang digunakan, ditunjukkan pada Tabel 3.18.

Tabel 3.18 Data Uji

TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin	Status Risiko
190	120	197	117	50	0	7.8	13.3	0.9	50

Selanjutnya adalah mencari derajat keanggotaan tiap atribut data uji terhadap aturan dengan persamaan (2-32). Sebagai contoh perihutngan derajat keanggotaan pada aturan pertama adalah sebagai berikut:

$$\mu_{11} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{11}-V_{11})^2}{2\sigma_1^2}} = e^{-\frac{(150-110{,}864)^2}{2(8{,}3666)^2}} = 3{,}74E-20$$

$$\mu_{12} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{12}-V_{12})^2}{2\sigma_2^2}} = e^{-\frac{(120-103{,}798)^2}{2(28{,}6234)^2}} = 0{,}85198$$

$$\mu_{13} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{13}-V_{13})^2}{2\sigma_3^2}} = e^{-\frac{(197-166{,}667)^2}{2(40{,}6300)^2}} = 0{,}7567$$

$$\mu_{14} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{14}-V_{14})^2}{2\sigma_4^2}} = e^{-\frac{(117-111{,}096)^2}{2(33{,}0756)^2}} = 0{,}9841$$

$$\mu_{15} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{15}-V_{15})^2}{2\sigma_5^2}} = e^{-\frac{(50-51{,}1364)^2}{2(17{,}6833)^2}} = 0{,}9979$$

$$\mu_{16} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{16}-V_{16})^2}{2\sigma_6^2}} = e^{-\frac{(0-0{,}837)^2}{2(0{,}447)^2}} = 0{,}1734$$

$$\mu_{17} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{17}-V_{17})^2}{2\sigma_7^2}} = e^{-\frac{(7{,}8-6{,}354)^2}{2(0{,}447)^2}} = 0{,}0054$$

$$\mu_{18} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{18}-V_{18})^2}{2\sigma_8^2}} = e^{-\frac{(13{,}3-24{,}5936)^2}{2(15{,}1656)^2}} = 0{,}7578$$

$$\mu_{11} = e^{-\frac{(X_{ij}-C_{kj})^2}{2\sigma_j^2}} = e^{-\frac{(X_{11}-V_{11})^2}{2\sigma_1^2}} = e^{-\frac{(0{,}9-1{,}392)^2}{2(0{,}661)^2}} = 0{,}7573$$

Hasil selengkapnya dari nilai derajat keanggotaan masing-masing aturan (μ_{rj}) ditunjukkan pada Tabel 3.19.

Tabel 3.19 Derajat Keanggotaan Aturan *Fuzzy*

R	TD	KG	KT	LDL	U	JK	AU	BUN	Kreatinin
R1	3.74E-20	0.85198	0.7567	0.9841	0.9979	0.1734	0.0054	0.7578	0.7573
R2	0.990616	0.14299	0.4766	0.0582	0.8344	0.9641	0.8757	0.8278	0.9971
R3	0.458562	0.58300	0.0411	0.1141	0.7099	0.8035	0.2273	0.7425	0.8058

Kemudian tahapan selanjutnya adalah mencari *fire strength* (α-predikat) untuk setiap aturan. Apabila digunakan operator *product* sebagai operator pada anteseden maka dapat diperoleh:

$[R1] : \alpha_1 = \mu_{11} \bullet \mu_{12} \bullet \mu_{13} \bullet \mu_{14} \bullet \ldots \bullet \mu_{19}$

$= 0{,}000 \bullet 0{,}851 \bullet 0{,}756 \bullet 0{,}984 \bullet \ldots \bullet 0{,}757 = 0{,}000$

$[R2] : \alpha_2 = \mu_{21} \bullet \mu_{22} \bullet \mu_{23} \bullet \mu_{24} \bullet \ldots \bullet \mu_{29}$

$= 0{,}990 \bullet 0{,}142 \bullet 0{,}476 \bullet 0{,}058 \bullet \ldots \bullet 0{,}997 = 0{,}002$

$[R3] : \alpha_3 = \mu_{31} \bullet \mu_{32} \bullet \mu_{33} \bullet \mu_{34} \bullet \ldots \bullet \mu_{39}$

$= 0{,}458 \bullet 0{,}583 \bullet 0{,}041 \bullet 0{,}114 \bullet \ldots \bullet 0{,}805 = 0{,}000$

Menentukan nilai *Z* untuk setiap aturan merupakan langkah selanjutnya pada proses inferensi fuzzy sugeno orde-satu. Berikut ini adalah perhitungan nilai *Z* dari ketiga aturan yang telah terbentuk:

$$Z_1 = (k_{11} \times x_{11}) + (k_{12} \times x_{12}) + \ldots + (k_{19} \times x_{19}) + k_{110}$$
$$Z_1 = (-3{,}5086 \times 190) + (-0{,}1860 \times 120) + \ldots + (-5{,}7172 \times 0{,}9) + -1{,}4243$$
$$Z_1 = 180{,}4001$$

Hasil nilai Z untuk tiap aturan ditunjukkan pada Tabel 3.20.

Tabel 3.20 Nilai Z Untuk Setiap Aturan

	Z
Z_1	180.4001281
Z_2	49.9999995
Z_3	-5.134065618

Langkah terakhir, yaitu menghitung nilai *Z* total (*defuzzy*) dengan metode *weighted average*.

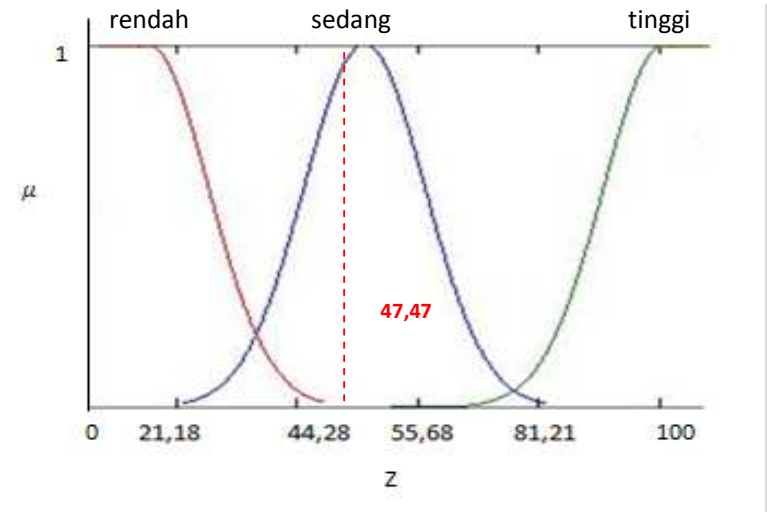
$$Z = \frac{(a_1 \times Z_1) + (a_2 \times Z_2) + (a_3 \times Z_3)}{a_1 + a_2 + a_3}$$

$$Z = \frac{(0{,}000000 \times 180{,}4001281) + (0{,}002286545 \times 49{,}9999995) + (0{,}000097473 \times (-5{,}134065618))}{0{,}000000 + 0{,}002286545 + 0{,}000097473}$$

$$Z = 47{,}74579141$$

Dengan demikian hasil akhir untuk proses pengujian dalam mendapatkan nilai risiko stroke dari data uji dengan nilai atribut tekanan darah (TD) = 190, kadar gula (KG) = 120, kolesterol total (KT) = 197, *Low Density Lipoprotein* (LDL) = 117, usia (U) = 50, jenis kelamin (JK) = 0 (perempuan), asam urat (AU) = 7.8, *Blood Urea Nitrogen* (BUN) = 13.3, kreatinin = 0.9 adalah sebesar 47,74579141.

Selanjutnya, dari nilai Z tersebut ditentukan masuk kelas rendah, sedang, atau tinggi dengan melihat grafik derajat keanggotaan tingkat risiko yang telah didapat sebelumnya. Adapun grafik derajat keanggotaan tingkat risiko adalah seperti berikut:

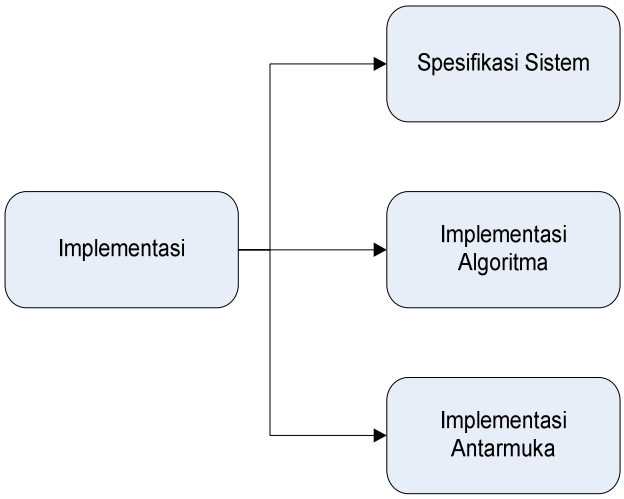


Gambar 3.27 Grafik Derajat Keanggotaan Tingkat Status Risiko

Dari gambar diatas, dapat dilihat bahwa nilai Z masuk dalam range kelas sedang, sehingga dapat dikatakan bahwa pasien tersebut mempunyai risiko yang sedang untuk terkena penyakit stroke.

BAB IV
IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai implementasi sistem berdasarkan proses perancangan. Diagram implementasi sebagai gambaran umum pokok bahasan pada bab ini ditunjukkan pada Gambar (4.1).



Gambar 4.1 Diagram Implementasi

Implementasi terdiri dari spesifikasi sistem, implementasi algoritma, dan implementasi antarmuka. Implementasi sistem deteksi dini risiko stroke ini dibangun dengan mengacu pada pemodelan menggunakan metode *fuzzy c-means clustering* dan metode *fuzzy inference system* sugeno orde-satu dan sesuai dengan model perancangan.

4.1. Spesifikasi Sistem

Spesifikasi sistem yang dimaksud meliputi spesifikasi perangkat keras (*hardware*) dan perangkat lunak (*software*) yang digunakan untuk implementasi.

4.1.1. Spesifikasi Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan sistem adalah sebuah *notebook* dengan spesifikasi sebagai berikut:

1. *Prosesor* Intel(R) Core(TM)2 Duo CPU T8300 @2.40GHz

2. *Memori* 3.00 Gb RAM
3. *Harddisk* 250 GB
4. *Monitor* 14”

4.1.2. Spesifikasi Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistem diantaranya adalah sebagai berikut:

1. Sistem operasi Windows XP professional Version 2002
2. Microsoft Visual Studio 2010
3. XAMPP 1.7.1

4.2. Implementasi Algoritma

Berdasarkan analisa dan perancangan sistem yang telah dibangun pada bab sebelumnya, maka pada bab implementasi ini akan dijelaskan cara menerapkan hasil perancangan ke dalam sebuah program yang di dalamnya berisikan *source code* yang membentuk sistem. Algoritma ini diimplementasikan ke dalam beberapa kelas. Secara umum kelas yang dibentuk merupakan kelas untuk mengimplementasikan tampilan antarmuka sistem dan kelas untuk mengoperasikan proses jalannya sistem. Penjelasan kelas-kelas pembentuk sistem deteksi dini risiko stroke dijelaskan pada Tabel (4.1).

Tabel 4.1 Deskripsi Kelas Program

Nama kelas	Deskripsi
FCM.cs	Kelas yang berisi proses pelatihan data latih dengan menggunakan *fuzzy c-means*.
VARIAN.cs	Kelas yang digunakan untuk memproses nilai batasan varian dari hasil *clustering*.
EKSTRAKSIATURANFUZZY.cs	Kelas yang digunakan untuk pembentukan aturan *fuzzy* dan menentukan nilai koefisien output.
PENGUJIANRANGE.cs	Kelas yang digunakan untuk menentukan *range* dari nilai status risiko.
FISSUGENO.cs	Kelas yang digunakan untuk proses pengujian terhadap data uji dengan menggunakan *fuzzy inference system*

	model Sugeno.
Form1.cs	Kelas ini digunakan untuk menampilkan antarmuka

4.2.1. Implementasi Proses *Clustering* Menggunakan *Fuzzy C-Means*

Kelas FCM.cs merupakan kelas yang yang dibuat untuk proses pembelajaran data latih menggunakan metode *fuzzy c-means clustering.* Proses FCM *clustering* diterapkan pada kelas FCM.cs sesuai dengan diagram alir pada Gambar (3.3), dimana pada kelas ini terdapat 7 proses utama yang diterapkan pada beberapa *method*. Penjelasan *method* pembentuk kelas FCM.cs dijelaskan pada Tabel (4.2).

Tabel 4.2 Method-method Pembentuk Kelas FCM.cs

Nama *Method*	Deskripsi
datalatih()	*Method* ini digunakan untuk membaca dan menampilkan data latih dari *database*.
matriksPartisiAwal()	*Method* ini digunakan untuk membentuk matriks partisi awal
CariPusat*Cluster*(double[,] matriksU, double[,] matriksData)	*Method* ini digunakan untuk menghitung nilai pusat *cluster*
HitungFungsiObyektif(double[,] matriksAwal, double[,] matriksV, double[,] matriksX)	*Method* ini digunakan untuk menghitung nilai fungsi objektif
perubahanMatriksPartisi(double[,] matriksAwal, double[,] matriksData, double[,] matriksV)	*Method* ini digunakan untuk menghitung perubahan matriks partisi sampai syarat kondisi berhenti terpenuhi
isBerhenti(double Pt, double PtMin1, int iterasiKe)	*Method* ini digunakan untuk mengecek syarat kondisi berhenti
tentukanIdx*Cluster*(double[,] matriksU)	*Method* ini digunakan untuk menentukan id *cluster* dari data latih

1. Implementasi Proses Baca Data Latih

Proses pertama pada *FCM clustering* adalah membaca data latih. Proses masukan data latih ini diterapkan kedalam sebuah *method* datalatih(). Untuk dapat

membaca data latih yang telah disimpan sebelumnya pada sebuah *database*, maka diperlukan sebuah *method* FCM() untuk mengoneksikan antara program dengan *database*. Penerapan untuk proses baca data latih ditunjukkan pada *Source code* 4.1.

```
public void datalatih()
        {
            try
            {
                koneksi.Open();

                string query = "select * from data_latih";
                comm = new MySqlCommand(query, koneksi);
                comm.ExecuteNonQuery();
                adapter = new MySqlDataAdapter(comm);
                read = comm.ExecuteReader();
                matriksData = new double[data, atribut];
                int i = 0;
                while (read.Read())
                {
                    int id = Convert.ToInt16(read["id_latih"]);
                    int TekananDarah =
Convert.ToInt16(read["tekanan_darah"]);
                    int KadarGula =
Convert.ToInt16(read["kadar_gula"]);
                    int Kolesterol =
Convert.ToInt16(read["kolesterol_total"]);
                    int LDL = Convert.ToInt16(read["ldl"]);
                    int Usia = Convert.ToInt16(read["usia"]);
                    int JenisKelamin =
Convert.ToInt16(read["jenis_kelamin"]);
                    double AsamUrat =
Convert.ToDouble(read["asam_urat"]);
                    double BUN = Convert.ToDouble(read["bun"]);
                    double Kreatinin =
Convert.ToDouble(read["kreatinin"]);
                    int StatusRisiko =
Convert.ToInt16(read["status"]);

                    matriksData[i, 0] = TekananDarah;
                    matriksData[i, 1] = KadarGula;
                    matriksData[i, 2] = Kolesterol;
                    matriksData[i, 3] = LDL;
                    matriksData[i, 4] = Usia;
                    matriksData[i, 5] = JenisKelamin;
                    matriksData[i, 6] = AsamUrat;
                    matriksData[i, 7] = BUN;
                    matriksData[i, 8] = Kreatinin;
                    matriksData[i, 9] = StatusRisiko;
                    i++;

                    f1.tabelDataLatih.Rows.Add(TekananDarah,
KadarGula, Kolesterol, LDL, Usia, JenisKelamin, AsamUrat, BUN,
Kreatinin, StatusRisiko);
                }
            }
```

```
            catch (Exception er)
            {
                Console.WriteLine("Error: " + er);
            }
            finally
            {
                koneksi.Close();
            }
        }
```

***Source code* 4.1** Implemetasi Baca Data Latih

2. Implementasi Proses Pembentukan Matriks Partisi Awal U

Proses pembentukan matriks awal dimulai dengan pembangkitan bilangan random yang diikuti dengan penjumlahan elemen setiap baris sejumlah data. Jumlah bilangan yang dibangkitkan yaitu jumlah data x jumlah *cluster*. Kemudian dilakukan perhitungan nilai elemen matriks dengan membagi nilai μik dengan Qi yang bersesuaian.

Kode program untuk proses pembentukan matriks partisi awal merujuk pada persamaan (2.1) dan (2.2). Implementasi untuk proses pembentukan matriks partisi awal adalah dengan membuat *method* matriksPartisiAwal() seperti yang terlihat pada *Source code* 4.2.

```
public void matriksPartisiAwal()
        {
            matriksAwal = new double[data,jmlcluster];
            Random r=new Random();
            double[] Qi = new double[data];

            for (int i=0; i<data; i++)
            {
                for (int k = 0; k < jmlcluster; k++)
                {
                    matriksAwal[i, k] = r.NextDouble();
                    Qi[i] += matriksAwal[i,k];
                }
            }

            for(int i=0; i<data; i++)
            {
                for (int k=0; k < jmlcluster; k++)
                {
                        matriksAwal[i,k]=matriksAwal[i,k]/Qi[i];
                }
            }
        }
```

***Source code* 4.2** Implementasi Pembentukan Matriks Partisi Awal

3. Implementasi Proses Perhitungan Pusat *Cluster*

Perhitungan pusat *cluster* dilakukan sesuai dengan persamaan (2.3) sehingga menghasilkan pusat *cluster* sejumlah *cluster* x jumlah atribut. Struktur penulisan kode program untuk implementasi penentuan pusat *cluster* ditunjukkan pada *Source code* 4.3.

```
Public double[,] CariPusatCluster(double[,] matriksU, double[,]
matriksData)
        {
            matriksV = new double[jmlcluster,atribut];

            for (int k = 0; k < jmlcluster; k++)
            {
                for (int j = 0; j < atribut; j++)
                {
                    double pembilang = 0, penyebut = 0;
                    for (int i = 0; i < data; i++)
                    {
                        pembilang += ((Math.Pow(matriksAwal[i,
k], w)) * matriksData[i, j]);
                        penyebut += (Math.Pow(matriksAwal[i, k],
w));
                    }
                    if (penyebut == 0.0) matriksV[k, j] = 0;
                    else matriksV[k, j] = pembilang/penyebut;
                }
            }
            return matriksV;
        }
```

***Source code* 4.3** Implementasi Penentuan Pusat *Cluster*

4. Implementasi Proses Perhitungan Fungsi Objektif

Proses perhitungan fungsi objektif akan menghasilkan *output* berupa nilai fungsi objektif iterasi ke-t (Pt). Perhitungan fungsi objektif dilakukan dengan menggunakan persamaan (2.4). Kode program implementasi perhitungan nilai fungsi objektif pada metode *clustering* FCM ditunjukkan oleh kode program pada *Source code* 4.4.

```
public double HitungFungsiObyektif(double[,] matriksAwal,
double[,] matriksV, double[,] matriksX)
        {
            double sigma1=0, sigma2=0, sigma3=0;

            for (int i = 0; i < data; i++)
            {
                for (int k = 0; k < jmlcluster; k++)
                {
                    for (int j = 0; j < atribut; j++)
                    {
                        sigma3 = sigma3 +
```

```
Math.Pow((matriksData[i,j] - matriksV[k,j]), 2);
                    sigma3 = sigma3 *
Math.Pow((matriksAwal[i,k]), w);
                }
                sigma2 = sigma2 + sigma3;
                sigma3 = 0;
            }
            sigma1 = sigma1 + sigma2;
            sigma2 = 0;
        }
        return sigma1;
    }
```

***Source code* 4.4** Implementasi Perhitungan Fungsi Objektif

5. Implementasi Perhitungan Perubahan Matriks Partisi U

Perhitungan perubahan matriks partisi dilakukan untuk memperbarui nilai matriks awal berdasarkan pusat *cluster* yang terbentuk sesuai dengan persamaan (2.5). Implementasi perhitungan perubahan matriks partisi ditunjukkan pada *Source code* 4.5.

```
public void perubahanMatriksPartisi(double[,] matriksAwal,
double[,] matriksData, double[,] matriksV)
        {
            double a=0, b=0, sigma3=0;
            for(int i =0; i<data; i++)
            {
                for(int k=0; k<jml_cluster; k++)
                {
                    for(int j = 0; j < atribut; j++)
                      a=a+Math.Pow((matriksData[i,j] -
matriksV[k,j]),2);
                        a=Math.Pow(a,(-1/(w-1)));

                    for(int l=0; l<jml_cluster; l++)
                    {
                        for(int j=0; j<atribut; j++)

sigma3=sigma3+Math.Pow((matriksData[i,j] - matriksV[l,j]),2);

                        b=b+Math.Pow(sigma3,(-1/(w-
1)));

                        sigma3=0;
                    }
                matriksAwal[i,k]=a/b;
                    a=0;b=0;
                }
            }
        }
```

***Source code* 4.5** Implementasi Perhitungan perubahan Matriks Partisi

6. Implementasi Pengecekan Kondisi Berhenti

Proses pengecekan kondisi berhenti dilakukan dengan cara menghitung selisih Pt dengan nilai Pt-1 kemudian dibandingkan dengan nilai *error* minimum (ξ) yang telah ditentukan sebelumnya. Jika |Pt – Pt-1| < ξ, maka perulangan pada proses *clustering* dihentikan. Jika tidak, maka pengecekan kondisi berhenti dapat mengacu pada jumlah iterasi yang telah dilakukan. Apabila jumlah iterasi telah memenuhi nilai maksimum iterasi yang telah ditentukan sebelumnya, makaproses *clustering* dapat dihentikan. Penerapan untuk proses pengecekan kondisi berhenti adalah dengan membuat *method* isBerhenti(double Pt, double PtMin1, int iterasiKe). Implementasi pengecekan kondisi berhenti ditunjukkan pada *Source code* 4.6.

```
public Boolean isBerhenti(double Pt, double PtMin1, int
iterasiKe)
        {
                if (Math.Abs (Pt-PtMin1) < errormin || iterasiKe
> maksIterasi)
                        return true;
                else
                        return false;
          }
```

***Source code* 4.6** Implementasi Pengecekan Kondisi Berhenti

7. Implementasi Pengelompokan Data

Proses pengelompokan data melibatkan *input* berupa data latih dan derajat keanggotaan. Proses ini menghasilkan *output* berupa kelompok data yang menyatakan *cluster* dari setiap data. Pada implementasinya, proses ini diterapkan kedalam sebuah *method* tentukanIdx*Cluster*(double[,] matriksU). Implementasi pengelompokan data dapat dilihat pada *Source code* 4.7.

```
public int[] tentukanIdxCluster(double[,] matriksU)
        {
            int[] idxCluster = new int[data];

            for (int i = 0; i < data; i++)
            {
                double max = 0;
                int idxMax = 0;

                for (int k=0; k<jmlcluster; k++)
                    if (matriksU[i,k] > max)
                    {
                        max = matriksU[i,k];
                        idxMax = k;
                    }
```

```
                idxCluster[i] = idxMax;
            }
            return idxCluster;
        }
```

***Source code* 4.7** Implementasi Pengelompokan Data

4.2.2. Implementasi Proses Perhitungan Varian

Kelas VARIAN.cs merupakan kelas untuk perhitungan nilai varian *cluster*. Perhitungan varian terbagi menjadi 3 jenis, yaitu proses perhitungan varian tiap *cluster*, proses perhitungan *varian within cluster*, dan proses perhitungan *varian between cluster*. Proses dari perhitungan varian adalah untuk mendapatkan nilai batasan varian dari hasil *cluster* yang terbentuk. Lewat nilai batasan varian inilah nantinya bisa dijadikan acuan untuk memilih hasil *cluster* mana yang efektif guna dijadikan bahan bagi proses ekstraksi aturan *fuzzy*. Penjelasan *method* pembentuk kelas VARIAN.cs dijelaskan pada Tabel (4.3).

Tabel 4.3 Method-method Pembentuk Kelas VARIAN.cs

Nama *method*	Deskripsi
jumlahDataTiap*Cluster*(int index*Cluster*)	*Method* yang digunakan untuk menghitung jumlah data pada tiap *cluster* yang terbentuk.
rataRataAtribut(int index*Cluster*, int indexAtribut)	Sebuah *method* yang digunakan menghitung nilai rata – rata pada tiap atribut data pada suatu *cluster* yang terbentuk.
standarDeviasi()	*Method* yang digunakan untuk menghitung nilai standar deviasi (sigma) dari data
varianTiapAtribut(int x, int y)	*Method* yang digunakan untuk mencari nilai varian tiap atribut pada suatu *cluster*.
varianTiap*Cluster*(int x)	*Method* yang digunakan untuk proses perhitungan varian tiap *cluster*.
varianWithin*Cluster*()	Sebuah *method* yang digunakan untuk proses perhitungan *variance within cluster*.
varianAntarAtribut(int x, int y)	Sebuah *method* yang digunakan untuk menghitung nilai varian antar atribut pada suatu *cluster*.
varianAntar*Cluster*(int x)	*Method* untuk menghitung nilai varian antar *cluster*.
varianBetween*Cluster*()	Sebuah *method* yang digunakan

	untuk menghitung nilai *variance between cluster*.
batasanVarian()	*Method* yang diperuntukan untuk mengitung nilai batasan varian.

1. Implementasi Proses Perhitungan Varian Tiap *Cluster*

Proses perhitungan varian tiap *cluster* diterapkan dengan membuat *method* varianTiap*Cluster*(), dimana dalam *method* tersebut perlu untuk memanggil *method* tambahan lainnya untuk membantu proses perhitungan, diantaranya adalah method jumlahDataTiap*Cluster*(), rataRataAtribut(), standarDeviasi() dan varianTiapAtribut(). Alur proses untuk *method* varianTiap*Cluster*() mengacu pada Persamaan (2.8). Implementasi proses perhitungan varian tiap *cluster* ditunjukkan pada *Source code* 4.8.

```
public int jumlahDataTiapCluster(int indexCluster)
        {
            int jumlahDataCluster = 0;
            int[]
idxHasilCluster=fcm.tentukanIdxCluster(fcm.matriksAwal);

            for (int i = 0; i < fcm.data; i++)
            {
                if (idxHasilCluster[i] == indexCluster)
                {
                    jumlahDataCluster++;
                }
            }
            return jumlahDataCluster;
        }

public double rataRataAtribut(int indexCluster, int
indexAtribut)
        {
            double rata2atribut = 0, totalData = 0;
            int[] idxHasilCluster =
fcm.tentukanIdxCluster(fcm.matriksAwal);
            int jumlahDataCluster = 0;
            for (int i = 0; i < fcm.data; i++)
            {
                if (idxHasilCluster[i] == indexCluster)
                {
                    totalData = totalData +
fcm.matriksData[i,indexAtribut];
                    jumlahDataCluster++;
                }
            }
            rata2atribut = totalData / jumlahDataCluster;
            return rata2atribut;
        }

public double[,] standarDeviasi()
        {
```

```
            standarDeviasi1 = new double
[fcm.jmlcluster,fcm.atribut];
            int[]
idxHasilCluster=fcm.tentukanIdxCluster(fcm.matriksAwal);

            for(int i=0;i<fcm.jmlcluster;i++)
            {
                for(int j=0;j<fcm.atribut;j++)
                {
                    double temp=0.0;
                    for(int k=0;k<fcm.data;k++)
                    {
                        if(idxHasilCluster[k]==i)
                        {

temp=temp+Math.Pow((fcm.matriksData[k,j]-
ratarataAtribut(i,j)),2);
                        }
                    }
                    if ((jumlahDataTiapCluster(i)-1)==0)
                    {
                        standarDeviasi1[i,j]=0;
                    }
                    else
                    {

standarDeviasi1[i,j]=Math.Sqrt(temp/(jumlahDataTiapCluster(i)-
1));
                    }
                }
            }
            return standarDeviasi1;
        }

private double varianTiapAtribut(int x, int y)
        {
            double varianAtribut = 0, rataAtribut = 0;
            int[] idxHasilCluster =
fcm.tentukanIdxCluster(fcm.matriksAwal);
            rataAtribut = ratarataAtribut(x, y);
            for (int k = 0; k < fcm.data; k++)
            {
                if (idxHasilCluster[k] == x)
                {
                    varianAtribut = varianAtribut +
Math.Pow(fcm.matriksData[k,y] - rataAtribut, 2);
                }
            }
            if (jumlahDataTiapCluster(x) == 1)
            {
                varianAtribut = 0;
            }
            else
            {
                varianAtribut = varianAtribut * 1 /
((jumlahDataTiapCluster(x)) - 1);
            }
            return varianAtribut;
        }
```

```
private double varianTiapCluster(int x)
        {
            double varian = 0, temp = 0;
            int[] idxHasilCluster =
fcm.tentukanIdxCluster(fcm.matriksAwal);
            for (int j = 0; j < fcm.atribut; j++)
            {
                temp = varianTiapAtribut(x, j);
                varian = varian + temp;
            }
            return varian;
        }
```

***Source code* 4.8** Implementasi Perhitungan Varian Tiap *Cluster*

2. Implementasi Proses Perhitungan *Varian Within Cluster*

Proses perhitungan *varian within cluster* nantinya melibatkan hasil dari *method* varianTiap*Cluster*(). Dalam implementasinya, proses perhitungan *varian within cluster* diterapkan dengan membuat *method* varianWithin*Cluster*(). Proses pada *method* ini mengacu pada Persamaan (2.9). Variabel *temp* digunakan untuk menampung jumlah total nilai varian tiap *cluster* yang terbentuk. Implementasi untuk proses perhitungan *varian within cluster* ditunjukkan pada *Source code* 4.9.

```
private double varianWithinCluster()
        {
            double VW = 0, temp = 0;
            for (int i = 0; i < fcm.jmlcluster; i++)
            {
                temp = temp + ((jumlahDataTiapCluster(i) - 1) *
varianTiapCluster(i));
            }
            double temp2 = fcm.data - fcm.jmlcluster;
            double temp3 = 1 / temp2;
            VW = temp3 * temp;
            return VW;
        }
```

***Source code* 4.9** Implementasi Perhitungan *Variant Within Cluster*

3. Implementasi Proses Perhitungan *Varian Between Cluster*

Proses perhitungan varian tiap *cluster* diterapkan dengan membuat *method* varianBetween*Cluster*(), dimana dalam *method* tersebut perlu untuk memanggil *method* tambahan lainnya untuk membantu proses perhitungan, diantaranya adalah method jumlahDataTiap*Cluster*(), rataRataAtribut(), dan varianAntarAtribut(). Implementasi untuk proses perhitungan varian *between cluster* ditunjukkan pada *Source code* 4.10.

```
private double varianAntarAtribut(int x, int y)
        {
            double varianAntarAtribut = 0, rataAtribut = 0;
            int[] idxHasilCluster =
fcm.tentukanIdxCluster(fcm.matriksAwal);
            rataAtribut = ratarataAtribut(x, y);
            for (int k = 0; k < fcm.data; k++)
            {
                if (idxHasilCluster[k] == x)
                {
                    varianAntarAtribut = varianAntarAtribut +
(Math.Pow(fcm.matriksData[k,y] - rataAtribut, 2)) *
jumlahDataTiapCluster(x);
                }
            }
            return varianAntarAtribut;
        }

private double varianAntarCluster(int x)
        {
            double varian = 0, temp = 0;
            int[] idxHasilCluster =
fcm.tentukanIdxCluster(fcm.matriksAwal);
            for (int j = 0; j < fcm.atribut; j++)
            {
                temp = varianAntarAtribut(x, j);
                varian = varian + temp;
            }
            return varian;
        }

private double varianBetweenCluster()
        {
            double VB = 0, temp = 0;
            for (int i = 0; i < fcm.jmlcluster; i++)
            {
                temp = temp + varianAntarCluster(i);
            }
            double temp2 = fcm.jmlcluster - 1;
            double temp3 = 1 / temp2;
            VB = temp3 * temp;
            return VB;
        }
```

***Source code* 4.10** Implementasi Perhitungan *Variant Between Cluster*

4. Implementasi Proses Perhitungan Batasan Varian

Perhitungan nilai batasan varian merupakan proses terakhir pada proses perhitungan varian. Nilai batasan varian didapatkan dari hasil bagi antara nilai *varian within cluster* dengan *varian between cluster* sesuai dengan Persamaan (2.11). Pada implementasinya, proses perhitungan batasan varian diterapkan pada sebuah *method* batasanVarian() yan ditunjukkan pada *Source code* 4.11.

```
public double batasanVarian()
        {
```

```
            double batasanVarian = 0;
            batasanVarian = varianWithin
Cluster() /
varianBetween
Cluster();
            return batasanVarian;
        }
```

***Source code* 4.11** Implementasi Perhitungan Batasan Varian

4.2.3. Implementasi Proses Ekstraksi Aturan *Fuzzy*

Kelas EKSTRAKSIATURANFUZZY.cs merupakan kelas untuk mendapatkan nilai koefisien output dan aturan *fuzzy*. Pada kelas ekstraksiAturanFuzzy() terdapat dua *method* utama, yaitu *method* hitungDerajatKeanggotaan() dan perhitunganKoefisienOutput().

1. Implementasi Perhitungan Nilai Derajat Keanggotaan

Proses perhitungan nilai derajat keanggotaan untuk tiap data terhadap pusat *cluster* dan sigma dengan menggunakan fungsi *gauss* sama seperti pada Persamaan (2.21). Pada implementasinya, proses ini diterapkan ke dalam *method* hitungDerajatKeanggotaan(). Implementasi Perhitungan Nilai Derajat Keanggotaan dalam *method* hitungDerajatKeanggotaan()ditunjukkan pada *Source code* 4.12.

```
public double[,] hitungDerajatKeanggotaan()
        {
            tabelMyu=new double[fcm.data,fcm.jml
cluster];
            for(int i=0;i<fcm.data;i++)
            {
                for(int k=0;k<fcm.jml
cluster;k++)
                {
                    double sigma=1;
                    for(int j=0;j<fcm.atribut;j++)
                    {
                        if(sd.standarDeviasi1[k,j]!=0)
                        {

sigma=sigma+((Math.Pow((fcm.matriksData[i,j]-
fcm.v[k,j]),2))/(2*(Math.Pow(sd.standarDeviasi1[k,j],2))));
                        }
                        else if(sd.standarDeviasi1[k,j]==0)
                        {
                            sigma=sigma+0.0;
                        }
                    }
                    tabelMyu[i,k]=Math.Exp((-1)*sigma);
                }
            }
            return tabelMyu;
        }
```

***Source code* 4.12** Implementasi Perhitungan Nilai Derajat Keanggotaan

2. Implementasi Perhitungan Koefisien *Output*

Implementasi proses perhitungan koefisien *output* diterapkan pada *method* perhitunganKoefisienOutput(). Proses perhitungan koefisien *output* memerlukan operasi matriks yang cukup kompleks, diantaranya seperti perkalian matriks, *transpose*, dan *inverse*. Implementasi perhitungan koefisien *ouput* pada *method* perhitunganKoefisienOutput()ditunjukkan pada *Source code* 4.13.

```
private double[,] hitungMatriksD()
        {
            double[,] d= new
double[fcm.data,fcm.atribut*fcm.jmlcluster];
            for(int i=0;i<fcm.data;i++)
            {
                int k=0,l=0,cekAtribut=1;
                for(int j=0;j<fcm.atribut*fcm.jmlcluster;j++)
                {
                    if(cekAtribut==fcm.atribut)
                    {
                        d[i,j]=tabelMyu[i,l];
                        k=0;
                        l++;
                        cekAtribut=0;
                    }
                    else
                    {
                    d[i,j]=fcm.matriksData[i,k]*tabelMyu[i,l];
                    k++;
                    }
                cekAtribut++;
                }
            }
            return d;
        }

private double[,] hitungMatriksU()
        {
            double [,] u = new
double[fcm.data,fcm.atribut*fcm.jmlcluster];
            double [] totalMiu = new double[fcm.data];
            double [,] d = hitungMatriksD();
            for(int i=0;i<fcm.data;i++)
            {
                totalMiu[i] = 0;
                for(int k=0;k<fcm.jmlcluster;k++)
                {
                    totalMiu[i] = totalMiu[i]+tabelMyu[i,k];
                }
                for(int j=0;j<fcm.atribut*fcm.jmlcluster;j++)
                {
                    u[i,j] = d[i,j]/totalMiu[i];
                }
            }
```

```
            return u;
        }

        private double[,] hitungMatriksUT()
        {
            double[,] u = hitungMatriksU();
            double[,] uT = u.Transpose();

            int x = u.GetLength(0);
            int y = u.GetLength(1);

            int x1 = uT.GetLength(0);
            int y1 = uT.GetLength(1);

            double[,] uTranspose = new double[x1,y1];

            for(int i=0; i<x1; i++)
            {
                for(int j=0; j<y1; j++)
                {
                    uTranspose[i,j]=uT[i,j];
                }
            }
            return uTranspose;
        }

private double[,] operasiMatriksUTxU()
        {
            double[,] uT = hitungMatriksUT();
            double[,] u = hitungMatriksU();
            double[,] uTU = uT.Multiply(u);

            int x1 = uTU.GetLength(0);
            int y1 = uTU.GetLength(1);

            double[,] uTransposeU = new double[x1,y1];

            for(int i=0; i<x1; i++)
            {
               for(int j=0; j<y1; j++)
               {
                   uTransposeU[i,j]=uTU[i,j];
               }
            }
            return uTransposeU;
        }

private double [,] operasiInverseU()
        {
            double[,] uTUI = operasiMatriksUTxU();
            double[,] uTUInverse = uTUI.PseudoInverse();

            int x = uTUI.GetLength(0);
            int y = uTUI.GetLength(1);

            int x1 = uTUInverse.GetLength(0);
            int y1 = uTUInverse.GetLength(1);

            double[,] inverseU = new double[x1,y1];
```

```
            for(int i=0; i<x1; i++)
            {
                for(int j=0; j<y1; j++)
                {
                    inverseU[i,j]=uTUInverse[i,j];
                }
            }
            return inverseU;
        }

public double [,] perhitunganKoefisienOutput()
        {
            double[,] u = hitungMatriksU();
            double[,] uT = u.Transpose();

            int x = u.GetLength(0);
            int y = u.GetLength(1);

            int x1 = uT.GetLength(0);
            int y1 = uT.GetLength(1);

            double[,] uTranspose = new double[x1, y1];
            for (int i = 0; i < x1; i++)
            {
                for (int j = 0; j < y1; j++)
                {
                    uTranspose[i, j] = uT[i, j];
                }
            }

            double[,] uTU = uT.Multiply(u);

            int x2 = uTU.GetLength(0);
            int y2 = uTU.GetLength(1);

            double[,] uTransposeU = new double[x2, y2];
            for (int i = 0; i < x2; i++)
            {
                for (int j = 0; j < y2; j++)
                {
                    uTransposeU[i, j] = uTU[i, j];
                }
            }

            double[,] ui = uTransposeU;
            double[,] uTUInverse = ui.PseudoInverse();

            int x3 = uTUInverse.GetLength(0);
            int y3 = uTUInverse.GetLength(1);

            double[,] inverseU = new double[x3, y3];
            for (int i = 0; i < x3; i++)
            {
                for (int j = 0; j < y3; j++)
                {
                    inverseU[i, j] = uTUInverse[i, j];
                }
            }
```

```
            double[,] inverseUxUT = uTUInverse.Multiply(uT);
            double[,] nk = new double[fcm.data, 1];
            for (int i = 0; i < fcm.data; i++)
            {
                nk[i, 0] = fcm.matriksData[i, fcm.atribut - 1];
            }

            double[,] targetOutput = nk;
            double[,] ko = inverseUxUT.Multiply(targetOutput);
            double[,] koefisienOutput = new
double[fcm.jmlcluster, fcm.atribut];
            int k = 0;

            for (int i = 0; i < fcm.jmlcluster; i++)
            {

                if (i == 0)
                {
                    for (int j = 0; j < fcm.atribut; j++)
                    {
                        koefisienOutput[i, j] = ko[j, 0];
                    }
                }

                else
                {
                    for (int j = 0; j < fcm.atribut; j++)
                    {
                        k = (i * fcm.atribut) + j;
                        koefisienOutput[i, j] = ko[k, 0];
                    }
                }
            }
            return koefisienOutput;
        }
```

***Source code* 4.13** Implementasi Perhitungan Koefisien Output

4.2.4. Implementasi Proses Penentuan *Range*

Sebelum dilakukan pengujian terhadap data uji, terlebih dahulu dilakukan proses penentuan *range* dengan menggunakan data latih untuk mendapatkan nilai *range* dari masing-masing status risiko. Implementasi proses penentuan *range* diterapkan ke dalam kelas PENGUJIANRANGE.cs, dimana di dalamnya terdapat 5 buah *method* diantaranya adalah derajatKeanggotaan(), alfaPredikat(), nilaiZ(), defuzzy(), dan tentukanRange(). *Method* derajatKeanggotaan(), alfaPredikat(), nilaiZ(), defuzzy(), nantinya juga terdapat pada kelas FISSUGENO.cs, sehingga pada proses ini hanya akan ditunjukkan implementasi penentuan *range* dalam *method* tentukanRange().Proses awal penetuan *range* ini yaitu menghitung jumlah total data pada masing-masing *output* atau status risiko untuk menentukan panjang

array yang akan digunakan untuk menampung nilai Z dari masing-masing status risiko. Dari masing-masing *array* tersebut akan dicari nilai minimun dan maksimumnya, sehingga nilai tersebutlah yang nantinya digunakan sebagai *range* dari masing-masing status risiko. Implementasi program untuk proses penentuan *range* ditunjukkan pada *Source code* 4.14.

```
public void tentukanRange()
        {
            double[,] nilaiZ = new double[fcm.data, 2];
            int jmlRendah = 0;
            int jmlSedang = 0;
            int jmlTinggi = 0;
            double[] z = pengujian();
            double output = 0;

            for (int i = 0; i < fcm.data; i++)
            {
                output = fcm.matriksData[i, 9];

                if (output == 0.0)
                {
                    jmlRendah++;
                }
                else if (output == 50.0)
                {
                    jmlSedang++;
                }
                else
                {
                    jmlTinggi++;
                }
            }

            sRendah = new double[jmlRendah];
            sSedang = new double[jmlSedang];
            sTinggi = new double[jmlTinggi];

            int R = 0;
            int S = 0;
            int T = 0;

            for (int i = 0; i < fcm.data; i++)
            {
                output = fcm.matriksData[i, 9];
                double zi = z[i];

                if (output == 0.0)
                {
                    sRendah[R] = zi;
                    R++;
                }
                else if (output == 50.0)
                {
                    sSedang[S] = zi;
                    S++;
                }
```

```
                else
                {
                    sTinggi[T] = zi;
                    T++;
                }
            }
            Rmin = sRendah.Min();
            Rmax = sRendah.Max();
            Smin = sSedang.Min();
            Smax = sSedang.Max();
            Tmin = sTinggi.Min();
            Tmax = sTinggi.Max();
        }
```

***Source code* 4.14** Implementasi Penentuan *Range*

4.2.5. Implementasi Proses *Fuzzy Inference System* Sugeno

Kelas FISSUGENO.cs merupakan kelas untuk pengujian akurasi terhadap data uji menggunakan *fuzzy inference system* model sugeno orde-satu dengan aturan yang telah dibangkitkan pada proses *clustering* dan ekstraksi aturan *fuzzy*. Implementasi pada proses ini diterapkan ke dalam kelas FISSUGENO.cs, dimana di dalamnya terdapat 6 buah *method* diantaranya adalah BacaDataUji(), getCenter(), hitungMiu(), hitungAlphaPredikat(), hitungZ(), dan defuzzy().

1. Implementasi Proses Pembacaan Data Uji

Langkah pertama dalam proses pengujian adalah memasukkan data uji. Dalam implementasinya, data uji disimpan ke dalam sebuah database yang nantinya dipanggil menggunakan *method* BacaDataUji(). Di dalam *method* BacaDataUji() terdapat *method* FISSUGENO() yang fungsinya sebagai konektor antara aplikasi dengan *database*. Implementasi proses baca data uji ditunjukkan pada *Source code* 4.15.

```
public void datauji()
        {
            try
            {
                koneksi.Open();

                string query = "select * from data_uji";
                MySqlCommand comm = new MySqlCommand(query,
koneksi);
                comm.ExecuteNonQuery();
                MySqlDataAdapter adapter = new
MySqlDataAdapter(comm);
                MySqlDataReader read = comm.ExecuteReader();
                matriksDataUji = new double[jmlDataUji, 12];
                int i = 0;
                while (read.Read())
                {
```

```
                    int TekananDarah =
Convert.ToInt16(read["tekanan_darah"]);
                    int KadarGula =
Convert.ToInt16(read["kadar_gula"]);
                    int Kolesterol =
Convert.ToInt16(read["kolesterol_total"]);
                    int LDL = Convert.ToInt16(read["ldl"]);
                    int Usia = Convert.ToInt16(read["usia"]);
                    int JenisKelamin =
Convert.ToInt16(read["jenis_kelamin"]);
                    double AsamUrat =
Convert.ToDouble(read["asam_urat"]);
                    double BUN = Convert.ToDouble(read["bun"]);
                    double Kreatinin =
Convert.ToDouble(read["kreatinin"]);
                    int StatusRisiko =
Convert.ToInt16(read["status"]);

                    matriksDataUji[i, 0] = TekananDarah;
                    matriksDataUji[i, 1] = KadarGula;
                    matriksDataUji[i, 2] = Kolesterol;
                    matriksDataUji[i, 3] = LDL;
                    matriksDataUji[i, 4] = Usia;
                    matriksDataUji[i, 5] = JenisKelamin;
                    matriksDataUji[i, 6] = AsamUrat;
                    matriksDataUji[i, 7] = BUN;
                    matriksDataUji[i, 8] = Kreatinin;
                    matriksDataUji[i, 9] = StatusRisiko;
                    i++;

                   fp.TabelLihatDataUji.Rows.Add(TekananDarah,
KadarGula, Kolesterol, LDL, Usia, JenisKelamin, AsamUrat, BUN,
Kreatinin, StatusRisiko);
                }
            }
            catch (Exception er)
            {
                Console.WriteLine("Error: " + er);
            }
            finally
            {
                koneksi.Close();
            }
        }
```

***Source code* 4.15** Implementasi Baca Data Uji

2. Implementasi Inisialisasi Pusat *Cluster*

Proses inisialisasi pusat *cluster* merupakan proses untuk mendapatkan nilai pusat *cluster* yang sebelumnya telah dihitung pada proses *clustering*. Proses ini diimplementasikan dengan membuat *method* getCenter(). Implementasi inisialisasi pusat *cluster* ditunjukkan pada *Source code* 4.16.

```
public double[,] getCenter()
        {
```

```
            double [,] pusatCluster=new
double[fcm.jmlcluster,fcm.atribut];
            for(int i=0;i<fcm.jmlcluster;i++)
            {
                for(int j=0;j<fcm.atribut;j++)
                {
                    pusatCluster[i,j]=fcm.v[i,j];
                }
            }
            return pusatCluster;
        }
```

***Source code* 4.16** Implementasi Inisialisasi Pusat *Cluster*

3. Implementasi Proses Perhitungan Derajat Keanggotaan

Proses perhitungan derajat keanggotaan dilakukan terhadap data uji yang memiliki 10 parameter. Nantinya 10 parameter tersebut dicari derajat keanggotaannya menggunakan fungsi *gauss*. Implementasi untuk proses perhitungan derajat keanggotaan diterapkan pada *method* hitungMiu() yang ditunjukkan pada *Source code* 4.17.

```
public double[,] hitungMiu(double [,] dataUji, int x)
        {
            double[,] miuDataUji = new
double[fcm.jmlcluster,fcm.atribut-1];
            double [,] center = getCenter();
            for(int i=0;i<fcm.jmlcluster;i++)
            {
                double tempSigma=0;
                for(int j=0;j<fcm.atribut-1;j++)
                {
                    tempSigma=((Math.Pow((dataUji[x,j]-
center[i,j]),2))/(2*(Math.Pow(sd.standarDeviasi1[i,j],2))));
                    miuDataUji[i,j]=Math.Exp((-1)*tempSigma);
                }
            }
            return miuDataUji;
        }
```

***Source code* 4.17** Implementasi Perhitungan Derajat Keanggotaan

4. Implementasi Perhitungan *Alpha Predikat*

Implementasi proses perhitungan *alpha predikat* diterapkan pada sebuah *method* hitungAlphaPredikat(). Implementasi proses perhitungan *alpha predikat* ditunjukkan pada *Source code* 4.18.

```
public double[] hitungAlphaPredikat(double[,] dataUji, int x)
        {
            double[] alphaPredikat = new double[fcm.jmlcluster];
            double[,] miuDataUji = hitungMiu(dataUji, x);
            for (int i = 0; i < fcm.jmlcluster; i++)
```

```
        {
            double temp = 1;
            for (int j = 0; j < fcm.atribut - 1; j++)
            {
                temp = temp * miuDataUji[i,j];
            }
            alphaPredikat[i] = temp;
        }
        return alphaPredikat;
    }
```

***Source code* 4.18** Implementasi Perhitungan Alpha Predikat

5. Implementasi Perhitungan Nilai *z*

Implementasi proses perhitungan nilai *z* untuk tiap aturan diterapkan pada sebuah *method* hitungZ() yang ditunjukkan pada *Source code* 4.19.

```
public double[] hitungZ(double[,] dataUji, int x)
    {
        double[] z = new double[fcm.jmlcluster];
        double[,] ko = aturan.perhitunganKoefisienOutput();
        for (int i = 0; i < fcm.jmlcluster; i++)
        {
            double temp = 0;
            for (int j = 0; j < fcm.atribut - 1; j++)
            {
                double temp2 = 0;
                temp2 = dataUji[x,j] * ko[i,j];
                temp = temp + temp2;
            }
            z[i] = temp + ko[i,fcm.atribut - 1];
        }
        return z;
    }
```

***Source code* 4.19** Implementasi Perhitungan Nilai z

6. Implementasi Proses *Defuzzy*

Proses *defuzzy* merupakan proses terakhir pada fase pengujian. Proses *defuzzy* merupakan proses merubah nilai *fuzzy* menjadi bentuk *crisp*. Pada implementasinya, proses *defuzzy* diterapkan ke dalam *method* defuzzy(). Untuk memudahkan alur penulisan kode program, maka diperlukan variabel *temp* untuk menampung jumlah perkalian *aplha predikat* dengan nilai *z*, sedangkan variabel *temp2* digunakan untuk menampung nilai total dari hasil penjumlahan setiap *alpha predikat* pada masing-masing aturan. Implementasi proses *defuzzy* ditunjukkan pada *Source code* 4.20.

```
public double defuzzy(double[,] dataUji, int x)
    {
        double zAkhir = 0;
        double[] a = hitungAlphaPredikat(dataUji, x);
```

```
            double[] b = hitungZ(dataUji, x);
            double temp = 0, temp2 = 0;
            for (int i = 0; i < fcm.jml cluster; i++)
            {
                temp = temp + (a[i] * b[i]);
                temp2 = temp2 + a[i];
            }
            zAkhir = temp / temp2;
            return zAkhir;
        }
```

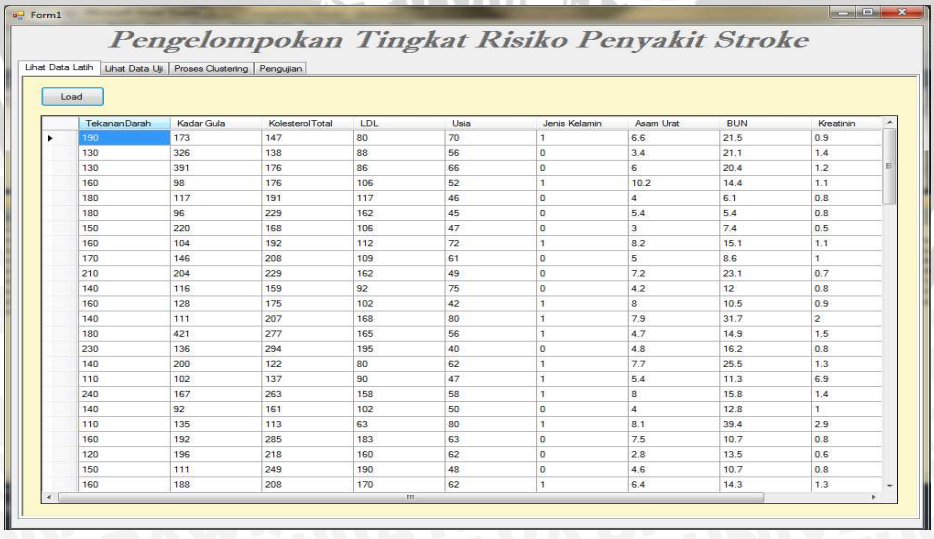
***Source code* 4.20** Implementasi Proses *Defuzzy*

4.3. Implementasi Antarmuka

Implementasi antarmuka diterapkan berdasarkan pada bab perancangan. Aplikasi sistem pembangkitan aturan secara otomatis untuk deteksi dini risiko penyakit stroke ini terdapat 4 menu, yaitu lihat data latih, lihat data uji, proses *clustering*, dan pengujian. Bahasa pemrograman yang digunakan untuk membuat aplikasi ini adalah C#. Berikut adalah implementasi antarmuka yang digunakan untuk proses *clustering*, pemilihan jumlah *cluster* ideal, pembangkitan aturan, dan proses inferensi *fuzzy* untuk pengelompokan tingkat risiko penyakit stroke.

4.3.1. Implementasi Antarmuka Lihat Data Latih

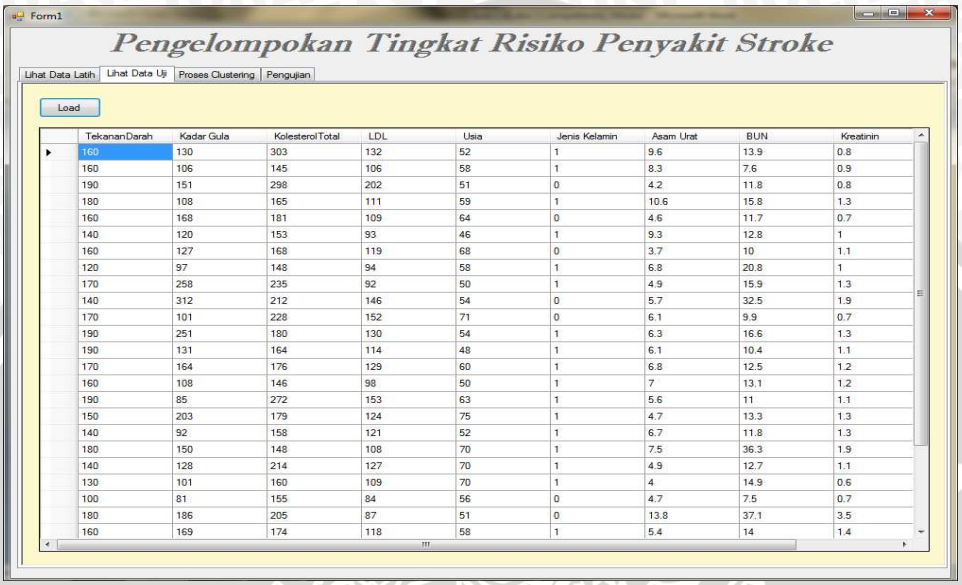
Implementasi antarmuka lihat data latih merupakan antarmuka yang akan tampil pertama kali saat program dijalankan. Antarmuka ini digunakan untuk melihat data latih yang akan digunakan pada proses pelatihan. Implementasi antarmuka data ditunjukkan pada Gambar 4.2.



Gambar 4.2 Implementasi Antarmuka Lihat Data Latih

4.3.2. Implementasi Antarmuka Data Uji

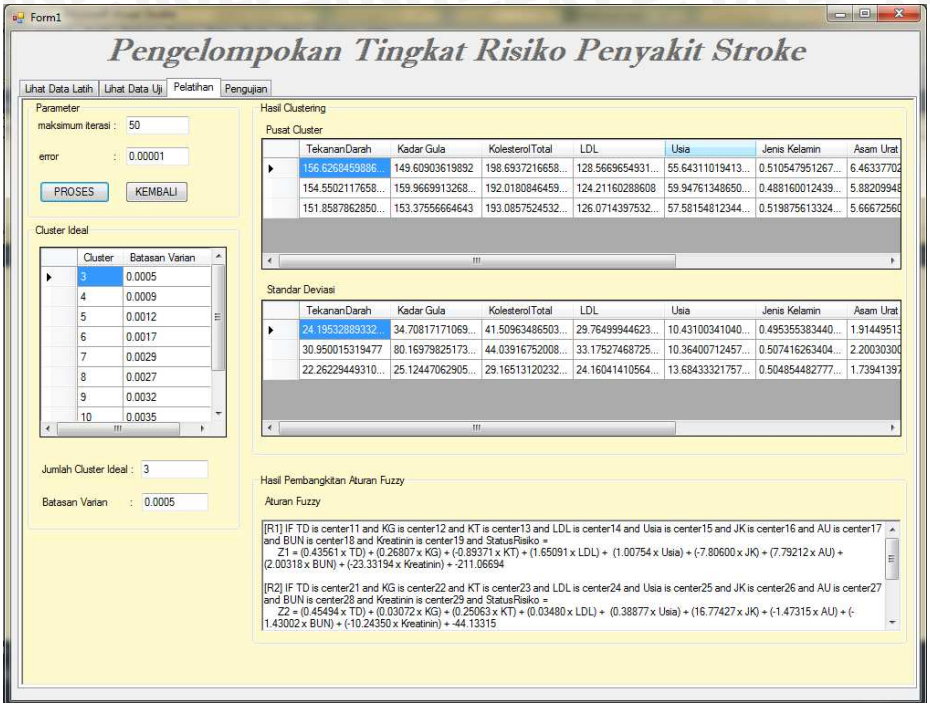
Implementasi antarmuka data uji digunakan untuk melihat data uji yang digunakan pada proses pengujian. Implementasi antarmuka data uji ditunjukkan pada Gambar 4.3.



Gambar 4.3 Implementasi Antarmuka Lihat Data Uji

4.3.3. Implementasi Antarmuka Proses Pelatihan

Antarmuka Proses pelatihan merupakan antarmuka yang digunakan untuk melakukan proses pelatihan menggunakan *fuzzy c-means clustering* terhadap data latih. Ada beberapa parameter yang dapat diubah, diantaranya adalah nilai error terkecil dan maksimum iterasi. Tombol “PROSES” merupakan tombol yang dapat digunakan untuk memproses data latih berdasarkan parameter yang talah ditentukan sebelumnya. Sedangkan tombol “KEMBALI” merupakan tombol untuk mengosongkan seluruh hasil dari proses *clustering* sehingga *user* dapat memulai proses *clustering* kembali. Implementasi antarmuka Proses *Clustering* dapat dilihat pada Gambar 4.4.



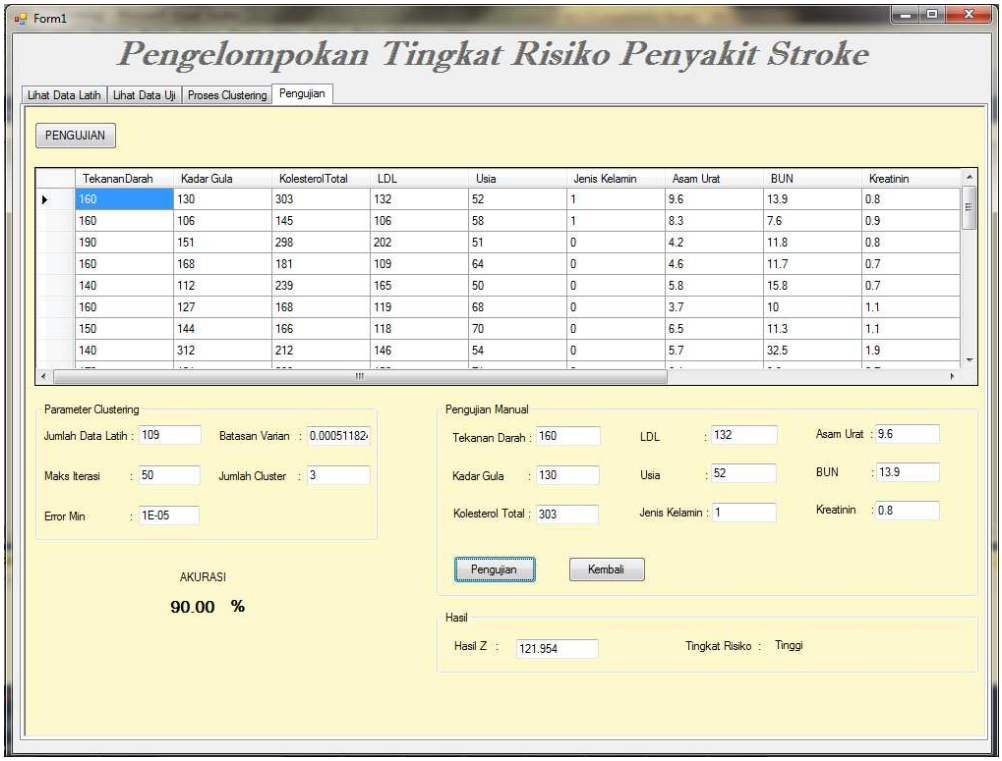
Gambar 4.4 Implementasi Antarmuka Pelatihan

Pada antarmuka ini disediakan informasi hasil dari proses setiap pelatihan yang dilakukan sebanyak 10 kali dengan jumlah *cluster* yang berbeda-beda. Hasil proses tersebut ditampilkan pada tabel *cluster* ideal, dimana terdapat nilai jumlah *cluster* beserta nilai batasan variannya. Sistem juga akan memilih secara otomatis hasil *cluster* mana yang akan dipakai pada proses ekstraksi aturan *fuzzy* berdasarkan analisa varian. Dari proses analisa varian, diberikan informasi terkait jumlah *cluster* terpilih, batasan varian terpilih, pusat *cluster* terpilih, standar deviasi *cluster* terpilih, dan aturan *fuzzy* beserta koefisien *output* yang terbentuk.

4.3.4. Implementasi Antarmuka Pengujian

Implementasi pengujian data uji merupakan antarmuka untuk menampilkan hasil deteksi dini risiko penyakit stroke berdasarkan ekstraksi aturan *fuzzy* pada proses pelatihan sebelumnya. Dalam form ini juga ditampilkan parameter-parameter yang digunakan pada proses pelatihan dengan FCM *clustering*. Pada implementasi pengujian ini juga ditampilkan akurasi hasil perhitungan data uji dengan menggunakan aturan *fuzzy* yang telah terbentuk dengan algoritma sugeno orde-satu. Pada antarmuka pengujian ini juga dapat

dilakukan perhitungan untuk deteksi dini risiko stroke pada data yang baru. Tampilan antarmuka untuk implementasi pengujian ditunjukkan pada Gambar 4.5.



Gambar 4.5 Implementasi Antarmuka Pengujian

BAB V
PENGUJIAN DAN ANALISIS

Pada bab ini akan dibahas tentang pengujian dan analisis implementasi algortima *fuzzy c-means clustering* untuk pembangkitan aturan fuzzy pada proses deteksi dini risiko penyakit stroke. Pengujian dilakukan sebanyak dua kali, yaitu pengujian jumlah cluster dan pengujian akurasi.

5.1. Pengujian Jumlah *Cluster*

Pengujian jumlah *cluster* adalah pengujian yang dilakukan untuk mendapatkan jumlah *cluster* ideal berdasarkan analisa varian. Tujuan dari pengujian jumlah *cluster* ideal adalah untuk menentukan jumlah *cluster* terpilih berdasarkan analisa varian. Dari jumlah *cluster* yang terpilih selanjutnya akan dilakukan pembangkitan aturan *fuzzy*. Aturan *fuzzy* yang terbentuk kemudian akan digunakan pada pengujian akurasi.

5.1.1. Skenario Pengujian Jumlah *Cluster*

Pengujian dilakukan sebanyak 10 kali dengan menggunakan data uji sebanyak 30 data. Pada proses pelatihan, menggunakan data latih sebanyak 109 data. Iterasi maksimum yang digunakan adalah 50 sedangkan eror minimum yang ditetapkan adalah 0,00001. Pelatihan dilakukan mulai dari jumlah *cluster* 3 (jumlah *cluster* minimum) hingga jumlah *cluster* 12. Untuk menentukan jumlah *cluster* yang ideal, maka dilakukan perhitungan varian untuk setiap jumlah *cluster*. Jumlah *cluster* ideal dipilih berdasarkan nilai batasan varian terkecil (nilai V minimum).

5.1.2. Hasil Pengujian Jumlah *Cluster*

Pada pengujian jumlah *cluster* dihasilkan jumlah *cluster* ideal yang terpilih berdasarkan analisa varian pada data latih. Hasil perhitungan nilai batasan varian mulai dari jumlah *cluster* 3 sampai dengan 12 ditunjukkan dalam tabel (5.1).

Tabel 5.1 Hasil Pengujian Pemilihan Jumlah Cluster Ideal

Jmlh *Clstr*	Nilai Batasan Varian Pada Pengujian ke-									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
3	**0.0005**	**0.0004**	**0.0005**	**0.0005**	**0.0005**	**0.0004**	**0.0005**	**0.0005**	**0.0005**	**0.0005**
4	0.0008	0.0007	0.0009	0.0009	0.0008	0.001	0.0009	0.0008	0.0008	0.0008
5	0.0015	0.0015	0.0014	0.001	0.0012	0.0013	0.0012	0.0015	0.0013	0.001
6	0.0018	0.0022	0.0015	0.0019	0.0018	0.0018	0.002	0.0013	0.0017	0.0015
7	0.0031	0.002	0.0024	0.0019	0.0026	0.002	0.0021	0.0022	0.0025	0.0023
8	0.0032	0.0029	0.0033	0.0021	0.0026	0.0023	0.0024	0.0026	0.0026	0.0028
9	0.0028	0.0033	0.0033	0.0033	0.0041	0.0031	0.0033	0.0026	0.0029	0.0032
10	0.0039	0.0035	0.0044	0.0046	0.0045	0.0029	0.0047	0.0048	0.0039	0.0035
11	0.0055	0.0042	0.0053	0.0044	0.0044	0.004	0.0052	0.0049	0.0048	0.0046
12	0.0058	0.005	0.0052	0.0061	0.0051	0.0049	0.0059	0.0057	0.0058	0.0046

Dari uji coba 1 hingga uji coba 10, didapatkan nilai batasan varian yang hampir sama. Nilai batasan varian terkecil, yaitu 0,0005 terletak pada jumlah *cluster* 3, sehingga jumlah *cluster* tersebut terpilih menjadi *cluster* ideal. Hasil dari *cluster* yang terpilih tersebut kemudian akan digunakan untuk proses pembangkitan aturan *fuzzy*. Jumlah aturan *fuzzy* yang terbentuk akan sesuai dengan jumlah *cluster* ideal yang terpilih.

Setelah proses uji coba pemilihan *cluster* ideal berdasarkan nilai batasan varian terkecil dilakukan, kemudian dilakukan kembali proses perhitungan akurasi terhadap jumlah *cluster* 3,4,5,6,7,8,9,10,11,12 sebagai pembanding. Perhitungan akurasi pada jumlah *cluster* tersebut dilakukan untuk mengetahui kesesuaian antar *cluster* ideal yang terpilih dan akurasinya. Perhitungan tersebut masing-masing dilakukan sebanyak 10 kali percobaan. Akurasi pengujian tiap jumlah *cluster* didapatkan berdasarkan rata-rata akurasi dari 10 kali percobaan tiap *cluster*nya. Akurasi hasil pengujian jumlah *cluster* ditunjukkan pada tabel (5.2).

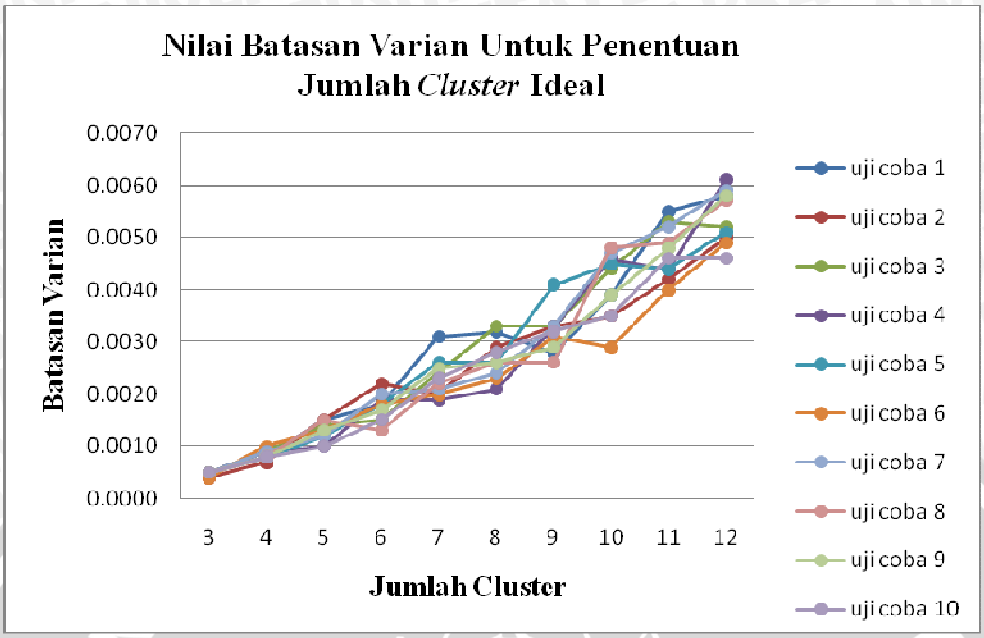
Tabel 5.2 Hasil Pengujian Jumlah *Cluster* terhadap Akurasi

Jmlh *Clstr*	Nilai Akurasi Pada Percobaan ke-										Rata-Rata
	1	2	3	4	5	6	7	8	9	10	
3	90.0%	93.3%	80.0%	90.0%	83.3%	80.0%	83.3%	83.3%	90.0%	90.0%	86.3%
4	70.0%	60.0%	73.3%	73.3%	93.3%	86.7%	76.7%	86.7%	80.0%	83.3%	78.3%
5	70.0%	40.0%	73.3%	80.0%	56.7%	80.0%	73.3%	73.3%	60.0%	66.7%	67.3%
6	50.0%	70.0%	46.7%	66.7%	66.7%	76.7%	43.3%	60.0%	60.0%	50.0%	59.0%
7	40.0%	36.7%	40.0%	33.3%	43.3%	36.7%	30.0%	53.3%	40.0%	43.3%	39.7%
8	46.7%	36.7%	53.3%	46.7%	30.0%	50.0%	16.7%	43.3%	40.0%	26.7%	39.0%
9	43.3%	53.3%	30.0%	23.3%	23.3%	33.3%	26.7%	50.0%	36.7%	50.0%	37.0%
10	36.7%	13.3%	40.0%	23.3%	43.3%	46.7%	20.0%	40.0%	33.3%	33.3%	33.0%
11	40.0%	43.3%	13.3%	30.0%	16.7%	46.7%	30.0%	13.3%	13.3%	43.3%	29.0%
12	16.7%	30.0%	10.0%	16.7%	20.0%	33.3%	10.0%	23.3%	23.3%	36.7%	22.0%

Berdasarkan tabel (5.2), tampak bahwa hasil akurasi dari 100 kali percobaan pada jumlah *cluster* tertentu tidak stabil. Oleh karena itu, dihitung rata-rata akurasi setiap jumlah *cluster* untuk mewakili akurasi setiap jumlah *cluster*. Jumlah *cluster* ideal 3 memiliki rata-rata akurasi sebesar 86,3%. Sedangkan rata-rata akurasi untuk jumlah *cluster* 4, 5, 6, 7, 8, 9, 10, 11 dan 12 secara berturut-turut adalah 78,3%; 67,3%; 59%; 39,7%; 39%; 37%; 33%; 29% dan 22%.

5.1.3. Analisis Hasil Pengujian Jumlah *Cluster*

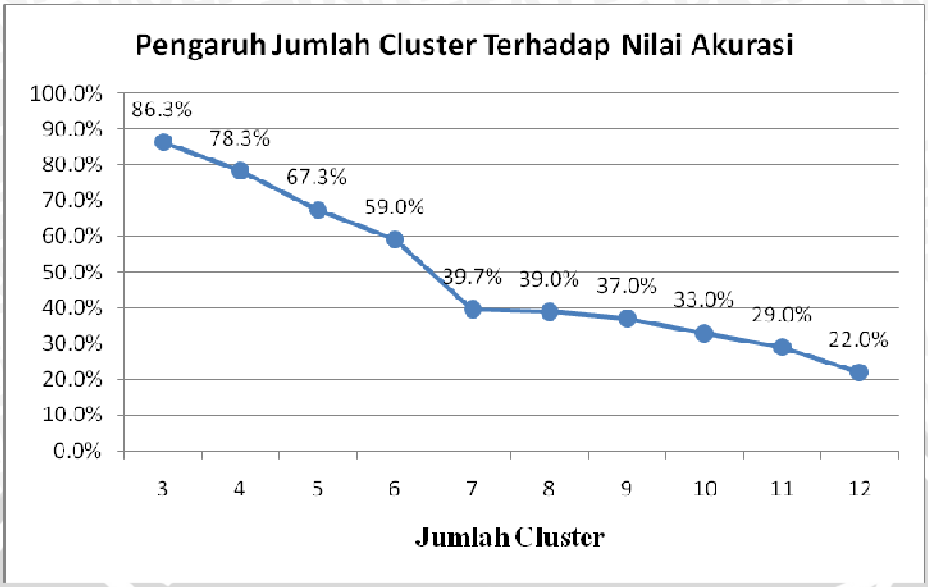
Pada proses pengujian jumlah *cluster* yang telah dilakukan sebanyak 10 kali dengan menggunakan jumlah data latih sebanyak 109 data, maka didapatkan jumlah *cluster* ideal yang terpilih. Grafik pemilihan jumlah *cluster* ideal ditunjukkan pada Gambar (5.1).



Gambar 5.1 Grafik Pemilihan Jumlah Cluster Ideal

Berdasarkan grafik pemilihan jumlah *cluster* ideal pada Gambar (5.1), dapat dilihat bahwa semakin banyak jumlah *cluster* maka nilai batasan varian akan semakin besar. Dari pengujian jumlah *cluster* ideal ini dihasilkan jumlah *cluster* ideal yang sama dari tiap pengujian yaitu 3. Hal tersebut dikarenakan dari 10 kali pengujian yang dilakukan, menunjukkan bahwa nilai batasan varian terkecil berada pada jumlah *cluster* 3 dengan nilai batasan varian sebesar 0,0004 dan 0,0005. Nilai batasan varian yang dihasilkan, dipengaruhi oleh nilai *variance between cluster* dan nilai *variance within cluster* pada tiap *cluster* yang dihasilkan. Sedangkan nilai *variance between cluster* dan nilai *variance within cluster* dipengaruhi oleh keanggotaan titik data terhadap *cluster* yang terbentuk.

Pada proses pengujian pengaruh jumlah *cluster* terhadap akurasi yang telah dilakukan sebanyak 100 kali, didapatkan nilai rata-rata akurasi dari 10 jumlah *cluster* yang berbeda. Grafik pengaruh jumlah *cluster* ideal yang terpilih ditunjukkan pada Gambar (5.2).



Gambar 5.2 Grafik Pengaruh Jumlah Cluster terhadap Nilai Akurasi

Berdasarkan grafik pada Gambar (5.2) menunjukkan bahwa semakin banyak jumlah *cluster* maka nilai akurasi cenderung akan semakin rendah. Pada pengujian ini nilai rata-rata akurasi tertinggi berada pada jumlah *cluster* 3, yaitu sebesar 86,3%. Hal tersebut menunjukkan bahwa metode analisa varian telah sesuai untuk penentuan jumlah *cluster* ideal. Kesesuaian tersebut terlihat dari jumlah *cluster* ideal yang terbentuk berdasarkan analisa varian sama dengan jumlah *cluster* yang mempunyai akurasi tertinggi. Jumlah *cluster* ideal yang telah terpilih akan digunakan untuk pembangkitan aturan *fuzzy*. Aturan *fuzzy* yang terbentuk kemudian akan digunakan untuk pengujian akurasi.

5.2. Pengujian Akurasi

Pengujian akurasi merupakan pengujian akurasi dari hasil aturan *fuzzy* yang terbentuk dari proses pelatihan dan pengujian jumlah *cluster* ideal yang terpilih. Tujuan pengujian akurasi adalah untuk mengetahui nilai akurasi hasil inferensi berdasarkan aturan *fuzzy* yang terbentuk dari proses pelatihan dengan menggunakan jumlah *cluster* ideal yang talah terpilih.

5.2.1. Skenario pengujian Akurasi

Pengujian akurasi dilakukan sebanyak 10 kali pada aturan *fuzzy* yang terbentuk dari proses pelatihan berdasarkan jumlah *cluster* ideal yang terpilih.

Aturan *fuzzy* diujikan pada data uji sebanyak 30 data. Pengujian akurasi dilakukan dengan membandingkan nilai risiko dari pakar dengan nilai risiko dari hasil pengujian data uji. Dari hasil pengujian yang dilakukan sebanyak 10 kali maka akan dihasilkan 10 nilai akurasi yang hasil akurasinya akan dirata-rata untuk mendapatkan akurasi sistem.

5.2.2. Hasil Pengujian Akurasi

Pada pengujian akurasi yang dilakukan sebanyak 10 kali dengan menggunakan data uji sebanyak 30 data, telah dihasilkan 10 jumlah *cluster* ideal dan 10 aturan *fuzzy*. Pada aturan-aturan yang terbentuk, terdapat parameter-parameter *output* dari tiap aturan. Parameter-parameter ini adalah penjumlahan dari tiap perkalian antara nilai koefisien output dengan nilai tiap atribut pada suatu data. Sebagai contoh, pada pengujian pertama, didapatkan nilai koefisien output seperti pada Tabel 5.3 dan Tabel 5.4.

Tabel 5.3 Koefisien *Output* (1)

Koefisien Output					
R	TD	KG	KT	LDL	Usia
R1	0.70742	0.07217	0.31965	0.36673	0.86157
R2	0.40668	0.06961	0.14198	0.18020	0.63604
R3	-1.87617	16.81343	10.99953	-9.56862	-0.71864

Tabel 5.4 Koefisien *Output* (2)

Koefisien Output					
R	JK	AU	BUN	Kreatinin	Status Risiko
R1	-23.52553	4.54403	0.13595	6.27773	-253.96035
R2	14.88980	-1.69911	0.18020	-11.16226	-83.29420
R3	53.02352	276.76403	-7.01919	3.95617	-4811.58855

Detail dari hasil pengujian akurasi dengan menggunakan data uji ditunjukkan pada Tabel (5.5).

Tabel 5.5 Hasil Pengujian Akurasi berdasarkan Aturan Fuzzy dengan Menggunakan Data Uji

Jenis Aturan	**Batasan Varian**	**Jumlah *Cluster***	**Akurasi**
Aturan 1	0.0005	3	93.33%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is			

center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.70742 x TD) + (0.07217 x KG) + (0.31965 x KT) + (0.36673 x LDL) + (0.86157 x Usia) + (-23.52553 x JK) + (4.54403 x AU) + (0.13595 x BUN) + (6.27773 x Kreatinin) + -253.96035 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.40668 x TD) + (0.06961 x KG) + (0.14198 x KT) + (0.18020 x LDL) + (0.63604 x Usia) + (14.88980 x JK) + (-1.69911 x AU) + (0.32394 x BUN) + (-11.16226 x Kreatinin) + -83.29420 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (-1.87617 x TD) + (16.81343 x KG) + (10.99953 x KT) + (-9.56862 x LDL) + (-0.71864 x Usia) + (53.02352 x JK) + (276.76403 x AU) + (-7.01919 x BUN) + (3.95617 x Kreatinin) + -4811.58855			
Aturan 2	0.0005	3	80.00%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (-2.32335 x TD) + (0.42154 x KG) + (1.27488 x KT) + (-2.68279 x LDL) + (3.25198 x Usia) + (-14.12524 x JK) + (25.15191 x AU) + (-11.97905 x BUN) + (35.36532 x Kreatinin) + 291.40067 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.57728 x TD) + (0.07480 x KG) + (0.32130 x KT) + (0.14820 x LDL) + (0.53770 x Usia) + (2.20581 x JK) + (-3.37940 x AU) + (0.07608 x BUN) + (0.02910 x Kreatinin) + -138.27910 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (0.83386 x TD) + (-0.09913 x KG) + (-0.72954 x KT) + (1.37778 x LDL) + (1.75516 x Usia) + (4.32772 x JK) + (9.34061 x AU) + (2.05891 x BUN) + (-25.24429 x Kreatinin) + -260.47072			
Aturan 3	0.0005	3	80.00%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.87019 x TD) + (0.21106 x KG) + (0.12237 x KT) + (0.41964 x LDL) + (0.69157 x Usia) + (-6.30279 x JK) + (2.91648 x AU) + (0.36130 x BUN) + (3.36906 x Kreatinin) + -248.89344 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.26428 x TD) + (0.02676 x KG) + (0.14151 x KT) + (0.17423 x LDL) + (0.59865 x Usia) + (15.92971 x JK) + (-1.39212 x AU) + (0.21031 x BUN) + (-9.94752 x Kreatinin) + -43.87233			

[R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (-2.13588 x TD) + (0.21954 x KG) + (-0.14301 x KT) + (-2.41373 x LDL) + (0.22517 x Usia) + (0.16356 x JK) + (14.98231 x AU) + (-5.06724 x BUN) + (127.90154 x Kreatinin) + 438.25116			
Aturan 4	0.0004	3	86.67%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (-4.05290 x TD) + (-0.29305 x KG) + (-1.56785 x KT) + (-8.09952 x LDL) + (-10.32471 x Usia) + (151.42591 x JK) + (-65.65370 x AU) + (10.89093 x BUN) + (-9.28517 x Kreatinin) + 2694.63014 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.25399 x TD) + (0.02530 x KG) + (0.15092 x KT) + (0.11966 x LDL) + (0.72294 x Usia) + (8.37575 x JK) + (-0.43278 x AU) + (0.20627 x BUN) + (-6.79421 x Kreatinin) + -50.87900 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (0.85353 x TD) + (0.21390 x KG) + (0.28032 x KT) + (0.33301 x LDL) + (0.94920 x Usia) + (-6.23346 x JK) + (0.89012 x AU) + (-0.39965 x BUN) + (4.73634 x Kreatinin) + -257.99386			
Aturan 5	0.0005	3	86.67%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.46993 x TD) + (0.62667 x KG) + (-0.80723 x KT) + (1.15762 x LDL) + (0.58951 x Usia) + (-24.26504 x JK) + (9.43573 x AU) + (1.42654 x BUN) + (-17.39891 x Kreatinin) + -190.53504 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.36915 x TD) + (0.04386 x KG) + (0.19246 x KT) + (0.23774 x LDL) + (0.42039 x Usia) + (11.95790 x JK) + (-2.28703 x AU) + (-1.07597 x BUN) + (-5.23423 x Kreatinin) + -59.02390 Z3 = (1.46436 x TD) + (-0.79293 x KG) + (1.92948 x KT) + (-0.47220 x LDL) + (3.26455 x Usia) + (17.90081 x JK) + (9.25685 x AU) + (-1.92895 x BUN) + (16.78886 x Kreatinin) + -617.99293			
Aturan 6	0.0005	3	90.00%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.99047 x TD) + (0.09809 x KG) + (0.29017 x KT) + (0.20617 x LDL) + (1.29242 x Usia) + (0.78846 x JK) + (3.54920 x AU) + (-0.55433 x BUN) +			

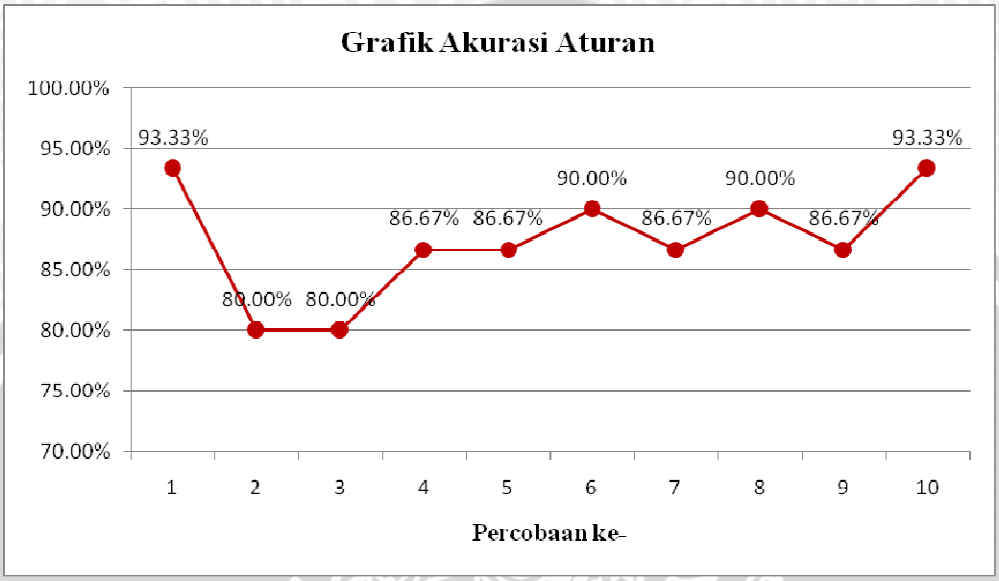
(6.27387 x Kreatinin) + -298.27989 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.62597 x TD) + (-0.73769 x KG) + (4.52925 x KT) + (-0.56065 x LDL) + (-4.95775 x Usia) + (0.52806 x JK) + (-56.75610 x AU) + (10.03526 x BUN) + (-148.80820 x Kreatinin) + -117.60681 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (0.26846 x TD) + (0.03651 x KG) + (0.13021 x KT) + (0.12865 x LDL) + (0.56320 x Usia) + (12.92272 x JK) + (-1.17612 x AU) + (0.48388 x BUN) + (-11.49542 x Kreatinin) + -39.57837			
Aturan 7	0.0005	3	86.67%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.64105 x TD) + (0.04060 x KG) + (0.06924 x KT) + (0.45512 x LDL) + (0.86206 x Usia) + (-11.76678 x JK) + (3.16092 x AU) + (0.54764 x BUN) + (2.58481 x Kreatinin) + -204.95867 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.26521 x TD) + (0.09848 x KG) + (0.11056 x KT) + (0.13476 x LDL) + (0.61444 x Usia) + (9.20452 x JK) + (-0.17658 x AU) + (0.55141 x BUN) + (-11.02463 x Kreatinin) + -60.48116 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (3.49517 x TD) + (-0.45770 x KG) + (0.52148 x KT) + (0.17912 x LDL) + (-1.40399 x Usia) + (39.15561 x JK) + (-8.37101 x AU) + (-2.94159 x BUN) + (-29.27717 x Kreatinin) + -325.90179			
Aturan 8	0.0005	3	90.00%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (-1.20923 x TD) + (0.09475 x KG) + (0.40540 x KT) + (-1.79587 x LDL) + (-2.76448 x Usia) + (14.06793 x JK) + (11.44768 x AU) + (0.59537 x BUN) + (-20.55836 x Kreatinin) + 418.88157 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (1.30160 x TD) + (0.13318 x KG) + (0.52284 x KT) + (0.65169 x LDL) + (2.03661 x Usia) + (-12.68318 x JK) + (4.24943 x AU) + (-0.29223 x BUN) + (-14.97684 x Kreatinin) + -444.74066 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko =			

Z3 = (0.28627 x TD) + (0.07239 x KG) + (0.12772 x KT) + (0.26701 x LDL) + (0.40510 x Usia) + (6.82323 x JK) + (-1.74744 x AU) + (0.24572 x BUN) + (-8.51495 x Kreatinin) + -60.40362			
Aturan 9	0.0005	3	86.67%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.98401 x TD) + (-0.38408 x KG) + (1.01565 x KT) + (-0.02905 x LDL) + (1.32390 x Usia) + (-2.04682 x JK) + (2.90523 x AU) + (-0.23801 x BUN) + (6.69590 x Kreatinin) + -328.19810 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.40644 x TD) + (0.66484 x KG) + (-0.80860 x KT) + (1.17168 x LDL) + (0.70586 x Usia) + (-31.21526 x JK) + (9.25921 x AU) + (0.89516 x BUN) + (-5.09984 x Kreatinin) + -199.52028 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (0.39842 x TD) + (0.05226 x KG) + (0.21006 x KT) + (0.11663 x LDL) + (0.59699 x Usia) + (18.31713 x JK) + (-2.39974 x AU) + (-0.80893 x BUN) + (-12.40476 x Kreatinin) + -60.22520			
Aturan 10	0.0005	3	93.33%
[R1] IF TD is center11 and KG is center12 and KT is center13 and LDL is center14 and Usia is center15 and JK is center16 and AU is center17 and BUN is center18 and Kreatinin is center19 and StatusRisiko = Z1 = (0.41555 x TD) + (0.08521 x KG) + (0.13953 x KT) + (0.25972 x LDL) + (0.59023 x Usia) + (13.20295 x JK) + (-1.24003 x AU) + (-0.72027 x BUN) + (-18.77664 x Kreatinin) + -74.89187 [R2] IF TD is center21 and KG is center22 and KT is center23 and LDL is center24 and Usia is center25 and JK is center26 and AU is center27 and BUN is center28 and Kreatinin is center29 and StatusRisiko = Z2 = (0.51204 x TD) + (0.16594 x KG) + (-0.33699 x KT) + (0.62544 x LDL) + (1.21978 x Usia) + (-3.24731 x JK) + (8.66020 x AU) + (-0.08157 x BUN) + (9.49424 x Kreatinin) + -183.68552 [R3] IF TD is center31 and KG is center32 and KT is center33 and LDL is center34 and Usia is center35 and JK is center36 and AU is center37 and BUN is center38 and Kreatinin is center39 and StatusRisiko = Z3 = (0.91840 x TD) + (0.11112 x KG) + (0.98792 x KT) + (0.00735 x LDL) + (0.90372 x Usia) + (-18.53118 x JK) + (5.37447 x AU) + (0.23374 x BUN) + (13.39467 x Kreatinin) + -398.34327			
Rata-Rata Akurasi			**87.33%**

Dari Tabel (5.5) dapat diketahui bahwa rata-rata nilai akurasi yang dihasilkan dari pengujian ini adalah 87,33%, dengan nilai akurasi terendah sebesar 80 % dan nilai akurasi tertinggi sebesar 93,33%.

5.2.3. Analisis Pengujian Akurasi

Pengujian akurasi dilakukan sebanyak 10 kali untuk mendapatkan 10 jenis aturan *fuzzy* yang terbentuk pada proses pelatihan. Grafik uji akurasi pada pengujian kedua ini ditunjukkan pada Gambar (5.3).



Gambar 5.3 Grafik Akurasi Aturan

Dari grafik akurasi aturan pada Gambar (5.3) dapat dilihat bahwa akurasi yang dihasilkan dari proses pengujian dengan menggunakan data uji sebanyak 30 data berdasarkan aturan *fuzzy* yang terbentuk bervariasi meskipun jumlah aturan sama yaitu 3.

Nilai akurasi tertinggi yang dihasilkan dari pengujian dengan menggunakan data uji adalah 93,33%, yaitu pada aturan 1 dan 10. Sedangkan nilai akurasi terendah adalah 80%, yaitu pada aturan 2 dan 3. Rata-rata nilai akurasi dari pengujian ini adalah 87,33%.

Dari penjelasan diatas, terdapat beberapa hal yang mengakibatkan hasil akurasi pada proses pengujian ini bervariasi. Pertama adalah pusat *cluster* yang dihasilkan pada proses *clustering*. Pusat *cluster* berperan besar pada keanggotaan suatu titik data terhadap *cluster* berdasarkan derajat keanggotaan. Sehingga dimungkinkan data uji yang sama memiliki derajat keanggotaan berbeda, tergantung pusat *cluster* yang terbentuk.

Kedua yaitu jumlah data latih. Karena minimnya jumlah data latih yang digunakan pada penelitian ini, maka dalam proses pembelajaran, pengetahuan yang dipelajari oleh sistem pun terbatas, sehingga akurasi yang dihasilkan kurang optimal.

Ketiga yaitu variasi data (variabel *output* rendah, sedang dan tinggi) pada data latih. Pada 109 data latih yang digunakan pada penelitian ini terdapat 23 data pada status rendah, 47 data pada status sedang, dan 39 data pada status tinggi. Variasi data yang tidak imbang inilah yang akan mempengaruhi proses pembelajaran sistem. Apabila jumlah variasi datanya seimbang maka pengetahuan sistem terhadap data kelompok rendah, sedang maupun tinggi juga seimbang. Sehingga dapat disimpulkan bahwa semakin banyak data latih yang digunakan dan semakin seimbang variasi data maka semakin baik tingkat akurasi yang dihasilkan.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil penelitian tentang pembangkitan aturan *fuzzy* menggunakan *fuzzy c-means clustering* untuk deteksi dini risiko penyakit stroke dapat disimpulkan bahwa :

1. Metode *fuzzy c-means* dapat diimplementasikan untuk pembangkitan aturan *fuzzy* pada deteksi dini risiko penyakit stroke.
2. Jumlah *cluster* ideal yang terbentuk dari 10 kali pengujian dengan jumlah data latih yang sama berdasarkan nilai batasan varian terkecil adalah jumah *cluster* 3. Hal tersebut menunjukkan bahwa penentuan jumlah *cluster* paling ideal telah sesuai dengan menggunakan metode analisa varian. Kesesuaian tersebut terlihat dari jumlah *cluster* ideal yang terbentuk berdasarkan analisa varian sama dengan jumlah *cluster* yang mempunyai akurasi tertinggi.
3. Akurasi rata-rata yang dihasilkan dari 10 kali pengujian dengan menggunakan data uji pada 10 aturan yang terbentuk adalah 87,33% dengan akurasi tertinggi yang dapat dihasilkan adalah 93,33%. Akurasi yang dihasilkan dari pengujian bervariasi karena faktor jumlah *cluster*, jumlah data latih dan variasi data. Karena minimnya jumlah data latih yang digunakan pada penelitian ini, maka dalam proses pembelajaran, pengetahuan yang dipelajari oleh sistem pun terbatas, sehingga akurasi yang dihasilkan kurang optimal. Kemudian yaitu variasi data (variabel *output* rendah, sedang dan tinggi) pada data latih. Jumlah data dengan *output* rendah, sedang dan tinggi seharusnya seimbang. Apabila jumlah variasi datanya seimbang maka pengetahuan sistem terhadap data kelompok rendah, sedang maupun tinggi juga seimbang.

6.2 Saran

Untuk pengembangan sistem lebih lanjut dari hasil penelitian yang telah dilakukan, berikut saran-saran yang dapat diberikan :

1. Data latih yang digunakan lebih banyak dan variasi data juga harus seimbang agar semakin banyak pengetahuan sistem dalam proses pembelajaran sehingga tingkat akurasi yang dihasilkan bias lebih baik dan stabil.
2. Dilakukan penelitian untuk nantinya dapat mengetahui jenis stroke yang diderita pasien, apakah stroke iskemik atau hemoragik.

DAFTAR RUSTAKA

[ABA-03] Abas, F.S., K. Martinez. 2003. "*Classification of Painting Cracks for Content-Based Analysis"*. http://eprints.ecs.soton.ac.uk/7294/1/ spie_cracks.pdf [19 Mei 2014].

[AKH-10] Arapoglou, Roi.Kolomvatos, Kostas dan Hadjiefthymiades, Stathes.*"Buye Agent Decision Process Based on Automatic FuzzyRules Generation Mehods"*. http://pcomp.di.uoa.gr/pubs/WCCI_f427.pdf. [19 Mei 2014].

[ANN-13] Annathasshia, Cindy.2013."Faktor Risiko terjadinya Stroke". http://www.pilihdokter.com/id/blog/Faktor-Resiko-Terjadinya-Stroke. [19 Mei 2014].

[ARA-10] Arapoglou, Roi, Kostas Kolomvatsos and Stathes Hadjiefthymiades. 2010. "*Buyer Agent Decision Process Based on Automatic Fuzzy Rules Generation Methods"*. http://p-comp.di.uoa.gr/ pubs/WCCI_f427.pdf [19 Mei 2014].

[DEW-12] dr Dewi. 2012. "Jenis-jenis stroke". http://poliklinik.ipdn.ac.id/home/artikel-kesehatan/jenis---jenis-stroke [15 Maret 2013].

[ILH-11] Ilham, B.Priyambodo. 2011. "*Impelementasi Metode Single Lingkage untuk Menentukan Kinerja Agen pada Call Centre Berbasis Asterisk for Java"*. Institut Teknologi Sepuluh Nopember (ITS). Surabaya.

[JAN-97] Jang, J.S.R., Sun, C.T., Mizutani,E. 1997. "Neuro-Fuzzy and Soft Computing". Prentice-Hall International, New Jersey.

[JEP-13] Jeperson,Hendra.2013."Memahami Apa Itu Kadar Gula Darah". http://www.deherba.com/memahami-apa-itu-kadar-gula-darah.html. [19 Mei 2014].

[KER-09] Kertia, Nyoman.2009."Asam Urat". Bentang Pustaka, Yogyakarta.

[KUS-10] Kusumadewi, S dan Purnomo, H. 2010. "Aplikasi Logika Fuzzy untuk Pendukung Keputusan". Jilid 2 Yogyakarta.: Graha Ilmu.

[KUS-10] Kusrini. 2008. Aplikasi Sistem Pakar Menentuan Faktor Kepastian Pengguna dengan Metode Kuantitatif Pertanyaan, Andi, Yogyakarta.

[LAR-05] Larose, Daniel T.2005. “Discovering Knowledge in Data: an Introduction to Data Mining”. John & Wiley & Sons, Inc. New Jersey.

[LUD-11] Ludviani, Resti. 2011. “Pembangkitan Aturan Fuzzy menggunakan Fuzzy C-Means (FCM) Clustering untuk Diagnosa Risiko Penyakit Jantung Koroner (PJK)”. Skripsi. Universitas Brawijaya. Malang.

[MUH-13] Muhlisin, Ahmad.2013.”Tekanan Darah”.http://mediskus.com/penyakit/tekanan-darah.html. [05 Mei 2014].

[NUG-06] Nugraha, Dany, dkk. 2006. “Diagnosis Gangguan Sistem Urinari pada Anjing dan Kucing Menggunakan VFI 5”. Institut Pertanian Bogor.

[QAD-13] Qadariah Arief, Ria.2013.”Kolesterol Total”. http://www.konsultankolesterol.com/kolesterol-total.html. [19 Mei 2014].

[SAN-10] Santoso, Singgih. 2010. “Statistika Multivariat”. PT Gramedia. Jakarta.

[SAY-14] Sayekti, Ely Ratna. 2014. “Implementasi Algoritma Fuzzy C-Means Clustering Untuk Pembangkitan Aturan Fuzzy Pada Pengelompokan Tingkat Resiko Penyakit Kanker Payudara”. Skripsi. Universitas Brawijaya. Malang.

[SHE-13] Shernend.2013. “Kreatinin Darah (Serum)”. http://www.amazine.co/27032/apa-itu-rasio-bun-kreatinin-prosedur-membaca-hasilnya/. [19 Mei 2014]

[SHO-12] Sholeh, Ahmad Fashel. 2012. “Aplikasi Pendukung Keputusan Untuk Deteksi Dini Risiko Penyakit Stroke Menggunakan Logika *Fuzzy* Mamdani: Studi Kasus Di RS XYZ”. Skripsi. Institut Tinggi Sepuluh November. Surabaya.

[YAS-11] Yastroki. 2011. “Sekilas Tentang Stroke”. Perhimpunan Dokter Spesialis Syaraf Indonesia dan Yayasan Stroke Indonesia.

LAMPIRAN

Lampiran 1 : Data Latih

Id Latih	T. Darah	Kadar Gula	Kolesterol Total	LDL	Usia	JK	Asam Urat	BUN	Krea-tinin	Risiko
1	190	173	147	80	70	1	6.6	21.5	0.9	100
2	130	326	138	88	56	0	3.4	21.1	1.4	50
3	130	391	176	86	66	0	6	20.4	1.2	50
4	130	108	187	123	63	1	6.7	20	1.5	0
5	160	98	176	106	52	1	10.2	14.4	1.1	50
6	140	228	195	127	30	1	6.2	20	1.5	50
7	130	106	189	128	57	0	6.2	20	1.5	50
8	180	117	191	117	46	0	4	6.1	0.8	50
9	180	96	229	162	45	0	5.4	5.4	0.8	50
10	150	220	168	106	47	0	3	7.4	0.5	50
11	160	104	192	112	72	1	8.2	15.1	1.1	100
12	170	146	208	109	61	0	5	8.6	1	50
13	120	106	195	127	55	1	6.2	9.6	1	0
14	210	204	229	162	49	0	7.2	23.1	0.7	100
15	140	116	159	92	75	0	4.2	12	0.8	50
16	160	128	175	102	42	1	8	10.5	0.9	50
17	110	86	153	105	72	1	6.2	27.4	1.7	0
18	140	111	207	168	80	1	7.9	31.7	2	100
19	180	421	277	165	56	1	4.7	14.9	1.5	100
20	230	136	294	195	40	0	4.8	16.2	0.8	100
21	140	200	122	80	62	1	7.7	25.5	1.3	50
22	110	102	137	90	47	1	5.4	11.3	6.9	0
23	240	167	263	158	58	1	8	15.8	1.4	100
24	140	92	161	102	50	0	4	12.8	1	50
25	110	135	113	63	80	1	8.1	39.4	2.9	0
26	160	192	285	183	63	0	7.5	10.7	0.8	100
27	120	196	218	160	62	0	2.8	13.5	0.6	50
28	120	147	139	78	61	1	9	13.2	1.3	0
29	140	64	193	122	44	0	7.5	30.7	1.8	0
30	150	111	249	190	48	0	4.6	10.7	0.8	100
31	160	188	208	170	62	1	6.4	14.3	1.3	100
32	130	92	221	164	48	0	4.1	6.9	0.6	50
33	200	253	249	187	49	0	5.2	14.1	1	100
34	110	117	111	52	73	1	7.3	31.6	2.1	0
35	160	136	143	84	71	1	8.9	23.4	1.9	50

36	170	240	264	163	52	1	6.9	15.3	2.5	100
37	140	328	269	108	50	0	6.6	9.5	0.6	100
38	150	181	186	137	75	0	3.5	6.2	0.5	50
39	130	119	150	108	40	0	4.9	7.4	1	0
40	170	86	152	106	80	1	2.8	9.6	0.8	100
41	130	128	237	154	50	0	4.5	12.4	0.7	50
42	190	100	190	124	62	0	6	14.8	1.2	50
43	160	163	221	148	52	1	8.4	11.1	0.8	100
44	120	163	132	75	73	1	6.5	24.1	1.4	0
45	160	105	157	102	56	0	4.5	11.6	0.7	50
46	140	137	244	187	40	1	6	9.3	0.8	50
47	170	107	138	93	65	1	6.8	9.5	1.1	100
48	170	192	209	150	59	1	8.2	34.7	1.9	100
49	110	95	210	153	62	1	5.2	7.7	1	0
50	180	206	270	184	70	0	3.8	19.5	1.3	100
51	180	257	195	137	56	0	3	14.7	0.7	100
52	110	109	137	92	65	1	8.8	38.7	3.6	0
53	130	115	229	173	71	1	5.5	12.3	1.1	50
54	120	405	252	166	57	1	3.9	13	1.2	100
55	120	117	198	113	36	1	9	9.8	1.1	0
56	192	231	183	134	76	1	9.8	17.9	1.2	100
57	180	140	194	134	69	1	8.6	28.7	1.7	100
58	180	107	215	150	65	0	5.4	14.6	0.9	100
59	120	97	159	103	80	0	3.4	12	0.9	50
60	150	178	252	135	72	1	4.4	10.9	0.9	100
61	200	138	267	188	63	0	6.4	13	0.9	100
62	160	102	187	117	60	1	8	14.4	1.1	50
63	150	132	167	95	75	0	4.8	39.2	1.5	50
64	120	91	162	91	55	0	4.8	20.2	1	0
65	130	100	148	96	59	1	5	10.3	1	0
66	140	120	257	178	64	0	7.5	44.1	2.6	100
67	170	70	205	151	43	0	10	26.5	3	50
68	110	155	115	90	52	0	5.2	11.7	0.5	0
69	160	78	243	182	59	0	4.9	11.7	0.6	100
70	180	136	177	136	51	1	7.3	13.5	1.2	50
71	160	79	238	157	50	0	4.4	11.9	0.7	100
72	100	155	195	127	35	0	6.2	20	1.5	0
73	200	169	195	127	65	1	6.2	19.9	1.6	100
74	220	174	195	127	56	1	6.2	12.9	0.9	100
75	180	247	195	127	55	0	6.2	20.6	1.1	50
76	106	121	161	56	69	0	4.4	29.5	1.5	0

77	130	129	141	78	73	1	2.3	8.8	0.6	0
78	140	108	222	146	40	0	3.3	11.6	0.5	50
79	170	98	147	95	52	1	2.8	21.5	1.5	50
80	160	110	194	108	60	1	10	20.4	1.5	50
81	140	101	269	195	58	0	5.1	21.4	0.8	100
82	140	188	195	127	60	1	6.2	10.5	1	50
83	190	156	154	63	52	1	7.8	14.6	1.3	50
84	150	115	122	70	62	0	2	7.5	0.7	0
85	190	119	195	127	63	1	6.2	13.2	0.7	100
86	240	305	232	167	40	1	12.2	25.8	2.4	100
87	200	243	282	182	54	0	6.2	11.5	0.7	100
88	200	186	195	127	64	0	6.2	13.6	0.9	100
89	150	155	195	127	66	0	6.2	30.6	1	50
90	154	175	219	162	48	0	6.2	19.8	1.1	50
91	190	127	133	86	43	0	3.8	18.2	0.5	0
92	130	110	153	95	78	1	6.5	20	1.5	50
93	130	118	195	127	47	1	6.2	10.4	1.2	50
94	150	87	167	107	60	1	6.7	20	1.5	50
95	170	155	195	127	60	1	6.2	20	1.5	50
96	180	82	130	84	72	0	4.7	12.7	0.5	50
97	190	120	197	117	50	0	7.8	13.3	0.9	50
98	140	155	195	127	46	0	6.2	20	1.5	50
99	160	386	260	168	55	0	3.1	6	1	100
100	150	155	195	127	60	1	6.2	20	1.5	50
101	100	100	195	127	28	1	6.2	16.2	0.6	0
102	170	130	207	137	52	0	3.6	10.4	1.2	50
103	100	155	173	133	56	1	7.2	31.6	1.9	50
104	170	152	195	127	55	1	6.2	14.9	1.2	50
105	110	85	97	53	35	0	6.2	48.9	2.2	0
106	170	223	375	205	60	0	6.2	79.2	3.8	100
107	130	155	195	127	61	0	6.2	11.6	0.5	50
108	190	260	172	128	39	0	2.2	10	0.5	100
109	170	112	189	121	47	0	5.2	11.1	0.8	100

Lampiran 2 : Data Uji

Id uji	T. Darah	Kadar Gula	Kolesterol Total	LDL	Usia	JK	Asam Urat	BUN	Krea-tinin	Risiko
1	160	130	303	132	52	1	9.6	13.9	0.8	100
2	160	106	145	106	58	1	8.3	7.6	0.9	50
3	190	151	298	202	51	0	4.2	11.8	0.8	100
4	160	168	181	109	64	0	4.6	11.7	0.7	50
5	140	120	153	93	46	1	9.3	12.8	1	0
6	160	127	168	119	68	0	3.7	10	1.1	50
7	120	97	148	94	58	1	6.8	20.8	1	0
8	150	135	297	206	57	0	6.3	20.5	0.8	100
9	190	135	195	143	85	1	6.3	10.8	1.2	100
10	170	101	228	152	71	0	6.1	9.9	0.7	100
11	190	251	180	130	54	1	6.3	16.6	1.3	100
12	190	131	164	114	48	1	6.1	10.4	1.1	50
13	140	529	195	127	67	0	6.2	16.7	0.9	100
14	150	144	166	118	70	0	6.5	11.3	1.1	50
15	140	171	165	113	46	1	8	20	1.5	50
16	160	108	146	98	50	1	7	13.1	1.2	50
17	190	85	272	153	63	1	5.6	11	1.1	100
18	150	203	179	124	75	1	4.7	13.3	1.3	100
19	140	92	158	121	52	1	6.7	11.8	1.3	50
20	180	150	148	108	70	1	7.5	36.3	1.9	100
21	140	128	214	127	70	1	4.9	12.7	1.1	50
22	130	101	160	109	70	1	4	14.9	0.6	50
23	100	81	155	84	56	0	4.7	7.5	0.7	0
24	180	186	205	87	51	0	13.8	37.1	3.5	100
25	160	169	174	118	58	1	5.4	14	1.4	50
26	130	243	202	141	66	0	8.5	54.9	1.5	100
27	150	128	176	116	71	1	5.6	14	1.1	50
28	140	98	154	98	69	1	8	35	1.9	50
29	155	149	200	161	49	1	5.2	9.1	0.9	50
30	110	155	195	127	54	0	6.2	20	1.5	50

Lampiran 3 : Hasil Wawancara

Narasumber : Dr. Farhad Bal'afif, Sp.BS.

Waktu : Rabu, 16 Juli 2014

1. **Dari 9 faktor risiko tersebut, faktor apa yang paling berpengaruh?**
   Faktor yang paling berpengaruh pada stroke penyumbatan yaitu kadar gula, sedangkan pada stroke pendarahan yaitu tekanan darah. Namun karena penelitian ini merupakan stroke secara umum, maka faktor-faktor yang paling bnyak berpengaruh yaitu kadar gula, tekanan darah, LDL, usia, dan jenis kelamin.
2. **Bagaimana cara menganalisa dan menentukan tingkat risiko stroke dari faktor-faktor tersebut?**
   Dalam menganalisa dan menentukan tingkat risiko stroke sebenarnya tidak hanya faktor medis saja, namu faktor non medis juga diperhatikan. Faktor non medis disini yaitu riwayat stroke, obesitas, dan merokok. Namun karena pada penelitian anda tidak ada faktor non medis, maka untuk faktor medis cara menganalisanya pertama yang dilihat terlebih dahulu yaitu kadar gula, tekanan darah dan LDL.
   1. Jika ketiga faktor tersebut normal, usia muda-parobaya dan dia wanita atau laki-laki berusia muda maka status risiko stroke rendah. Namun jika dia laki-laki berusia parobaya-tua, dan wanita berusia tua maka status risiko stroke sedang.
   2. Jika kadar gula dan tekanan darah tinggi, faktor lainnya normal,dan dia adalah wanita berusia muda maka status risiko rendah, namun jika dia laki-laki berusia muda maka status risiko sedang. Kemudian jika dia laki-laki berusia parobaya-tua maka status risiko tinggi.
   3. Jika kadar gula dan tekanan darah tinggi, faktor lainnya sedang, dan dia adalah wanita berusia muda maka status risiko sedang. Kemudian jika dia wanita atau laki-laki berusia parobaya-tua maka status risiko tinggi.
3. **Seberapa besar pengaruh BUN, asam urat dan kreatinin terhadap risiko stroke?**
   Berpengaruh namun sangat kecil.