PREDIKSI CURAH HUJAN MENGGUNAKAN METODE *ADAPTIVE NEURO FUZZY INFERENCE SYSTEM* (ANFIS)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Muhammad Isradi Azhar
NIM : 145150201111037

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

PREDIKSI CURAH HUJAN MENGGUNAKAN METODE *ADAPTIVE NEURO FUZZY INFERENCE SYSTEM* (ANFIS)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Muhammad Isradi Azhar
NIM: 145150201111037

Skripsi ini telah diuji dan dinyatakan lulus pada
6 Juni 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D
NIP. 19720919 199702 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 6 Juni 2018

Muhammad Isradi Azhar

NIM: 145150201111037

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas berkat, rahmat, ridho dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "Prediksi Curah Hujan Menggunakan Metode *Adaptive Neuro Fuzzy Inference System*" sebagai salah satu persyaratan untuk menyelesaikan studi di Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya. Dengan adanya tugas akhir ini diharapkan mahasiswa dapat menerapkan ilmu yang dipelajari semasa kuliah dan memberikan manfaat yang berkelanjutan bagi masyarakat.

Penulis menyadari bahwa tugas akhir ini dapat terselesaikan berkat bantuan, petunjuk, bimbingan dan dukungan dari berbagai pihak yang telah banyak membantu proses penyelesaian tugas akhir ini. Oleh karena itu penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dosen Pembimbing tunggal dan Dekan Fakultas Ilmu Komputer Universitas Brawijaya yang dengan sabar memberikan arahan, bimbingan, ilmu, saran, dan kritik selama penyusunan laporan dan pengerjaan tugas akhir ini.
2. Bapak tercinta H.Huzaili Yamani, S.Pd, M.Si dan Ibu tercinta Hj.Taty Indriani selaku orang tua dari penulis yang selalu memberikan do'a, kasih sayang, dukungan moril dan materil yang tidak ada henti-hentinya sejak awal kuliah sampai saat ini penulis menyelesaikan tugas akhir.
3. Kakak tersayang Indra Irawan, S.T, M.Sc dan Irfan Adhitia,S.E yang selalu memberikan semangat, dukungan serta motivasi kepada penulis sehingga penulis dapat menyelesaikan tugas akhir ini.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya yang telah memberikan kesempatan untuk pelaksanaan skripsi.
5. Bapak dan Ibu dosen serta seluruh staf Fakultas Ilmu Komputer Universitas Brawijaya yang telah mendidik selama perkuliahan dan memberi inspirasi dalam pengerjaan skripsi ini.
6. Teman-Teman Himpunan Mahasiswa Informatika khususnya teman seperjuangan angkatan 2014 yang telah memberikan ilmu dan pengalaman, serta membantu dalam penyusunan laporan skripsi ini.
7. Nirmala Fa'izah Saraswati sebagai partner pengerjaan skripsi yang selalu memberikan dukungan, masukan, motivasi, dan memberikan semangat dalam pengerjaan skripsi ini.
8. Nur Adli Ari Darmawand, S.Kom, Ali Syahrawardi, S.Kom, Muhammad Noor Taufiq, S.Kom sebagai senior Kalimantan dan sahabat penulis yang selalu

memberikan motivasi, do'a, masukan, dan hiburan disaat penulis sedang mengalami kesulitan dalam mengerjakan skripsi.

9. Muhammad Ismail sebagai sahabat penulis yang bersedia menemani penulis dari awal kuliah sampai selesai pengerjaan skripsi ini, dan selalu mengingatkan penulis untuk terus bersabar dan berdoa kepada Allah SWT disaat penulis mengalami kesulitan dalam mengerjakan skripsi.
10. Sahabat BE'M Filkom yang setia memberikan dukungan, semangat, dan motivasi untuk segera menyelesaikan pengerjaan skripsi.
11. Semua pihak yang tidak dapat penulis sebutkan satu persatu yang terlibat secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Semoga jasa dan amal baik mendapatkan balasan dari Tuhan Yang Maha Esa. Dan semoga penulisan laporan skripsi ini bermanfaat baik untuk pembaca dan pengembangan penelitian selanjutnya. Penulis menyadari bahwa skripsi ini masih jauh dari sempurna dan tidak terlepas dari kekurangan dan kesalahan karena keterbatasan materi, kemampuan, pengalaman, dan pengetahuan yang dimiliki penulis. Oleh karena itu, segala kritik dan saran yang bersifat membangun, penulis harapkan untuk penyempurnaan skripsi ini dapat disampaikan melalui email penulis.

Malang, 6 Juni 2018

penulis
azhar.isradi@gmail.com

ABSTRAK

Curah hujan adalah banyaknya hujan yang tercurah atau turun di suatu daerah dalam jangka waktu tertentu. Informasi curah hujan berguna dalam berbagai bidang salah satunya pertanian. Dalam bidang pertanian, informasi curah hujan dapat memengaruhi masa tanam setiap tahun dan jenis tanaman apa yang cocok untuk ditanam. Kabupaten Malang adalah salah satu daerah di Indonesia yang memiliki lahan pertanian luas yaitu 36.359 Ha dan menghasilkan produksi beras sebesar 470.285 Ton. Tanaman padi memiliki kriteria umum untuk menentukan awal musim tanam padi, yaitu dengan jumlah curah hujan sebesar lebih dari 50 mm dalam tiga dasarian berturut-turut. Namun musim penghujan saat ini tidak menentu yang mengakibatkan proses penanaman padi terganggu. Oleh karena itu, diperlukan prediksi curah hujan untuk membantu petani agar tidak mengalami kerugian.

Ditinjau dari permasalahan tersebut, metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) dapat digunakan untuk melakukan prediksi curah hujan dengan memanfaatkan data dasarian curah hujan, suhu udara, kelembapan udara, dan kecepatan angin. Metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) merupakan gabungan dari jaringan saraf tiruan dan logika fuzzy. Pada proses pembelajaran metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) terdapat algoritme *backpropagation steepest descent* dan *least square estimator* (LSE). Berdasarkan hasil pengujian dengan menggunakan parameter-parameter terbaik, didapatkan nilai RMSE sebesar 1,88.

Kata kunci: curah hujan, prediksi, adaptive neuro fuzzy inference system, anfis

ABSTRACT

Rainfall is the amount of rain that occurs in an area within a certain period of time. Information of rainfall is useful in various fields such as agriculture. In the field of agriculture, the information of rainfall can affect the annual planting period and also can determine what kind of crops that are suitable to be planted. Malang Regency is one areas in Indonesia which has 36.359 Ha farming area and produce rice equal to 470.285 Ton. Rice crops have a common criteria for determining the beginning of the rice growing season, with the amount of rainfall more than 50 mm in three consecutive dasarians. But the current wet season is uncertain which resulted in the process of rice cultivation is disrupted. Therefore, rainfall prediction is needed to help farmers to reduce the possibility of loss.

In view of these problems, Adaptive Neuro Fuzzy Inference System (ANFIS) method can be used to predict rainfall by utilizing dasarian data of rainfall, temperature, humidity, and wind speed. Adaptive Neuro Fuzzy Inference System (ANFIS) is a combination of neural network and fuzzy logic. In the learning process of Adaptive Neuro Fuzzy Inference System (ANFIS), there is backpropagation steepest descent and least square estimator (LSE) algorithm. Based on the test results using the best parameters, it obtain best RMSE value of 1.88.

Keywords: rainfall, prediction, adaptive neuro fuzzy inference system, anfis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE SUMBER

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Informasi curah hujan merupakan hal yang sangat berguna dalam kegiatan sehari-hari. Kegiatan tersebut meliputi produksi pertanian, perkebunan, perikanan, dan transportasi. Pada bidang pertanian, curah hujan dapat mempengaruhi masa tanam setiap tahun dan jenis tanaman apa yang cocok untuk ditanam. Selain itu informasi curah hujan juga dapat mengantisipasi kemungkinan kejadian ekstrim yang menyebabkan kegagalan produksi pertanian. Kejadian ekstrim ini merupakan fenomena *El Nino* yang mengakibatkan kekeringan dan fenomena *La Nina* yang mengakibatkan banjir. Dampak yang terjadi dari fenomena tersebut disebabkan kurangnya informasi tentang curah hujan yang akurat dan tingkat kemampuan peramalan yang masih belum baik (Wigena, 2006). Setiap tahun di Indonesia mengalami kekeringan dan banjir dengan intensitas yang berbeda. Oleh karena itu prediksi curah hujan sangat berpengaruh dalam kegiatan sosial ekonomi Indonesia.

Indonesia merupakan negara agraris dimana penduduknya sebagian besar bergerak di bidang pertanian. Salah satu daerah dengan lahan pertanian yang cukup besar di Indonesia yaitu Jawa Timur. Menurut data dari Badan Pusat Statistik (BPS) Provinsi Jawa Timur, Kabupaten Malang mempunyai luas sawah 36.359 Ha dan menghasilkan produksi beras sebesar 470.285 Ton. Tanaman padi adalah tanaman yang dipengaruhi oleh curah hujan. Tanaman padi memiliki kriteria umum untuk menentukan awal musim tanam padi yaitu awal musim hujan (MH) dengan jumlah curah hujan sebesar lebih dari 50 mm dalam tiga dasarian berturut-turut (Surmaini & Syahbuddin, 2016). Namun musim penghujan saat ini tidak menentu dan curah hujan sulit untuk diprediksi yang mengakibatkan proses penanaman padi terganggu.

Pada umumnya prediksi curah hujan dilakukan pada data masa lampau yang dianalisis dengan menggunakan cara-cara tertentu. Curah hujan mempunyai kecenderungan pola yang berulang dari masa ke masa. Data masa lampau dikumpulkan, dipelajari, dan dianalisis. Hasil analisis tersebut mencoba menyatakan sesuatu yang akan terjadi di masa yang akan datang, jelas dalam hal ini ada ketidakpastian sehingga akan ada faktor keseksamaan yang harus diperhitungkan. Yang jelas tidak akan selalu didapatkan hasil ramalan dengan ketepatan 100 %. Ini tidak berarti ramalan menjadi percuma, sebaliknya terbukti bahwa ramalan telah banyak digunakan dan membantu dengan baik dalam berbagai manajemen sebagai dasar-dasar perencanaan, pengawasan, dan pengambilan keputusan. Salah satunya adalah peramalan curah hujan atau dalam dunia klimatologi biasa disebut dengan prakiraan atau prediksi curah hujan.

Untuk melakukan prediksi curah hujan pada suatu wilayah ada banyak metode yang digunakan salah satunya dengan *metode Adaptive Neuro-Fuzzy Inference System* (ANFIS) yang merupakan gabungan dari Jaringan Saraf Tiruan

(*Artificial Neural Network*) dan Logika Fuzzy (*Fuzzy Inference System*). Metode ANFIS dipilih karena pada dasarnya *Neural Network* dapat belajar dari pengalaman/data sebelumnya. Sama seperti *Neural Network*, Logika Fuzzy dapat menyediakan perhitungan fungsi tanpa pemodelan matematis sebagaimana output bergantung dengan input. Sebagai tambahan, *Neuro-Fuzzy memiliki low-level learning* dan kekuatan komputasional dari *Neural Network* serta keuntungan dari *high-level human like thinking* dari Fuzzy sistem, sehingga menjadikan keduanya lebih baik untuk masalah *non-linear prediction*.

Pada metode ANFIS diperlukan metode *clustering* untuk membagi data menjadi beberapa kelompok. Data curah hujan memiliki 3 kelompok yaitu rendah, menengah, dan tinggi (BMKG, 2017). Salah satu metode *clustering* yang bisa digunakan yaitu *K-Means Clustering*. *K-Means Clustering* adalah algoritme sederhana dari metode *unsupervised learning* yang umum digunakan untuk menyelesaikan permasalahan klasterisasi karena mudah untuk diimplementasikan (Hung, Wu, Chang, & Yang, 2005).

Penelitian prediksi curah hujan pernah dilakukan oleh Fatyanosa dan Mahmudy (2016) dengan metode *Real Coded Genetic Fuzzy System* (RCGFS). Metode penelitian tersebut menggabungkan logika fuzzy dan algoritme genetika. Nilai *Root Mean Square Error* (RMSE) pada penelitian tersebut sebesar 8,45. Prediksi curah hujan juga dilakukan pada daerah Tengger oleh Wahyuni, Mahmudy, dan Iriany (2016) dengan metode *Tsukamoto Fuzzy Inference System* (FIS). Penelitian tersebut menggunakan data dasarian curah hujan sebagai data masukan. Nilai *Root Mean Square Error* (RMSE) yang didapat sebesar 8,64. Selain itu, pada daerah yang sama telah dilakukan penelitian prediksi curah hujan oleh Wahyuni dan Mahmudy (2017) menggunakan metode *hybrid Tsukamoto FIS* dengan algoritma genetika. Nilai RMSE yang didapatkan lebih kecil dari sebelumnya yaitu sebesar 6,63. Metode-metode yang digunakan untuk prediksi curah hujan sudah cukup baik, namun memiliki beberapa kekurangan yaitu cukup tingginya nilai *Root Mean Square Error* (RMSE) yang didapat dan data yang digunakan sebagai data acuan prediksi hanya satu data curah hujan.

Pada tahun (2017), Santika, Mahmudy dan Naba melakukan penelitian tentang peramalan beban listrik dengan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS). Dari penelitian tersebut didapatkan RMSE sebesar 0,0298. Pada penelitian selanjutnya dilakukan Wahyuni, Mahmudy dan Iriany (2017) tentang prediksi curah hujan menggunakan metode hybrid *Adaptive Neuro-Fuzzy Inference System* (ANFIS) dan Algoritme Genetika. Hasil dari penelitian tersebut didapatkan nilai RMSE sebesar 5,41.

Oleh karena itu metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) yang merupakan kombinasi dari Jaringan Saraf Tiruan (*Artificial Neural Network*) dan Logika Fuzzy (*Fuzzy Inference System*) diharapkan dapat menghasilkan suatu sistem yang mampu belajar secara terus menerus dan mampu memberikan hasil keluaran berupa prediksi curah hujan dengan tingkat akurasi yang baik.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah diuraikan, maka didapat rumusan masalah sebagai berikut:

1. Bagaimana mengimplementasikan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan?
2. Bagaimana hasil pengujian dan analisis metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan?

1.3 Tujuan

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan.
2. Menguji hasil prediksi dengan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) menggunakan RMSE (*Root Mean Square Error*).

1.4 Manfaat

Manfaat dari penelitian ini adalah:

1. Dapat membantu petani dalam memprediksi hasil produksi pertanian, karena curah hujan berpengaruh dalam proses pengairan lahan pertanian.
2. Dapat membantu pemerintah dalam mengantisipasi persediaan sumber air disaat kemarau panjang.
3. Dapat membantu bidang penerbangan dalam memprediksi curah hujan untuk keselamatan penerbangan.

1.5 Batasan masalah

Untuk merumuskan permasalahan yang lebih terfokus maka dibuat batasan-batasan yang ditentukan pada penelitian, antara lain:

1. Data curah hujan yang digunakan berasal dari BMKG Karang Ploso Kabupaten Malang.
2. Data curah hujan yang digunakan terdiri dari beberapa indikator diantaranya suhu udara, kelembapan udara, dan kecepatan angin.

1.6 Sistematika pembahasan

Sistematika penulisan yang disusun dalam penelitian ini adalah sebagai berikut:

BAB 1 PENDAHULUAN

Menjelaskan tentang latar belakang dilakukannya penilitian prediksi curah hujan menggunakan metode *Adaptive Neuro Fuzzy Inference System*

(ANFIS), identifikasi masalah, rumusan masalah, tujuan penelitian, manfaat penilitian, batasan masalah, serta sistematika dari penulisan.

BAB 2 DASAR TEORI

Menjelaskan dasar teori dan referensi apa saja yang dibutuhkan dalam pemahaman permasalahan yang dibahas dalam pembuatan tugas akhir. Teori-teori yang terdapat dalam bab ini mencakup prediksi curah hujan, metode *Adaptive Neuro Fuzzy Inference System* (ANFIS), Jaringan Saraf Tiruan, dan Logika Fuzzy.

BAB 3 METODOLOGI

Menjelaskan langkah-langkah yang dilakukan dalam penelitian dan meliputi studi literatur terkait prediksi curah hujan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS), pengumpulan data, analisa kebutuhan sistem, perancangan sistem, implementasi, pengujian dan pengambilan kesimpulan.

BAB 4 PERANCANGAN

Menjelaskan perancangan user interface untuk mengembangkan perangkat lunak prediksi curah hujan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS). Selain itu pada bab ini juga menjelaskan proses implementasi *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan, algoritme dan diagram alir sistem yang akan diimplementasikan.

BAB 5 IMPLEMENTASI

Menjelaskan proses implementasi yang dilakukan yaitu implementasi menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan.

BAB 6 PENGUJIAN

Menjelaskan cara pengujian serta menganalisis tingkat akurasi hasil pada prediksi curah hujan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS).

BAB 7 PENUTUP

Berisi kesimpulan dan saran dari keseluruhan laporan. Bagian Saran berisi kritik dan saran untuk pengembangan selanjutnya dan bagian kesimpulan berisi hasil kesimpulan yang didapat dari proses penelitian ini.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisi landasan kepustakaan yang meliputi kajian pustaka dan dasar teori yang diperlukan untuk penelitian ini. Kajian pustaka membahas penelitian yang telah ada dan diusulkan. Dasar teori membahas tentang teori yang diperlukan untuk menyusun penelitian yang diusulkan.

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini akan membahas tentang beberapa penelitian yang sebelumnya telah dilakukan. Beberapa penelitian tersebut akan digunakan peneliti untuk mendukung penelitian ini. Referensi pertama adalah Prediksi curah hujan pada daerah Tengger menggunakan metode *hybrid Tsukamoto FIS* dan Algoritme Genetika. Pada penelitian ini menggunakan data dasarian curah hujan sebagai data masukan. Algoritme genetika pada penelitian ini dilakukan untuk membentuk fungsi keanggotaan. Dari hasil pengujian yang dilakukan didapatkan hasil RMSE sebesar 6,78 untuk wilayah Tosari dan nilai RMSE sebesar 6,63 untuk wilayah Tutur (Wahyuni & Mahmudy, 2017).

Penelitian kedua yaitu memprediksi beban listrik dengan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS). Untuk memprediksi beban listrik dapat dilakukan dengan jangka pendek, jangka menengah dan jangka panjang. Penelitian ini menggunakan data beban listrik, suhu dan populasi untuk data masukan. Fungsi keanggotaan fuzzy yang digunakan pada penelitian ini adalah funggsi keanggotaan *generalized bell*. Hasil penelitian yaitu prediksi beban listrik dengan nilai RMSE sebesar 0,0298 (Santika, Mahmudy, & Naba, 2017).

Penelitian ketiga yaitu memprediksi curah hujan menggunakan metode *Hybrid Adaptive Neuro Fuzzy Inference System* (ANFIS) dan Algoritme Genetika. Pada penelitian ini digunakan data masukan curah hujan dari 1 dasarian sebelumnya dan 2 dasarian sebelumnya. Algoritme genetika pada penelitian ini digunakan untuk melakukan optimasi nilai prediksi curah hujan. Dari hasil pengujian yang dilakukan didapatkan hasil RMSE sebesar 5,41 untuk wilayah Sumber dan nilai RMSE sebesar 5,87 untuk wilayah Puspo (Wahyuni, Mahmudy, & Iriany, 2017).

2.2 Curah Hujan

Curah hujan merupakan salah satu komponen penting dalam kehidupan sehari-hari. Seperti yang sudah disampaikan dalam latar belakang bahwa curah hujan digunakan dalam berbagai sektor, salah satunya adalah sektor pertanian. Menurut Kamus Besar Bahasa Indonesia (KBBI) curah hujan adalah banyaknya hujan yang tercurah atau turun di suatu daerah dalam jangka waktu tertentu atau bisa juga disebut endapan/deposit air dalam bentuk cair maupun padat yang berasal dari atmosfer. Hal ini berarti curahan mencakup tetes hujan, salju, batu es, embun, dan embun kristal.

Derajat curah hujan dinyatakan dengan jumlah curah hujan dalam suatu satuan waktu tertentu, misalnya mm/jam. Dalam dunia meteorologi, butiran hujan yang mempunyai diameter antara 0.5-0.1 mm disebut gerimis dan butiran hujan yang mempunyai diameter lebih dari 0.5 mm disebut hujan. Ukuran butiran hujan berbanding lurus dengan kecepatan jatuhnya hujan, semakin besar ukuran butiran hujan maka semakin besar pula kecepatan jatuhnya, begitu juga sebaliknya.

Ketelitian alat ukur curah hujan yang sudah ada saat ini adalah 1/10 mm. banyaknya curah hujan dinyatakan dengan satuan milimeter. Di beberapa negara banyaknya curah hujan masih dinyatakan dengan inci. Biasanya pembacaan dilakukan satu kali dalam sehari dan dicatat sebagai curah hujan hari terdahulu atau kemarin. Data curah hujan yang terkumpul nantinya dapat digunakan sebagai acuan untuk prediksi curah hujan pada kemudian hari (Siswanti, 2011).

2.2.1 Faktor-Faktor yang Mempengaruhi Curah Hujan

Teori tentang curah hujan dan faktor-faktor yang mempengaruhi curah hujan telah dibahas oleh Siswanti (2011) dan Wilson (1990). Faktor-faktor tersebut adalah:

a. Kelembapan Udara

   Kelembapan adalah perbandingan antara massa uap dalam suatu satuan volum dengan massa uap yang jenuh dalam satuan volum itu pada suhu yang sama. Secara umum kelembapan menyatakan banyaknya kadar air yang ada di udara. Banyaknya uap yang bergerak di dalam atmosfer berpengaruh terhadap lamanya hujan, besarnya hujan, dan intensitas curah hujan. Kelembapan tertinggi umumnya terjadi pada musim penghujan dan paling rendah pada musim kemarau. Variasi kelembapan bergantung dari suhu udara, jika siang hari dan suhu tinggi maka kelembapan akan lebih rendah jika dibandingkan pada pagi hari saat suhu rendah. Umumnya semakin tinggi suatu daerah dari permukaan laut maka kelembapan udaranya semakin tinggi. Semakin tinggi kelembapan udara akan dapat menyebabkan bertambah banyak uap air yang dapat diserap awan. Uap air itu akan menghasilkan tekanan yang dinyatakan dengan satuan tinggi air raksa (1 mmHg = 1,33 milibar). Tekanan yang diberikan oleh uap air disebut dengan tekanan uap air (Siswanti & Wutsqa, 2011).

b. Suhu Udara

   Suhu udara adalah keadaan panas atau dinginnya udara. Suhu juga disebut temperatur yang diukur dengan alat termometer. Beberapa faktor yang mempengaruhi suhu udara yaitu tinggi suatu tempat, daratan/lautan, radiasi matahari, indeks datang matahari dan angin. Pengukuran biasa dinyatakan dalam skala Celcius (C), Reamur (R), dan Fahrenheit (F). Suhu udara tertinggi di permukaan bumi adalah di daerah tropis atau daerah sekitar garis ekuator. Sehingga, semakin ke arah kutub maka suhu semakin dingin (Wilson, 1990).

c. Kecepatan angin

Angin adalah udara yang bergerak akibat adanya perbedaan tekanan udara dengan arah aliran angin dari tempat yang memiliki tekanan tinggi ke tempat yang bertekanan rendah atau dari daerah yang memiliki suhu atau *temperature* rendah ke wilayah bersuhu tinggi. Angin dapat disebabkan oleh pergerakan benda sehingga mendorong udara disekitarnya untuk bergerak ke tempat lain. Angin juga memiliki hubungan yang erat dengan sinar matahari karena daerah yang terkena banyak paparan sinar matahari akan memiliki suhu yang lebih tinggi serta tekanan udara yang lebih rendah dari daerah lain di sekitarnya sehingga menyebabkan terjadinya aliran udara (Siswanti & Wutsqa, 2011).

d. Tekanan Udara

Tekanan udara merupakan tenaga yang bekerja untuk menggerakkan massa udara dalam setiap satuan luas tertentu. Tekanan udara diukur dengan menggunakan alat bernama barometer. Satuan tekanan udara adalah milibar (mb). Garis yang menghubungkan tempat-tempat yang sama tekanan udaranya disebut juga isobar. Tekanan udaara dibatasi oleh ruang dan waktu. Artinya pada tempat dan waktu yang berbeda, besarnya juga berbeda. Semakin tinggi suatu tempat maka tekanan udaranya semakin menurun, sedangkan tekanan udara pada daerah yang mempunyai rata-rata ketinggian sama maka tekanan udara udara dipengaruhi oleh suhu udara. Daerah yang bersuhu udara rendah tekanannya tinggi dan daerah yang suhu udaranya tinggi mempunyai tekanan rendah (Siswanti & Wutsqa, 2011).

e. Lama Penyinaran Matahari

Lama Penyinaran matahari adalah lamanya waktu pancaran terang atau cahaya yang dipancarkan oleh matahari ke permukaan bumi. Intensitas atau lama pencahayaan matahari sangat berpengaruh dengan suhu dan kelembapan yang ada di sekitarnya, dengan kata lain lama penyinaran matahari juga berpengaruh terhadap intensitas curah hujan yang terjadi. Ada banyak alat ukut yang digunakan untuk mengukur lama penyinaran matahari, salah satunya adalah *Campbell Stokes Recorder*. Alat tersebut merupakan alat pengukur lama penyinaran matahari yang secara resmi digunakan oleh Badan Meteorologi, Klimatologi, dan Geofisika (BMKG).

2.3 Jaringan Saraf Tiruan

Jaringan Saraf Tiruan adalah paradigma pemprosesan suatu informasi yang terinspirasi oleh sistim sel saraf biologi, sama seperti otak yang memproses suatu informasi. Elemen mendasar dari paradigma tersebut adalah struktur yang baru dari sistem pemprosesan informasi. Jaringan Saraf Tiruan, seperti manusia, belajar dari suatu contoh. Jaringan ini dibentuk untuk memecahkan suatu masalah tertentu seperti pengenalan pola atau klasifikasi karena proses pembelajaran.

Jaringan saraf tiruan atau *artificial neural network* adalah sistem pengolah informasi yang memiliki karakter seperti jaringan saraf biologis, yaitu jaringan otak manusia. Pada jaringan saraf tiruan terdapat istilah *neuron* atau lebih dikenal dengan *node*. Setiap *neuron* terhubung dengan *neuron* lainnya melalui *layer* dengan bobot tertentu. Bobot melambangkan informasi yang digunakan jaringan untuk menyelesaikan permasalahan. Setiap *neuron* memiliki *internal state* yang disebut dengan fungsi aktivasi. Fungsi aktivasi merupakan fungsi dari input yang diterima *neuron*. Satu *neuron* akan mengirimkan sinyal ke *neuron* yang lain (Kuncahyo, Ginardi, & Arieshanti, 2012).

Jaringan Saraf Tiruan (JST) ditentukan oleh tiga hal, yaitu (Darmawan, 2009):

1. Pola hubungan antar *neuron* (disebut arsitektur jaringan)
2. Metode untuk menentukan bobot penghubung (disebut metode pembelajaran atau *training*)
3. Fungsi aktivasi.

2.3.1 Pemodelan Jaringan Saraf Tiruan (JST)

Secara umum, proses pemodelan JST terbagi menjadi dua bagian, yaitu proses *training* dan proses *testing*. Proses *training* merupakan proses pembelajaran dari sistem jaringan saraf yang mengatur nilai *input* serta bagaimana pemetaannya pada *output* sampai diperoleh model yang sesuai. Proses *training* terjadi pada saat pengaturan bobot dan bias. Sedangkan proses *testing* atau pengujian adalah proses menguji ketelitian dari model yang sudah diperoleh dari proses *training*. Untuk mengoptimalkan pembagian data, sebaiknya pembagian data dibagi menjadi 80% untuk data *training* dan 20% untuk data *testing* (Yao & Tan, 2000).

Dalam setiap lapisan, nilai *input* ditransformasi ke dalam lapisan secara nonlinear oleh elemen-elemen proses dan kemudian diproses maju ke lapisan berikutnya. Akhirnya, nilai-nilai *output* $\hat{y}$, yang dapat berupa nilai-nilai skalar atau vector, dihitung pada lapisan *output* dengan persamaan berikut.

$$\hat{y}(k) = f^o\left[\left[\sum_{j=1}^{q} v_j^o f_j^h\left(\sum_{j=1}^{p} w_{j,i}^h x_{i(k)} + b_j^h\right) + b^o\right]\right] \qquad (2.1)$$

Keterangan:

$x_{i(k)}$ = Perubahan *input* ke-*i*, (*i* = 1, 2, …, p) dimana p merupakan jumlah *input*

$\hat{y}(k)$ = Nilai dugaan dari peubah *output*

k = Indeks pasangan data *input*-target $(x_{i(k)}, \hat{y}(k))$,

k = 1, 2, …, n. dimana n merupakan pola

$w_{j,i}^h$ = Bobot dari *input* ke-*I* yang menuju *neuron* ke-*j* pada lapisan tersembunyi

b_j^h = Bias pada *neuron* ke-*j* pada lapisan tersembunyi,

(*j = 1, 2, …, q*) dimana *q* merupakan jumlah *node* lapisan tersembunyi

f_j^h = Fungsi aktivasi di *neuron* ke-*j* pada lapisan tersembunyi

v_j^o = Bobot dari *neuron* ke-*j* dilapisan tersembunyi menuju *neuron output*

b^o = Bias pada *neuron* di lapisan *output*

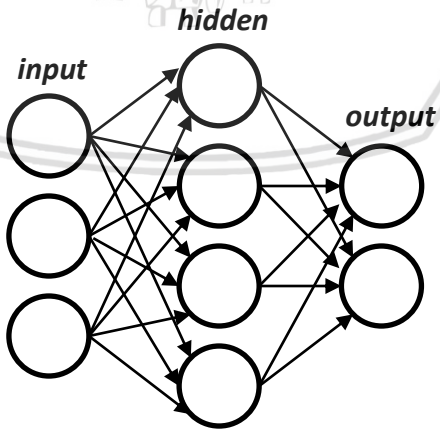
f^o = Fungsi aktivasi pada *neuron* di lapisan *output*.

2.3.2 Arsitektur Jaringan Saraf Tiruan (JST)

JST memiliki 3 lapisan utama, yaitu: lapisan *input* yang terhubung dengan lapisan tersembunyi yang selanjutnya terhubung dengan lapisan *output*.

1. Aktivasi unit-unit lapisan *input* menunjukkan informasi dasar yang kemudian digunakan dalam JST.
2. Aktivasi setiap unit-unit lapisan tersembunyi ditentukan oleh aktivitas dari *input* dan bobot dari koneksi antara unit-unit *input* dan unit-unit lapisan tersembunyi.
3. Karakterisktik dari unit-unit *output* tergantung dari aktivitas unit-unit lapisan tersembunyi dan bobot antara unit-unit lapisan tersembunyi dan unit-unit *output*.

Jaringan dengan banyak lapisan memiliki satu atau lebih lapisan yang terletak di antara lapisan *input* dan lapisan *output*, seperti terlihat pada Gambar 2.1. Umumnya, ada lapisan bobot-bobot yang terletak antara dua lapisan yang bersebelahan. Jaringan dengan banyak lapisan ini dapat menyelesaikan permasalahan yang lebih sulit dibandingkan dengan lapisan tunggal, tentu saja dengan pembelajaran yang lebih rumit. Namun demikian, pada banyak kasus, pembelajaran pada jaringan dengan banyak lapisan ini lebih sukses dalam menyelesaikan masalah (Suhartono, 2007).



Gambar 2.1 Arsitektur Jaringan Saraf Tiruan Dengan Banyak Lapisan

Sumber: (Kusumadewi & Hartati, Neuro Fuzzy Integrasi Sistem Fuzzy & Jaringan Saraf, 2010)

2.3.3 Proses Pembelajaran

Umumnya, jika menggunakan Jaringan Saraf Tiruan, hubungan antara *input* dan *output* harus diketahui secara pasti dan jika hubungan tersebut telah diketahui maka dapat dibuat suatu model. Hal lain yang penting adalah proses belajar hubungan *input*/*output* dilakukan dengan pembelajaran. Ada dua tipe pembelajaran yang dikenal yaitu pembelajaran terawasi dan pembelajaran tidak terawasi.

a. Pembelajaran terawasi (*supervised learning*)
   Metode pembelajaran pada jaringan saraf disebut terawasi jika *output* yang diharapkan telah diketahui sebelumnya.
   Contoh:
   Misalkan terdapat operasi OR sebagai jaringan saraf yang digunakan untuk mengenali pola:

0	0	0
0	1	1
1	0	1
1	1	1

   Pada proses pembelajaran, satu pola *input* akan diberikan ke satu *neuron* pada lapisan *input*. Pola ini akan dirambatkan disepanjang jaringan saraf hingga sampai ke *neuron* pada lapisan *output*. Lapisan *output* ini akan membangkitkan pola *output* yang nantinya akan dicocokkan dengan pola *output* targetnya. Apabila terjadi perbedaan antara pola *output* hasil pembelajaran dengan pola target, maka disini akan muncul *error*. Apabila nilai *error* ini masih cukup besar, mengindikasikan bahwa masih perlu dilakukan lebih banyak pembelajaran lagi.

b. Pembelajaran tak terawasi (*unsupervised learning*)
   Pada metode pembelajaran tak terawasi ini tidak memerlukan target *output*. Pada metode ini, tidak dapat ditentukan hasil yang seperti apakah yang diharapkan selama proses pembelajaran. Selama proses pembelajaran, nilai bobot disusun dalam suatu *range* tertentu tergantung pada nilai *input* yang diberikan. Tujuan pembelajaran ini adalah mengelompokkan unit-unit yang hampir sama dalam suatu area tertentu. Pembelajaran ini biasanya sangat cocok untuk pengelompokan (*clustering*) pola.

2.4 Logika Fuzzy

Logika fuzzy pertama kali dikenalkan oleh Zadeh pada tahun 1965. Dasar logika fuzzy adalah teori himpunan fuzzy. Pada teori himpunan fuzzy, peranan derajat keanggotaan sebagai penentu keberadaan elemen dalam suatu himpunan sangatlah penting. Pada himpunan tegas (*crisp*), nilai keanggotaan hanya terdapat dua kemungkinan, yaitu 0 dan 1. Pada himpunan fuzzy, nilai keanggotaan terletak pada rentang 0 sampai 1, dimana 0 adalah nilai minimum dan 1 adalah nilai maksimum. Apabila x memiliki nilai keanggotaan fuzzy $\mu A(x) = 0$, berarti x tidak menjadi anggota himpunan A, apabila x memiliki nilai keanggotaan fuzzy $\mu A(x) = 1$, berarti x menjadi anggota penuh pada himpunan A.

Dalam hal yang lain, logika fuzzy digunakan sebagai suatu cara untuk menetapkan permasalahan dari input menuju output yang diharapkan. Logika fuzzy dapat dianggap sebagai kotak hitam (*black box*) yang menghubungkan antara ruang input menuju ke ruang output (Kamsyakawuni, 2012).

2.4.1 Himpunan Fuzzy

Himpunan *fuzzy* memiliki dua atribut, yaitu linguistik dan numerik. Atribut linguistik adalah atribut yang digunakan untuk penamaan suatu *group* yang mewakili suatu keadaan atau kondisi tertentu dengan menggunakan bahasa alami, seperti muda, parubaya, tua. Sedangkan atribut numeris adalah suatu nilai yang menunjukkan ukuran dari suatu variable. Beberapa hal yang perlu diketahui dalam memahami sistem *fuzzy* yaitu:

1. Variabel *fuzzy*
   Variabel *fuzzy* merupakan suatu lambang atau kata yang menunjuk kepada suatu yang tidak tertentu dalam sistem *fuzzy*.
2. Himpunan *fuzzy*
   Himpunan *fuzzy* merupakan suatu *group* yang mewakili suatu kondisi atau keadaan tertentu dalam suatu variable *fuzzy*.
3. Semesta pembicaraan
   Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variable *fuzzy*. Semesta pembicaraan merupakan himpunan bilangan *real* yang senantiasa bertambah secara tetap. Nilai semesta pembicaraan dapat berupa bilangan positif ataupun negatif.
4. Domain
   Domain himpunan *fuzzy* adalah keseluruhan nilai yang diijiknkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *fuzzy*. Seperti halnya semesta pembicaraan, domain merupakan himpunan bilangan *real* yang senantiasa bertambah secara tetap. Nilai domain dapat berupa bilangan positif ataupun negatif.

2.4.2 Fungsi Keanggotan *Generallized Bell*

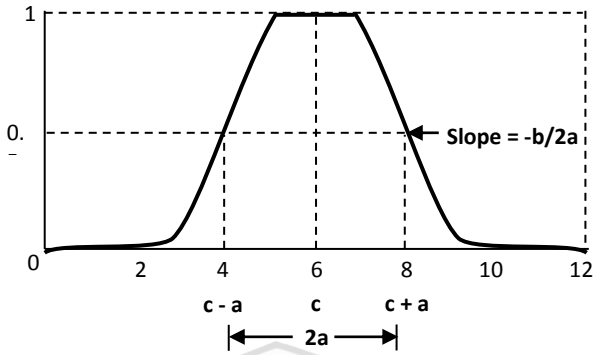
Fungsi keanggotaan atau *membership function* (MF) merupakan suatu kurva yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaan yang memiliki nilai interval antara 0 dan 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi (Kuncahyo, Ginardi, & Arieshanti, 2012). Fungsi yang digunakan adalah fungsi keanggotaan *generalized bell*. Fungsi keanggotaan *generalized bell* merupakan fungsi yang *smooth* dan non linear serta memiliki parameter *{a,b,c}* yang didefinisikan seperti pada Persamaan 2.2.

$$bell(x; a, b, c) = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}} \quad (2.2)$$

Keterangan:
x = data input
a = standar deviasi
b = 1 (*slopes*)

c = rata-rata

Parameter b selalu positif, agar kurva menghadap ke bawah, seperti yang terlihat pada gambar berikut:

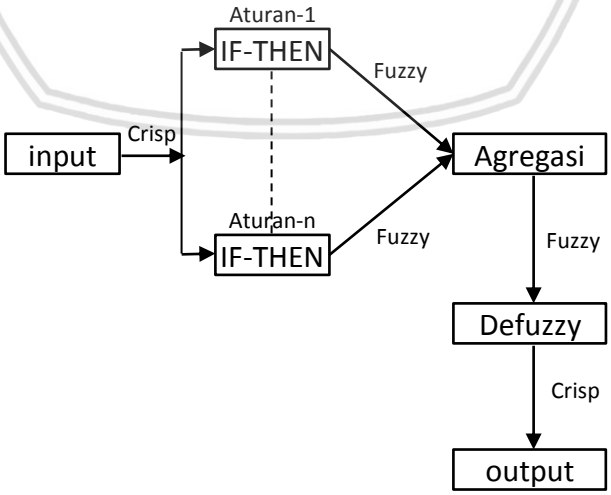


Gambar 2.2 Fungsi Keanggotaan *Generalized Bell*

Sumber : (Mancini, Lioi, Caniani, & Masi, 2012)

2.5 Sistem Inferensi Fuzzy

Sistem Inferensi *Fuzzy* (*Fuzzy Inference System*) atau disebut juga dengan *Fuzzy Inference Engine* adalah sistem yang dapat melakukan penalaran hampir serupa dengan manusia. Tiga komponen konsep *fuzzy inference system* yaitu baris aturan (mengandung seleksi dari aturan-aturan *fuzzy*), basis data dan mekanisme penalaran. Sistem Inferensi *fuzzy* pada dasarnya mendefinisikan pemetaan non linear dari vector *input* menjadi skalar *output*. Proses pemetaan melibatkan *input-output* fungsi keanggotaan, operator-operator *fuzzy*, aturan *fuzzy IF-THEN*, agregasi dari himpunan *output* dan defuzifikasi. Model umum dari sistem inferensi *fuzzy* ditunjukkan oleh Gambar 2.3.



Gambar 2.3 Diagram Blok Sistem Inferensi *Fuzzy*

Sumber: (Kusumadewi & Hartati, Neuro Fuzzy Integrasi Sistem Fuzzy & Jaringan Saraf, 2010)

Terdapat 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.

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*).

Metode yang terakhir adalah metode *sugeno*. Pada penelitian ini digunakan metode *sugeno* orde-satu. Penalaran metode *sugeno* hampir sama dengan penalaran *mamdani*, hanya saja *output* 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 (Kusumadewi & Purnomo, 2010).

2.5.1 Metode Takagi Sugeno Kang (TSK)

Metode TSK adalah salah satu metode dalam sistem inferensi *fuzzy*. Dalam metode TSK ini terdiri atas 2 jenis, yaitu (Kusumadewi & Purnomo, 2010):

1. Model *fuzzy* Sugeno orde-nol

   Secara umum bentuk model *fuzzy* sugeno orde-nol adalah:

   $$IF(x_1\ is\ A_1) \circ (x_2\ is\ A_2) \circ (X_3\ is\ A_3) \circ \ldots \circ (x_n\ is\ A_n)\ THEN\ z = k \quad (2.3)$$

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

2. Model *fuzzy* Sugeno orde-satu

   Secara umum bentuk model *fuzzy* sugeno orde-satu adalah:

   $$IF(x_1\ is\ A_1) \circ \ldots \circ (x_n\ is\ A_n)\ THEN\ z = p_1 * x_1 + \ldots + p_n * x_n + q \quad (2.4)$$

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

2.6 Algoritme K-Means

K-Means adalah suatu metode penganalisisan data yang melakukan proses pemodelan tanpa *supervise* (*unsupervised*) dan merupakan salah satu metode yang melakukan pengelompokan data dengan sistem partisi. Metode *k-means* berusaha mengelompokkan data yang ada ke dalam beberapa kelompok, dimana data dalam satu kelompok mempunyai karakteristik yang sama antara satu dengan lainnya dan mempunyai karakteristik yang berbeda dengan data yang berada di dalam kelompok lain. Dengan kata lain, metode ini berusaha untuk meminimalkan variasi antar data yang ada di dalam suatu *cluster* dan memaksimalkan variasi dengan data yang ada pada *cluster* lainnya.

K-Means diperkenalkan oleh J. B. MacQueen pada tahun 1976, yang merupakan salah satu algoritme *clustering* yang sangat umum untuk mengelompokkan data sesuai dengan karakteristik atau ciri-ciri bersama yang serupa. *Group* data ini disebut sebagai *cluster*. Di dalam suatu *cluster* mempunyai ciri-ciri (atau fitur, karakteristik, atribut, properti) serupa dan tidak serupa dengan data *cluster* lain (Husnita, 2012).

Berikut merupakan langkah-langkah dalam melakukan pengelompokan data menggunakan metode *k-means*:

1. Standarisasi/normalisasi data yang dikelompokkan (menentukan bobot dari data mentah yang telah didapatkan). Hal ini digunakan agar data mempunyai skala yang sama, sehingga pengelompokan akan stabil. Normalisasi data dilakukan dengan menggunakan persamaan seperti pada Persamaan 2.5.

$$x'_{ij} = \frac{x_{ij} - xMin_j}{xMax_j - xMin_j} \quad (2.5)$$

   Keterangan:

   x_{ij}' = data ke-*i* parameter ke-*j* yang telah distandarisasi

   $xMin_j$ = nilai min pada parameter ke-*j*

   $xMax_j$ = nilai max pada parameter ke-*j*

   x_{ij} = data ke-*I* parameter ke-*j*

2. Melakukan pengelompokan dengan metode *k-means clustering*, dengan langkah-langkah sebagai berikut:

   a. Tentukan jumlah *cluster* (*k*) dan nilai *threshold*.

   b. Inisialisasi pusat *cluster* dapat dilakukan dengan berbagai cara, yang paling sering dilakukan adalah dengan cara *random*. Pusat-pusat *cluster* diberi nilai awal dengan angka-angka *random*.

   c. Tempatkan setiap data/objek ke *cluster* terdekat. Kedekatan dua objek ditentukan jarak antara data dengan pusat *cluster*. Dalam tahap ini perlu dihitung jarak tiap data ke tiap pusat *cluster*. Jarak paling dekat antara satu data dengan data satu *cluster* tertentu akan menentukan suatu data masuk ke dalam *cluster* yang mana. Menentukan ukuran kemiripan atau

ketidakmiripan antar data dengan metode jarak *Euclidean* seperti pada Persamaan 2.6:

$$d(x,y) = \|x-y\|^2 = \sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2} \quad (2.6)$$

Keterangan:

d(x,y) = ukuran ketidakmiripan

X = (x_1, x_2, …, x_j) adalah *variable* data

Y = (y_1, y_2, …, y_j) adalah pada titik pusat *cluster*

3. Hitung kembali pusat *cluster* dengan keanggotaan *cluster* saat ini. Pusat *cluster* adalah rata-rata dari semua data dalam *cluster* tertentu. Persamaan 2.7 merupakan persamaan yang digunakan untuk pencarian nilai rata-rata.

$$C_i X_j^{(t+1)} = \frac{x_1+x_2+x_3+\cdots+x_n+C_i X_j^{(t)}}{n+1} \quad (2.7)$$

Keterangan:

C_i = cluster ke-i

x_j = variabel ke-j

t = iterasi

x_n = data inputan ke-n pada *cluster* tertentu

n = jumlah data pada *cluster* tertentu

4. Tugaskan lagi setiap objek dengan memakai pusat *cluster* yang baru. Jika pusat *cluster* sudah tidak berubah lagi, maka proses pengelompokan *cluster* selesai. Jika pusat *cluster* masih berubah, kembali lagi ke langkah ketiga sampai pusat *cluster* tidak berubah.

5. Setelah selesai, maka didapatkanlah data yang telah dikelompokan berdasarkan berdasarkan *cluster*, di mana data tersebut akan dimasukkan untuk pencarian *mean* dan standar deviasi pada Persamaan 2.8 dan Persamaan 2.9.

$$\tilde{x} = \frac{x_1 + x_2 + \ldots + x_n}{n} \quad (2.8)$$

$$\sigma = \sqrt{\frac{\sum(x_i - \tilde{x})^2}{(n-1)}} \quad (2.9)$$

Keterangan:

$\tilde{x}$ = rata-rata (*mean*) setiap *cluster*

σ = standar deviasi setiap *cluster*

x_i = data input ke-*I* setiap *cluster*

n = banyaknya data setiap *cluster*

2.7 *Adaptive Neuro Fuzzy Inference System* (ANFIS)

Adaptive Neuro Fuzzy Inference System (ANFIS) adalah jaringan yang berbasis pada *system inference fuzzy*. Parameter ANFIS dapat dipisahkan menjadi dua, yaitu parameter premis dan konsekuen yang dapat diadaptasikan dengan pelatihan *hybrid*. Pelatihan *hybrid* dilakukan dalam dua langkah, yaitu langkah maju dan langkah mundur.

ANFIS merupakan arsitektur yang secara fungsional sama dengan *fuzzy rule base* model Sugeno dan juga sama dengan jaringan saraf tiruan fungsi radial dengan sedikit batasan tertentu (Kusumadewi S. , 2006). ANFIS adalah penggabungan mekanisme *fuzzy inference system* yang digambarkan dalam arsitektur jaringan saraf. Sistem inferensi yang digunakan adalah sistem inferensi *fuzzy* model Takagi-Sugeno-Kang (TSK) orde satu dengan pertimbangan kesederhanaan dan kemudahan komputasi.

Arsitektur ANFIS juga sama dengan jaringan saraf dengan fungsi radial dengan batasan tertentu. Agar jaringan dengan fungsi basis radial ekuivalen dengan *fuzzy* berbasis aturan model Sugeno orde satu, maka diperlukan batasan:

1. Keduanya harus memiliki metode agregasi yang sama (rata-rata terbobot atau penjumlahan terbobot) untuk menurunkan semua *output*nya.
2. Jumlah fungsi aktivasi harus sama dengan jumlah aturan *fuzzy* (*IF-THEN*).
3. Jika ada beberapa *input* pada basis aturannya, maka tiap-tiap fungsi aktivasi harus sama dengan fungsi keanggotaan tiap-tiap inputnya.
4. Fungsi aktivasi dan aturan-aturan *fuzzy* harus memiliki fungsi yang sama untuk *neuron-neuron* dan aturan-aturan yang ada di sisi outputnya.

2.7.1 Struktur ANFIS

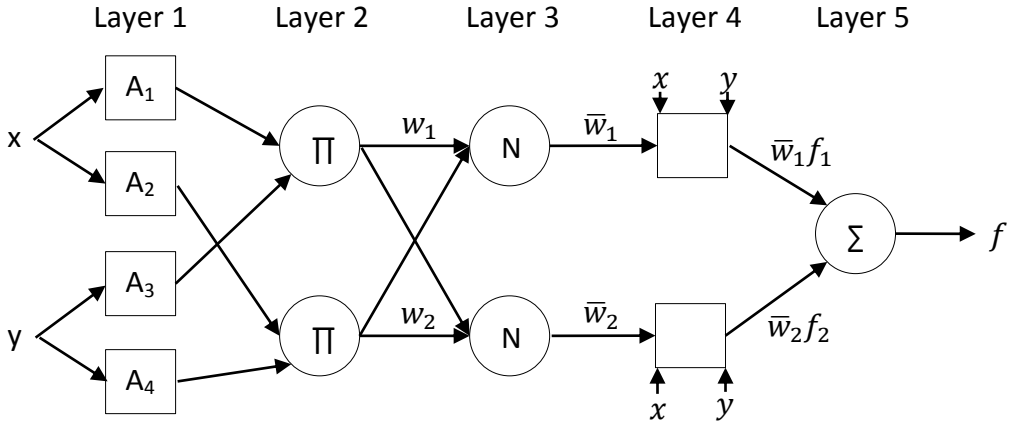
Struktur ANFIS yang menggambarkan sistem inferensi *fuzzy* Takagi-Sugeno-Kang (TSK) terdapat pada Gambar 2.4. Misalkan terdapat 2 *input* x_1, x_2 dan 1 *output* y. Ada 2 aturan pada basis aturan Sugeno seperti pada Persamaan 2.10.

$$IF\ x_1\ is\ A_1\ AND\ x_2\ is\ B_1\ THEN\ y_1 = c_{11}x_1 + c_{12}x_2 + c_{10}$$
$$IF\ x_1\ is\ A_2\ AND\ x_2\ is\ B_2\ THEN\ y_2 = c_{21}x_1 + c_{22}x_2 + c_{20} \qquad (2.10)$$

Keterangan:

y_i = *output*

$c_{i,j}$ = parameter konsekuen



Gambar 2.4 Arsitektur ANFIS

Sumber: (Suyanto, 2008)

Berikut adalah penjelasan untuk tiap-tiap *layer* atau lapisan pada arsitektur jaringan ANFIS:

1. Lapisan 1

   Semua simpul pada lapisan ini adalah simpul adaptif (parameter dapat berubah) dengan fungsi simpul pada Persamaan 2.11.

   $$O_{1,i} = \mu A_i(x), untuk\ i = 1,2, atau$$
   $$O_{1,i} = \mu B_{i-2}(y), untuk\ i = 3,4 \quad (2.11)$$

   Dengan x dan y adalah masukan pada simpul i, A_i atau (B_{i-2}) adalah fungsi keanggotaan masing-masing simpul. Simpul $O_{1,i}$ berfungsi untuk menyatakan derajat keanggotaan tiap masukan terhadap himpunan *fuzzy* A dan B. Fungsi keanggotaan yang digunakan adalah jenis *generalized bell* (gbell). Parameter a, b, c, pada fungsi keanggotaan gbell dinamakan parameter yang adaptif.

2. Lapisan 2

   Semua simpul pada lapisan ini adalah nonadaptif (parameter tetap). Fungsi simpul ini adalah mengalikan setiap sinyal masukan yang datang. Fungsi simpul seperti pada Persamaan 2.12.

   $$O_{2,i} = w_i = \mu A_i(x)\mu B_i(y), i = 1,2 \quad (2.12)$$

   Tiap keluaran simpul menyatakan derajat pengaktifan (*firing strength*) tiap aturan *fuzzy*. Fungsi ini dapat diperluas apabila bagian premis memiliki lebih dari dua himpunan *fuzzy*. Banyaknya simpul pada lapisan ini menunjukkan banyaknya aturan yang dibentuk.

3. Lapisan 3

   Setiap simpul pada lapisan ini adalah simpul nonadaptif yang menampilkan fungsi derajat pengaktifan ternormalisasi (*normalized firing strength*) yaitu rasio keluaran simpul ke-I pada lapisan sebelumnya terhadap seluruh

keluaran lapisan sebelumnya, dengan bentuk fungsi simpul seperti pada Persamaan 2.13.

$$O_{3,i} = \overline{w} = \frac{w_i}{w_1+w_2}, i = 1,2 \quad (2.13)$$

Apabila dibentuk lebih dari dua aturan, fungsi dapat diperluas dengan membagi w_i dengan jumlah total w untuk semua aturan.

4. Lapisan 4

Setiap simpul pada lapisan ini adalah simpul adaptif dengan fungsi simpul seperti pada Persamaan 2.14.

$$O_{4,i} = \overline{w_i} f_i = \overline{w_i}(c_{11}x_1 + c_{i2}x_2 + c_{i0}) \quad (2.14)$$

Dengan $\overline{w_i}$ adalah derajat pengaktifan ternormalisasi dari lapisan 3 dan parameter c_{i1}, c_{i2}, c_{i0} menyatakan parameter konsekuen yang adaptif.

5. Lapisan 5

Pada lapisan ini hanya ada satu simpul tetap yang fungsinya untuk menjumlahkan semua masukan. Fungsi simpul seperti pada Persamaan 2.15.

$$O_{5,i} = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (2.15)$$

Jaringan adaptif dengan lima lapisan tersebut ekivalen dengan sistem inferensi *fuzzy* TSK.

2.7.2 *Least Square Estimator* (LSE)

Jika diketahui keluaran dari model linear y yang diekspresikan melalui Persamaan 2.16.

$$y = \theta_1 f_1(u) + \theta_2 f_2(u) + \ldots + \theta_n f_n(u) \quad (2.16)$$

Keterangan:

y = $[y_1, y_2, \ldots, y_m]^T$ model vektor *output*

u = $[u_1, u_2, \ldots, u_p]^T$ model vektor *input*

f_n = fungsi u yang diketahui

θ_n = parameter yang diestimasi

Dengan menggunakan notasi matriks didapatkan:

$$A\theta = y \quad (2.17)$$

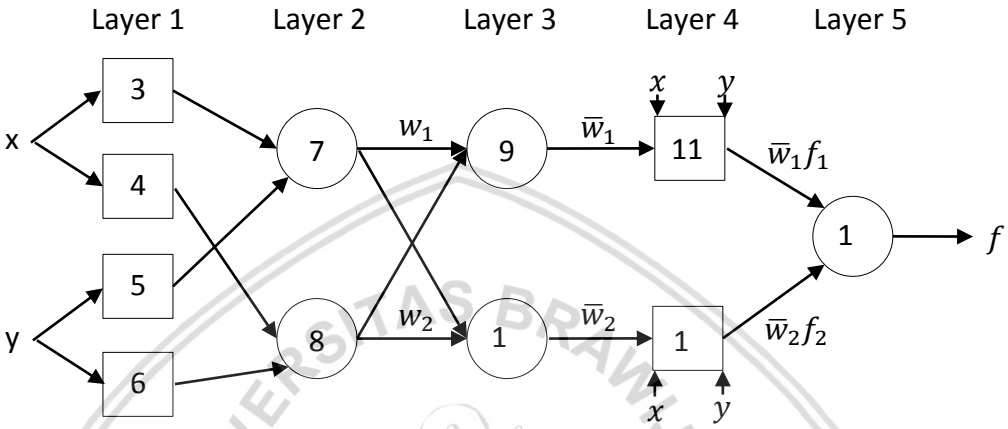
Penyelesaian terbaik untuk θ, yang meminimalkan $\|A\theta - y\|^2$ adalah *least square estimator* (LSE) θ^*:

$$\theta^* = (A^T A)^{-1} A^T y \quad (2.18)$$

Di mana A^T adalah bentuk *transpose* dari A.

2.7.3 Model Propagasi *Error*

Jaringan adaptif di latih untuk mendapatkan nilai parameter a dan c. Untuk melakukan perbaikan terhadap a dan c tersebut, digunakan model propagasi *error* dengan konsep *steepest descent*. Pada arsitektur metode ANFIS terdapat lima lapisan. Pada setiap lapisan dalam arsitektur ANFIS memiliki rumus *error* masing-masing untuk masing-masing lapisannya. Pada Gambar 2.5 merupakan struktur ANFIS yang akan dicari rumus-rumus *error* pada tiap lapisan.



Gambar 2.5 Arsitektur ANFIS

Sumber: (Suyanto, 2008)

***Error* pada lapisan 5**

Apabila kita memilih jaringan adaptif seperti pada Gambar 2.4, yang hanya memiliki 1 *neuron* pada lapisan *output* (*neuron* 13), maka propagasi *error* yang menuju pada lapisan ke-5 seperti pada Persamaan 2.19.

$$\varepsilon_{13} = \frac{\partial E_p}{\partial x_{13}} = -2(d_{13} - x_{13}) = -2(y_p - y'_p) \quad (2.19)$$

Dengan y_p adalah target *output* data pelatihan ke-p, dan y'_p adalah *output* jaringan pada data pelatihan ke-p.

***Error* pada lapisan 4**

Jaringan adaptif pada Gambar 2.5. Propagasi *error* yang menuju pada lapisan ke-4, yaitu *neuron* 11 dan *neuron* 12 adalah seperti pada Persamaan 2.20 dan Persamaan 2.21.

$$\varepsilon_{11} = \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{11}}\right) = \varepsilon_{13}\left(\frac{\partial f_{13}}{\partial x_{11}}\right) = \varepsilon_{13}(1) = \varepsilon_{13} \quad (2.20)$$

Karena $f_{13} = \overline{w_1} f_1 + \overline{w_2} f_2$, maka $\frac{\partial f_{13}}{\partial(\overline{w}_2 f_2)} = 1$.

$$\varepsilon_{12} = \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{12}}\right) = \varepsilon_{13}\left(\frac{\partial f_{13}}{\partial x_{12}}\right) = \varepsilon_{13}(1) = \varepsilon_{13} \quad (2.21)$$

Karena $f_{13} = \overline{w_1} f_1 + \overline{w_2} f_2$, maka $\frac{\partial f_{13}}{\partial(\overline{w}_2 f_2)} = 1$.

***Error* pada lapisan 3**

Jaringan adaptif pada Gambar 2.5. Propagasi *error* yang menuju lapisan ke-3, yaitu *neuron* 9 dan *neuron* 10 seperti pada Persamaan 2.22 dan Persamaan 2.23.

$$\varepsilon_9 = \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{11}}\right)\left(\frac{\partial f_{11}}{\partial x_9}\right) = \varepsilon_{11}\left(\frac{\partial f_{11}}{\partial x_9}\right) = \varepsilon_{11} f_{11} \tag{2.22}$$

Dan

$$\varepsilon_{10} = \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{12}}\right)\left(\frac{\partial f_{12}}{\partial x_{10}}\right) = \varepsilon_{12}\left(\frac{\partial f_{12}}{\partial x_{10}}\right) = \varepsilon_{12} f_{12} \tag{2.23}$$

***Error* pada lapisan 2**

Jaringan adaptif pada Gambar 2.5. Propagasi *error* yang menuju pada lapisan ke-2, yaitu *neuron* 7 dan *neuron* 8 seperti pada Persamaan 2.24 dan Persamaan 2.25.

$$\begin{aligned}\varepsilon_7 &= \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{11}}\right)\left(\frac{\partial f_{11}}{\partial x_9}\right)\left(\frac{\partial f_9}{\partial x_7}\right) + \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{12}}\right)\left(\frac{\partial f_{12}}{\partial x_{10}}\right)\left(\frac{\partial f_{10}}{\partial x_7}\right) \\ &= \varepsilon_9\left(\frac{\partial f_9}{\partial x_7}\right) + \varepsilon_{10}\left(\frac{\partial f_{10}}{\partial x_7}\right) \\ &= \varepsilon_9\left(\frac{w_2}{(w_1+w_2)^2}\right) + \varepsilon_{10}\left(-\frac{w_2}{(w_1+w_2)^2}\right) \\ &= \frac{w_2}{(w_1+w_2)^2}(\varepsilon_9 - \varepsilon_{10})\end{aligned} \tag{2.24}$$

$$\begin{aligned}\varepsilon_8 &= \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{12}}\right)\left(\frac{\partial f_{12}}{\partial x_{10}}\right)\left(\frac{\partial f_{10}}{\partial x_8}\right) + \left(\frac{\partial E_p}{\partial x_{13}}\right)\left(\frac{\partial f_{13}}{\partial x_{11}}\right)\left(\frac{\partial f_{11}}{\partial x_9}\right)\left(\frac{\partial f_9}{\partial x_8}\right) \\ &= \varepsilon_{10}\left(\frac{\partial f_{10}}{\partial x_8}\right) + \varepsilon_9\left(\frac{\partial f_9}{\partial x_8}\right) \\ &= \varepsilon_{10}\left(\frac{w_2}{(w_1+w_2)^2}\right) + \varepsilon_9\left(-\frac{w_2}{(w_1+w_2)^2}\right) \\ &= \frac{w_2}{(w_1+w_2)^2}(\varepsilon_{10} - \varepsilon_9)\end{aligned} \tag{2.25}$$

***Error* pada lapisan 1**

Jaringan adaptif pada gambar 2.4. Propagasi *error* yang menuju pada lapisan ke-1, yaitu *neuron* 3, 4, 5, dan 6 seperti pada Persamaan 2.26, Persamaan 2.27, Persamaan 2.28, dan Persamaan 2.29.

$$\varepsilon_3 = \varepsilon_7\left(\frac{\partial f_7}{\partial x_3}\right) = \varepsilon_7 \mu_{B1}(x_2) \tag{2.26}$$

$$\varepsilon_4 = \varepsilon_8\left(\frac{\partial f_8}{\partial x_4}\right) = \varepsilon_8 \mu_{B2}(x_2) \tag{2.27}$$

$$\varepsilon_5 = \varepsilon_7\left(\frac{\partial f_7}{\partial x_5}\right) = \varepsilon_7 \mu_{A1}(x_1) \tag{2.28}$$

$$\varepsilon_6 = \varepsilon_8\left(\frac{\partial f_8}{\partial x_6}\right) = \varepsilon_8 \mu_{A2}(x_1) \tag{2.29}$$

Selanjutnya, *error* tersebut dapat digunakan untuk mencari informasi *error* terhadap parameter a dan c, seperti pada Persamaan 2.30 dan Persamaan 2.31.

$$\varepsilon_{aik} = \frac{2(x_i - c_{ik})^2}{a_{ik}^2\left(1+\left(\frac{x_i - c_{ik}}{a_{ik}}\right)^2\right)^2} * \varepsilon_i \quad (2.30)$$

Dan

$$\varepsilon_{cik} = \frac{2(x_i - c_{ik})^2}{a_{ik}^3\left(1+\left(\frac{x_i - c_{ik}}{a_{ik}}\right)^2\right)^2} * \varepsilon_i \quad (2.31)$$

Dari persamaan di atas, perubahan nilai parameter a_{ik} dan c_{ik} (Δa_{ik} dan Δc_{ik}) dapat ditentukan seperti pada Persamaan 2.32.

$$\Delta a_{ik} = \eta \varepsilon_{aik} \text{ , dan } \Delta c_{ik} = \eta \varepsilon_{cik} x_i \quad (2.32)$$

Dengan η adalah laju pembelajaran yang terletak pada interval [0, 1] sehingga nilai a_{ik} dan c_{ik} yang baru dapat dihitung seperti pada Persamaan 2.33 dan Persamaan 2.34.

$$a_{ik} = a_{ik(lama)} + \Delta a_{ik}, \text{ dan} \quad (2.33)$$

$$c_{ik} = c_{ik(lama)} + \Delta c_{ik} \quad (2.34)$$

2.8 Proses Prediksi dengan ANFIS

Metode ANFIS merupakan gabungan dari *fuzzy* dan *artificial neural network*. Proses yang terdapat dalam metode ANFIS adalah sebagai berikut (Dewi & Himawati, 2015):

1. Data yang ada akan dinormalisasi terlebih dahulu sebelum dilakukan *clustering* dengan menggunkan metode *k-means*. Proses *clustering* ini bertujuan untuk mengelompokkan data kedalam beberapa *cluster*/kelompok.
2. Data yang telah ternormalisasi digunakan untuk mencari nilai *mean* dan juga nilai standar deviasi yang kemudian akan digunakan untuk menghitung derajat keanggotaan menggunakan fungsi keanggotaan *generalized bell*.
3. Menghitung *fire strength*, yaitu Perkalian derajat keanggotaan dengan koefisien pada tiap parameter.
4. Melakukan normalisasi pada nilai *fire strength*.
5. Perhitungan *matriks* desain yang digunakan untuk *input* LSE.
6. Penentuan parameter konsekuen adaptif menggunakan LSE.
7. Menghitung nilai *output* jaringan.
8. Membandingkan *output* jaringan dengan target *output* untuk mencari nilai *error* jaringan.
9. Apabila nilai *error* yang didapatkan lebih besar dari nilai *error* yang diharapkan, lakukan perbaikan parameter premis menggunakan algoritme *steepest descent*.

10. Membandingkan nilai target *output* dengan *output* jaringan yang terbentuk yang kemudian dicocokan dengan nilai *error*.

2.9 Akurasi Hasil Pengujian

Metode yang digunakan untuk mengukur tingkat akurasi adalah *Root Mean Square Error* atau yang biasa disebut dengan RMSE. RMSE merupakan akar rata-rata total kuadrat *error* yang terjadi antara *output* proses dan *output* target, semakin kecil nilai RMSE maka semakin besar tingkat keberhasilan proses pelatihan (Husnita, 2012).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \bar{y}_i)^2}{N}} \quad (2.35)$$

Keterangan:

N = banyaknya data

y_i = *output* actual atau target jaringan

$\bar{y}_i$ = *output* prediksi atau *output* jaringan.

2.10 *Missing Value*

Missing value adalah kondisi di mana terdapat data atau informasi yang tidak ditemukan pada suatu atribut tertentu dalam dataset (Hidayatullah, Prasetyo, & Sari, 2014). *Missing value* dapat terjadi karena nilainya tidak relevan untuk kasus tertentu, tidak bisa dicatat pada saat data dikumpulkan, atau disebabkan adanya privasi (Gimpy, Vohra, & Minakshi, 2014). Untuk mengatasi *missing value*, dapat dilakukan beberapa hal seperti melakukan pengurangan objek data, memperkirakan nilai *missing values*, tidak melibatkan *missing values* dalam analisis data, dan mencari nilai rata-rata pada atribut yang memiliki *missing value*. Penanganan *missing value* dapat dilakukan dengan *mean imputation*. *Mean imputation* adalah metode yang cukup sering digunakan. Metode ini mengganti *missing value* pada atribut dengan nilai rata-rata yang diperoleh dari seluruh atribut yang diketahui nilainya.

BAB 3 METODOLOGI

Pada bab ini membahas tentang metode penelitian yang digunakan untuk memprediksi curah hujan dengan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS). Metodologi pada penelitian ini dilakukan dengan beberapa tahapan yaitu sebagai berikut:

1. Mempelajari literatur yang berhubungan dengan curah hujan, Jaringan Saraf Tiruan (JST), *Fuzzy Inference System*, *clustering* dan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS).
1. Melakukan pengumpulan data yang berupa data sekunder berisi curah hujan beserta 3 indikator diantaranya suhu udara, kelembapan udara, dan kecepatan angin.
2. Melakukan analisis dan perancangan sistem yang diperlukan dalam pembangunan sistem untuk memprediksi curah hujan dengan metode *Adaptive Neuro Fuzzy Inference System*.
3. Mengimplementasikan sistem berdasarkan hasil dari analisis dan perancangan yang telah dibuat sebelumnya.
4. Melakukan pengujian terhadap perangkat lunak yang telah dibuat.
5. Melakukan pengambilan kesimpulan berdasarkan hasil pengujian yang dilakukan pada tahap sebelumnya.

3.1 Tipe Penelitian

Tipe penelitian pada penelitian ini adalah nonimplementatif yang bersifat analitik. Penelitian ini berfokus pada penerapan metode komputasi cerdas untuk menyelesaikan masalah dari objek penelitian. Tipe penelitian ini memfokuskan investigasi terhadap fenomena atau situasi tertentu yang menghasilkan analisa ilmiah atau hasil investigasi. Produk atau artefak utama dari penelitian ini adalah hasil analisis.

3.2 Strategi Penelitian

Penelitian ini menggunakan strategi/metode penelitian eksperimen. Metode penelitian eksperimen adalah suatu cara untuk mencari hubungan sebab akibat (hubungan kausal) antara dua faktor yang sengaja ditimbulkan oleh peneliti dengan mengeliminasi atau mengurangi atau menyisihkan faktor-faktor lain yang mengganggu (Arikunto, 2006). Metode eksperimen yang digunakan pada penelitian ini adalah *Adaptive Neuro Fuzzy Inference System* (ANFIS).

3.3 Partisipan Penelitian

Partisipan yang terlibat dalam penelitian ini adalah stasiun klimatologi Karang Ploso BMKG Kabupaten Malang. Stasiun klimatologi Karang Ploso memberikan data harian berupa curah hujan, suhu udara, kelembapan udara, dan kecepatan angin.

3.4 Lokasi Penelitian

Penelitian ini dilakukan di Fakultas Ilmu Komputer Universitas Brawijaya dan stasiun klimatologi Karang Ploso Kabupaten Malang. Lokasi ini dipilih peneliti agar memperoleh data yang diperlukan dan tercapainya tujuan penelitian yaitu mengimplementasikan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan .

3.5 Teknik Pengumpulan Data

Data yang digunakan pada penelitian ini adalah data curah hujan, suhu udara, kelembapan udara, dan kecepatan angin Kabupaten Malang yang telah didata oleh Stasiun Klimatologi Karang Ploso Kabupaten Malang. Data yang diperoleh adalah data harian curah hujan, suhu udara, kelembapan udara, dan kecepatan angin yang selanjutnya akan diolah menjadi data dasarian oleh peneliti.

3.6 Implementasi Algoritme

Pada penelitian ini implementasi algoritme menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS). Pada metode ANFIS diperlukan metode *clustering* untuk membagi data menjadi beberapa kelompok. Proses prediksi curah hujan menggunakan metode ANFIS dilakukan dengan cara mengelompokkan data ke dalam 3 *cluster* dengan menggunakan algoritme *K-means*. Pengelompokan data menggunakan algoritme *K-Means* bertujuan untuk mendapatkan nilai parameter premis awal. Setelah proses *clustering*, proses selanjutnya adalah menghitung *output* lapisan 1 yaitu derajat keanggotaan dengan menggunakan fungsi keanggotaan *generalized bell*, Setelah itu menghitung *output* lapisan 2 yaitu nilai *fire strength* yang didapatkan dari hasil perkalian nilai derajat keanggotaan masing-masing cluster pada tiap parameter. Pada lapisan 3 dilakukan perhitungan normalisasi terhadap nilai *fire strength*. Setelah itu menghitung parameter konsekuen pada lapisan 4 dengan menggunakan algoritme *Least Square Error* (LSE). Kemudian pada lapisan 5 menghitung *output* jaringan. Setelah itu dilakukan proses perbaikan premis dengan menggunakan algoritme *steepest descent*. Pada proses *training* akan didapatkan parameter premis dan konsekuen yang nantinya akan digunakan untuk proses *testing* menggunakan data uji. Proses *testing* dilakukan dengan cara menghitung *output* lapisan 1 sampai lapisan 5 menggunakan parameter premis dan parameter konsekuen yang telah didapatkan dari proses *training*.

3.7 Jadwal Penelitian

Pelaksanaan penelitian diatur berdasarkan jadwal sebagaimana terlihat pada Tabel 3.1.

Tabel 3.1 Jadwal Penelitian

No	Kegiatan	Jadwal Penelitian																							
		Januari				Februari				Maret				April				Mei				Juni			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Konsultasi dan Penyusunan Proposal																								
2	Penyerahan Proposal																								
3	Perbaikan/revisi proposal penelitian																								
4	Pengumpulan Data																								
5	Penyusunan Laporan Penelitian																								
6	Bimbingan dan Konsultasi hasil penelitian																								
7	Seminar Hasil Penelitian																								
8	Perbaikan hasil penelitian																								
9	Sidang Skripsi																								
10	Perbaikan hasil sidang penelitian																								

BAB 4 PERANCANGAN

4.1 Formulasi Permasalahan

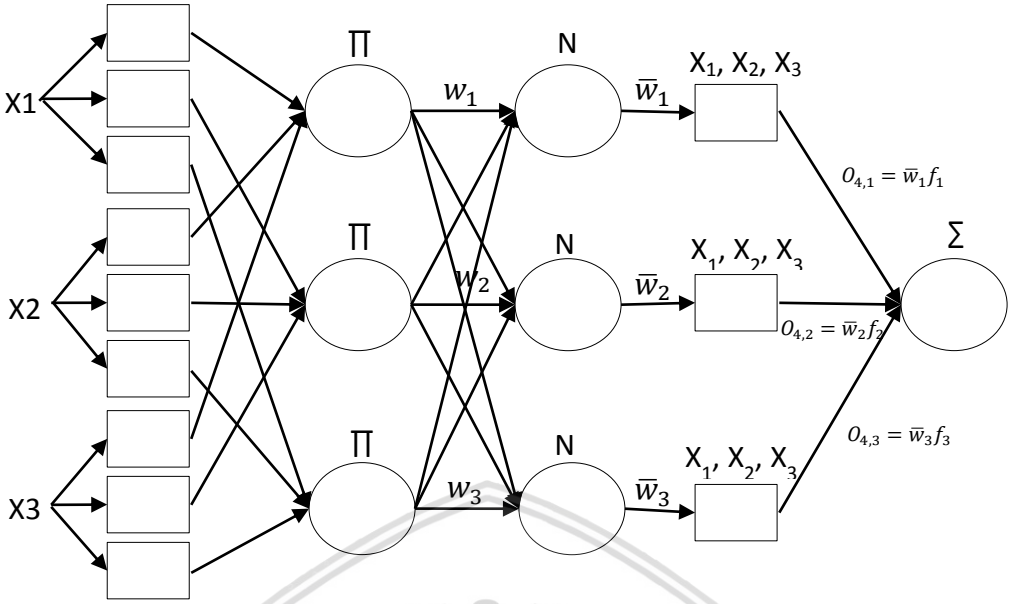
Permasalahan yang akan diselesaikan pada penelitian ini adalah prediksi curah hujan. Informasi curah hujan dalam suatu daerah dapat dilihat dalam website Badan Meteorologi, Klimatologi, dan Geofisikia (BMKG) yang telah diukur setiap hari. Dengan adanya informasi tersebut para petani dapat mengetahui seberapa besar curah hujan sehingga dapat menentukan waktu masa tanam dan tanaman apa yang cocok untuk ditanam. Namun dengan adanya iklim yang tidak menentu mengakibatkan petani salah memprediksi dan terjadi gagal produksi. Dari permasalahan tersebut perlu adanya sistem untuk memprediksi curah hujan agar dapat membantu dan mempermudah para petani dalam menentukan waktu masa tanam yang benar.

Sistem yang dibangun dalam penelitian ini adalah perangkat lunak yang dapat memprediksi curah hujan dengan menggunakan 3 variabel input berupa data suhu udara, tekanan udara, dan kecepatan angin. Data yang digunakan adalah data dasarian (10 harian), sehingga hasil prediksi juga akan menghasilkan data curah hujan untuk 10 hari kedepan. Dari hasil prediksi tersebut dapat ditarik kesimpulan apakah curah hujan mengalami kenaikan, penurunan, atau tetap untuk satu dasarian kedepan.

Pada peneliatian ini, yang harus dilakukan pertama kali yaitu mengolah data curah hujan, suhu udara, kelembapan udara, dan kecepatan angin yang masih berupa data harian menjadi data per sepuluh harian. Selanjutnya dilakukan proses pembelajaran (*training)* terhadap data suhu udara, kelembapan udara, dan kecepatan angin sebagai data *input* dalam system.

4.2 Rancangan Arsitektur

Arisektur pada metode *Adaptive Neuro Fuzzy Inference System (ANFIS)* menggunakan sistem inferensi *fuzzy* Takagi Sugeno Kang (TSK) yang terdiri dari 5 lapisan dan memiliki fungsi masing-masing pada setiap lapisannya. Pada Gambar di bawah ini menjelaskan arsitektur *Adaptive Neuro Fuzzy Inference System (ANFIS)* dalam memprediksi curah hujan berdasarkan 3 variabel input berupa (suhu udara (x_1), kelembapan udara (x_2), dan kecepatan angin (x_3)).



Gambar 4.1 Desain Arsitektur ANFIS Untuk Prediksi Curah Hujan

Lapisan 1 : Pada lapisan ini dibangkitkan derajat keanggotaan (*membership degree*) dengan menggunakan fungsi derajat keanggotaan *generalized bell* dari 3 variabel inputan (suhu udara (x_1), kelembapan udara (x_2), dan kecepatan angin (x_3)) dan 3 buah *cluster* berupa rendah, sedang dan tinggi.

Lapisan 2 : Pada lapisan ini dibangkitkan nilai *fire strength* yang didapatkan dengan menggunakan hasil keluaran dari Lapisan 1 sebagai *input*.

Lapisan 3 : Pada Lapisan ini dilakukan normalisasi terhadap nilai *fire strength* yang di dapatkan dari keluaran pada lapisan 2. Nilai tersebut merupakan *normalized firing strength*.

Lapisan 4 : Pada lapisan ini dilakukan perhitungan untuk mendapatkan nilai parameter konsekuen.

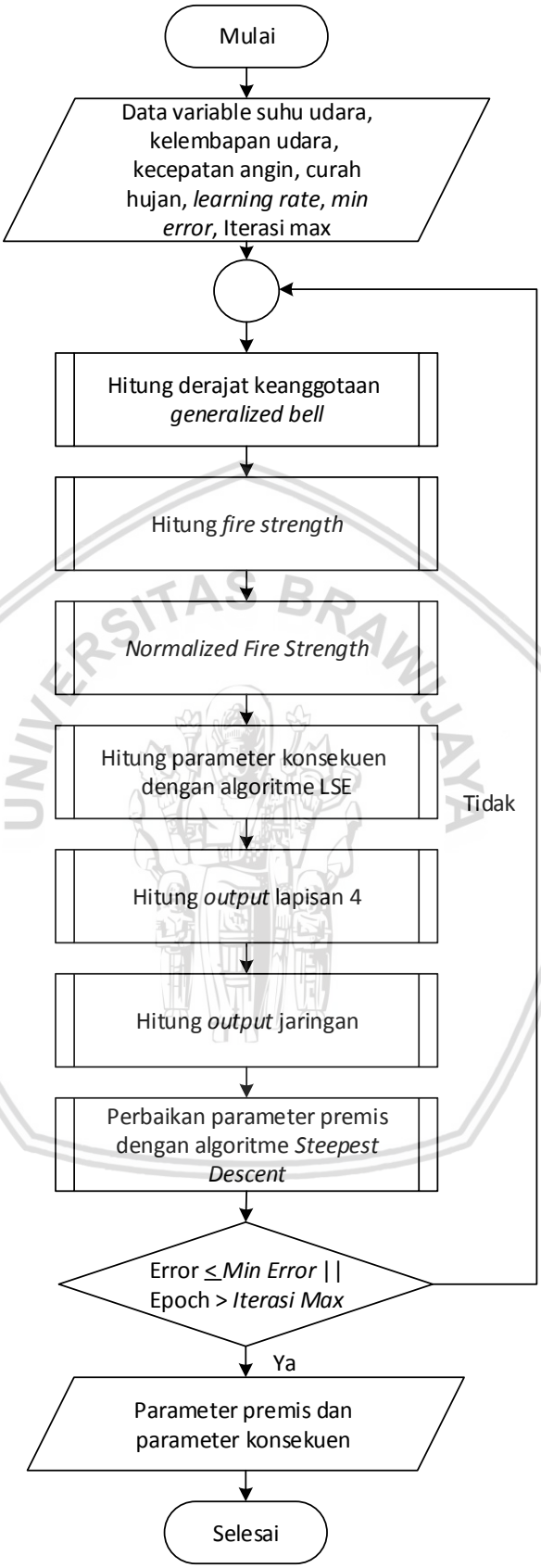
Lapisan 5 : Pada lapisan ini dilakukan perhitungan sinyal keluaran jaringan ANFIS dengan menjumlahkan seluruh sinyal yang masuk.

4.3 Siklus Algoritme ANFIS

Dalam metode *Adaptive Neuro Fuzzy Inference System (ANFIS),* terdapat beberapa langkah yang harus dilakukan untuk melakukan proses prediksi curah hujan agar didapatkan hasil prediksi yang akurat. Langkah-langkah pelatihan ANFIS yang harus dilakukan adalah sebagai berikut:

1. *Input* data berupa 3 variabel input (suhu udara, kelembapan udara, kecepatan angin) serta inisialisasi nilai parameter *learning rate*, *maximum error* dan iterasi maksimum.
2. Menghitung derajat keanggotaan berdasarkan data *input* dengan menggunakan persamaan 2.11.
3. Melakukan proses *firing strength*. Nilai *fire strength* didapatkan dari proses perkalian antara nilai dari setiap derajat keanggotaan yang didapat dengan menggunakan persamaan 2.12.
4. Melakuakan proses standarisasi nilai *fire strength* untuk mendapatkan nilai *normalized fire strength* dengan menggunakan persamaan 2.13.
5. Menghitung nilai parameter konsekuen menggunakan algoritme *Least Square Estimator* (LSE) sesuai dengan persamaan 2.18.
6. Menghitung nilai *output* pada lapisan 4 dengan menggunakan persamaan 2.14.
7. Menjumlahkan seluruh nilai pada lapisan 4 untuk mendapatkan *output* pada lapisan 5 (*output* jaringan).
8. Menghitung *propagation* error jaringan. Jika *error* yang dihasilkan pada jaringan lebih kecil atau sama dengan nilai *maximum error* yang ditetapkan dan nilai *epoch* lebih besar dari iterasi maksimum maka lanjutkan ke proses selanjutnya. Apabila tidak maka lakukan proses perbaikan pada parameter konsekuen dan parameter premis dengan menggunakan algoritme *steepest descent* sampai didapatkan nilai yang sesuai.
9. Didapatkan nilai parameter premis dan parameter konsekuen berupa nilai *mean* (c) dan standar deviasi (a).

Berdasarkan langkah-langkah di atas, proses pelatihan dengan metode ANFIS untuk prediksi curah hujan dapat digambarkan dalam siklus diagram alir pada Gambar 4.2 berikut.



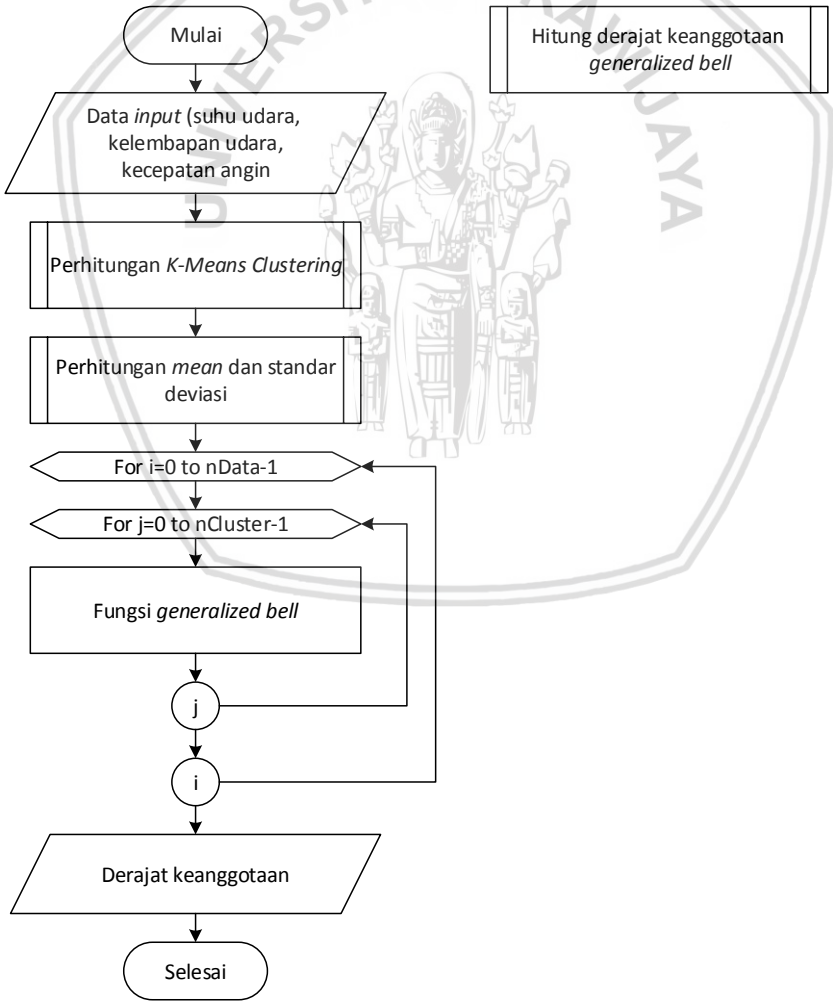
Gambar 4.2 Diagram Alir Perancangan ANFIS

4.3.1 Perhitungan Derajat Keanggotaan

Proses perhitungan derajat keanggotaan pada metode ANFIS menggunakan fungsi keanggotaan *generalized bell* dengan langkah-langkah sebagai berikut:

1. Memasukkan data varibel input berupa (suhu udara (x_1), kelembapan udara (x_2), dan kecepatan angin (x_3)).
2. Mengelompokkan data *input* dengan menggunakan metode *clustering k-means* untuk menghitung nilai *mean* dan standar deviasi.
3. Melakukan perulangan sampai proses *clustering* selesai.
4. Melakukan perhitungan derajat keanggotaan dengan menggunakan fungsi keanggotaan *generalized bell* menggunakan persamaan 2.2.
5. Didapatkan *output* lapisan 1 pada jaringan ANFIS berupa nilai derajat keanggotaan.

Proses perhitungan derajat keanggotaan dapat digambarkan dalam siklus diagram alir seperti pada Gambar 4.3 berikut.



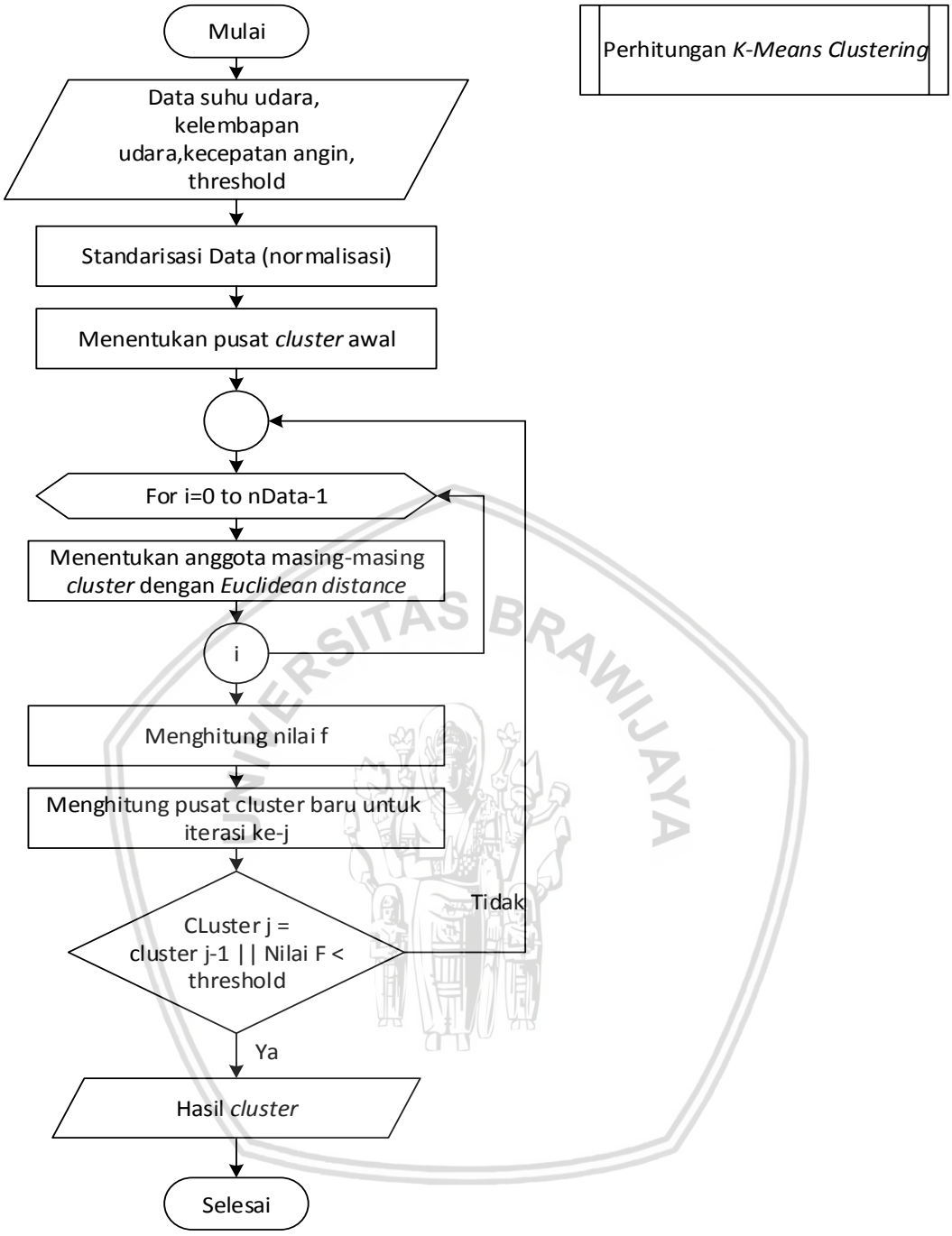
Gambar 4.3 Diagram Alir Perhitungan Derajat Keanggotaan *Generalized Bell*

4.3.1.1 Perhitungan *K-Means Clustering*

Proses pengelompokan data menggunakan metode *k-means clustering* bertujuan untuk mengelompokkan data ke dalam 3 cluster, yaitu naik, tetap dan turun. Proses *clustering* dijelaskan dalam langkah-langkah berikut:

1. Menginputkan data suhu udara, kelembapan udara, kecepatan angin dan nilai *threshold*.
2. Melakukan standarisasi data (normalisasi data) menggunakan persamaan 2.5.
3. Menentukan pusat cluster awal (*centroid*) secara random.
4. Menentukan anggota setiap *cluster* menggunakan rumus jarak *Euclidean distance* sesuai pada persamaan 2.6.
5. Menghitung pusat cluster baru sesuai pada persamaan 2.7.
6. Melakukan proses iterasi sampai tidak ada perubahan pada anggota setiap *cluster* yang terbentuk atau nilai f kurang dari nilai *threshold*.
7. Didapatkan *output* berupa anggota setiap *cluster*.

Proses pengelompokan data dengan menggunakan *k-means clustering* dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.4 berikut.



Gambar 4.4 Diagram Alir Perhitungan *K-Means Clustering*

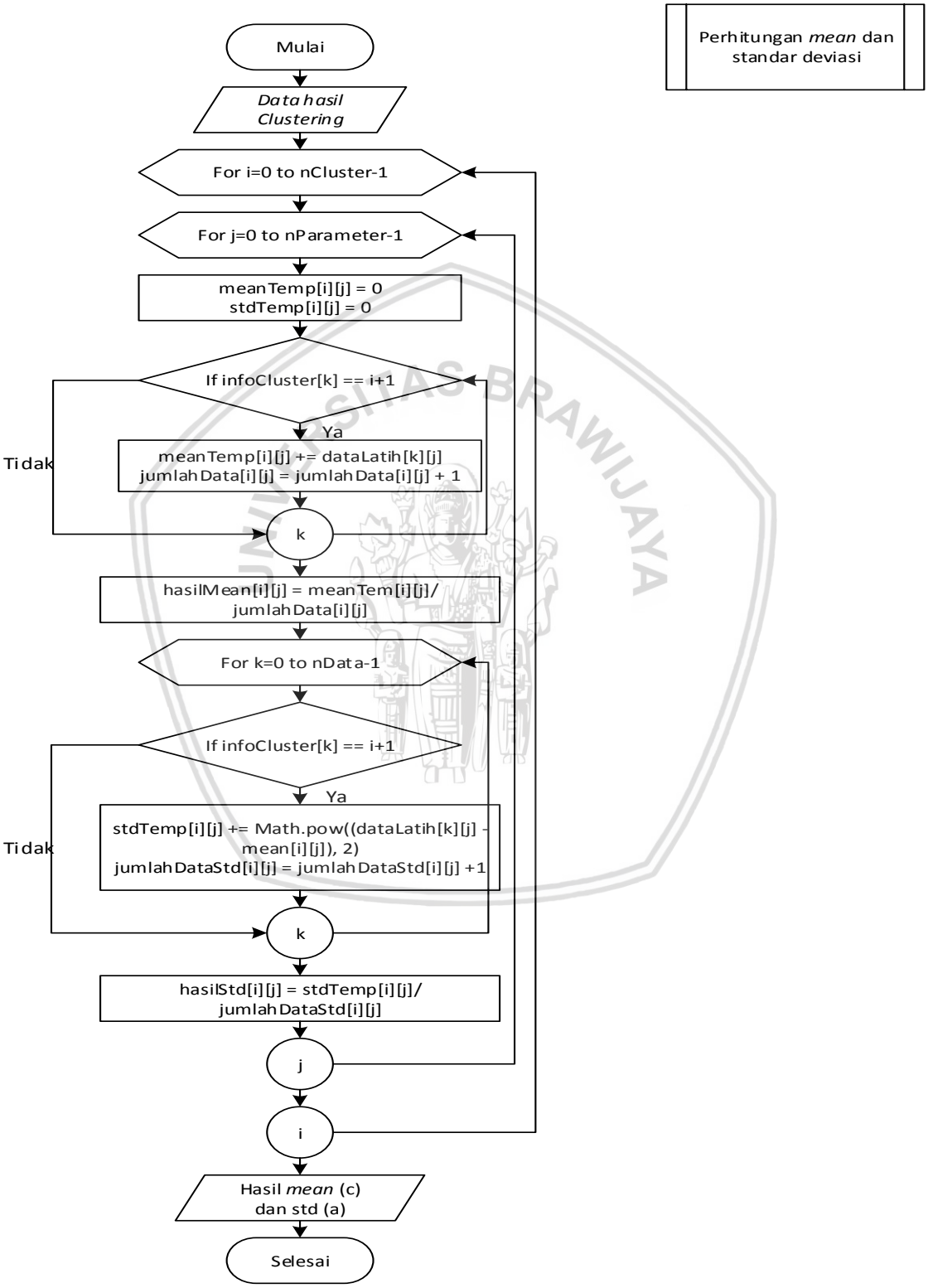
4.3.1.2 Perhitungan *Mean* dan Standar Deviasi

Hasil dari perhitungan *mean* dan standar deviasi digunakan sebagai nilai awal parameter premis a dan c. Berikut adalah langkah-langkahnya:

1. Data hasil *k-means clustering* digunakan untuk mencari *mean* dan standar deviasi untuk masing-masing data *input*.
2. Menghitung nilai *mean* menggunakan persamaan 2.8 dan menghitung nilai standar deviasi menggunakan persamaan 2.9.

3. Melakukan perulangan hingga iterasi selesai dan didapatkan nilai *mean* dan standar deviasi.

Proses perhitungan *mean* dan standar deviasi dapat digambarkan dalam siklus diagram alir seperti pada Gambar 4.5 berikut.



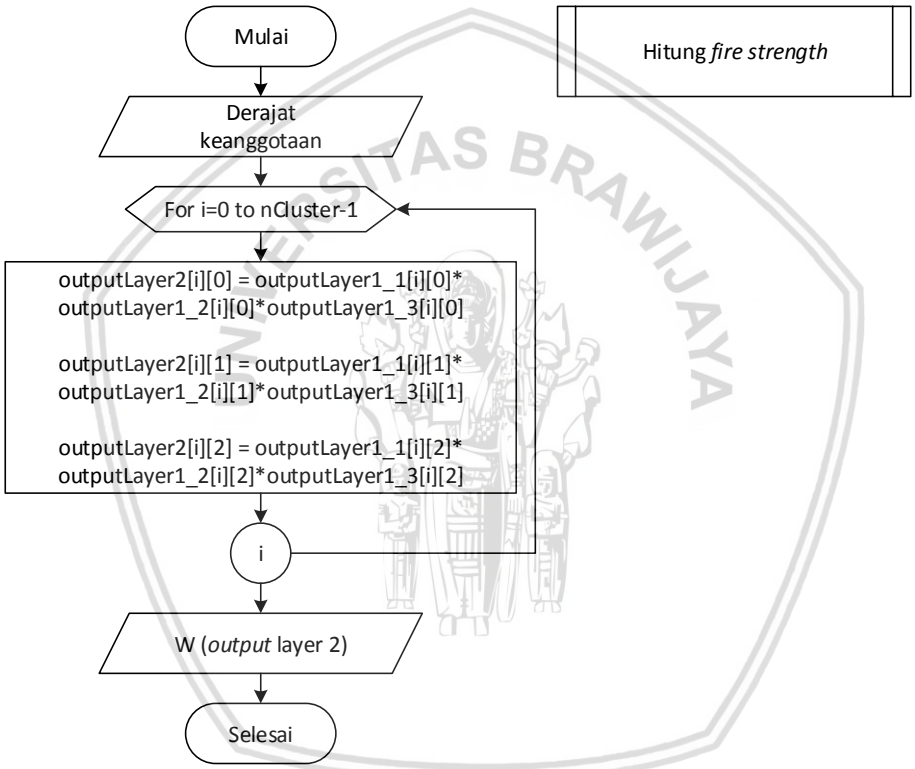
Gambar 4.5 Diagram Alir Perhitungan *Mean* dan Standar Deviasi

4.3.2 Menghitung Nilai *Fire Strength*

Nilai *fire strength* didapatkan dari hasil *output* lapisan 2. Proses perhitungan *fire strength* dijelaskan pada langkah-langkah berikut:

1. *Input* yang didapatkan dari *output* pada lapisan 1.
2. Melakukan perulangan terhadap parameter, *cluster* dan jumlah data yang digunakan.
3. Menghitung *output* lapisan 2 menggunakan persamaan 2.12.
4. Melakukan iterasi hingga didapatkan nilai bobot (*w*) untuk setiap parameter.

Langkah-langkah di atas dapat digambarkan pada diagram alir seperti pada Gambar 4.6 berikut.



Gambar 4.6 Diagram Alir Perhitungan *Fire Strength*

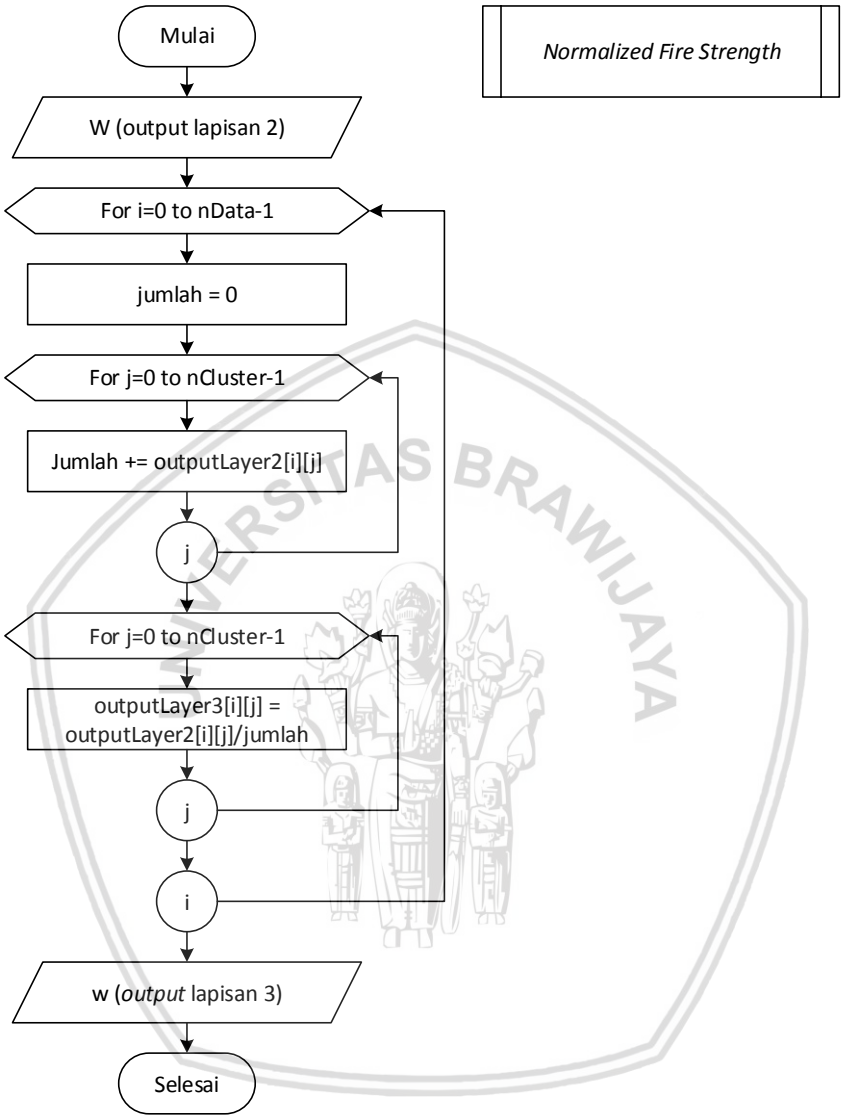
4.3.3 *Normalized Fire Strength*

Nilai fire strength dari lapisan 2 akan dinormalisasi menjadi *output* pada lapisan 3. Langkah perhitungan normalisasi *fire strength* dijelaskan pada langkah-langkah berikut:

1. *Input* yang digunakan adalah nilai *fire strength* yang didapatkan dari *output* pada lapisan 2.
2. Melakukan perulangan terhadap parameter, *cluster* dan jumlah data yang digunakan.
3. Menghitung nilai normalisasi *fire strength* menggunakan persamaan 2.13.

4. Melakukan iterasi hingga didapatkan nilai *fire strength* yang ternormalisasi untuk setiap parameter.

Proses perhitungan *normalized fire strength* dapat digambarkan ke dalam bentuk diagram alir seperti pada Gambar 4.7 berikut.



Gambar 4.7 Diagram Alir Perhitungan *Normalized Fire Strength*

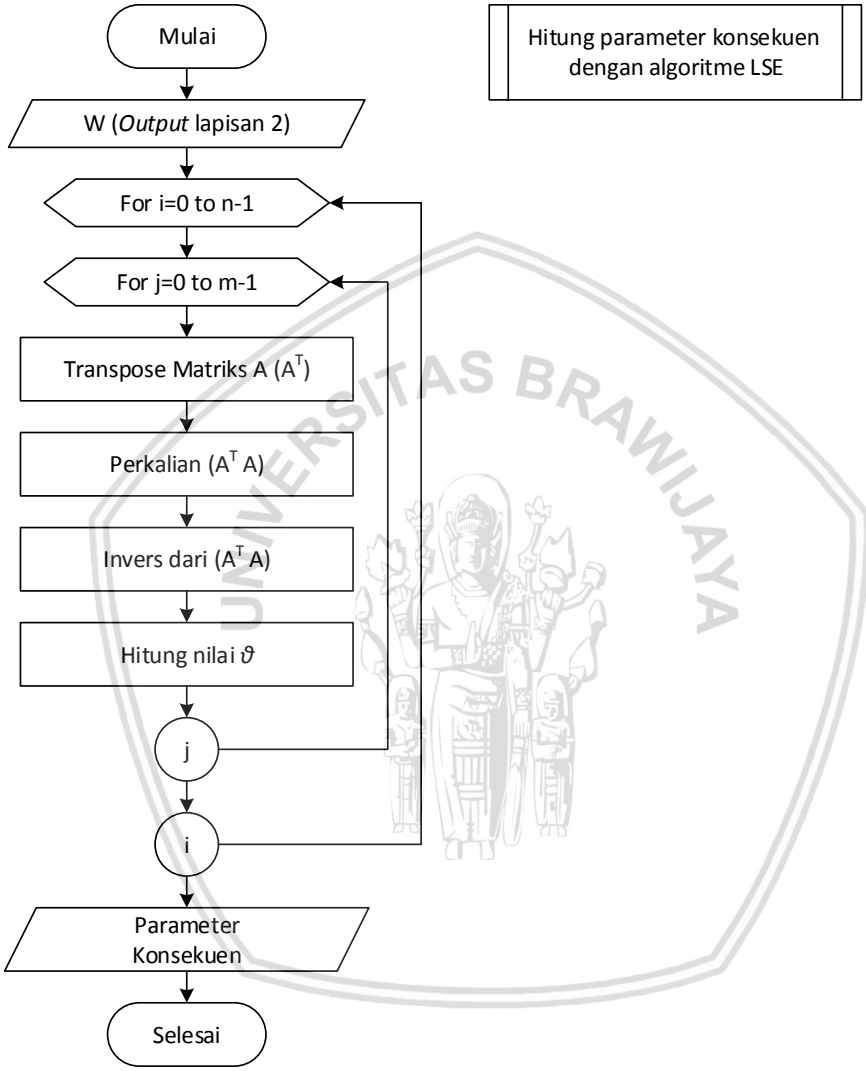
4.3.4 Perhitungan Parameter Konsekuen dengan Algoritme LSE

Proses perhitungan parameter konsekuen menggunakan LSE dijelaskan pada langkah-langkah berikut:

1. *Input* berupa matriks desain (A).
2. Melakukan transpose terhadap matriks desain (A^T).
3. Melakukan perkalian antara matriks desain (A) dengan matriks desain yang telah di transpose (A^T).

4. Menghitung matriks *invers* dari hasil perkalian (A) dan (A^T).
5. Menghitung nilai parameter konsekuen menggunakan persamaan 2.18.
6. Didapatkan nilai parameter konsekuen (θ).

Proses perhitungan parameter konsekuen menggunakan LSE dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.8 berikut.



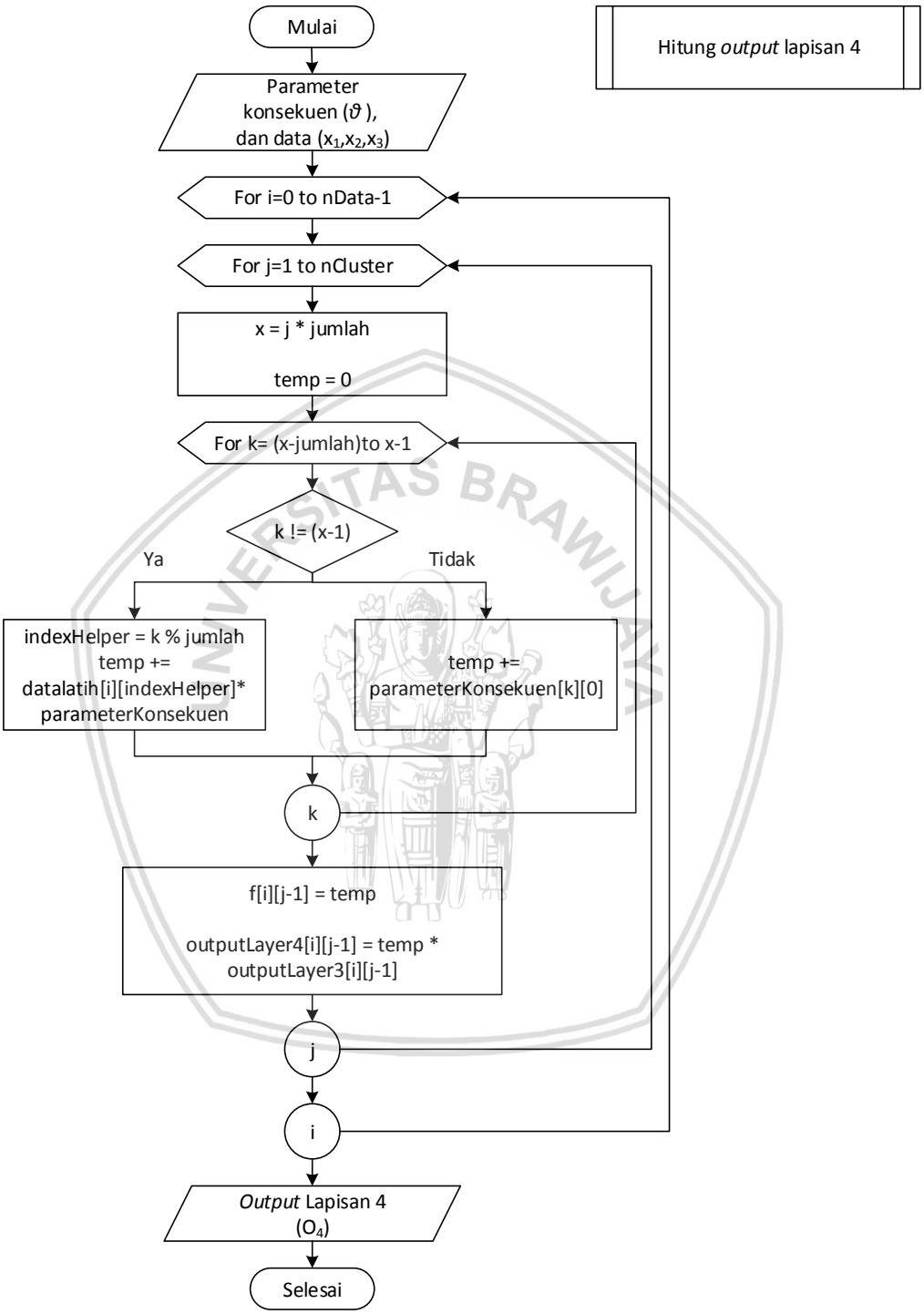
Gambar 4.8 Diagram Alir Perhitungan Parameter Konsekuen dengan LSE

4.3.5 Perhitungan *Output* Lapisan 4

Perhitungan *output* lapisan 4 dilakukan dengan langkah-langkah berikut:

1. *Input* yang digunakan untuk menghitung *output* lapisan 4 adalah parameter konsekuen serta parameter penentu curah hujan, yaitu suhu udara, kelembapan udara, dan kecepatan angin.
2. Menghitung *output* lapisan 4 menggunakan persamaan 2.14.
3. Didapatkan *output* lapisan 4.

Proses perhitungan *output* lapisan 4 dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.9 berikut.



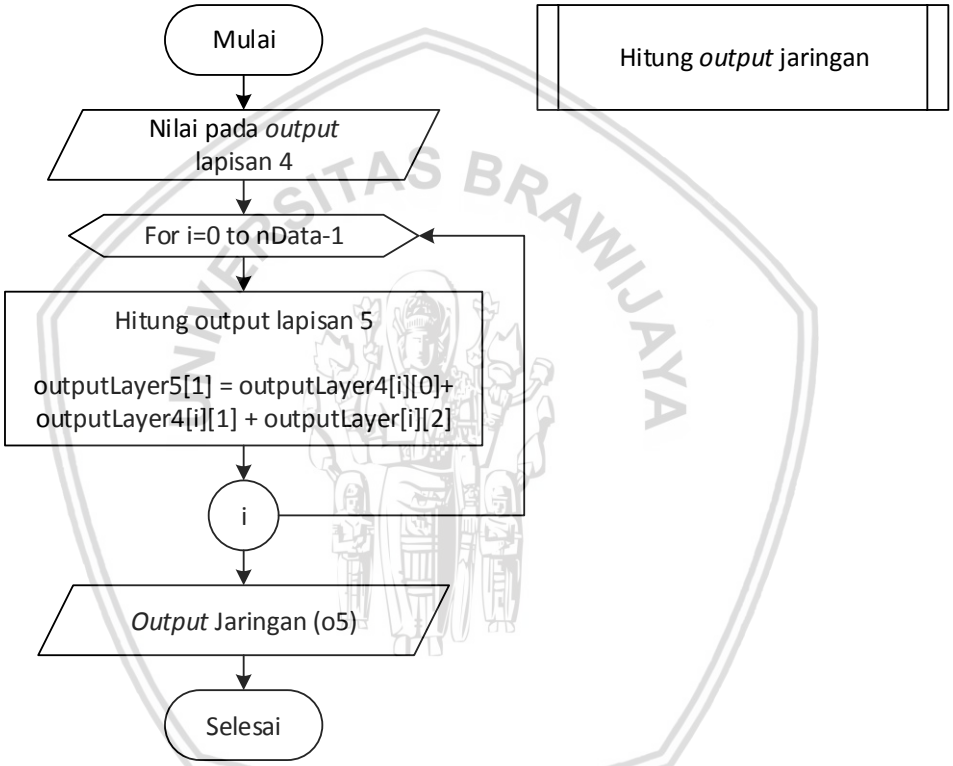
Gambar 4.9 Diagram Alir Perhitungan *Output* Lapisan 4

4.3.6 Perhitungan *Output* Jaringan

Output jaringan merupakan nilai yang dihasilkan sebagai *output* pada lapisan 5. Untuk mendapatkan nilai *output* jaringan dilakukan langkah-langkah berikut:

1. Menggunakan *output* pada lapisan 4 sebagai *input* untuk menghitung nilai *output* jaringan.
2. Menghitung nilai pada lapisan 5 menggunakan persamaan 2.15.
3. Didapatkan hasil *output* jaringan yang merupakan nilai dari lapisan 5.

Proses perhitungan *output* jaringan dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.10 berikut.



Gambar 4.10 Diagram Alir Perhitungan *Output* Jaringan

4.3.7 Perbaikan Parameter Premis Menggunakan *Steepest Descent*

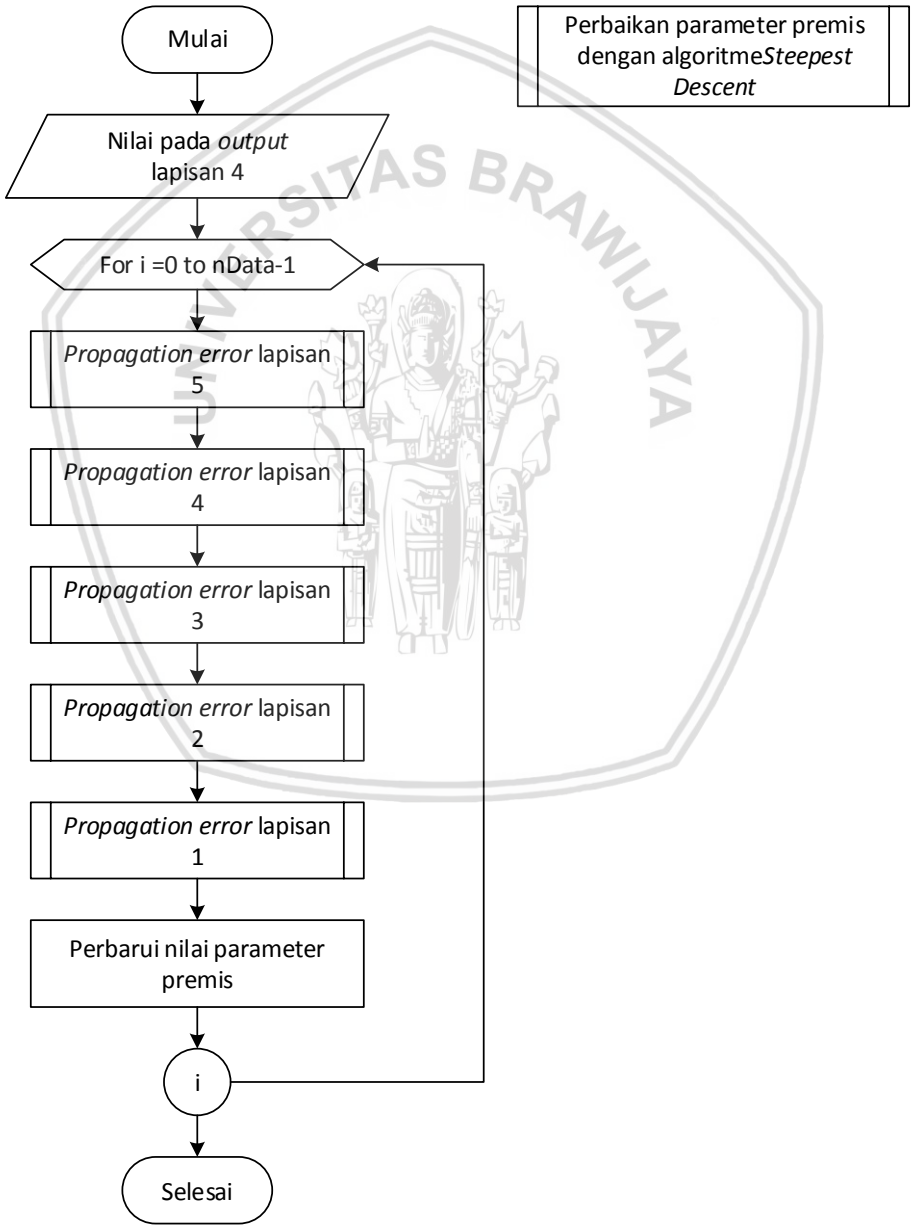
Untuk melakukan perbaikan terhadap parameter premis dilakukan langkah-langkah berikut:

1. *Output* jaringan yang didapatkan pada lapisan ke 5 digunakan sebagai *input* untuk melakukan perbaikan parameter premis.
2. Menghitung *propagation error* lapisan 5 menggunakan persamaan 2.19.
3. Menghitung *propagation error* lapisan 4 menggunakan persamaan 2.20 dan 2.21.

4. Menghitung *propagation error* lapisan 3 menggunakan persamaan 2.22 dan 2.23.
5. Menghitung *propagation error* lapisan 2 menggunakan persamaan 2.24 dan 2.25.
6. Menghitung *propagation error* lapisan 1 menggunakan persamaan 2.26, 2.27, 2.28,dan 2.29.
7. Perbarui nilai parameter premis.

Proses perbaikan parameter premis dengan menggunakan algoritme *steepest descent* dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.11 berikut.

.



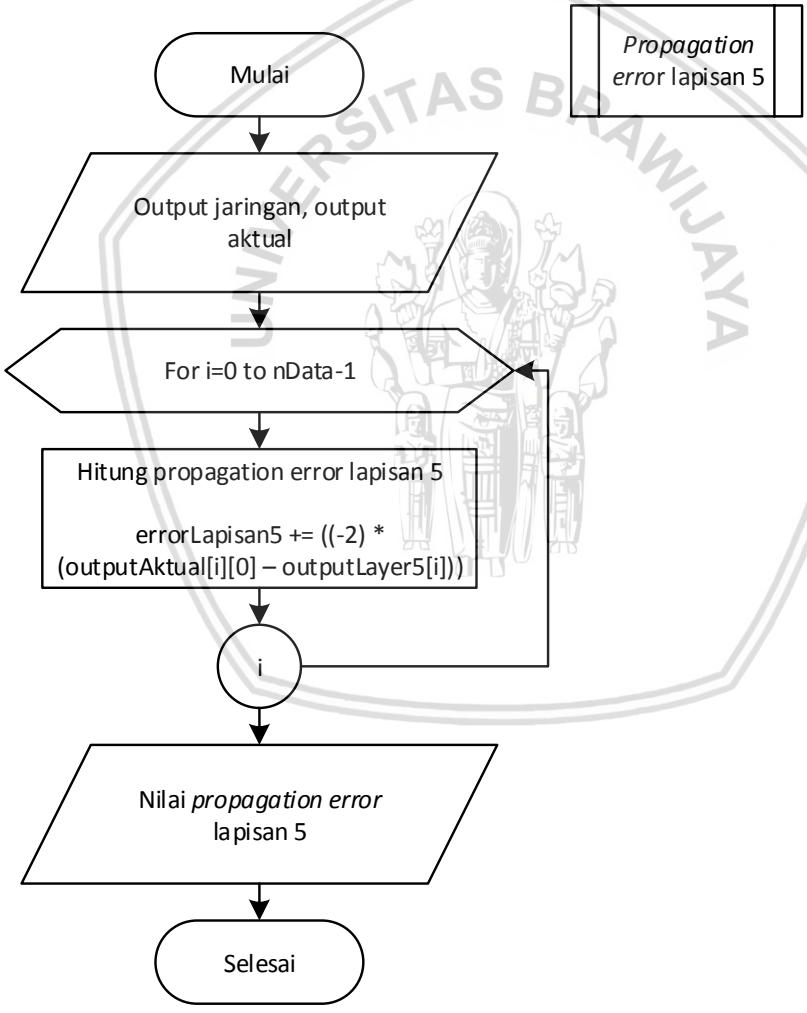
Gambar 4.11 Diagram Alir Perbaikan Parameter Premis dengan *Steepest Descent*

4.3.7.1 Perhitungan *Propagation Error* Lapisan 5

Perhitungan nilai *propagation error* pada lapisan 5 dilakukan dengan langkah-langkah berikut:

1. Menggunakan *output* jaringan dan *output* aktual sebagai *input*.
2. Melakukan iterasi sebanyak jumlah data yang ada.
3. Menghitung nilai *error* pada lapisan 5 menggunakan persamaan 2.19.
4. Didapatkan nilai *error* lapisan 5.

Proses perhitungan *propagation error* lapisan 5 dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.12 berikut.



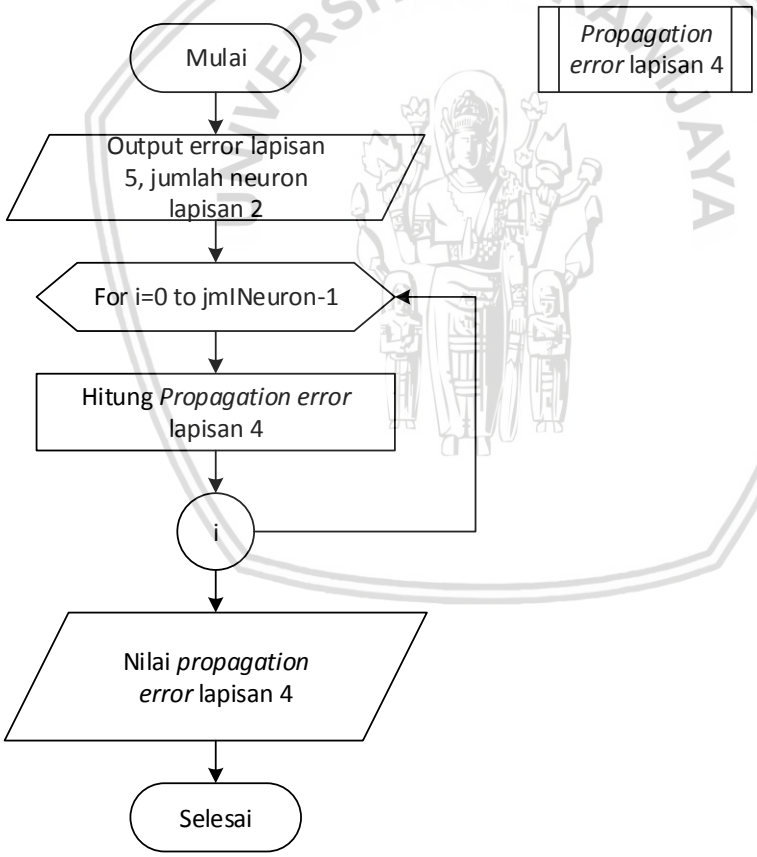
Gambar 4.12 Diagram Alir Perhitungan *Propagation Error* Lapisan 5

4.3.7.2 Perhitungan *Propagation Error* Lapisan 4

Untuk mendapatkan nilai *propagation error* pada lapisan 4 digunakan langkah-langkah berikut:

1. Menggunakan jumlah *neuron* pada lapisan 2 dan *output* dari hasil perhitungan *propagation error* lapisan 5 sebagai *input*.
2. Melakukan iterasi sebanyak jumlah data yang ada pada *neuron* lapisan 2.
3. Melakukan perhitungan *propagation error* lapisan 4 menggunakan persamaan 2.20 dan 2.21
4. Didapatkan nilai *propagation error* lapisan 4 sebanyak jumlah *neuron* pada lapisan 2.

Proses perhitungan nilai *propagation error* pada lapisan 4 dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.13 berikut.



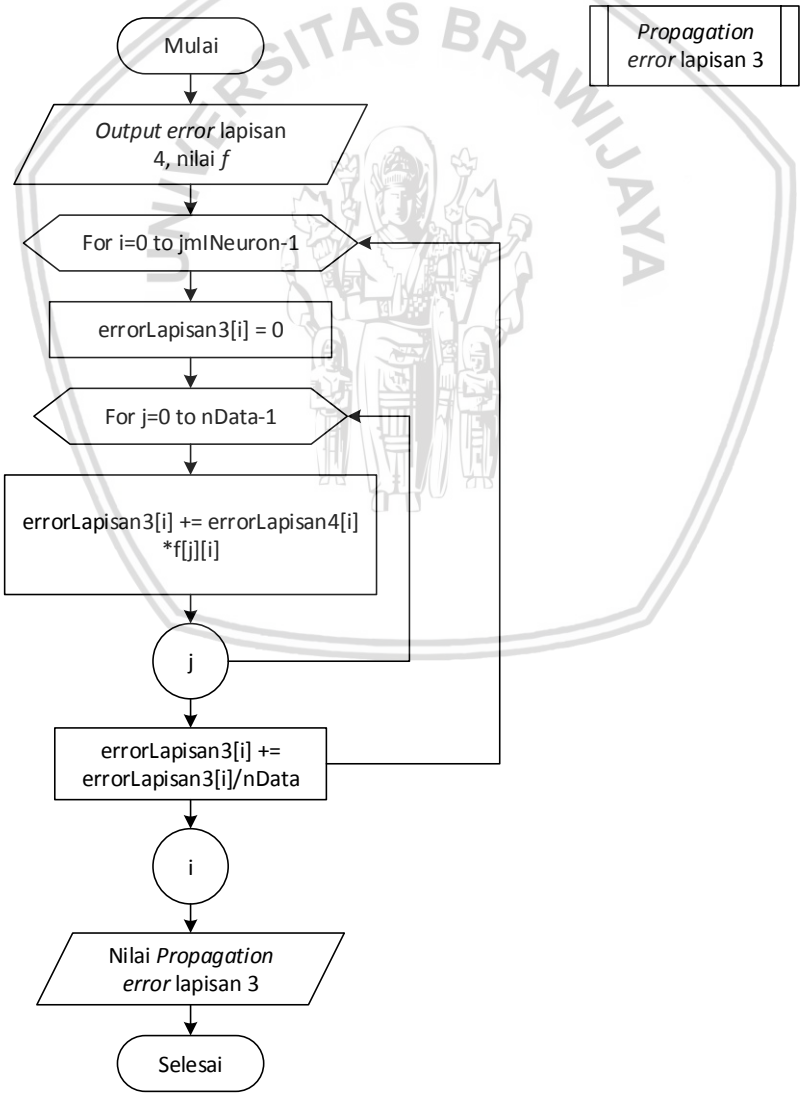
Gambar 4.13 Diagram Alir Perhitungan *Propagation Error* Lapisan 4

4.3.7.3 Perhitungan *Propagation Error* Lapisan 3

Untuk mendapatkan nilai *propagation error* pada lapisan 3 digunakan langkah-langkah berikut:

1. Menggunakan *output error* pada lapisan 4 dan nilai f yang di dapatkan pada lapisan 4.
2. Melakukan perulangan sebanyak jumlah *neuron*.
3. Menghitung nilai *propagation error* lapisan 3 menggunakan persamaan 2.22 dan 2.23.
4. *Output* nilai *error* lapisan 3.

Proses perhitungan nilai *propagation error* lapisan 3 dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.14 berikut.



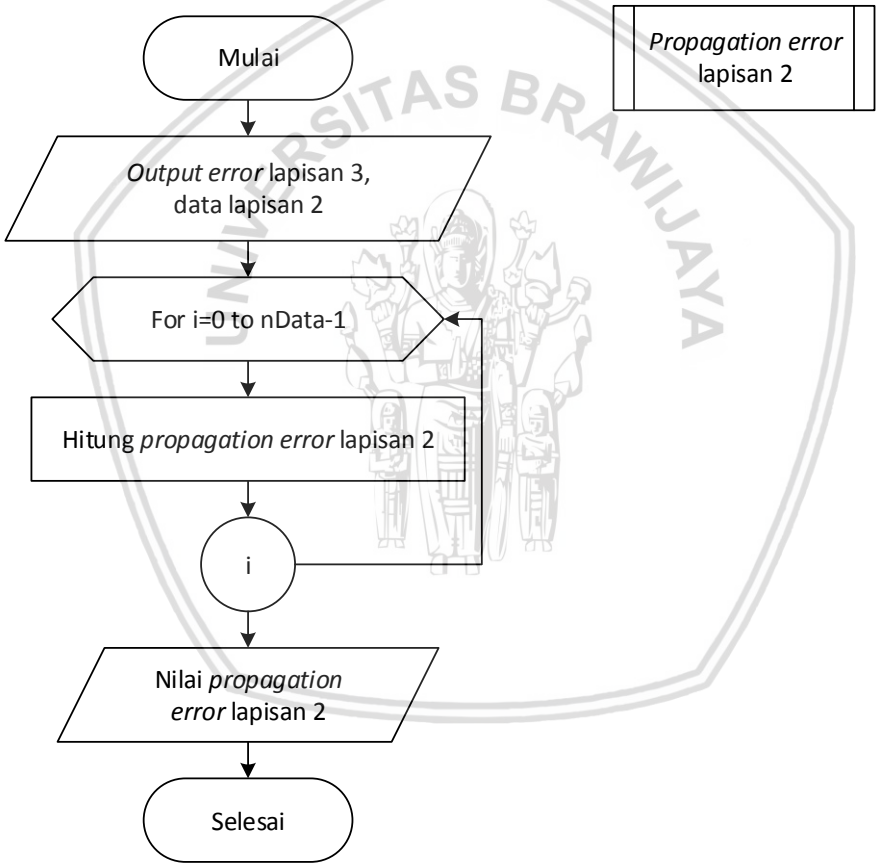
Gambar 4.14 Diagram Alir Perhitungan *Propagation Error* Lapisan 3

4.3.7.4 Perhitungan *Propagation Error* Lapisan 2

Untuk mendapatkan nilai *propagation error* lapisan 2 digunakan langkah-langkah berikut:

1. Menggunakan *output error* lapisan 3 dan data pada lapisan 2 sebagai *input*.
2. Melakukan iterasi sebanyak jumlah data.
3. Menghitung nilai *propagation error* lapisan 2 menggunakan persamaan 2.24 dan 2.25.
4. *Output propagation error* lapisan 2.

Proses perhitungan nilai *propagation error* lapisan 2 dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.15 berikut.



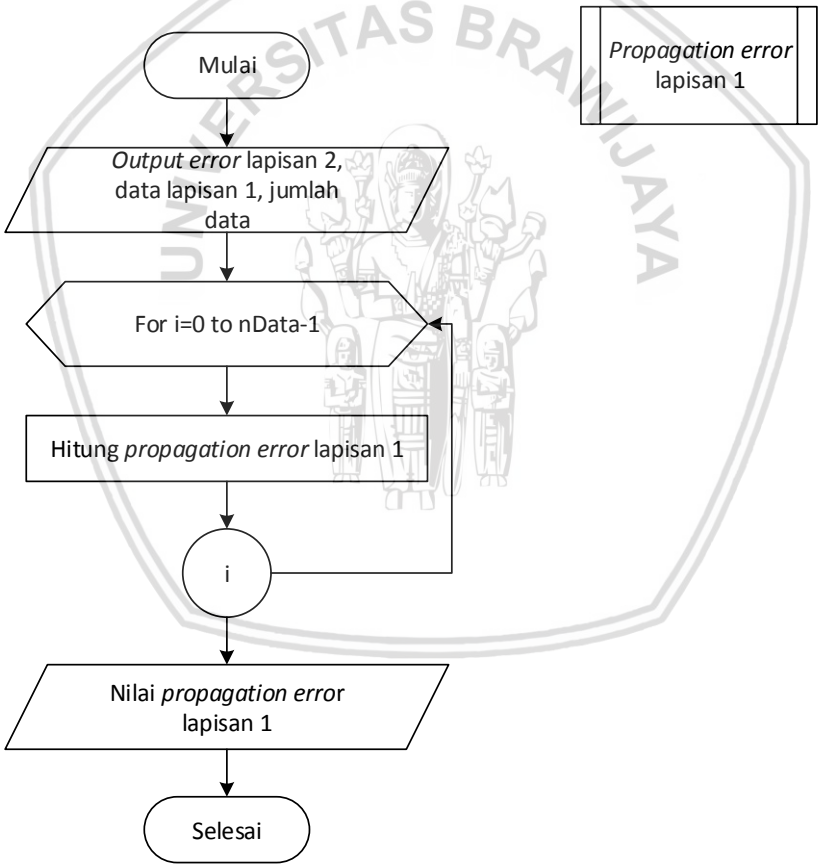
Gambar 4.15 Diagram Alir Perhitungan *Propagation Error* Lapisan 2

4.3.7.5 Perhitungan *Propagation Error* Lapisan 1

Untuk mendapatkan nilai *propagation error* pada lapisan 1 digunakan langkah-langkah berikut:

1. Data pada lapisan 1, jumlah data dan *output error* lapisan 2 digunakan sebagai *input*.
2. Melakukan proses iterasi sebanyak jumlah data pada neuron lapisan 1.
3. Melakukan perhitungan nilai *propagation error* lapisan 1 menggunakan persamaan 2.26, 2.27, 2.28, dan 2.29
4. *Output* nilai *propagation error* lapisan 1.

Proses perhitungan nilai *propagation error* lapisan 1 dapat digambarkan dalam bentuk diagram alir seperti pada Gambar 4.16 berikut.



Gambar 4.16 Diagram Alir Perhitungan *Propagation Error* Lapisan 1

4.4 Penyelesaian Masalah Menggunakan Algoritme ANFIS

Berikut ini adalah perhitungan manual dalam memprediksi curah hujan menggunakan metode ANFIS berdasarkan tiga buah variable *input* yaitu, suhu (x_1), kelembapan udara(x_2), kecepatan angin (x_3). Tabel 4.1 merupakan tablel yang menyatakan data *input*.

Tabel 4.1 Data Input

Tahun	Waktu	*Input*			*Output*
		Suhu Udara (x1)	Kelembapan Udara (x2)	Kecepatan Angin (x3)	
2009-2010	12 sept-21 sept	23,48	77,7	6,5	0,4
	22 sept-1 okt	23,37	70	5,9	0,1
	2 okt-11 okt	23,77	74,7	5,7	0
	12 okt-21 okt	24,67	68,5	6,6	2,2
	22 okt-31 okt	24,97	67,3	5,8	1,1
	1 nov-10 nov	25,89	63,4	8,3	0
	11 nov-20 nov	25,02	73,2	6,2	12,1
	21 nov-30 nov	23,82	80,8	4,1	7,9
	1 des-10 des	24,66	75,4	4	6,6
	11 des-20 des	24,45	73,5	5,2	0,7
	21 des-30 des	23,88	85	4,3	12,4
	31 des-9 jan	24,18	83,6	4	12,1
Min		23,37	63,4	4	0
Max		25,89	85	8,3	12,4

4.4.1 Pengelompokan data dengan K-Means *Clustering*

Algoritme *K-Means Clustering* diperlukan untuk mengelompokkan data dalam memprediksi curah hujan menggunakan metode ANFIS. Pengelompokan data dilakukan dengan tujuan untuk mencari nilai *mean* (c) serta nilai standar deviasi (a) yang kemudian akan digunakan untuk melakukan perhitungan *fuzzy generalized bell* yang menghasilkan derajat keanggotaan. Hasil tersebut akan digunakan untuk melakukan perhitungan pada setiap lapisan arsitektur ANFIS. Pada tahap pengelompokan data dengan k-means *clustering*, terdapat 12 data di atas yang akan dikelompokkan ke dalam 3 cluster yaitu tinggi, sedang dan rendah. Langkah-langkah dari proses *clustering* adalah sebagai berikut:

1. Normalisasi data *input* menggunakan Persamaan 2.5. Berikut contoh perhitungan normalisasi data:

$$x_{ij} = \frac{x_{ij} - xMinj}{xMaxj - xMinj}$$

$$x_{11} = \frac{23{,}48 - 23{,}37}{25{,}89 - 23{,}37} = 0{,}04365$$

$$x_{12} = \frac{77{,}7 - 63{,}4}{85 - 63{,}4} = 0{,}66204$$

$$x_{13} = \frac{6{,}5 - 4}{8{,}3 - 4} = 0{,}58140$$

Lakukan normalisasi untuk setiap data. Tabel 4.2 merupakan tabel hasil data yang telah dinormalisasi. Dari data yang telah ternormalisasi, dipilih 3 buah data secara *random* yang akan digunakan sebagai pusat *cluster* awal.

Tabel 4.2 Data *Input* Hasil Normalisasi

Tahun	Waktu	*Input*			*Output*
		x1	x2	x3	
2009-2010	12 sept-21 sept	0,04365	0,66204	0,58140	0,03226
	22 sept-1 okt	0	0,30556	0,44186	0,00806
	2 okt-11 okt	**0,15873**	**0,52315**	**0,39535**	**0**
	12 okt-21 okt	0,51587	0,23611	0,60465	0,17742
	22 okt-31 okt	**0,63492**	**0,18056**	**0,41860**	**0,08871**
	1 nov-10 nov	1	0	1	0
	11 nov-20 nov	0,65476	0,45370	0,51163	0,97581
	21 nov-30 nov	**0,17857**	**0,80556**	**0,02326**	**0,63710**
	1 des-10 des	0,51190	0,55556	0	0,53226
	11 des-20 des	0,42857	0,46759	0,27907	0,05645
	21 des-30 des	0,20238	1	0,06977	1
	31 des-9 jan	0,32143	0,93519	0	0,97581

Nilai dari *cluster* awal yang dipilih secara random ditunjukkan pada tabel 4.3 berikut ini.

Tabel 4.3 Nilai Pusat *Cluster* Awal

	x1	x2	x3
c1	0,15873	0,52315	0,39535
c2	0,63492	0,18056	0,41860
c3	0,17857	0,80556	0,02326

2. Melakukan perhitungan iterasi pertama dengan menghitung jarak antara tiap data dengan nilai pusat cluster awal dan inisialisasi awal nilai *threshold* = 0,8. Perhitungan jarak menggunakan rumus *Euclidean Distance*, kemudian mencari nilai minimum (jarak terdekat) dari masing-masing data untuk menentukan anggota baru setiap cluster. Tabel 4.4 merupakan hasil dari perhitungan iterasi pertama.

Tabel 4.4 Data Hasil Iterasi Pertama

No	dc1	dc2	dc3	MIN	*Cluster*	Jarak Ke Pusat *Cluster*		
						C1	C2	C3
1	0,25913	0,77970	0,59188	0,25913	1	*		
2	0,27332	0,64753	0,67610	0,27332	1	*		
3	0	0,58708	0,46755	0	1	*		
4	0,50373	0,22775	0,88094	0,22775	2		*	
5	0,58708	0	0,86901	0	2		*	
6	1,16061	0,70986	1,50920	0,70986	2		*	
7	0,51419	0,28924	0,76751	0,28924	2		*	
8	0,46755	0,86901	0	0	3			*
9	0,53112	0,57532	0,41732	0,41732	3			*
10	0,29903	0,38005	0,49210	0,29903	1	*		
11	0,57905	0,99008	0,20134	0,20134	3			*
12	0,59376	0,91814	0,19430	0,19430	3			*

3. Untuk melanjutkan iterasi selanjutnya dilakukan perhitungan untuk mencari nilai pusat *cluster* yang baru. Nilai dari pusat *cluster* yang baru diperoleh dengan menghitung nilai rata-rata data tiap anggota *cluster* dan nilai pusat *cluster* sebelumnya dengan menggunakan Persamaan 2.7. Setelah didapatkan nilai pusat *cluster* yang baru, dihitung kembali jarak tiap data dengan nilai pusat *cluster* yang baru. Berikut adalah contoh perhitungan nilai pusat *cluster* baru.

   Contoh perhitungan nilai pusat *cluster* baru :

$$C_i X_j^{(t+1)} = \frac{x_1 + x_2 + x_3 + \cdots + x_n + C_i X_j^{(t)}}{n+1}$$

$$C_1 X_1^{(2)} = \frac{0{,}25913+0{,}27332+0+0{,}29903+0{,}15873}{5} = 0{,}19804$$

$$C_1 X_2^{(2)} = \frac{0{,}77970+0{,}64753+0{,}58708+0{,}38005+0{,}52315}{5} = 0{,}58350$$

$$C_1 X_3^{(2)} = \frac{0{,}59188 + 0{,}67610 + 0{,}46755 + 0{,}49210 + 0{,}39535}{5} = 0{,}52460$$

   Tabel 4.5 merupakan tabel hasil perhitungan nilai pusat *cluster* yang baru dan table 4.6 merupakan hasil *clustering* untuk iterasi kedua.

Tabel 4.5 Pusat *Cluster* Baru pada Iterasi Kedua

	C1 (baru)	C2 (baru)	C3 (baru)
X1	**0,19804**	0,68011	0,47001
X2	**0,58350**	0,28148	0,83162
X3	**0,52460**	0,88905	0,16724

Tabel 4.6 Data *Clustering* Hasil Iterasi Kedua

No	dc1	dc2	dc3	MIN	*Cluster* Baru	*Cluster* Lama
1	0,18229	0,80284	0,61811	0,18229	1	1
2	0,35117	0,81431	0,75701	0,35117	1	1
3	0,14796	0,75762	0,49405	0,14796	1	1
4	0,47760	0,33154	0,74031	0,33154	2	2
5	0,60371	0,48327	0,71712	0,48327	2	2
6	1,09983	0,44031	1,29072	0,44031	2	2
7	0,47498	0,41564	0,54365	0,41564	2	2
8	0,54866	1,12951	0,32611	0,32611	3	3
9	0,61196	0,94542	0,32548	0,32548	3	3
10	0,35618	0,68556	0,38306	0,35618	1	1
11	0,61673	1,18984	0,33087	0,33087	3	3
12	0,64351	1,16034	0,24652	0,24652	3	3
Total	1,03760	1,67075	0,57739			

4. Pada iterasi kedua tidak terdapat perubahan terhadap anggota setiap *cluster*. Selanjutnya menghitung nilai fungsi objektif (F). Berikut merupakan perhitungan nilai fungsi objektif (F):

   $Threshold = 0{,}8$

   $F\ Lama\ = 0$

   $F\ Baru\ = 1{,}03760 + 1{,}67075 + 0{,}57739 = 3{,}28574$

   $Delta = |F\ Baru - F\ Lama| = |3{,}28574 - 0| = 3{,}28574 > 0{,}8$

   Pada perhitungan nilai fungsi objektif ini, Delta lebih besar daripada nilai *threshold*. Terdapat dua kondisi iterasi *K-Means Clustering* dapat dihentikan yaitu tidak terdapat perubahan anggota *cluster* lama dan anggota *cluster* baru atau nilai dari fungsi objektif lebih kecil dari nilai *threshold*. Oleh karena itu pada iterasi kedua, iterasi akan dihentikan karena salah satu kondisi permberhentian iterasi sudah terpenuhi yaitu tidak terdapat perubahan anggota *cluster* lama dan anggota *cluster* baru.

4.4.2 Menghitung *Mean* dan Standar Deviasi

Nilai *mean* dan standar deviasi didapatkan dari hasil *clustering* data yang telah dilakukan sebelumnya. Nilai *mean* dan standar deviasi digunakan sebagai *input* untuk menghitung nilai pada lapisan 1 dalam arsitektur ANFIS. Berikut langkah-langkah untuk mendapatkan nilai *mean* dan standar deviasi.

1. Buat pengelompokan data aktual berdasarkan hasil akhir *clustering* data. Tabel 4.7 merupakan tabel data aktual berdasarkan pengelompokan k-means *clustering*.

Tabel 4.7 Data Aktual Berdasarkan Pengelompokan K-Means *Clustering*

Waktu	x1	x2	x3	*Cluster*
12 sept-21 sept	23,48	77,7	6,5	1
22 sept-1 okt	23,37	70	5,9	1
2 okt-11 okt	23,77	74,7	5,7	1
12 okt-21 okt	24,67	68,5	6,6	2
22 okt-31 okt	24,97	67,3	5,8	2
1 nov-10 nov	25,89	63,4	8,3	2
11 nov-20 nov	25,02	73,2	6,2	2
21 nov-30 nov	23,82	80,8	4,1	3
1 des-10 des	24,66	75,4	4	3
11 des-20 des	24,45	73,5	5,2	1
21 des-30 des	23,88	85	4,3	3
31 des-9 jan	24,18	83,6	4	3

2. Hitung nilai *mean* (c) terhadap setiap variable menggunakan Persamaan 2.8. Berikut contoh perhitungan niai *mean*:

$$\tilde{x} = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

$$C_1(x_1) = \frac{23{,}48 + 23{,}37 + 23{,}77 + 24{,}45}{4} = 23{,}768$$

$$C_1(x_2) = \frac{77{,}7 + 70 + 74{,}7 + 73{,}5}{4} = 73{,}975$$

$$C_1(x_3) = \frac{6{,}5 + 5{,}9 + 5{,}7 + 5{,}2}{4} = 5{,}825$$

Tabel 4.8 merupakan hasil dari perhitungan nilai *mean* (c) pada setiap variabel *input*:

Tabel 4.8 Hasil Perhitungan *Mean*

mean (c)	x1	x2	x3
c1	**23,768**	**73,975**	**5,825**
c2	25,138	68,100	6,725
c3	24,135	81,200	4,100

3. Hitung nilai standar deviasi (a) menggunakan Persamaan 2.9. Berikut contoh perhitungan standar deviasi:

$$\sigma = \sqrt{\frac{\sum(x_i - \tilde{x})^2}{(n-1)}}$$

$$a_1(x_1) = \sqrt{\frac{(23{,}48-23{,}768)^2+(23{,}37-23{,}768)^2+(23{,}77-23{,}768)^2+(24{,}45-23{,}768)^2}{(4-1)}} = 0{,}48527$$

Tabel 4.9 merupakan hasil dari perhitungan nilai standar deviasi (a) pada setiap variabel *input*:

Tabel 4.9 Hasil Perhitungan Standar Deviasi

Standar deviasi (a)	x1	x2	x3
a1	**0,48527**	3,18473	0,53774
a2	0,52494	4,03733	1,09962
a3	0,38380	4,24264	0,14142

4.4.3 *Output* Lapisan 1 (Derajat Keanggotaan *Generalized Bell*)

Output lapisan 1 merupakan nilai derajat keanggotaan yang didapatkan dengan menggunakan fungsi keanggotaan *generalized bell* seperti pada Persamaan 2.2. Nilai *mean* (c) dan standar deviasi (a) yang telah didapatkan pada proses sebelumnya digunakan sebagai data *input* untuk menghitung derajat keanggotaan yang merupakan *output* pada lapisan 1. Berikut contoh perhitungan *output* lapisan 1:

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

$$\mu A_1(x_1) = \frac{1}{1+\left|\frac{23{,}48-23{,}768}{0{,}48572}\right|^{2}} = 0{,}74020$$

$$\mu A_2(x_1) = \frac{1}{1+\left|\frac{23{,}48-25{,}138}{0{,}52494}\right|^{2}} = 0{,}09116$$

$$\mu A_3(x_1) = \frac{1}{1+\left|\frac{23{,}48-24{,}135}{0{,}38380}\right|^{2}} = 0{,}25558$$

$$\mu B_1(x_2) = \frac{1}{1+\left|\frac{77{,}7-73{,}975}{3{,}18473}\right|^{2}} = 0{,}42229$$

$$\mu B_2(x_2) = \frac{1}{1+\left|\frac{77{,}7-68{,}1}{4{,}03733}\right|^{2}} = 0{,}15029$$

$$\mu B_3(x_2) = \frac{1}{1+\left|\frac{77{,}7-81{,}2}{4{,}24264}\right|^{2}} = 0{,}59504$$

$$\mu C_1(x_3) = \frac{1}{1+\left|\frac{6{,}5-5{,}825}{0{,}53774}\right|^{2}} = 0{,}38825$$

$$\mu C_2(x_3) = \frac{1}{1+\left|\frac{6{,}5-6{,}725}{1{,}09962}\right|^{2}} = 0{,}95981$$

$$\mu C_3(x_3) = \frac{1}{1+\left|\frac{6{,}5-4{,}1}{0{,}14142}\right|^{2}} = 0{,}00346$$

Tabel 4.10 merupakan hasil perhitungan derajat keanggotaan dengan menggunakan fungsi keanggotaan *generalized bell*.

Tabel 4.10 Hasil Perhitungan Nilai *Output* Lapisan 1

gbell	μA1	μA2	μA3	μB1	μB2	μB3	μC1	μC2	μC3
1	**0,74020**	**0,09116**	**0,25558**	**0,42229**	**0,15029**	**0,59504**	**0,38825**	**0,95981**	**0,00346**
2	0,59846	0,08106	0,20109	0,39095	0,81868	0,12549	0,98092	0,63984	0,00613
3	0,99997	0,12843	0,52509	0,95073	0,27230	0,29876	0,94874	0,53508	0,00775
4	0,22428	0,55768	0,33977	0,25282	0,99028	0,10040	0,32498	0,98724	0,00319
5	0,14005	0,90759	0,17442	0,18543	0,96222	0,08522	0,99784	0,58561	0,00687
6	0,04968	0,32734	0,04564	0,08315	0,42459	0,05376	0,04508	0,32771	0,00113
7	0,13052	0,95229	0,15830	0,94409	0,38525	0,21951	0,67281	0,81437	0,00451
8	0,98843	0,13700	0,59751	0,17881	0,09178	0,99119	0,08857	0,14928	1,00000
9	0,22818	0,54722	0,34829	0,83319	0,23423	0,34857	0,07988	0,14003	0,66667
10	0,33579	0,36829	0,59751	0,97824	0,35856	0,23289	0,42538	0,34208	0,01626
11	0,94900	0,14840	0,69375	0,07702	0,05399	0,55487	0,11059	0,17055	0,33333
12	0,58053	0,23110	0,98644	0,09868	0,06354	0,75758	0,07988	0,14003	0,66667

4.4.4 *Output* Lapisan 2 (*Fire Strength*)

Nilai *output* lapisan 2 merupakan nilai derajat pengaktifan atau biasa disebut dengan nilai *fire strength* (w). Nilai *fire strength* didapatkan dengan cara mengalikan setiap sinyal masukan yang datang dari lapisan sebelumnya seperti pada Persamaan 2.12. Berikut contoh perhitungan nilai *output* lapisan 2 (*fire strength*):

$$O_{2,i} = w_i = \mu A_i(x)\mu B_i(y), i$$

$$O_{2,1} = 0{,}74020 \times 0{,}42229 \times 0{,}38825 = 0{,}12136$$

$$O_{2,2} = 0{,}09116 \times 0{,}15029 \times 0{,}95981 = 0{,}01315$$

$$O_{2,3} = 0{,}25558 \times 0{,}59504 \times 0{,}00346 = 0{,}00053$$

Tabel 4.11 merupakan hasil perhitungan nilai *output* lapisan 2 (*fire strength*).

Tabel 4.11 Hasil Perhitungan *Output* Lapisan 2 (*Fire Strength)*

Data Ke-	O2,1	O2,2	O2,3
1	**0,12136**	**0,01315**	**0,00053**
2	0,22950	0,04246	0,00015
3	0,90197	0,01871	0,00122
4	0,01843	0,54521	0,00011
5	0,02591	0,51142	0,00010
6	0,00019	0,04555	0,00000
7	0,08291	0,29877	0,00016
8	0,01565	0,00188	0,59224
9	0,01519	0,01795	0,08093
10	0,13973	0,04517	0,00226

Data Ke-	O2,1	O2,2	O2,3
11	0,00808	0,00137	0,12831
12	0,00458	0,00206	0,49820

4.4.5 *Output* Lapisan 3 (*Normalized Fire Strength*)

Pada lapisan ini akan dilakukan perhitungan nilai *normalized fire strength*. Nilai *normalized fire strength* didapatkan dengan cara melakukan standarisasi/normalisasi terhadap nilai *fire strength* yang didapatkan dari lapisan sebelumnya seperti pada Persamaan 2.13. *Output* dari lapisan ke-3 ini adalah nilai *fire strength* yang sudah terstandarisasi/ternormalisasi. Berikut contoh perhitungan *normalized fire strength*:

$$O_{3,i} = \overline{w} = \frac{w_i}{w_1 + w_2}, i$$

$$O_{3,1} = \frac{0{,}12136}{0{,}12136 + 0{,}01315 + 0{,}00053} = 0{,}89872$$

$$O_{3,2} = \frac{0{,}01315}{0{,}12136 + 0{,}01315 + 0{,}00053} = 0{,}09738$$

$$O_{3,3} = \frac{0{,}00053}{0{,}12136 + 0{,}01315 + 0{,}00053} = 0{,}00390$$

Tabel 4.12 merupakan hasil dari normalisasi nilai *fire strength* yang merupakan nilai output 3.

Tabel 4.12 Hasil Perhitungan *Output* Lapisan 3 *(Normalized Fire Strength)*

Data ke-	O3,1	O3,2	O3,3
1	**0,89872**	**0,09738**	**0,00390**
2	0,84340	0,15603	0,00057
3	0,97838	0,02030	0,00132
4	0,03269	0,96712	0,00019
5	0,04822	0,95159	0,00019
6	0,00407	0,99587	0,00006
7	0,21713	0,78246	0,00041
8	0,02567	0,00308	0,97125
9	0,13314	0,15735	0,70951
10	0,74656	0,24135	0,01209
11	0,05867	0,00992	0,93141
12	0,00906	0,00407	0,98686

4.4.6 Perhitungan Parameter Konsekuen dengan *Least Square Estimator* (LSE)

Perhitungan nilai parameter konsekuen dengan algoritme LSE berguna untuk mencari nilai *output* pada lapisan ke-4. Berikut langkah-langkah yang harus dilakukan untuk mendapatkan nilai parameter konsekuen:

1. Menghitung matriks desain (A) dengan cara melakukan perkalian antara nilai *output* pada lapisan 3 dengan nilai data aktual. Contoh perhitungan matriks desain:

$$C_{11} = 0{,}89872 \times 23{,}48 = 21{,}10206$$

$$C_{12} = 0{,}89872 \times 77{,}7 = 69{,}83092$$

$$C_{13} = 0{,}89872 \times 6{,}5 = 5{,}84171$$

$$C_{10} = 0{,}89872$$

Tabel 4.13 merupakan hasil matriks desain (A) dari 3 variabel *input*, sehingga didapatkan matriks desain dengan dimensi 12 x 12.

Tabel 4.13 Matriks Desain (A)

No	c11	c12	c13	c10	c21	c22	c23	c20	c31	c32	c33	c30
1	**21,10206**	**69,83092**	**5,84171**	**0,89872**	2,28644	7,56628	0,63296	0,09738	0,09150	0,30281	0,02533	0,00390
2	19,71022	59,03787	4,97605	0,84340	3,64649	10,92230	0,92059	0,15603	0,01330	0,03982	0,00336	0,00057
3	23,25617	73,08522	5,57678	0,97838	0,48248	1,51625	0,11570	0,02030	0,03135	0,09854	0,00752	0,00132
4	0,80637	2,23901	0,21573	0,03269	23,85887	66,24777	6,38300	0,96712	0,00476	0,01322	0,00127	0,00019
5	1,20395	3,24493	0,27965	0,04822	23,76130	64,04227	5,51924	0,95159	0,00475	0,01279	0,00110	0,00019
6	0,10541	0,25813	0,03379	0,00407	25,78302	63,13802	8,26570	0,99587	0,00157	0,00385	0,00050	0,00006
7	5,43252	15,89370	1,34619	0,21713	19,57720	57,27623	4,85127	0,78246	0,01028	0,03007	0,00255	0,00041
8	0,61150	2,07428	0,10525	0,02567	0,07333	0,24874	0,01262	0,00308	23,13517	78,47698	3,98212	0,97125
9	3,28324	10,03877	0,53256	0,13314	3,88018	11,86398	0,62939	0,15735	17,49658	53,49725	2,83805	0,70951
10	18,25346	54,87236	3,88213	0,74656	5,90096	17,73909	1,25501	0,24135	0,29558	0,88854	0,06286	0,01209
11	1,40107	4,98707	0,25229	0,05867	0,23686	0,84311	0,04265	0,00992	22,24206	79,16982	4,00506	0,93141
12	0,21919	0,75783	0,03626	0,00906	0,09848	0,34050	0,01629	0,00407	23,86233	82,50167	3,94745	0,98686

2. Langkah selanjutnya yaitu menentukan nilai *transpose* (A^T) dari matriks desain A. Tabel 4.14 merupakan bentuk *transpose* dari matriks desain A.

Tabel 4.14 Matriks *Transpose* dari Matriks Desain A (A^T)

No	c11	c12	c13	c10	c21	c22	c23	c20	c31	c32	c33	c30
1	21,10206	19,71022	23,25617	0,80637	1,20395	0,10541	5,43252	0,61150	3,28324	18,25346	1,40107	0,21919
2	69,83092	59,03787	73,08522	2,23901	3,24493	0,25813	15,89370	2,07428	10,03877	54,87236	4,98707	0,75783
3	5,84171	4,97605	5,57678	0,21573	0,27965	0,03379	1,34619	0,10525	0,53256	3,88213	0,25229	0,03626
4	0,89872	0,84340	0,97838	0,03269	0,04822	0,00407	0,21713	0,02567	0,13314	0,74656	0,05867	0,00906
5	2,28644	3,64649	0,48248	23,85887	23,76130	25,78302	19,57720	0,07333	3,88018	5,90096	0,23686	0,09848
6	7,56628	10,92230	1,51625	66,24777	64,04227	63,13802	57,27623	0,24874	11,86398	17,73909	0,84311	0,34050
7	0,63296	0,92059	0,11570	6,38300	5,51924	8,26570	4,85127	0,01262	0,62939	1,25501	0,04265	0,01629
8	0,09738	0,15603	0,02030	0,96712	0,95159	0,99587	0,78246	0,00308	0,15735	0,24135	0,00992	0,00407
9	0,09150	0,01330	0,03135	0,00476	0,00475	0,00157	0,01028	23,13517	17,49658	0,29558	22,24206	23,86233
10	0,30281	0,03982	0,09854	0,01322	0,01279	0,00385	0,03007	78,47698	53,49725	0,88854	79,16982	82,50167
11	0,02533	0,00336	0,00752	0,00127	0,00110	0,00050	0,00255	3,98212	2,83805	0,06286	4,00506	3,94745
12	0,00390	0,00057	0,00132	0,00019	0,00019	0,00006	0,00041	0,97125	0,70951	0,01209	0,93141	0,98686

3. Langkah selanjutnya yaitu melakukan perkalian antara matriks *transpose* A^T dengan matriks desain A (A^TA). Tabel 4.15 merupakan hasil perkalian antara matriks *transpose* A^T dengan matriks desain A.

Tabel 4.15 Perkalian Matriks *Transpose* (A^T) dengan Matriks Desain A (A^T A)

No	c11	c12	c13	c10	c21	c22	c23	c20	c31	c32	c33	c30
1	1752,61523	5471,98132	431,90968	73,77070	409,11088	1222,70058	98,25609	16,82233	116,36883	378,51453	20,16897	4,79212
2	5471,98132	17112,99618	135,95462	230,39314	1222,70058	3664,23999	293,87994	50,32371	378,51453	1237,18221	65,72868	15,60324
3	431,90968	1351,95462	107,35585	18,20307	98,25609	293,87994	23,87418	4,04394	20,16897	65,72868	3,53881	0,83139
4	73,77070	230,39314	18,20307	3,10608	16,82233	50,32371	4,04394	0,69224	4,79212	15,60324	0,83139	0,19739
5	409,11088	1222,70058	98,25609	16,82233	2250,58176	6060,34471	606,24502	89,51953	79,69002	247,64721	13,20577	3,24376
6	1222,70058	3664,23999	293,87994	50,32371	6060,34471	16392,31824	1620,86062	241,33582	247,64721	771,34963	41,06949	10,08502
7	98,25609	293,87994	23,87418	4,04394	606,24502	1620,86062	166,29627	24,06277	13,20577	41,06949	2,20126	0,53771
8	16,82233	50,32371	4,04394	0,69224	89,51953	241,33582	24,06277	3,56223	3,24376	10,08502	0,53771	0,13205
9	116,36883	378,51453	20,16897	4,79212	79,69002	247,64721	13,20577	3,24376	1905,58334	6481,47333	325,08063	79,15335
10	378,51453	1237,18221	65,72868	15,60324	247,64721	771,34963	41,06949	10,08502	6481,47333	22095,87193	1107,14855	269,34712
11	20,16897	65,72868	3,53881	0,83139	13,20577	41,06949	2,20126	0,53771	325,08063	1107,14855	55,53938	13,50808
12	4,79212	15,60324	0,83139	0,19739	3,24376	10,08502	0,53771	0,13205	79,15335	269,34712	13,50808	3,28832

4. Setelah didapatkan hasil perkalian dari matriks *transpose* A^T dan matriks desain A (A^TA), langkah selanjutnya adalah menghitung nilai *invers* dari matriks (A^TA). Tabel 4.16 merupakan hasil *invers* dari matriks $(A^TA)^{-1}$.

Tabel 4.16 Matriks *Inverse* $(A^T A)^{-1}$

No	c11	c12	c13	c10	c21	c22	c23	c20	c31	c32	c33	c30
1	45,45232	-3,17275	39,45020	-1073,09668	-10,32872	-3,04809	0,47398	456,33332	10,11578	-0,71701	28,73010	-298,39436
2	-3,17275	0,28647	-2,97040	71,34317	0,56931	0,20556	0,00222	-27,76880	-0,78923	0,05475	-2,34205	23,78925
3	39,45020	-2,97040	36,75620	-929,99398	-7,95490	-2,65942	-0,07929	374,84781	9,51743	-0,67524	27,28483	-281,76897
4	-1073,09668	71,34317	-929,99398	25592,00905	249,61928	72,55322	-11,11497	-10960,32929	-237,06250	16,89721	-667,41186	6961,34045
5	-10,32872	0,56931	-7,95490	249,61928	6,94480	0,80222	-1,71526	-215,91634	-2,60092	0,18899	-6,36390	72,28683
6	-3,04809	0,20556	-2,65942	72,55322	0,80222	0,26544	0,07108	-38,18102	-0,70839	0,04980	-1,97053	20,76736
7	0,47398	0,00222	-0,07929	-11,11497	-1,71526	0,07108	1,20350	30,12757	0,30729	-0,02489	0,35724	-6,77197
8	456,33332	-27,76880	374,84781	-10960,32929	-215,91634	-38,18102	30,12757	7746,30215	109,73311	-7,84582	286,66590	-3132,33692
9	10,11578	-0,78923	9,51743	-237,06250	-2,60092	-0,70839	0,30729	109,73311	10,43244	0,12320	14,56386	-319,92893
10	-0,71701	0,05475	-0,67524	16,89721	0,18899	0,04980	-0,02489	-7,84582	0,12320	0,06865	-0,84165	-5,20996
11	28,73010	-2,34205	27,28483	-667,41186	-6,36390	-1,97053	0,35724	286,66590	14,56386	-0,84165	56,55821	-510,80263
12	-298,39436	23,78925	-281,76897	6961,34045	72,28683	20,76736	-6,77197	-3132,33692	-319,92893	-5,20996	-510,80263	10193,63718

5. Langkah selanjutnya adalah menghitung nilai dari matriks parameter konsekuen dengan cara mengalikan antara proses matriks $(A^T A)^{-1}$ dengan matriks *y* yang merupakan target output. Tabel 4.18 merupakan matriks parameter konsekuen yang didapatkan dengan menggunakan algoritma LSE seperti pada Persamaan 2.18.

Tabel 4.17 Target Output

Output
0,4
0,1
0
2,2
1,1
0
12,1
7,9
6,6
0,7
12,4
12,1

Tabel 4.18 Parameter Konsekuen

c11	-9,04604
c12	0,23380
c13	-5,75361
c10	230,11031
c21	9,44113
c22	2,89952
c23	0,37513
c20	-431,22165
c31	2,66904
c32	1,11277
c33	-1,64962
c30	-138,94706

4.4.7 *Output* Lapisan 4

Nilai *output* lapisan 4 merupakan hasil perkalian antara nilai dari *output* pada lapisan 3 dengan parameter konsekuen seperti pada Persamaan 2.14. Langkah-langkah untuk mendapatkan nilai *output* lapisan 4 adalah sebagai berikut:

1. Menghitung nilai f dengan menggunakan parameter konsekuen.

Contoh perhitungan nilai f:

$$f = C_{i1}x_1 + C_{i2}x_2 + C_{i3}x_3 + C_{i0})$$

$$f_1 = \big((-9{,}04604 \times 23{,}48) + (0{,}23380 \times 77{,}7) + (-5{,}75361 \times 6{,}5) + (230{,}11031)\big) = -1{,}52316$$

$$f_2 = ((9{,}44113 \times 23{,}48) + (2{,}89952 \times 77{,}7) + (0{,}37513 \times 6{,}5) + (-431{,}22165)) = 18{,}18688$$

$$f_3 = \big((2{,}66904 \times 23{,}48) + (1{,}11277 \times 77{,}7) + (-1{,}64962 \times 6{,}5) + (-138{,}94706)\big) = -0{,}53834$$

Tabel 4.19 merupakan hasil perhitungan nilai f.

Tabel 4.19 Hasil Perhitungan Nilai f

f1	f2	f3
-1,52316	**18,18688**	**-0,53834**
1,12382	-5,40300	-8,41049
-0,24502	11,92615	-1,78293
-15,01426	2,78379	-7,76462
-13,40574	1,83660	-6,97954
-37,02395	0,15216	-12,98787
-14,78008	19,56586	-0,94059
9,93463	29,48504	7,77781
1,64881	21,72069	4,17581
-3,80008	14,67913	-0,47850
9,22310	42,30451	12,28166
7,90805	40,96498	12,01938

2. Menghitung nilai *output* lapisan 4 dengan melakukan perkalian antara nilai *output* lapisan 3 dengan nilai f. Berikut perhitungan nilai *output* lapisan 4:

   $O_{4,1} = 0{,}89872 \times (-1{,}52316) = -1{,}36891$

   $O_{4,2} = 0{,}09738 \times 18{,}18688 = 1{,}77100$

   $O_{4,3} = 0{,}00390 \times (-0{,}53834) = -0{,}00210$

   Tabel 4.20 merupakan hasil perhitungan nilai *output* lapisan 4.

Tabel 4.20 Nilai *Output* Lapisan 4

No	O4,1	O4,2	O4,3
1	**-1,36891**	**1,77100**	**-0,00210**
2	0,94783	-0,84305	-0,00478
3	-0,23972	0,24207	-0,00235
4	-0,49076	2,69226	-0,00150
5	-0,64637	1,74770	-0,00133
6	-0,15074	0,15153	-0,00079
7	-3,20915	15,30954	-0,00039
8	0,25504	0,09077	7,55419
9	0,21952	3,41769	2,96279
10	-2,83700	3,54278	-0,00578
11	0,54113	0,41962	11,43925
12	0,07169	0,16685	11,86147

4.4.8 *Output* Lapisan 5

Output lapisan 5 merupakan nilai *output* jaringan yang didapatkan dari hasil penjumlahan seluruh sinyal yang masuk pada lapisan sebelumnya dengan menggunakan Persamaan 2.15. Berikut merupakan perhitungan *output* lapisan 5:

$$O_{5,i} = \sum_i wifi = \frac{\sum_i wifi}{\sum_i wi}$$

$$O_{5,1} = (-1{,}36891) + 1{,}77100 + (-0{,}00210) = 0{,}40000000002$$

Tabel 4.21 merupakan hasil perhitungan nilai output lapisan 5.

Tabel 4.21 Hasil *Output* Lapisan 5

No	O5
1	**0,40000000002**
2	0,09999999988
3	0,00000000010
4	2,20000000002
5	1,09999999991
6	0,00000000008
7	12,10000000003
8	7,89999999998
9	6,60000000000
10	0,70000000010
11	10,20334
12	11,87539

4.4.9 Model Propagasi *Error*

Model propagasi *error* bertujuan untuk melakukan perbaikan parameter premis yaitu perbaikan nilai *mean* (c) dan nilai standar deviasi (a) dengan menggunakan algoritme *steepest descent*. Model propagasi *error* dilakukan dengan menghitung nilai *error* pada setiap lapisan, dimulai dari nilai propagasi *error* pada lapisan 5 hingga lapisan 1. Berikut adalah langkah-langkah untuk melakukan perbaikan parameter premis menggunakan algoritme *steepest descent*:

1. **Menghitung Propagasi *Error* Lapisan 5**

Tabel 4.22 merupakan perbandingan *output* jaringan dan *output* aktual.

Tabel 4.22 Perbandingan *Output* Jaringan dan *Output* Aktual

Waktu	O5	aktual	*error*
1 Jan - 10 Jan	0,40000000002	0,4	1,81001E-11
11 Jan - 20 Jan	0,09999999988	0,1	-1,2473E-10
21 Jan - 30 Jan	0,00000000010	0	1,01059E-10
31 Jan - 9 Feb	2,20000000002	2,2	2,04423E-11
10 Feb - 19 Feb	1,09999999991	1,1	-9,39531E-11
20 Feb - 1 Mar	0,00000000008	0	8,26128E-11
2 Mar - 11 Mar	12,10000000003	12,1	2,98623E-11
12 Mar - 21 Mar	7,89999999998	7,9	-2,1549E-11
22 Mar - 31 Mar	6,60000000000	6,6	1,40332E-13
1 Apr - 10 Apr	0,70000000010	0,7	1,01874E-10
11 Apr - 20 Apr	12,40000000001	12,4	1,20881E-11
21 Apr - 30 Apr	12,09999999998	12,1	-2,25828E-11

Berikut adalah contoh perhitungan propagasi *error* lapisan 5 dengan menggunakan Persamaan 2.19.

$\varepsilon_5 = -2(y_p - y'_p)$

$\varepsilon_5 = -2(0{,}4 - 0{,}40000000002) = 3{,}62002\text{E} - 11$

Tabel 4.23 merupakan hasil perhitungan dari nilai propagasi *error* pada lapisan 5.

Tabel 4.23 Nilai Propagasi *Error* Lapisan 5

ε5,1	**3,62002E-11**
ε5,2	-2,49459E-10
ε5,3	2,02119E-10
ε5,4	4,08846E-11
ε5,5	-1,87906E-10
ε5,6	1,65226E-10
ε5,7	5,97247E-11
ε5,8	-4,3098E-11
ε5,9	2,80664E-13
ε5,10	2,03748E-10
ε5,11	2,41762E-11
ε5,12	-4,51656E-11
Rata-Rata	**1,72274E-11**

2. **Menghitung Propagasi *Error* Lapisan 4**

Propagasi *error* lapisan 4 didapatkan dari Persamaan 2.20 dan Persamaan 2.21. Berdasarkan persamaan tersebut, nilai propagasi *error* 4 didapatkan dari nilai propagasi *error* lapisan 5. Tabel 4.24 berikut merupakan tabel nilai propagasi *error* lapisan 4.

Tabel 4.24 Nilai Propagasi *Error* Lapisan 4

ε4a	**1,72274E-11**
ε4b	**1,72274E-11**
ε4c	**1,72274E-11**

3. **Menghitung Propagasi *Error* Lapisan 3**

Propagasi *error* lapisan 3 dihitung menggunakan persamaan 2.22 dan persamaan 2.23. Berikut adalah contoh perhitungan propagasi *error* lapisan 3:

$$\varepsilon_{3a} = 1{,}72274E-11 \times -1{,}52316 = -2{,}62402E-11$$

$$\varepsilon_{3b} = 1{,}72274E-11 \times 18{,}18688 = 3{,}13313E-10$$

$$\varepsilon_{3c} = 1{,}72274E-11 \times -0{,}53834 = -9{,}27421E-12$$

Tabel 4.25 merupakan hasil perhitungan dari nilai propagasi *error* pada lapisan 3.

Tabel 4.25 Nilai Propagasi *Error* Lapisan 3

Data ke-	ε3a	ε3b	ε3c
1	**-2,62402E-11**	**3,13313E-10**	**-9,27421E-12**
2	1,93606E-11	-9,30799E-11	-1,44891E-10
3	-4,22106E-12	2,05457E-10	-3,07153E-11
4	-2,58657E-10	4,79575E-11	-1,33765E-10
5	-2,30947E-10	3,16399E-11	-1,2024E-10
6	-6,37828E-10	2,62131E-12	-2,23748E-10
7	-2,54623E-10	3,3707E-10	-1,6204E-11
8	1,71148E-10	5,07952E-10	1,33992E-10
9	2,84047E-11	3,74192E-10	7,19385E-11
10	-6,54657E-11	2,52884E-10	-8,24329E-12
11	1,5889E-10	7,28799E-10	2,11582E-10
12	1,36236E-10	7,05722E-10	2,07063E-10
Rata-Rata	**-8,03285E-11**	**2,84544E-10**	**-5,20878E-12**

4. **Menghitung Propagasi *Error* Lapisan 2**

Propagasi *error* lapisan 2 dihitung menggunakan Persamaan 2.24 dan Persamaan 2.25. Berikut adalah contoh perhitungan nilai propagasi *error* lapisan 2:

$$\varepsilon_{2a} = \frac{\left(\left(-8{,}03285\text{E}-11(0{,}01315+0{,}00053)\right)-(2{,}84544\text{E}-10\times 0{,}01315)-(-5{,}20878\text{E}-12\times 0{,}00053)\right.}{(0{,}12136+0{,}01315+0{,}00053)^2}$$

$$= -2{,}65294\text{E}-10$$

$$\varepsilon_{2b} = \frac{\left(\left(2{,}84544\text{E}-10(0{,}12136+0{,}00053)\right)-(-8{,}03285\text{E}-11\times 0{,}12136)-(-5{,}20878\text{E}-12\times 0{,}00053)\right)}{(0{,}12136+0{,}01315+0{,}00053)^2}$$

$$= 2{,}43681\text{E}-09$$

$$\varepsilon_{3b} = \frac{\left(\left(-5{,}20878\text{E}-12(0{,}01315+0{,}12136)\right)-(2{,}84544\text{E}-10\times 0{,}01315)-(-8{,}03285\text{E}-11\times 0{,}12136)\right)}{(0{,}12136+0{,}01315+0{,}00053)^2}$$

$$= 2{,}91014E-10$$

Tabel 4.26 merupakan hasil perhitungan nilai propagasi *error* pada lapisan 2.

Tabel 4.26 Propagasi *Error* Lapisan 2

Data ke-	ε2a	ε2b	ε2c
1	**-2,65294E-10**	**2,43681E-09**	**2,91014E-10**
2	-2,09376E-10	1,13149E-09	6,66804E-11
3	-8,14105E-12	3,87644E-10	7,3343E-11
4	-6,25969E-10	2,12545E-11	-4,92719E-10
5	-6,46081E-10	3,28372E-11	-5,06306E-10
6	-7,94509E-09	3,28668E-11	-6,3026E-09
7	-7,47793E-10	2,07796E-10	-5,51057E-10
8	-1,21493E-10	4,76882E-10	1,69975E-12
9	-9,70539E-10	2,22811E-09	-3,12002E-10
10	-4,75354E-10	1,47412E-09	-7,39973E-11
11	-5,34154E-10	2,11441E-09	1,11304E-11
12	-1,4979E-10	5,72968E-10	-9,88805E-13
Rata-Rata	**-1,05826E-09**	**9,26432E-10**	**-6,4965E-10**

5. **Menghitung Propagasi *Error* Lapisan 1**

Propagasi *error* lapisan 1 adalah perkalian antara nilai propagasi *error* lapisan 4 dengan nilai *output* lapisan 1. Propagasi *error* lapisan 1 dihitung menggunakan Persamaan 2.26, 2.27, 2.28, dan 2.29. Berikut adalah contoh perhitungan propagasi *error* lapisan 1:

$$\varepsilon_{1a} = -1{,}05826\text{E}-09\times 0{,}74020 = -7{,}83317\text{E}-10$$

$$\varepsilon_{1b} = 9{,}26432\text{E}-10\times 0{,}09116 = 8{,}44517\text{E}-11$$

$$\varepsilon_{1c} = -6{,}4965\text{E}-10\times 0{,}25558 = -1{,}66041\text{E}-10$$

$$\varepsilon_{2a} = -1{,}05826\text{E}-09\times 0{,}42229 = -4{,}46886\text{E}-10$$

$$\varepsilon_{2b} = 9{,}26432\text{E}-10\times 0{,}15029 = 1{,}3923\text{E}-10$$

$$\varepsilon_{2c} = -6{,}4965\text{E}-10\times 0{,}59504 = -3{,}86569\text{E}-10$$

$$\varepsilon_{3a} = -1{,}05826\text{E}-09\times 0{,}38825 = -4{,}1087\text{E}-10$$

$$\varepsilon_{3b} = 9{,}26432\text{E}-10\times 0{,}95981 = 8{,}89203\text{E}-10$$

$$\varepsilon_{3c} = -6{,}4965\text{E}-10\times 0{,}00346 = -2{,}24792\text{E}-12$$

Tabel 4.27 merupakan hasil perhitungan nilai propagasi *error* pada lapisan 1.

Tabel 4.27 Nilai Propagasi *Error* Lapisan 1

Data ke-	ε1a	ε1b	ε1c	ε2a	ε2b	ε2c	ε3a	ε3b	ε3c
1	**-7,83317E-10**	**8,44517E-11**	**-1,66041E-10**	**-4,46886E-10**	**1,3923E-10**	**-3,86569E-10**	**-4,1087E-10**	**8,89203E-10**	**-2,24792E-12**
2	-6,33321E-10	7,50927E-11	-1,30635E-10	-4,13727E-10	7,58455E-10	-8,15233E-11	-1,03806E-09	5,92769E-10	-3,98558E-12
3	-1,05823E-09	1,1898E-10	-3,41123E-10	-1,00612E-09	2,52269E-10	-1,94086E-10	-1,00401E-09	4,95714E-10	-5,03605E-12
4	-2,37344E-10	5,16654E-10	-2,20733E-10	-2,67544E-10	9,17427E-10	-6,52223E-11	-3,43915E-10	9,14613E-10	-2,07225E-12
5	-1,48207E-10	8,40823E-10	-1,1331E-10	-1,96229E-10	8,91431E-10	-5,53653E-11	-1,05597E-09	5,4253E-10	-4,46495E-12
6	-5,25706E-11	3,03257E-10	-2,9651E-11	-8,79978E-11	3,93354E-10	-3,49232E-11	-4,77042E-11	3,03597E-10	-7,3573E-13
7	-1,38124E-10	8,8223E-10	-1,02838E-10	-9,99092E-10	3,5691E-10	-1,42606E-10	-7,12002E-10	7,54457E-10	-2,93296E-12
8	-1,04601E-09	1,26922E-10	-3,88169E-10	-1,89224E-10	8,50321E-11	-6,43926E-10	-9,37311E-11	1,38301E-10	-6,4965E-10
9	-2,41472E-10	5,06958E-10	-2,26266E-10	-8,81726E-10	2,16997E-10	-2,26447E-10	-8,45387E-11	1,29732E-10	-4,331E-10
10	-3,55356E-10	3,41194E-10	-3,88169E-10	-1,03523E-09	3,32179E-10	-1,51296E-10	-4,50156E-10	3,1691E-10	-1,05634E-11
11	-1,00428E-09	1,37482E-10	-4,50693E-10	-8,15029E-11	5,00177E-11	-3,60472E-10	-1,17031E-10	1,58003E-10	-2,1655E-10
12	-6,14352E-10	2,14101E-10	-6,4084E-10	-1,04427E-10	5,88612E-11	-4,92159E-10	-8,45387E-11	1,29732E-10	-4,331E-10
Rata-Rata	**-5,26049E-10**	**3,45679E-10**	**-2,66539E-10**	**-4,75808E-10**	**3,71013E-10**	**-2,36216E-10**	**-4,53544E-10**	**4,4713E-10**	**-1,47037E-10**

6. **Menghitung Nilai Rata-Rata tiap variabel pada data *input***

Perhitungan nilai rata-rata untuk setiap variabel pada data *input* dilakukan dengan cara menjumlahkan seluruh data kemudian dibagi dengan jumlah data. Berikut adalah contoh perhitungan nilai rata-rata variable pada data *input*:

$$avrg_1 = \frac{23{,}48 + 23{,}37 + 23{,}77 + 24{,}67 + 24{,}97 + 25{,}89 + 25{,}02 + 23{,}82 + 24{,}66 + 24{,}45 + 23{,}88 + 24{,}18}{12}$$
$$= 24{,}34667$$

Tabel 4.28 merupakan perhitungan nilai rata-rata tiap variable pada data *input*:

Tabel 4.28 Nilai Rata-Rata Data *Input*

avrg_x1	**24,34667**
avrg_x2	74,42500
avrg_x3	5,55000

7. **Menghitung *Error* Terhadap Parameter *Mean* (c)**

Nilai *error* terhadap parameter *mean* (c) dihitung menggunakan Persamaan 2.31. Berikut adalah contoh perhitungan nilai *error* terhadap parameter *mean*:

$$\varepsilon_{cik} = \frac{2(24{,}34667-23{,}768)^2}{0{,}48527^3\left(1+\left(\frac{24{,}34667-23{,}768}{0{,}48527}\right)^2\right)^2} \times -5{,}26049\text{E}-10 = -5{,}25402\text{E}-10$$

Tabel 4.29 merupakan hasil perhitungan nilai *error* terhadap parameter *mean* (c).

Tabel 4.29 Nilai *Error* Parameter Mean

εcik	x1	x2	x3
c1	**-5,25402E-10**	4,16619E-12	-1,62907E-10
c2	-3,84869E-10	3,78034E-11	-1,06937E-10
c3	-4,22658E-10	4,2649E-11	-1,94093E-11

8. **Menghitung *Error* Terhadap Parameter Standar Deviasi (a)**

Nilai *error* terhadap parameter standar deviasi (a) dihitung menggunakan Persamaan 2.30. Berikut adalah contoh perhitungan nilai *error* terhadap parameter standar deviasi:

$$\varepsilon_{aik} = \frac{2(24{,}34667 - 23{,}768)^2}{0{,}48527^2(1 + (\frac{24{,}34667 - 23{,}768}{0{,}48527})^2)^2} \times -5{,}26049\text{E} - 10$$
$$= -2{,}54965\text{E} - 10$$

Tabel 4.30 merupakan hasil perhitungan nilai *error* terhadap parameter standar deviasi (a).

Tabel 4.30 Nilai *Error* Terhadap parameter Standar Deviasi

εaik	x1	x2	x3
a1	**-2,54965E-10**	1,32682E-11	-8,76021E-11
a2	-2,02032E-10	1,52625E-10	-1,1759E-10
a3	-1,62215E-10	1,80944E-10	-2,7449E-12

9. **Menghitung Perubahan Parameter *Mean* (c)**

Perubahan parameter *mean* (c) dihitung menggunakan Persamaan 2.32, dengan nilai *learning rate* 0,0001. Berikut adalah contoh perhitungan perubahan parameter *mean*:

$$\Delta C\ C_1X_1 = 0{,}0001 \times (-5{,}25402\text{E} - 10) \times 24{,}34667 = -1{,}27918\text{E} - 12$$

Tabel 4.31 merupakan hasil perhitungan perubahan parameter *mean*.

Tabel 4.31 Perubahan Nilai Parameter Mean

Δc	x1	x2	x3
c1	-1,27918E-12	3,10069E-14	-9,04135E-14
c2	-9,37027E-13	2,81352E-13	-5,93502E-14
c3	-1,02903E-12	3,17415E-13	-1,07722E-14

10. **Menghitung Perubahan Parameter Standar Deviasi (a)**

Perubahan parameter standar deviasi (a) dihitung menggunakan Persamaan 2.32, dengan nilai *learning rate* 0,0001. Berikut adalah contoh perhitungan perubahan parameter standar deviasi:

$$\Delta a\ a_1x_1 = 0{,}0001 \times (-2{,}54965\text{E} - 10) = -2{,}54965\text{E} - 14$$

Tabel 4.32 merupakan hasil perhitungan perubahan parameter standar deviasi.

Tabel 4.32 Perubahan Nilai Parameter Standar Deviasi

Δa	x1	x2	x3
a1	-2,54965E-14	1,32682E-15	-8,76021E-15
a2	-2,02032E-14	1,52625E-14	-1,1759E-14
a3	-1,62215E-14	1,80944E-14	-2,7449E-16

11. **Menghitung Nilai *Mean* yang Baru**

Nilai *mean* yang baru merupakan nilai perbaikan parameter (c). Nilai *mean* yang baru dihitung menggunakan Persamaan 2.34. Berikut adalah contoh perhitungan nilai *mean* baru:

$$c_{ik}(baru) = c_{ik}(lama) + \Delta c$$

$$c_{11}(baru) = 23{,}768 + (-1{,}27918\text{E} - 12) = 23{,}7675$$

Tabel 4.33 merupakan hasil perhitungan nilai parameter *mean* (c) yang baru.

Tabel 4.33 Nilai Parameter *Mean* Baru

cik (baru)	x1	x2	x3
c1	**23,7675**	73,97500	5,82
c2	25,1375	68,10000	6,72
c3	24,1350	81,20000	4,10

12. **Menghitung Nilai Standar Deviasi yang Baru**

Nilai standar deviasi yang baru merupakan nilai perbaikan parameter (a). Nilai standar deviasi yang baru dihitung menggunakan Persamaan 2.33. Berikut adalah contoh perhitungan nilai standar deviasi baru:

$$a_{ik}(baru) = a(lama) + \Delta a$$

$$a(baru) = 0{,}48527 + (-2{,}54965\text{E} - 14) = 0{,}485275$$

Tabel 4.34 merupakan hasil perhitungan nilai parameter standar deviasi (a) yang baru.

Tabel 4.34 Nilai Parameter Standar Deviasi Baru

aik (baru)	x1	x2	x3
a1	0,485275	3,184729	0,537742
a2	0,524937	4,037326	1,099621
a3	0,383797	4,242641	0,141421

4.5 Perancangan Pengujian

Pada tahap pengujian ini digunakan untuk melakukan pengujian tingkat akurasi dari sistem ANFIS dalam prediksi curah hujan. Pengujian akurasi dihitung dengan menggunakan *Root Mean Square Error* (RMSE). Terdapat 3 skenario pengujian yang akan dilakukan, yaitu pengujian jumlah data latih, pengujian jumlah iterasi, dan pengujian *learning rate*.

Pengujian jumlah data latih dilakukan dengan mengubah jumlah data latih yang digunakan pada setiap percobaan. Jumlah data latih yang digunakan adalah 50 data sampai dengan 250 data. Jumlah data latih yang menghasilkan nilai RMSE terbaik akan digunakan untuk skenario pengujian selanjutnya. Berikut merupakan tabel rancangan pengujian jumlah data latih.

Tabel 4.35 Rancangan Pengujian Jumlah Data Latih

Jumlah Data Latih	Nilai RMSE					Rata-rata RMSE	Rata-rata Lama Komputasi (msec)
	Percobaan ke-i						
	1	2	3	4	5		
50							
100							
150							
200							
250							

Skenario pengujian selanjutnya adalah pengujian jumlah iterasi. Pengujian ini dilakukan untuk mengetahui pengaruh jumlah iterasi terhadap nilai RMSE yang akan dihasilkan. Pengujian iterasi dilakukan dengan mengubah jumlah iterasi untuk masing-masing percobaan. Jumlah iterasi yang digunakan adalah 100 sampai dengan 1000. Jumlah iterasi yang menghasilkan nilai RMSE terbaik akan digunakan pada scenario selanjutnya. Berikut merupakan tabel rancangan pengujian jumlah iterasi.

Tabel 4.36 Rancangan Pengujian Jumlah Iterasi

Jumlah Iterasi	Nilai RMSE					Rata-rata RMSE	Rata-rata Lama Komputasi (msec)
	Percobaan ke-i						
	1	2	3	4	5		
100							
200							
300							

Jumlah Iterasi	Nilai RMSE					Rata-rata RMSE	Rata-rata Lama Komputasi (msec)
	Percobaan ke-i						
	1	2	3	4	5		
400							
500							
600							
700							
800							
900							
1000							

Skenario pengujian terakhir adalah pengujian perubahan *learning rate* dilakukan dengan mengubah nilai *learning rate* untuk setiap percobaan. Kemudian melihat pengaruh perubahan nilai *learning rate* terhadap nilai RMSE yang dihasilkan. Nilai dari *learning rate* yang digunakan adalah 0,1 sampai dengan 0,9. *Learning rate* yang menghasilkan nilai RMSE terbaik akan digunakan untuk skenario pengujian selanjutnya. Berikut merupakan tabel rancangan pengujian *learning rate*.

Tabel 4.37 Rancangan Pengujian *Learning Rate*

Learning Rate	Nilai RMSE					Rata-rata RMSE
	Percobaan ke-i					
	1	2	3	4	5	
0,1						
0,2						
0,3						
0,4						
0,5						
0,6						
0,7						
0,8						
0,9						

BAB 5 IMPLEMENTASI

Pada bab ini akan membahas tentang penerapan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan berdasarkan perancangan yang telah dibuat pada bab sebelumnya. Tujuan dari implementasi adalah agar aplikasi yang telah dirancang dapat dipergunakan oleh pengguna (*user*).

5.1 Lingkungan Implementasi

Lingkungan implementasi terdiri dari lingkungan perangkat lunak, yaitu perangkat lunak yang digunakan dalam proses implementasi aplikasi yang akan dibuat, serta lingkungan perangkat keras, yaitu perangkat keras yang digunakan dalam proses pembangunan aplikasi.

5.1.1 Lingkungan Perangkat Lunak

Dalam mengimplementasikan metode ANFIS dalam memprediksi curah hujan digunakan perangkat lunak dengan spesifikasi sebagai berikut:

1. NetBeans IDE 8.2

   NetBeans merupakan sebuah aplikasi lintas *platform* yang digunakan untuk pengembangan dalam Java yang dapat diterapkan pada OS *Microsoft Windows*, MAC OS X, Linux, Solaris dan serambi-serambi lain yang mendukung suatu JVM (*Java Virtual Machine*) yang sepadan.

2. Bahasa Pemrograman JAVA

   Bahasa pemrograman JAVA merupakan salah satu bahasa pemrograman tingkat tinggi yang popular digunakan oleh para *programmer* dalam membangun sebuah aplikasi, karena memiliki banyak kelebihan dibandingkan dengan bahasa pemrograman lainnya.

5.1.2 Lingkungan Perangkat Keras

Dalam pengembangan aplikasi prediksi curah hujan digunakan perangkat keras dengan spesifikasi sebagai berikut:

1. *Processor* : Intel Core i5-2467M CPU @ 1.60Ghz
2. *Installed Memory* (RAM) : 4.00 GB
3. *System Type* : 64-bit *Operating System*

5.2 Implementasi Algoritme

Implementasi algoritme merupakan implementasi yang berdasarkan hasil perancangan yang telah dibuat pada bab perancangan. Pada bagian ini akan dijelaskan proses-proses yang ada dalam memprediksi curah hujan dengan menggunakan metode *Adaptive Neuro Fuzzy Inference System*. Implementasi pembuatan aplikasi dibuat dengan menggunakan bahasa pemrograman JAVA.

5.2.1 Implementasi K-Means *Clustering*

Implementasi K-Means *Clustering* merupakan bentuk realisasi dari perancangan Gambar 4.4. Sesuai pada Gambar 4.4, proses pertama yang dilakukan untuk melakukan *clustering* data adalah membaca data *input*. Pembacaan data *input* dilakukan menggunakan *library* java Scanner dan File. Pada baris 9-20 merupakan proses dimana setiap baris yang ada pada data input akan dibaca dan dimasukkan ke dalam variabel matriks. Variabel matriks merupakan array multidimensi, dengan i menyatakan baris dan j menyatakan kolom. Kolom pertama menyimpan *input* parameter suhu, kolom kedua menyimpan *input* parameter kelembapan udara, kolom ketiga menyimpan *input* parameter kecepatan angin, dan kolom terakhir menyimpan nilai target curah hujan. Kode Sumber 5.1 berikut merupakan implementasi pembacaan data input.

No	Kode Sumber
1	`public double[][] readCsv(String file, int baris, int kolom)`
2	`{`
3	`   double[][] matriks = new double[baris][kolom];`
4	`   String[] data = new String[50];`
5	`   String delimiter = ";";`
6	`   try {`
7	`       Scanner sc = new Scanner(new File(file));`
8	`       int index = 0;`
9	`       while (sc.hasNextLine()) {`
10	`           String line = sc.nextLine();`
11	`           data = line.split(delimiter);`
12	`           for (int i = index; i < baris; i++) {`
13	`               for (int j = 0; j < kolom; j++) {`
14	`                   matriks[i][j] =`
15	`Double.parseDouble(data[j]);`
16	`               }`
17	`               break;`
18	`           }`
19	`           index++;`
20	`       }`
21	`       int temp = 0;`
22	`       for (int y = 0; y < baris; y++) {`
23	`           for (int x = 0; x < kolom; x++) {`
24	`               System.out.print(matriks[y][x] + " ");`
25	`               temp++;`
26	`               if (temp % kolom == 0) {`
27	`                   System.out.println("");`
28	`               }}}`
29	`   } catch (FileNotFoundException ex) {`
30	
31	`Logger.getLogger(ANFIS.class.getName()).log(Level.SEVERE,`
32	`null, ex);`
33	`       }`
34	`       return matriks;`
35	`  }`

Kode Sumber 5.1 Implementasi Pembacaan Data *Input*

Setelah proses pembacaan data input, proses selanjutnya adalah normalisasi data. Pada baris 29 – 40 merupakan proses normalisasi data. Terdapat dua *method* yang membantu dalam proses normalisasi. Method pertama adalah nilaiMinimum yang berfungsi untuk mencari nilai minimum pada data yang tersimpan pada *array* multidimensi. Method kedua adalah nilaiMaksimum yang berfungsi untuk mencari nilai maksimum pada data yang tersimpan pada *array* multidimensi. Kode Sumber 5.2 berikut merupakan implementasi untuk menormalisasi data.

No	Kode Sumber
1	`public double[] nilaiMinimum(double data[][]) {`
2	`   for (int i = 0; i < data[0].length; i++) {`
3	`       double min = data[0][i];`
4	`       for (int j = 0; j < data.length; j++) {`
5	`           if (data[j][i] < min) {`
6	`               min = data[j][i];`
7	`           }`
8	`           if (j == (data.length - 1)) {`
9	`               hasilMin[i] = min;`
10	`           }`
11	`       }`
12	`   }`
13	`   return hasilMin;`
14	`}`
15	`public double[] nilaiMaksimum(double data[][]) {`
16	`   for (int i = 0; i < data[0].length; i++) {`
17	`       double max = data[0][i];`
18	`       for (int j = 0; j < data.length; j++) {`
19	`           if (data[j][i] > max) {`
20	`               max = data[j][i];`
21	`           }`
22	`           if (j == (data.length - 1)) {`
23	`               hasilMax[i] = max;`
24	`           }`
25	`       }`
26	`   }`
27	`   return hasilMax;`
28	`}`
29	`public double[][] normalisasi(double data[][], double min[],`
30	`double max[]) {`
31	`   System.out.println("data" + data.length);`
32	`   System.out.println("dataa" + data[0].length);`
33	`   for (int i = 0; i < data.length; i++) {`
34	`       for (int j = 0; j < data[0].length; j++) {`
35	`           hasilNormalisasi[i][j] = (data[i][j] - min[j]) /`
36	`(max[j] - min[j]);`
37	`       }`
38	`   }`
39	`   return hasilNormalisasi;`
40	`}`

Kode Sumber 5.2 Implementasi Normalisasi Data

Proses selanjutnya setelah melakukan normalisasi data adalah menuntukan *centroid* awal. Pemilihan *centroid* awal dilakukan dengan memilih indeks dari sebanyak jumlah data secara acak. Proses ini ditunjukkan pada baris 3 – 6 dan proses pemilihan *centroid* awal ditunjukkan pada baris 7-15. Kode Sumber 5.3 merupakan implementasi pemilihan *centroid* awal.

No	Kode Sumber
1	`public double[][] centroid(int k, double data[][]) {`
2	`    LinkedList<Integer> index = new LinkedList<Integer>();`
3	`    for (int i = 0; i < data.length; i++) {`
4	`        index.add(i);`
5	`    }`
6	`    Collections.shuffle(index);`
7	`    double centroid[][] = new double[k][data[0].length];`
8	`    for (int i = 0; i < k; i++) {`
9	`        for (int j = 0; j < centroid[0].length; j++) {`
10	`            int baris = index.get(i);`
11	`            System.out.println("data " + baris + " menjadi`
12	`pusat cluster");`
13	`                centroid[i][j] = data[baris][j];`
14	`        }`
15	`    }`
16	`    return centroid;`
17	`}`

Kode Sumber 5.3 Implementasi Pemilihan *Centroid* Awal

Setelah *centroid* awal sudah didapatkan, proses selanjutnya adalah menghitung jarak antara data dengan pusat *cluster*. Jarak dihitung menggunakan algoritme *Euclidean distance*. Pada baris 3 – 13 merupakan proses perhitungan jarak. Setelah menghitung jarak antara data dengan pusat *cluster*, selanjutnya menentukan hasil *cluster* berdasarkan jarak data dengan pusat *cluster* yang paling dekat. Kode Sumber 5.4 berikut merupakan implementasi untuk menentukan anggota masing-masing *cluster*.

No	Kode Sumber
1	`public double[][] hitungJarak(double data[][], double[][]`
2	`centroid) {`
3	`   hasilJarak = new double[data.length][centroid.length];`
4	`   for (int i = 0; i < data.length; i++) {`
5	`       for (int j = 0; j < hasilJarak[i].length; j++) {`
6	`           double jarak = 0;`
7	`           for (int k = 0; k < centroid.length; k++) {`
8	`               jarak = jarak + Math.pow(data[i][k] -`
9	`centroid[j][k], 2);`
10	`           }`
11	`           hasilJarak[i][j] = Math.sqrt(jarak);`
12	`       }`
13	`   }`
14	`   return hasilJarak;`
15	`}`
16	`public int[] cluster(double[][] jarak) {`
17	`   hasilCluster = new int[jarak.length];`
18	`   minJarak = new double[jarak.length];`
19	`   for (int i = 0; i < jarak.length; i++) {`

No	Kode Sumber
20	double minTemp = 1000;
21	for (int j = 0; j < jarak[i].length; j++) {
22	if (jarak[i][j] < minTemp) {
23	minJarak[i] = jarak[i][j];
24	minTemp = minJarak[i];
25	}
26	}
27	}
28	for (int i = 0; i < jarak.length; i++) {
29	for (int j = 0; j < jarak[i].length; j++) {
30	if (jarak[i][j] == minJarak[i]) {
31	hasilCluster[i] = (j + 1);
32	}
33	}
34	}
35	System.out.println("");
36	return hasilCluster;
37	}

Kode Sumber 5.4 Implementasi Menentukan Anggota Masing-Masing *Cluster*

Proses selanjutnya adalah menghitung *centroid* baru. Terdapat variabel *centroidLama* yang digunakan untuk menghitung *centroid* yang baru. Kode Sumber 5.5 berikut merupakan implementasi perhitungan *centroid* baru.

No	Kode Sumber
1	public double[][] centroidBaru(double[][] centroidLama,
2	double[][] jarak) {
3	
4	double[][] centroidBr = new
5	double[centroidLama.length][centroidLama[0].length];
6	
7	int[][] jumlahData = new int[3][3];
8	double[][] centroidTemp = new
9	double[centroidLama.length][centroidLama[0].length];
10	
11	for (int i = 0; i < centroidLama.length; i++) {
12	for (int j = 0; j < centroidLama[i].length; j++) {
13	for (int k = 0; k < jarak.length; k++) {
14	if (hasilCluster[k] == i + 1) {
15	centroidTemp[i][j] += jarak[k][j];
16	jumlahData[i][j] = jumlahData[i][j] + 1;
17	}
18	}
19	}
20	}
21	for (int i = 0; i < centroidLama.length; i++) {
22	for (int j = 0; j < centroidLama[i].length; j++) {
23	centroidTemp[i][j] = (centroidTemp[i][j] +
24	centroidLama[i][j]) / (jumlahData[i][j] + 1);
25	centroidBr[i][j] = centroidTemp[i][j];
26	}
27	}
28	System.out.println("");
29	return centroidBr;

No	Kode Sumber
30	}

Kode Sumber 5.5 Implementasi Perhitungan *Centroid* Baru

Proses selanjutnya adalah melakukan proses cek *cluster* dengan membandingkan hasil *cluster* sebelumnya dan hasil *cluster* baru, proses ini akan terus berlanjut hingga hasil *cluster* sebelumnya dan hasil *cluster* baru sama. Kode Sumber 5.6 berikut merupakan implementasi pengecekan hasil *cluster*.

No	Kode Sumber
1	`public boolean cekCluster(int[] clusterLama, int[]`
2	`clusterBaru) {`
3	`   boolean cekClstr = true;`
4	`   for (int i = 0; i < clusterLama.length; i++) {`
5	`       if (clusterLama[i] != clusterBaru[i]) {`
6	`            cekClstr = false;`
7	`        }`
8	`    }`
9	`   return cekClstr;`
10	`}`

Kode Sumber 5.6 Implementasi Pengecekan Hasil *Cluster*

5.2.2 Implementasi Perhitungan *Mean* dan Standar Deviasi

Implementasi perhitungan *mean* dan standar deviasi merupakan bentuk realisasi dari perancangan Gambar 4.5. Pada baris 6 – 20 merupakan proses perhitungan *mean* didapatkan dari penjumlahan setiap nilai data dibagi dengan jumlah data. Nilai *mean* disimpan pada variabel hasilMean. Pada baris 27 – 41 merupakan proses perhitungan standar deviasi. Kode Sumber 5.7 berikut merupakan implementasi perhitungan *mean* dan standar deviasi.

No	Kode Sumber
1	`public double[][] Mean(double[][] data, int[] cluster) {`
2	`   double[][] hasilMean = new double[3][3];`
3	`   double[][] meanTemp = new double[3][3];`
4	`   double[][] jumlahData = new double[3][3];`
5	
6	`   for (int i = 0; i < hasilMean.length; i++) {`
7	`       for (int j = 0; j < hasilMean[i].length; j++) {`
8	`           meanTemp[i][j] = 0;`
9	`           for (int k = 0; k < data.length; k++) {`
10	`               if (cluster[k] == i + 1) {`
11	`                   meanTemp[i][j] += data[k][j];`
12	`                   jumlahData[i][j] = jumlahData[i][j] + 1;`
13	`               }`
14	`           }`
15	`      }`
16	`   }`
17	`   for (int i = 0; i < hasilMean.length; i++) {`
18	`      for (int j = 0; j < hasilMean[i].length; j++) {`
19	`        meanTemp[i][j] = meanTemp[i][j] / jumlahData[i][j];`
20	`           hasilMean[i][j] = meanTemp[i][j];`
21	`       }`

No	Kode Sumber
22	} return hasilMean; }
25	public double[][] StDeviasi(double[][] data, int[] cluster,
26	double[][] mean) {
27	double[][] hasilStd = new double[3][3];
28	double[][] stdTemp = new double[3][3];
29	double[][] jumlahData = new double[3][3];
30	
31	for (int i = 0; i < hasilStd.length; i++) {
32	for (int j = 0; j < hasilStd[i].length; j++) {
33	stdTemp[i][j] = 0;
34	for (int k = 0; k < data.length; k++) {
35	if (cluster[k] == i + 1) {
36	stdTemp[i][j] += Math.pow((data[k][j] -
37	mean[i][j]), 2);
38	jumlahData[i][j] = jumlahData[i][j] + 1;
39	}
40	}
41	}
42	}

Kode Sumber 5.7 Implementasi Perhitungan *Mean* dan Standar Deviasi

5.2.3 Implementasi Perhitungan *Output* Lapisan 1

Implementasi perhitungan *output* lapisan 1 merupakan bentuk realisasi dari perancangan Gambar 4.3. Setelah mendapatkan nilai parameter premis (*mean* dan standar deviasi), Selanjutnya adalah menghitung nilai *output* lapisan 1. *Output* lapisan 1 merupakan derajat keanggotaan *generalized bell* yang dihitung dengan menggunakan Persamaan 2.2. Kode Sumber 5.8 berikut merupakan implementasi perhitungan *output* lapisan 1.

No	Kode Sumber
1	public void layer1(double[][] data, double[][] mean,
2	double[][] std) {
3	outputLayer1_1 = new double[data.length][3];
4	outputLayer1_2 = new double[data.length][3];
5	outputLayer1_3 = new double[data.length][3];
6	for (int i = 0; i < data.length; i++) {
7	for (int j = 0; j < 3; j++) {
8	outputLayer1_1[i][j] = 1 / (1 +
9	(Math.pow((Math.abs((data[i][0] - mean[j][0]) /
10	std[j][0])), 2)));
11	outputLayer1_2[i][j] = 1 / (1 +
12	(Math.pow((Math.abs((data[i][1] - mean[j][1]) /
13	std[j][1])), 2)));
14	outputLayer1_3[i][j] = 1 / (1 +
15	(Math.pow((Math.abs((data[i][2] - mean[j][2]) /
16	std[j][2])), 2)));
17	}
18	}
19	System.out.println("");
20	System.out.println("Output Lapisan 1 --> Nilai Derajat
21	Keanggotaan");

No	Kode Sumber
22	`    for (int i = 0; i < data.length; i++) {`
23	`        for (int j = 0; j < 3; j++) {`
24	`            System.out.print(outputLayer1_1[i][j] + "\t");`
25	`        }`
26	`        for (int j = 0; j < 3; j++) {`
27	`            System.out.print(outputLayer1_2[i][j] + "\t");`
28	`        }`
29	`        for (int j = 0; j < 3; j++) {`
30	`            System.out.print(outputLayer1_3[i][j] + "\t");`
31	`        }`
32	`        System.out.println("");`
33	`    }`
34	`}`
35	` public double[][] getOutputayer1_1() {`
36	`   return outputLayer1_1;`
37	` }`
38	` public double[][] getOutputayer1_2() {`
39	`   return outputLayer1_2;`
40	` }`
41	` public double[][] getOutputayer1_3() {`
42	`   return outputLayer1_3;`
43	` }`

Kode Sumber 5.8 Implementasi Perhitungan *Output* Lapisan 1

5.2.4 Implementasi Perhitungan *Output* Lapisan 2

Implementasi perhitungan *output* lapisan 2 merupakan bentuk realisasi dari perancangan Gambar 4.6. *Output* lapisan 2 merupakan nilai *fire strength* yang didapatkan dengan menggunakan Persamaan 2.12. Baris 6 - 13 merupakan implementasi dari Persamaan 2.12 untuk mendapatkan nilai *fire strength* .Kode Program 5.9 berikut merupakan implementasi perhitungan *output* lapisan 2.

No	Kode Sumber
1	`public double[][] layer2(double[][] data, double[][]`
2	`outputLayer1_1, double[][] outputLayer1_2, double[][]`
3	`outputLayer1_3) {`
4	`    double[][] outputLayer2 = new double[data.length][3];`
5	
6	`    for (int i = 0; i < outputLayer2.length; i++) {`
7	`        outputLayer2[i][0] = outputLayer1_1[i][0] *`
8	`outputLayer1_2[i][0] * outputLayer1_3[i][0];`
9	`        outputLayer2[i][1] = outputLayer1_1[i][1] *`
10	`outputLayer1_2[i][1] * outputLayer1_3[i][1];`
11	`        outputLayer2[i][2] = outputLayer1_1[i][2] *`
12	`outputLayer1_2[i][2] * outputLayer1_3[i][2];`
13	`    }`
14	`    System.out.println("");`
15	`    return outputLayer2;`
16	`}`

Kode Sumber 5.9 Implementasi Perhitungan *Output* Lapisan 2

5.2.5 Implementasi Perhitungan *Output* Lapisan 3

Implementasi perhitungan *output* lapisan 3 merupakan bentuk realisasi dari perancangan Gambar 4.7. *Output* Lapisan 3 merupakan nilai *fire strength* yang telah ternormalisasi. Untuk mendapatkan nilai *fire strength* yang telah ternormalisasi digunakan Persamaan 2.13. Kode Sumber 5.10 berikut merupakan implementasi perhitungan *output* lapisan 3.

No	Kode Sumber
1	`public double[][] layer3(double[][] data, double[][]`
2	`outputLayer2) {`
3	`   double[][] outputLayer3 = new double[data.length][3];`
4	
5	`   for (int i = 0; i < outputLayer2.length; i++) {`
6	`       double jumlah = 0;`
7	`       for (int j = 0; j < outputLayer2[i].length; j++) {`
8	`           jumlah += outputLayer2[i][j];`
9	`       }`
10	`       for (int j = 0; j < outputLayer2[i].length; j++) {`
11	`          outputLayer3[i][j] = outputLayer2[i][j] / jumlah;`
12	`       }`
13	`   }`
14	`   System.out.println("");`
15	`   return outputLayer3;`
16	`}`

Kode Sumber 5.10 Implementasi Perhitungan *Output* Lapisan 3

5.2.6 Implementasi Perhitungan Parameter Konsekuen

Implementasi perhitungan parameter konsekuen merupakan bentuk realisasi dari perancangan Gambar 4.8. Untuk mengimplementasikan perhitungan parameter konsekuen, digunakan salah satu *library* untuk memudahkan mengoperasikan matriks. *Library* yang digunakan adalah Jama Matrix versi 1.0.3. Pada baris 4 – 38 merupakan proses pembentukan matriks A. Pada baris 48 – 63 merupakan proses perhitungan matriks A transpose dikalikan dengan matriks A. Pada baris 64 adalah proses inverse dari matriks A transpose dikalikan dengan matriks A. Pada baris 75-74 merupakan proses untuk mendapatkan parameter konsekuen. Kode Sumber 5.11 berikut merupakan implementasi perhitungan parameter konsekuean.

No	Kode Sumber
1	`public double[][] LSE(double[][] dataLatih, double[][]`
2	`outputLayer3) {`
3	`   double[][] parameterKonsekuen = new`
4	`double[dataLatih[0].length * 3][1];`
5	`   double matriksA[][] = new`
6	`double[dataLatih.length][dataLatih[0].length * 3];`
7	
8	`   for (int i = 0; i < matriksA.length; i++) {`
9	`       int indexHelper = 0;`
10	`       int indexHelper2 = 0;`
11	`       for (int j = 0; j < matriksA[i].length; j++) {`
12	`           if (j < dataLatih[i].length) {`

No	Kode Sumber
13	if (j == (dataLatih[i].length - 1)) {
14	matriksA[i][j] = outputLayer3[i][0] * 1;
15	} else {
16	matriksA[i][j] = outputLayer3[i][0] *
17	dataLatih[i][j];
18	}
19	} else if (j >= (dataLatih[i].length) && j <
20	(dataLatih[i].length * 2)) {
21	if (j == ((dataLatih[i].length * 2) - 1)) {
22	matriksA[i][j] = outputLayer3[i][1] * 1;
23	} else {
24	matriksA[i][j] = outputLayer3[i][1] *
25	dataLatih[i][indexHelper];
26	}
27	indexHelper++;
28	} else {
29	if (j == ((dataLatih[i].length * 3) - 1)) {
30	matriksA[i][j] = outputLayer3[i][2] * 1;
31	} else {
32	matriksA[i][j] = outputLayer3[i][2] *
33	dataLatih[i][indexHelper2];
34	}
35	indexHelper2++;
36	}
37	}
38	}
39	
40	System.out.println("");
41	System.out.println("Matriks Desain A");
42	for (int i = 0; i < matriksA.length; i++) {
43	for (int j = 0; j < matriksA[i].length; j++) {
44	System.out.print(matriksA[i][j] + "\t");
45	}
46	System.out.println("");
47	}
48	double[][] matriksAT = transposeMatrix(matriksA);
49	
50	Matrix AT = new Matrix(matriksAT);
51	Matrix A = new Matrix(matriksA);
52	Matrix ATxA = AT.times(A);
53	
54	double[][] matriksATxA = ATxA.getArray();
55	
56	System.out.println("");
57	System.out.println("Hasil Perkalian Matriks AT x A");
58	for (int i = 0; i < matriksATxA.length; i++) {
59	for (int j = 0; j < matriksATxA[i].length; j++) {
60	System.out.print(matriksATxA[i][j] + "\t");
61	}
62	System.out.println("");
63	}
64	Matrix inverseATxA = ATxA.inverse();
65	double[][] matriksInverseATxA = inverseATxA.getArray();
66	System.out.println("");
67	System.out.println("Hasil Invers Matriks ATxA");
68	for (int i = 0; i < matriksInverseATxA.length; i++) {
69	for (int j = 0; j < matriksInverseATxA[i].length;

No	Kode Sumber
70	j++) {
71	System.out.print(matriksInverseATxA[i][j] + "\t");
72	}
73	System.out.println("");
74	}
75	Matrix INVERSExAT = inverseATxA.times(AT);
76	double[][] matriksINVERSExAT = INVERSExAT.getArray();
77	double y[][] = new double[dataLatih.length][1];
78	
79	for (int i = 0; i < dataLatih.length; i++) {
80	y[i][0] = dataLatih[i][dataLatih[i].length - 1];
81	}
82	Matrix target = new Matrix(y);
83	Matrix theta = INVERSExAT.times(target);
84	parameterKonsekuen = theta.getArray();
85	return theta.getArray();
86	}
87	public double[][] transposeMatrix(double[][] data) {
88	double[][] temp = new
89	double[data[0].length][data.length];
90	for (int i = 0; i < data.length; i++) {
91	for (int j = 0; j < data[0].length; j++) {
92	temp[j][i] = data[i][j];
93	}
94	}
95	return temp;
96	}

Kode Sumber 5.11 Implementasi Perhitungan Parameter Konsekuen

5.2.7 Implementasi Perhitungan *Output* Lapisan 4

Implementasi perhitungan *output* lapisan 4 merupakan bentuk realisasi dari perancangan Gambar 4.9. Perhitungan nilai *f* di tunjukkan pada baris 8-20. Realisasi dari persamaan 2.14 untuk menghitung nilai *output* lapisan 4 ditunjukkan pada baris ke-21. Kode Sumber 5.12 merupakan implementasi perhitungan *output* lapisan 4.

No	Kode Sumber
1	public double[][] layer4(double[][] data, double[][]
2	outputLayer3, double[][] paramKonsekuen) {
3	double[][] outputLayer4 = new double[data.length][3];
4	f = new double[data.length][3];
5	int jumlah = data[0].length + 1;
6	
7	for (int i = 0; i < data.length; i++) {
8	for (int j = 1; j <= outputLayer4[0].length; j++) {
9	int x = j * jumlah;
10	double temp = 0;
11	for (int k = (x - jumlah); k < x; k++) {
12	if (k != (x - 1)) {
13	int indexHelper = k % jumlah;
14	temp += data[i][indexHelper] *
15	paramKonsekuen[k][0];
16	} else {

No	Kode Sumber
17	temp += paramKonsekuen[k][0];
18	}
19	}
20	f[i][j - 1] = temp;
21	outputLayer4[i][j - 1] = temp *
22	outputLayer3[i][j - 1];
23	}
24	}
25	System.out.println("");
26	System.out.println("Nilai F");
27	for (int i = 0; i < f.length; i++) {
28	for (int j = 0; j < f[i].length; j++) {
29	System.out.print(f[i][j] + "\t");
30	}
31	System.out.println("");
32	}
33	System.out.println("");
34	return outputLayer4;
35	}
36	public double[][] getNilaiF() {
37	return f;
38	}

Kode Sumber 5.12 Implementasi Perhitungan *Output* Lapisan 4

5.2.8 Implementasi Perhitungan *Output* Lapisan 5

Implementasi perhitungan *output* lapisan 5 merupakan bentuk realisasi dari perancangan Gambar 4.10. Nilai *output* lapisan 5 di dapatkan dengan cara menjumlahkan seluruh sinyal yang diterima dari lapisan sebelumnya sesuai dengan persamaan 2.15. Baris 5 - 8 merupakan implementasi dari Persamaan 2.15. Kode sumber 5.13 berikut merupakan implementasi perhitungan *output* lapisan 5.

No	Kode Sumber
1	public double[] layer5(double[][] data, double[][]
2	outputLayer4) {
3	double[] outputLayer5 = new double[data.length];
4	
5	for (int i = 0; i < outputLayer5.length; i++) {
6	outputLayer5[i] = outputLayer4[i][0] +
7	outputLayer4[i][1] + outputLayer4[i][2];
8	}
9	System.out.println("");
10	return outputLayer5;
11	}

Kode Sumber 5.13 Implementasi Perhitungan *Output* Lapisan 5

5.2.9 Implementasi Perhitungan *Propagation Error*

Implementasi perhitungan *propogation error* merupakan bentuk realisasi dari perancangan Gambar 4.11 sampai dengan Gambar 4.16. Perhiungan nilai *error* lapisan dilakukan untuk memperbaiki parameter premis nantinya. Pada implementasi perhitungan nilai *error*, akan di hitung nilai *error* mulai dari *error* lapisan 5 hingga *error* lapisan 1. Implementasi perhitungan nilai *propagation error* lapisan 5 mengacu pada Gambar 4.12, untuk implementasinya ditunjukkan pada Kode Sumber 5.14 pada baris 1-15. Implementasi perhitungan *propagation error* lapisan 4 mengacu pada Gambar 4.13 dan implementasi perhitungan *propagation error* lapisan 4 ditunjukkan pada baris 17-24. Pengimplementasian nilai *error* lapisan 3 mengacu pada Gambar 4.14, implementasi *error* lapisan 3 ditunjukkan pada baris ke 26-39. Implementasi nilai *error* lapisan 2 mengacu pada Gambar 4.15 dan ditunjukkan pada baris ke 41-69. Nilai *error* lapisan 1 diimplementasikan dengan mengacu kepada Gambar 4.16 dan implementasi dari nilai *error* lapisan 1 ditunjukkan pada baris ke 71-93. Kode Sumber 5.14 berikut merupakan implementasi perhitungan *propagation error*.

No	Kode Sumber
1	`public double propagationError5(double[][] data,`
2	`double[][] outputAktual, double[] outputLayer5) {`
3	`  double errorLapisan5 = 0;`
4	
5	`    for (int i = 0; i < data.length; i++) {`
6	`        errorLapisan5 += ((-2) * (outputAktual[i][0] -`
7	`outputLayer5[i]));`
8	`        System.out.println("error = " + errorLapisan5);`
9	`    }`
10	`    errorLapisan5 = errorLapisan5 / data.length;`
11	`    System.out.println("");`
12	`    System.out.println("Error Lapisan 5 = " +`
13	`errorLapisan5);`
14	`    return errorLapisan5;`
15	`}`
16	`public double[] propagationError4(double errorLapisan5) {`
17	`    double[] errorLapisan4 = new double[3];`
18	
19	`    for (int i = 0; i < errorLapisan4.length; i++) {`
20	`        errorLapisan4[i] = errorLapisan5;`
21	`    }`
22	`    return errorLapisan4;`
23	`}`
24	`public double[] propagationError3(double[] errorLapisan4,`
25	`double[][] f) {`
26	`    double[] errorLapisan3 = new double[3];`
27	
28	`    for (int i = 0; i < errorLapisan3.length; i++) {`
29	`        errorLapisan3[i] = 0;`
30	`        for (int j = 0; j < dataLatih.length; j++) {`
31	`            errorLapisan3[i] += errorLapisan4[i] * f[j][i];`
32	`        }`
33	`        errorLapisan3[i] = errorLapisan3[i] /`
34	`dataLatih.length;`
35	`    }`
36	`    return errorLapisan3;`

No	Kode Sumber
37	}
38	public double[] propagationError2(double[] errorLapisan3,
39	double[][] outputLayer2) {
40	double[] errorLapisan2 = new double[3];
41	errorLapisan2[0] = errorLapisan2[1] = errorLapisan2[2]
42	= 0;
43	for (int i = 0; i < dataLatih.length; i++) {
44	double jumlah = Math.pow((outputLayer2[i][0] +
45	outputLayer2[i][1] + outputLayer2[i][2]), 2);
46	errorLapisan2[0] += ((errorLapisan3[0] *
47	(outputLayer2[i][1] + outputLayer2[i][2])) -
48	(errorLapisan3[1] * outputLayer2[i][1]) -
49	(errorLapisan3[2] * outputLayer2[i][2])) / jumlah;
50	errorLapisan2[1] += ((errorLapisan3[1] *
51	(outputLayer2[i][0] + outputLayer2[i][2])) -
52	(errorLapisan3[0] * outputLayer2[i][0]) -
53	(errorLapisan3[2] * outputLayer2[i][2])) / jumlah;
54	errorLapisan2[2] += ((errorLapisan3[2] *
55	(outputLayer2[i][1] + outputLayer2[i][0])) -
56	(errorLapisan3[1] * outputLayer2[i][1]) -
57	(errorLapisan3[0] * outputLayer2[i][0])) / jumlah;
58	}
59	errorLapisan2[0] = errorLapisan2[0] /
60	dataLatih.length;
61	errorLapisan2[1] = errorLapisan2[1] /
62	dataLatih.length;
63	errorLapisan2[2] = errorLapisan2[2] /
64	dataLatih.length;
65	return errorLapisan2;
66	}
67	
68	public double[][] propagationError1(double[]
69	errorLapisan2) {
70	double[][] errorLapisan1 = new double[3][3];
71	for (int i = 0; i < errorLapisan1.length; i++) {
72	errorLapisan1[0][i] = errorLapisan1[1][i] =
73	errorLapisan1[2][i] = 0;
74	for (int j = 0; j < dataLatih.length; j++) {
75	errorLapisan1[0][i] += errorLapisan2[i] *
76	outputLayer1_1[j][i];
77	errorLapisan1[1][i] += errorLapisan2[i] *
78	outputLayer1_2[j][i];
79	errorLapisan1[2][i] += errorLapisan2[i] *
80	outputLayer1_3[j][i];
81	}
82	errorLapisan1[0][i] = errorLapisan1[0][i] /
83	dataLatih.length;
84	errorLapisan1[1][i] = errorLapisan1[1][i] /
85	dataLatih.length;
86	errorLapisan1[2][i] = errorLapisan1[2][i] /
87	dataLatih.length;
88	}
89	return errorLapisan1;
90	}

Kode Sumber 5.14 Implementasi Perhitungan *Propagation Error*

5.2.10 Implementasi Perbaikan Parameter Premis

Implementasi perhitungan *propogation error* merupakan bentuk realisasi dari perancangan Gambar 4.11. Setelah menghitung nilai *propagation error* tiap lapisan, selanjutnya dilakukan perbaikan parameter premis. Kode Sumber 5.15 berikut merupakan implementasi perbaikan parameter premis.

No	Kode Sumber
1	`public double[][] updateMean(double[][] data, double[][]`
2	`mean, double[][] std, double learningRate, double[][]`
3	`errorLapisan1) {`
4	`   double[][] errorMean = new`
5	`double[mean.length][mean[0].length];`
6	`   double[][] newMean = new`
7	`double[mean.length][mean[0].length];`
8	`   double[] avgX = new double[data[0].length - 1];`
9	`   for (int i = 0; i < avgX.length; i++) {`
10	`       avgX[i] = 0;`
11	`       for (int j = 0; j < data.length; j++) {`
12	`           avgX[i] += data[j][i];`
13	`       }`
14	`       avgX[i] = avgX[i] / data.length;`
15	`   }`
16	`   for (int i = 0; i < avgX.length; i++) {`
17	`       for (int j = 0; j < data[0].length - 1; j++) {`
18	`           double pembilangMean = 2 * Math.pow((avgX[j] -`
19	`mean[i][j]), 2) * errorLapisan1[i][j];`
20	`           double penyebutMean = Math.pow(std[i][j], 3) *`
21	`Math.pow(1 + (Math.pow(((avgX[i] - mean[i][j]) /`
22	`std[i][j]), 2)), 2);`
23	`           errorMean[i][j] = learningRate * (pembilangMean`
24	`/ penyebutMean) * avgX[i];`
25	`           System.out.println("error mean = " +`
26	`errorMean[i][j]);`
27	`           System.out.println("mean = " + mean[i][j]);`
28	`           newMean[i][j] = (mean[i][j]) +`
29	`(errorMean[i][j]);`
30	`           System.out.println("new mean = " +`
31	`newMean[i][j]);`
32	`       }`
33	`   }`
34	`   return newMean;`
35	` }`
36	
37	`public double[][] updateStd(double[][] data, double[][]`
38	`mean, double[][] std, double learningRate, double[][]`
39	`errorLapisan1) {`
40	`   double[][] errorStd = new`
41	`double[std.length][std[0].length];`
42	`   double[][] newStd = new`
43	`double[std.length][std[0].length];`
44	`   double[] avgX = new double[data[0].length - 1];`
45	`   for (int i = 0; i < avgX.length; i++) {`
46	`       avgX[i] = 0;`
47	`       for (int j = 0; j < data.length; j++) {`
48	`           avgX[i] += data[j][i];`
49	`       }`
50	`       avgX[i] = avgX[i] / data.length;`

No	Kode Sumber

51	}
	for (int i = 0; i < avgX.length; i++) {
	for (int j = 0; j < data[0].length - 1; j++) {
	double pembilangStd = 2 * Math.pow((avgX[j] - mean[i][j]), 2) * errorLapisan1[i][j];
	double penyebutStd = Math.pow(std[i][j], 2) * Math.pow(1 + (Math.pow(((avgX[i] - mean[i][j]) / std[i][j]), 2)), 2);
	errorStd[i][j] = learningRate * (pembilangStd / penyebutStd);
	newStd[i][j] = std[i][j] + errorStd[i][j];
	}
	}
	return newStd;
	}

Kode Sumber 5.15 Implementasi Perbaikan Parameter Premis

5.2.11 Implementasi Akurasi Pengujian

Implementasi akurasi pengujian ini berfungsi untuk menguji nilai RMSE yang dihasilkan dari hasil prediksi curah hujan menggunakan metode ANFIS. Implementasi akurasi pengujian pada Kode Sumber 5.16 ini merupakan realisasi dari persamaan 2.35. Kode sumber 5.16 berikut merupakan implementasi akurasi pengujian.

No	Kode Sumber
1	public void RMSE(double[][] targetAktual, double
2	outputLayer5[]) {
3	double rmse = 0;
4	int x = targetAktual[0].length - 1;
5	for (int i = 0; i < targetAktual.length; i++) {
6	rmse += Math.pow(targetAktual[i][x] -
7	outputLayer5[i], 2);
8	}
9	rmse = Math.sqrt(rmse / targetAktual.length);
10	System.out.println("");
11	System.out.println("RMSE = " + rmse);
12	}

Kode Sumber 5.16 Implementasi Akurasi Pengujian

BAB 6 PENGUJIAN

Pada bab ini membahas tentang pengujian hasil dari prediksi curah hujan menggunakan metode *Adaptive Neuro Fuzzy Inference System*. Pada penelitian ini menggunakan RMSE (*Root Mean Square Error*) sebagai hasil akurasi dari metode *Adaptive Neuro Fuzzy Inference System* untuk memprediksi curah hujan. Seperti dijelaskan pada Bab 2, RMSE adalah akar rata-rata total kuadrat *error* yang terjadi antara *output* proses dan *output* target, semakin kecil nilai RMSE maka semakin besar tingkat keberhasilan proses pelatihan. Pada pengujian ini menggunakan 280 data curah hujan per dasarian, yang mana 250 data tersebut merupakan data latih dan 30 data uji. Terdapat 3 skenario pengujian yang dilakukan, yaitu:

1. Pengujian pengaruh jumlah data latih terhadap RMSE
2. Pengujian pengaruh jumlah iterasi terhadap RMSE
3. Pengujian pengaruh *learning rate* terhadap RMSE

6.1 Pengujian Pengaruh Jumlah Data Latih Terhadap RMSE

Pengujian jumlah data latih bertujuan untuk mendapatkan jumlah data yang terbaik untuk digunakan dalam peramalan curah hujan. Pengujian dilakukan sesuai dengan rancangan pengujian jumlah data latih pada Tabel 4.35. Rincian parameter yang digunakan pada pengujian jumlah data latih adalah sebagai berikut:

a. Jumlah data uji : 30 data
b. Jumlah Iterasi ANFIS : 1000
c. *Learning rate* : 0,3
d. Nilai *minimum error* : 0,9

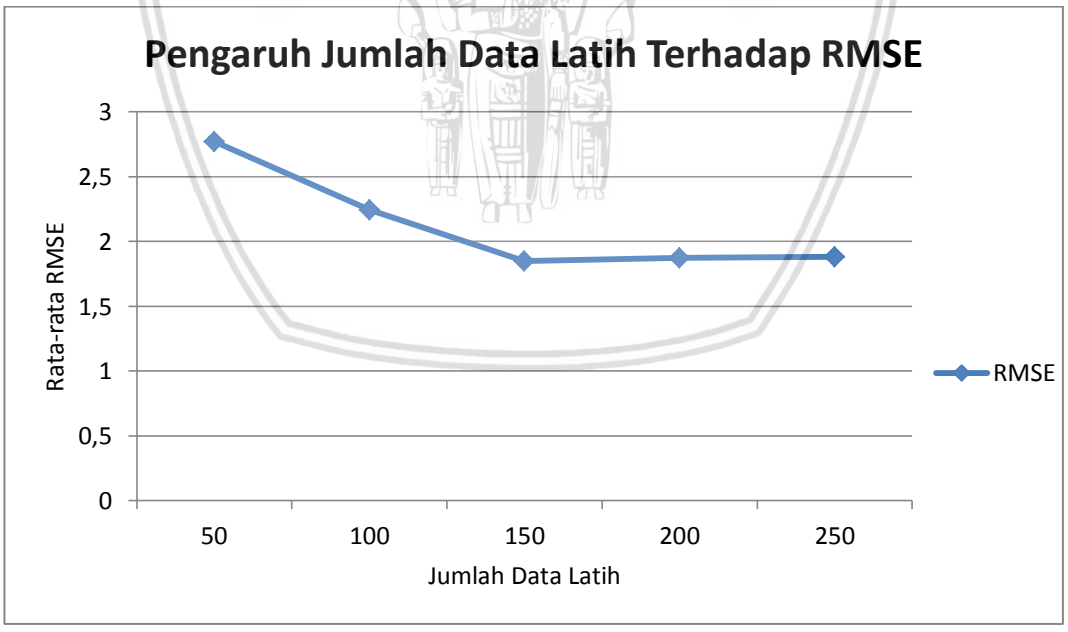
Pengujian pengaruh jumlah data latih terhadap RMSE dilakukan sebanyak 5 kali pada masing-masing jumlah data latih. Jumlah data latih yang digunakan yaitu sejumlah 50, 100, 150, 200, dan 250 data. Tabel 6.1 berikut merupakan hasil pengujian pengaruh jumlah data latih terhadap RMSE.

Tabel 6.1 Pengujian Jumlah Data Latih

Jumlah Data Latih	Nilai RMSE					Rata-Rata RMSE	Rata-rata Lama Komputasi (msec)
	Percobaan ke-i						
	1	2	3	4	5		
50	2,398	2,633	3,181	2,540	3,101	2,771	1002
100	2,027	2,448	2,148	2,402	2,186	2,242	1430
150	1,707	2,012	1,829	1,943	1,754	1,849	1813,6

Jumlah Data Latih	Nilai RMSE					Rata-Rata RMSE	Rata-rata Lama Komputasi (msec)
	Percobaan ke-i						
	1	2	3	4	5		
200	2,035	1,972	1,850	1,716	1,792	1,873	2808,8
250	1,919	2,059	1,785	1,808	1,834	1,881	2969,4

Berdasarkan tabel 6.1, pada percobaan pertama dengan 50 data latih menghasilkan RMSE senilai 2,771. Percobaan selanjutnya dengan jumlah data latih 100, terjadi penurun nilai RMSE yaitu senilai 2,242. Pada percobaan ke-3 terjadi penurunan RMSE yang cukup jauh yaitu senilai 1,849. Pada percobaan ke-4 dan ke-5, nilai RMSE cenderung stabil. Nilai RMSE yang terbaik didapatkan pada jumlah data latih 150 karena diperoleh hasil RMSE yang paling rendah dan lama komputasi yang tidak terlalu lama, dengan nilai rata-rata RMSE sebesar 1,849 dan rata-rata lama komputasi sebesar 1813,6 msec. Banyaknya waktu yang diperlukan untuk menjalankan sebuah metode berpengaruh terhadap hasil peramalan. Semakin lama waktu yang diperlukan maka metode tersebut tidak efisien (Priscilia, Umbara, & Jondri, 2014). Jumlah data latih 150 digunakan sebagai jumlah data latih terbaik yang akan digunakan pada skenario pengujian selanjutnya. Gambar 6.1 merupakan grafik yang menunjukkan lebih jelas pengaruh perubahan jumlah data latih terhadap nilai RMSE.



Gambar 6.1 Pengaruh Jumlah Data Latih Terhadap RMSE

6.2 Pengujian Pengaruh Jumlah Iterasi Terhadap RMSE

Pengujian pengaruh jumlah iterasi bertujuan untuk mengetahui pengaruh perubahan jumlah iterasi terhadap nilai RMSE yang akan dihasilkan. Pengujian dilakukan sesuai dengan rancangan pengujian jumlah iterasi pada Tabel 4.36. Rincian parameter yang digunakan pada pengujian jumlah iterasi adalah sebagai berikut:

a. Jumlah data latih : 150 data
b. Jumlah data uji : 30 data
c. *Learning rate* : 0,3

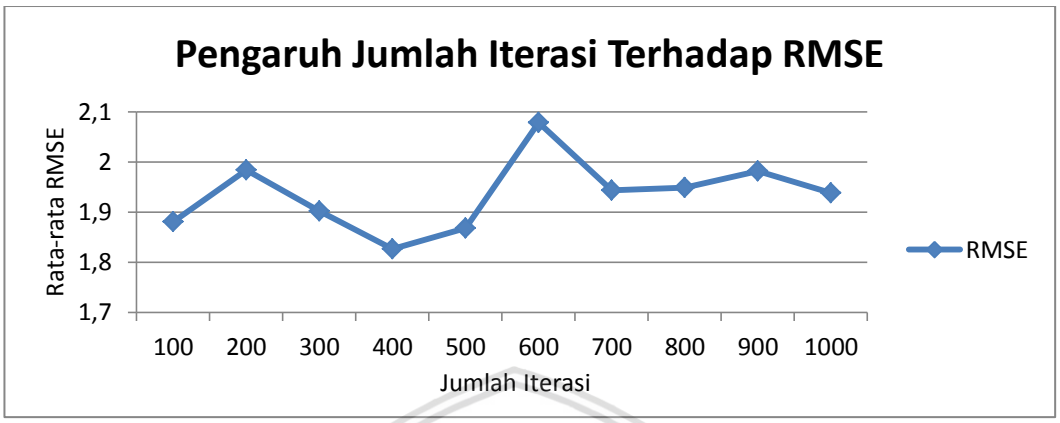
Pengujian pengaruh jumlah iterasi terhadap RMSE dilakukan sebanyak 5 kali pada masing – masing jumlah iterasi. Jumlah iterasi yang digunakan adalah 100, 200, 300, 400, 500, 600, 700, 800, 900, dan 1000. Tabel 6.2 berikut merupakan hasil pengujian pengaruh jumlah iterasi terhadap RMSE.

Tabel 6.2 Pengujian Jumlah Iterasi

Jumlah Iterasi	Nilai RMSE					Rata-rata RMSE	Rata-rata Lama Komputasi (msec)
	Percobaan ke-i						
	1	2	3	4	5		
100	1,826	1,890	1,960	1,878	1,850	1,881	1589
200	2,000	2,231	1,908	1,790	1,992	1,984	1669,8
300	1,761	1,845	2,095	1,974	1,834	1,902	1680
400	1,803	1,828	1,960	1,766	1,779	1,827	1825,2
500	1,778	1,982	1,879	1,996	1,706	1,868	2339,2
600	1,844	2,244	2,291	1,769	2,244	2,079	2564
700	1,819	2,142	1,834	1,913	2,006	1,943	2578,8
800	1,939	2002	1,888	1,963	1,953	1,949	2696,8
900	1,911	2,172	2,035	2,040	1,750	1,982	2798,2
1000	1,881	2,131	1,879	2,043	1,756	1,938	2817,6

Berdasarkan tabel 6.2, pada percobaan pertama jumlah iterasi 100 menghasilkan RMSE senilai 1,881. Percobaan selanjutnya dengan jumlah iterasi 200, nilai RMSE naik yaitu 1,984. Pada percobaan ke-3 dan ke-4 dengan jumlah iterasi 300 dan 400, terjadi penurunan nilai RMSE yaitu 1,902 dan 1,827. Berdasarkan pengujian iterasi, didapatkan hasil yang berubah-ubah terhadap RMSE. Nilai RMSE terbaik didapatkan pada iterasi 400 karena diperoleh hasil RMSE yang paling rendah dan lama komputasi yang tidak terlalu lama. Sehingga iterasi 400 digunakan sebagai parameter iterasi terbaik yang akan digunakan

pada skenario pengujian selanjutnya. Gambar 6.2 merupakan grafik yang menunjukkan lebih jelas pengaruh perubahan jumlah iterasi terhadap nilai RMSE.



Gambar 6.2 Pengaruh Jumlah Iterasi Terhadap RMSE

6.3 Pengujian Pengaruh *Learning Rate* Terhadap RMSE

Pengujian pengaruh *learning rate* bertujuan untuk mendapatkan nilai *learning rate* terbaik yang digunakan dalam perhitungan parameter premis. Pengujian dilakukan sesuai dengan rancangan pengujian learning rate pada Tabel 4.37. Rincian parameter yang digunakan pada pengujian learning rate adalah sebagai berikut:

a. Jumlah data latih : 150 data
b. Jumlah data uji : 30 data
c. Jumlah Iterasi : 400

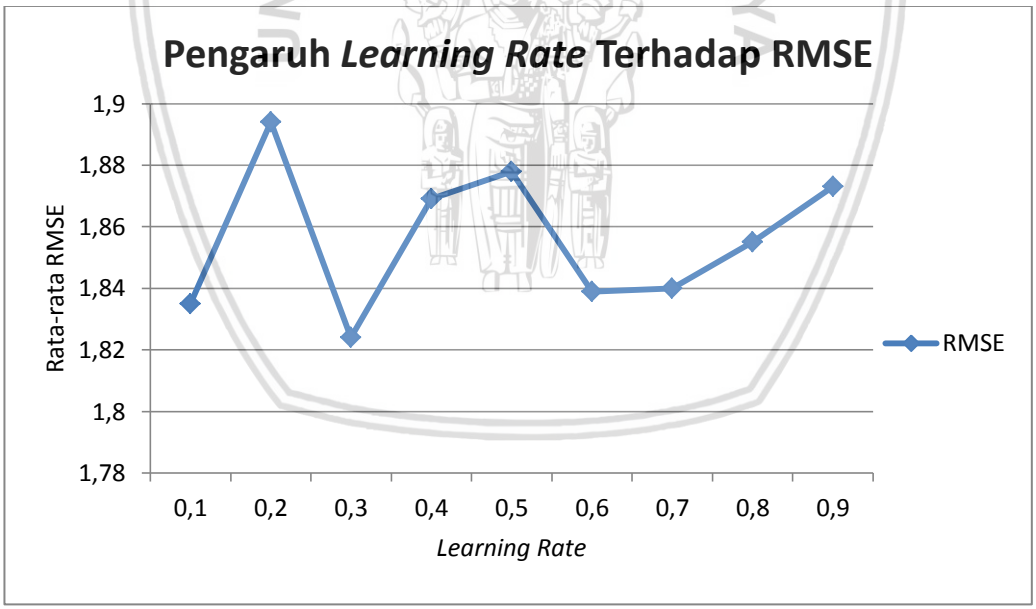
Pengujian pengaruh *learning rate* terhadap RMSE dilakukan sebanyak 5 kali pada masing – masing *learning rate*. Nilai *learning rate* yang digunakan adalah 0,1, 0,2, 0,3, 0,4, 0,5, 0,6, 0,7, 0,8, dan 0,9. Tabel 6.3 berikut merupakan hasil pengujian pengaruh *learning rate* terhadap RMSE.

Tabel 6.3 Pengujian *Learning Rate*

Learning Rate	**Nilai RMSE**					**Rata-rata RMSE**
	Percobaan ke-i					
	1	**2**	**3**	**4**	**5**	
0,1	1,970	1,865	1,780	1,763	1,797	1,835
0,2	1,866	1,793	1,920	1,906	1,983	1,894
0,3	1,865	1,773	1,852	1,806	1,822	1,824
0,4	1,810	1,874	1,790	1,809	2,062	1,869

Learning Rate	Nilai RMSE					Rata-rata RMSE
	Percobaan ke-i					
	1	2	3	4	5	
0,5	1,871	1,773	1,710	1,946	2,090	1,878
0,6	1,684	2,152	1,796	1,797	1,767	1,839
0,7	1,936	1,826	1,799	1,809	1,831	1,840
0,8	1,768	1,846	1,980	1,807	1,875	1,855
0,9	1,837	1,879	1,994	1,766	1,889	1,873

Berdasarkan tabel 6.3, pada percobaan pertama dengan *learning rate* 0,1 menghasilkan RMSE nilai 1,835. Percobaan selanjutnya dengan nilai *learning rate* 0,2, nilai RMSE naik menjadi 1,894. Percobaan ke-3 dengan *learning rate* 0,3, terjadi penurunan RMSE yaitu 1,824. Berdasarkan pengujian *learning rate*, Nilai RMSE yang dihasilkan cenderung tidak berubah, dan nilai RMSE terbaik didapatkan pada *learning rate* 0,3. Sehingga *learning rate* 0,3 akan digunakan pada skenario pengujian selanjutnya. Gambar 6.3 merupakan grafik yang menunjukkan lebih jelas pengaruh perubahan *learning rate* terhadap nilai RMSE.



Gambar 6.3 Pengaruh *Learning Rate* Terhadap RMSE

6.4 Hasil Prediksi Curah Hujan Menggunakan Metode *Adaptive Neuro Fuzzy Inference System*

Berdasarkan hasil pengujian yang telah dilakukan, telah didapatkan nilai *learning rate*, jumlah iterasi, serta jumlah data latih yang paling baik. Parameter-parameter tersebut akan digunakan untuk menghitung hasil prediksi curah hujan dengan menggunakan metode *Adaptive Neuro Fuzzy Inference System*. Nilai-nilai parameter terbaik tersebut antara lain:

1. *Learning Rate* = 0,3
2. Data Latih = 150 Data
3. Data Uji = 30 Data
4. Iterasi = 400

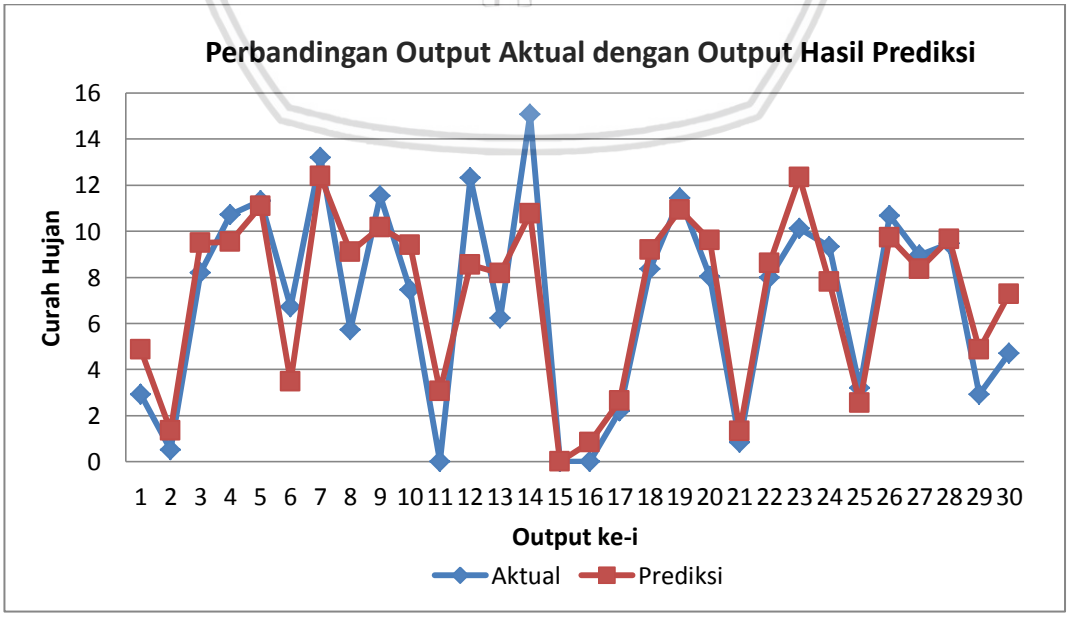
Dengan menggunakan nilai terbaik pada setiap parameter di atas, didapatkan hasil prediksi curah hujan. Tabel 6.4 adalah tabel hasil prediksi curah hujan beserta perbandingan dengan *output* aktualnya.

Tabel 6.4 Perbandingan *Output* Aktual dengan *Output* Hasil Prediksi

No	*Output* Aktual (mm)	*Output* Hasil Prediksi (mm)
1	2,92	4,869
2	0,525	1,347
3	8,21	9,500
4	10,73	9,544
5	11,32	11,094
6	6,71	3,469
7	13,2	12,379
8	5,72	9,096
9	11,53	10,173
10	7,47	9,407
11	0	3,064
12	12,32	8,538
13	6,24	8,171
14	15,07	10,756
15	0	0,019
16	0	0,841
17	2,21	2,649

No	*Output* Aktual (mm)	*Output* Hasil Prediksi (mm)
18	8,36	9,187
19	11,44	10,920
20	8,03	9,615
21	0,85	1,336
22	7,98	8,622
23	10,12	12,352
24	9,34	7,811
25	3,2	2,556
26	10,67	9,717
27	8,96	8,361
28	9,47	9,664
29	2,92	4,869
30	4,7	7,285

Hasil prediksi curah hujan menggunakan metode ANFIS sudah cukup baik dengan nilai RMSE yang dihasilkan sebesar 1,88. Penelitian sebelumnya oleh Wijaya (2018) yang melakukan prediksi curah hujan pada Kabupaten Malang menggunakan metode *High Order Fuzzy Time Series Multi Factors* diperoleh nilai RMSE sebesar 23,23. Hal ini menunjukkan bahwa metode *Adaptive Fuzzy Inference System* (ANFIS) lebih baik dibandingkan metode pada penelitian sebelumnya. Gambar 6.4 berikut merupakan grafik yang menggambarkan lebih jelas perbandingan antara *output* aktual dengan *output* hasil prediksi.



Gambar 6.4 Perbandingan *Output* Aktual dengan *Output* Hasil Prediksi

BAB 7 PENUTUP

Berdasarkan hasil pengujian yang mengacu pada impelementasi serta perancangan prediksi curah hujan dengan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) didapatkan beberapa kesimpulan sebagai berikut:

7.1 Kesimpulan

1. Proses prediksi curah hujan menggunakan metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) dilakukan dengan cara mengelompokkan data ke dalam 3 *cluster* dengan menggunakan algoritme *K-means*. Pengelompokan data menggunakan algoritme *K-Means* bertujuan untuk mendapatkan nilai parameter premis awal. Setelah proses *clustering*, proses selanjutnya adalah menghitung *output* lapisan 1 yaitu derajat keanggotaan dengan menggunakan fungsi keanggotaan *generalized bell*, Setelah itu menghitung *output* lapisan 2 yaitu nilai *fire strength* yang didapatkan dari hasil perkalian nilai derajat keanggotaan masing-masing cluster pada tiap parameter. Pada lapisan 3 dilakukan perhitungan normalisasi terhadap nilai *fire strength*. Setelah itu menghitung parameter konsekuen pada lapisan 4 dengan menggunakan algoritme *Least Square Error* (LSE). Kemudian pada lapisan 5 menghitung *output* jaringan. Setelah itu dilakukan proses perbaikan premis dengan menggunakan algoritme *steepest descent*. Pada proses *training* akan didapatkan parameter premis dan konsekuen yang nantinya akan digunakan untuk proses *testing* menggunakan data uji. Proses *testing* dilakukan dengan cara menghitung *output* lapisan 1 sampai lapisan 5 menggunakan parameter premis dan parameter konsekuen yang telah didapatkan dari proses *training*.
2. Berdasarkan hasil pengujian yang telah dilakukan, didapatkan:
    a. Parameter jumlah data latih berpengaruh terhadap nilai RMSE. Berdasarkan hasil pengujian, jumlah data yang menghasilkan nilai RMSE terbaik adalah 150 data.
    b. Perubahan nilai iterasi berpengaruh terhadap nilai RMSE disetiap percobaannya. Nilai RMSE yang dihasilkan fluktuatif, namun RMSE terbaik didapatkan dengan jumlah iterasi 400.
    c. Perubahan nilai learning rate mempengaruhi nilai RMSE yang dihasilkan. Nilai learning rate terbaik yang didapatkan berdasarkan hasil pengujian adalah 0,3.
    d. Ukuran besarnya kesalahan dan tingkat keberhasilan implementasi metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) untuk memprediksi curah hujan menggunakan Root Mean Square Error (RMSE) dan perhitungan akurasi data. Nilai RMSE yang dihasilkan oleh metode *Adaptive Neuro Fuzzy Inference System* (ANFIS) lebih kecil dibandingkan dengan metode *High Order Fuzzy Time Series Multi Factors.* Nilai RMSE yang dihasilkan pada metode *High Order Fuzzy Time Series Multi Factors* adalah 23,23. Sedangkan pada metode ANFIS

dihasilkan nilai RMSE sebesar 1,88. Penelitian ini membuktikan bahwa metode Adaptive Neuro Fuzzy Inference System (ANFIS) cukup baik digunakan untuk memprediksi curah hujan. Nilai RMSE dan tingkat akurasi yang dihasilkan menunjukkan bahwa metode *Adaptive Fuzzy Inference System* (ANFIS) cocok untuk diterapkan dalam menyelesaikan permasalah prediksi curah hujan.

7.2 Saran

Untuk kepentingan pengembangan metode, objek penelitian dan aplikasi yang dibangun, penulis memberikan saran, antara lain:

1. Penelitian selanjutnya diharapkan menggunakan data yang lebih lengkap. Karena data yang digunakan penulis saat ini terdapat beberapa data yang tidak terukur oleh alat ukur.
2. Penelitian selanjutnya diharapkan menggunakan algoritme clustering yang lain seperti fuzzy C-Means . Hal tersebut mungkin bisa berpengaruh untuk mendapatkan hasil pengelompokkan data dan hasil prediksi yang lebih baik.

DAFTAR PUSTAKA

Arikunto. (2006). *Prosedur Penelitian Suatu Pendekatan Praktek.* Jakarta: PT. Rineka Cipta.

BMKG. (2017). *Badan Meteorologi Klimatologi dan Geofisika*. Dipetik April 06, 2018, dari Stasiun Klimatologi Deli Serdang: http://bmkgsampali.net/normal-hujan-bulanan/

Darmawan, R. (2009). Perbandingan Jaringan Syaraf Tiruan Algoritma Backpropagation dan Algoritma Genetika Untuk Peramalan Data Time Series. Malang: Program Studi Statistika Jurusan Matematika Fakultas MIPA Universitas Brawijaya.

Dewi, C., & Himawati, W. W. (2015). Prediksi Tingkat Pengangguran Menggunakan Adaptive Neuro Fuzzy Inference System (ANFIS). Bali: ResearceGate.

Fatyanosa, T. N., & Mahmudy, W. F. (2016). Implementation of Real Coded Genetic Fuzzy System for Rainfall Forecasting. *International Conference on Sustainable Information Engineering and Technology (SIET)*, 24-33.

Gimpy, Vohra, R., & Minakshi. (2014). Estimation of Missing Values Using Decision Tree Approach. *(IJCSIT) International Journal of Computer Science and Information Technologies, 5*, 5216-5220.

Hidayatullah, A. F., Prasetyo, A. D., & Sari, D. P. (2014). Analisis Kualitas Data dan Klasifikasi Data Pasien Kanker. *Seminar Nasional Informatika Medis (SNIMed), V*.

Hung, M. C., Wu, J., Chang, J. H., & Yang, D.-L. (2005). An efficient K-Means Clustering Algorithm Using Simple Partitioning. *Journal Of Information Science and Engineering*, 1157-1177.

Husnita, S. D. (2012). Adaptive Neuro Fuzzy Inference System (ANFIS) Untuk Diagnosa Tingkatan Risiko Bagi Penderita Penyakit Jantung Koroner (PJK). Malang: Ilmu Komputer FMIPA Universitas Brawijaya.

Kuncahyo, B. T., Ginardi, R. V., & Arieshanti, I. (2012). Penerapan Metode Adaptive Neuro-Fuzzy Inference System Untuk Memprediksi Nilai Post Test Mahasiswa Pada Jurusan Teknik Informatika FTIF ITS. Surabaya: Fakultas Teknologi Informasi, ITS.

Kusumadewi, S. (2006). *Fuzzy Multi-Attribute Decision Making (FUZZY MADM).* Yogyakarta: Graha Ilmu.

Kusumadewi, S., & Hartati, S. (2010). *Neuro Fuzzy Integrasi Sistem Fuzzy & Jaringan Saraf* (Edisi 2 ed.). Yogyakarta: Graha Ilmu.

Kusumadewi, S., & Purnomo, H. (2010). *Aplikasi Logika Fuzzy untuk Pendukung Keputusan* (Edisi 2 ed.). Yogyakarta: Graha Ilmu.

Mancini, I. M., Lioi, D. S., Caniani, D., & Masi, S. (2012). *Fuzzy Logic and Neuro-Fuzzy Networks for Environmental Hazard Assessment.* Potenza: INTECH Open Access Publisher.

Priscilia, K., Umbara, R. F., & Jondri. (2014). Perbandingan Peramalam Indeks Harga Saham Gabungan Menggunakan Support Vector Machines Dan Jaringan Saraf Tiruan. *Seminar Nasional Ilmu Komputer & Teknik Informatika*.

Santika, G. D., Mahmudy, W. F., & Naba, A. (2017). Electrical Load Forecasting using Adaptive Neuro-Fuzzy Inference System. *International Journal of Advances in Soft Computing and its Applications, 9*(1).

Siswanti, K. Y., & Wutsqa, D. U. (2011). Peramalan Curah Hujan di Kota Yogyakarta dengan Model Fungsi Transfer Multivariant. *Prosiding Seminar Nasional Penelitian, Pendidikan dan Penerapan MIPA*, 343-358.

Siswanti, Y. K. (2011). Model Fungsi Transfer Multivariant dan Aplikasinya untuk Meramalkan Curah Hujan di Kota Yogyakarta. *Fakultas Matematika Dan Ilmu Pengetahuan Alam - Universitas Negeri Yogyakarta*, 1-180.

Suhartono. (2007). Feedforward Neural Networks Untuk Pemodelan Runtun Waktu. Yogyakarta: Disertasi Universitas Gajah Mada.

Surmaini, E., & Syahbuddin, H. (2016). Kriteria Awal Musim Tanam : Tinjauan Prediksi Waktu Tanam Padi Di Indonesia. *Jurnal Litbang Pertanian, 35*, 47-56.

Suyanto. (2008). *Soft Computing.* Bandung: Informatika.

Wahyuni, I., & Mahmudy, W. F. (2017). Rainfall Prediction in Tengger, Indonesia Using Hybrid Tsukamoto FIS and Genetic Algorithm Method. *Journal of ICT Research and Applications*, 38-55.

Wahyuni, I., Mahmudy, W. F., & Iriany, A. (2016). Rainfall Prediction in Tengger Region Indonesia using Tsukamoto Fuzzy Inference System. *International Conference on International Technology, Information Systems and Electrical Engineering (ICITISEE)*, 130-135.

Wahyuni, I., Mahmudy, W. F., & Iriany, A. (2017). Rainfall Prediction Using Hybrid Adaptive Neuro-Fuzzy Inference System (ANFIS) and Genetic Algorithm. *Journal of Telecommunication, Electronic and Computer Engineering, 9*(2-8), 51-56.

Wigena, A. H. (2006). *Pemodelan Statistical Downscaling dengan Regresi Projection Pursuit untuk Peramalan Curah Hujan Bulanan (Kasus Curah Hujan Bulanan di Indramayu.* Bogor.

Wijaya, A. B., Dewi, C., & Rahayudi, B. (2018). Peramalan Curah Hujan Menggunakan Metode High Order Fuzzy Time Series Multi Factors. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 930-939.

Wilson, E. M. (1990). *Engineering Hydrology Fourth Edition.* British: ELBS Macmillan Education.

Yao, J., & Tan, C. (2000). A Case Study on Using Neural Network to Perform Technical Forcasting of Forex. New Zealand: Department of Information System Massey University.