PREDIKSI JUMLAH PERMINTAAN KORAN MENGGUNAKAN METODE *EXTREME LEARNING MACHINE*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Amalia Ramadhanty
NIM: 125150100111016

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2017

PENGESAHAN

PREDIKSI JUMLAH PERMINTAAN KORAN MENGGUNAKAN METODE EXTREME LEARNING MACHINE

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Amalia Ramadhanty
NIM: 125150100111016

Skripsi ini telah diuji dan dinyatakan lulus pada
16 Januari 2017
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin, S.Si, M.Kom	Candra Dewi, S.Kom, M.Sc.
NIK: 201201 850719 1 001	NIP: 19771114 200312 2 001

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, 29 Desember 2016

Amalia Ramadhanty

NIM: 125150100111016

KATA PENGANTAR

Puji syukur kehadirat Tuhan YME yang telah melimpahkan rahmat-Nya sehingga laporan skripsi yang berjudul "Prediksi Jumlah Permintaan Koran Menggunakan Metode Extreme Learning Machine" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Imam Cholissodin, S.Si, M.Kom dan Ibu Candra Dewi, S.Kom, M.Sc selaku dosen pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
2. Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika
3. Bapak Novanto Yudistira, S.Kom, M.Sc selaku dosen penasehat akademik yang selalu memberikan nasehat kepada penulis selama menempuh masa studi.
4. Keluarga Tri Artono, serta seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
5. Diandra, Rozaq, Putra, Virgi, Khabib, Putri, Alfin, Nabilla, Redila, dan Sabrina, serta teman – teman Informatika Angkatan 2012 lainnya atas dukungan, masukan dan semangat yang diberikan kepada penulis sehingga terselesaikannya skripsi ini.
6. Seluruh civitas akademika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 29 Desember 2016

Penulis

amaliaramadhanty7@gmail.com

ABSTRAK

Kebutuhan masyarakat akan informasi semakin meningkat. Koran merupakan sumber berita dan informasi yang masih tetap digunakan oleh masyarakat Indonesia. Meskipun terdapat berbagai alternatif sumber berita lain, namun koran mempunyai nilai yang tidak dapat digantikan oleh berbagai jenis media elektronik lainnya karena kebenaran informasi yang terjamin dan lebih akurat. Sehingga minat masyarakat untuk membeli koran sebagai sumber berita juga masih besar. Namun tidak bisa dipastikan jumlah permintaan koran setiap harinya karena jumlah permintaan eceran yang tidak menentu. Hal ini menyebabkan semakin sulit bagi perusahaan koran untuk menentukan jumlah koran yang akan diproduksi. Dengan adanya perencanaan produksi yang tepat akan menghasilkan efisiensi jumlah koran dan meningkatkan keuntungan perusahaan. Prediksi yang akurat dan efektif dapat membantu menentukan jumlah permintaan koran yang efisien. Banyak metode yang telah digunakan dalam kasus prediksi. Metode ELM adalah salah satu metode yang bisa digunakan dalam kasus prediksi. Metode ELM dapat digunakan untuk memprediksi jumlah permintaan koran yang menghasilkan nilai *error* kecil. Sesuai dengan pengujian yang telah dilakukan dengan menggunakan data permintaan koran Jawa Pos Radar Madura tahun 2015 tingkat kesalahan yang rendah dihitung dengan MSE yaitu sebesar 0.009329.

Kata kunci: Koran, *Extreme Learning Machine*, *Mean Square Error* (MSE)

ABSTRACT

The public need for information is increasing. Newspapers are a source of news and information is still being used by Indonesian people. While there are many other alternative news sources, but the newspaper has a value that can not be replaced by various other types of electronic media for the truth of the information secure and more accurate. So that the public interest to buy newspapers as news source is still great. However it is not certain amount of requests daily newspaper for the number of retail demand is erratic. This causes more difficult for newspaper companies to determine the amount of paper to be produced. With the proper planning of production will generate efficiencies number of newspapers and increase corporate profits. Accurate and effective prediction can help determine the amount of demand for efficiency of newspaper. Many methods have been used in the case of prediction. ELM method is one method that can be used in the case of prediction. ELM methods can be used to predict the number of requests that newspapers generate an error value is small. In accordance with the testing that has been done using demand data newspaper Jawa Pos Radar Madura 2015 low error rate calculated by MSE is equal to 0.009329.

Keywords: *Newspapers, Extreme Learning Machine*, *Mean Square Error* (MSE)

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Koran menurut Kamus Besar Bahasa Indonesia (KBBI) adalah lembaran-lembaran kertas bertuliskan kabar (berita) dan sebagainya, yang terbagi dalam kolom-kolom (8—9 kolom), dan terbit setiap hari atau secara periodik. Radar Madura adalah salah satu perusahaan koran harian yang berpusat di Bangkalan, Madura, Jawa Timur. Radar Madura merupakan bagian dari Jawa Pos Group, Jawa Pos Group ini sendiri tersebar pada seluruh daerah di Indonesia(Jawa Pos Radar Madura, 2016). Jawa Pos Radar Madura merupakan koran harian terbesar di Madura dan merupakan koran harian dengan tingkat oplah terbesar di Madura, pada tahun 2015 rata-rata mencapai 244.946 eksemplar per bulannya (Wijono, 2016).

Koran atau surat kabar merupakan sumber berita yang masih tetap digunakan oleh masyarakat Indonesia. Sebagai contoh berdasar hasil riset lembaga riset Australia, Roy Morgan, pada tahun 2014 Jawa Pos setiap harinya dibaca rata-rata 1,4 juta orang, terbanyak dibanding dengan tiga koran besar lainnya yaitu Kompas sebanyak 1,2 juta orang, Poskota sebanyak 600 ribu orang, dan Suara Merdeka sebanyak 400 ribu orang. Hal ini membuktikan bahwa meskipun terdapat berbagai alternatif sumber berita lain seperti radio, televisi dan internet, namun koran mempunyai nilai yang tidak dapat digantikan oleh berbagai jenis media elektronik lainnya karena kebenaran informasi yang terjamin dan lebih akurat (JPNN, 2015).

Pembaca koran bisa membaca koran dengan berbagai cara, mulai dengan cara berlangganan, membeli eceran, membaca bersama di kantor, dan meminjam koran orang lain yang berlangganan. Berdasarkan data survei yang dilansir oleh Mars Indonesia tahun 2009, membaca koran dengan cara berlangganan sebanyak 15, 1 %, sementara itu pembaca yang membeli eceran sebanyak 54, 9 %, membaca bersama di kantor sebanyak 19, 6 %, dan meminjam koran orang lain yang berlangganan sebanyak 10, 4 %. Bisa dilihat minat masyarakat untuk membeli koran masih lebih besar di banding pembaca yang hanya meminjam atau membaca bersama di kantor. Sebesar 70% pembaca koran mendapatkan koran dengan cara membeli koran. Meski pembeli koran banyak namun tidak bisa dipastikan jumlah pembeli koran setiap harinya karena jumlah permintaan koran yang tidak menentu (Zumar, 2009).

Selama ini permintaan koran pada Radar Madura jumlahnya tidak sama setiap harinya. Jumlah permintaan koran untuk tahun 2015 bulan Januari rata-rata mencapai 8.010 eksemplar per hari, bulan Februari rata-rata sebanyak 8.304 per hari, bulan Maret rata-rata mencapai 8.437 eksemplar per hari, bulan April rata-rata sebanyak 8.130 per hari, bulan Mei rata-rata mencapai 8.414 eksemplar per hari, bulan Juni rata-rata sebanyak 8.082 per hari, bulan Juli rata-rata mencapai 7.127 eksemplar per hari paling sedikit di antara bulan lain, kemudian bulan Agustus rata-rata sebanyak 7.978 per hari, bulan September rata-rata

mencapai 7.865 eksemplar per hari, bulan Oktober rata-rata sebanyak 8.024 per hari, bulan November rata-rata mencapai 8.063 eksemplar per hari, dan paling banyak pada bulan Desember rata-rata mencapai 10.193 per hari (Jawa Pos Radar Madura, 2016).

Permintaan koran seperti yang disebutkan ada permintaan langganan atau permintaan tetap dan ada permintaan eceran. Permintaan eceran merupakan permintaan yang tidak pasti atau selalu berubah setiap harinya. Sehingga jumlah permintaan dan produksi koran setiap harinya tidak tetap. Sangat penting untuk perusahaan melakukan prediksi terhadap permintaan koran dengan tujuan mendapatkan jumlah produksi koran yang efisien.

Untuk mencapai tingkat laba optimal Radar Madura harus bisa menentukan jumlah koran yang akan diproduksi secara optimal, karena adanya resiko kelebihan dan kekurangan produksi. Dalam menentukan jumlah produksinya perusahaan melihat berdasarkan permintaan langganan, eceran dan pemberian gratis untuk pembuat iklan. Namun Radar Madura belum memiliki metode khusus dan belum dilengkapi perhitungan-perhitungan dalam menentukan prediksi jumlah permintaan koran. Prediksi yang akurat dan efektif dapat membantu Radar Madura dalam menentukan jumlah permintaan koran, sehingga koran yang akan diproduksi akan menghasilkan laba penjualan yang maksimal. Maka dari itu perlu dibuat suatu komputasi yang mampu memprediksi jumlah permintaan koran.

Pada skripsi ini akan diaplikasikan suatu metode dari JST yaitu *Extreme Learning Machine*(ELM). ELM adalah jaringan syaraf tiruan *feedforward* sederhana dengan satu *hidden layer* yang lebih dikenal dengan istilah *Single Hidden Layer Feedforward Neural Networks* (SLFNs). Metode ELM mempunyai kelebihan dalam learning speed dan mempunyai tingkat akurasi hasil prediksi yang lebih baik dibandingkan dengan metode konvensional seperti *Moving Average* dan *Exponential Smoothing* sehingga dengan menerapkan ELM pada *forecasting* mampu menghasilkan prediksi yang lebih efektif(Agustina dkk, 2009).

Metode ELM telah cukup banyak digunakan pada penelitian untuk membantu memecahkan masalah dalam kasus prediksi. Salah satu hasil penelitian menunjukkan bahwa metode ELM dapat diterapkan pada permasalahan peramalan permintaan. Hasil penelitian lainnya juga menunjukkan bahwa metode ELM ini sangat menjanjikan untuk permasalahan *time series prediction* (Singh & Balasundaram, 2007). Penelitian lain juga menunjukkan proses peramalan menggunakan metode ELM menunjukkan tingkat kesalahannya lebih kecil daripada metode *Backpropagation* (Khotimah dkk, 2010).

Berdasarkan latar belakang masalah yang telah dijabarkan di atas, penulis akan merancang perhitungan berbasis komputer agar memberikan solusi dalam memprediksi jumlah permintaan koran yang berjudul "Prediksi Jumlah Permintaan Koran Menggunakan Metode *Extreme Learning Machine*". Penulis berharap hasil penelitian ini dapat membantu perusahaan koran dalam

memprediksi jumlah permintaan koran sehingga dapat memproduksi koran secara optimal serta mendapatkan keuntungan yang maksimal.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan di atas, maka rumusan masalah yang dibahas adalah sebagai berikut:

1. Bagaimana implementasi metode *Extreme Learning Machine* (ELM) untuk memprediksi jumlah permintaan koran?
2. Bagaimana tingkat *error* hasil prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine* (ELM) yang diukur dengan perhitungan *Mean Square Error* (MSE)?

1.3 Tujuan

Adapun tujuan dari skripsi ini adalah sebagai berikut:

1. Mengimplementasikan metode *Extreme Learning Machine* (ELM) untuk memprediksi jumlah permintaan koran.
2. Menguji tingkat *error* hasil prediksi jumlah permintaan koran menggunakan metode Extreme Learning Machine (ELM) yang diukur dengan perhitungan *Mean Square Error* (MSE).

1.4 Manfaat

Adapun manfaat yang akan dapat diambil dari skripsi ini yaitu dapat membantu memprediksi jumlah permintaan koran agar perusahaan mendapatkan laba optimal dari produksi koran.

1.5 Batasan masalah

Mengingat banyaknya dan luasnya permasalahan, maka dalam skripsi ini dilakukan pembatasan masalah yaitu:

1. Data yang digunakan dalam skripsi ini menggunakan data permintaan koran harian selama satu tahun (tahun 2015) yang diperoleh dari Radar Madura.
2. Tidak menghiraukan hari-hari tertentu dalam melakukan prediksi jumlah permintaan koran.

1.6 Sistematika pembahasan

Adapun sistematika pembahasan pada skripsi ini adalah sebagai berikut:

BAB I Pendahuluan

Berisi tentang Latar Belakang Masalah, Rumusan Masalah, Tujuan, Manfaat Penulisan, Batasan Masalah, dan Sistematika Pembahasan.

BAB II Landasan Kepustakaan

Berisi uraian atau pembahasan tentang teori, konsep, dan metode atau sistem yang diperoleh dari literatur ilmiah, yang berkaitan dengan tema dan masalah atau pertanyaan penelitian.

BAB III Metodologi

Berisi mengenai studi literatur, pengumpulan data, analisis kebutuhan, perancangan sistem, pengujian sistem, dan kesimpulan.

Bab IV Perancangan

Membahas tentang bagaimana rancangan sistem yang akan dibuat dapat memberikan informasi prediksi jumlah permintaan koran di masa yang akan mendatang.

Bab V Implementasi

Membahas tentang implementasi prediksi jumlah permintaan koran menggunakan metode Extreme Learning Machine (ELM).

Bab VI Pengujian dan Pembahasan

Membahas tentang hasil pengujian akurasi pada metode yang telah digunakan yaitu metode *Extreme Learning Machine*.

Bab VII Penutup

Memuat kesimpulan yang diperoleh dari pengujian hasil prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine* (ELM) yang dikembangkan dalam tugas akhir ini serta saran untuk pengembangan yang lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini akan dijabarkan mengenai beberapa teori terkait metode *Extreme Learning Machine* (ELM), koran, permintaan, dan *Mean Square Error* (MSE) untuk menghitung tingkat *error* hasil prediksi jumlah permintaan koran. Penelitian-penelitian sebelumnya yang menyangkut implementasi dari metode *Extreme Learning Machine* (ELM) juga akan dibahas dalam kajian pustaka.

2.1 Kajian Pustaka

Metode *Extreme Learning Machine* (ELM) telah cukup banyak digunakan pada penelitian untuk membantu memecahkan masalah dalam kasus prediksi. Salah satu penelitian sebelumnya adalah mengenai peramalan permintaan. Penelitian ini dilakukan pada suatu perusahaan "Cak Cuk Shop" Surabaya yang memproduksi kaos dan pin. Hasil penelitian menunjukkan bahwa metode ELM dapat diterapkan pada permasalahan peramalan permintaan. Metode ELM menghasilkan *output* peramalan dengan tingkat kesalahan yang rendah yaitu 0,0042 % untuk kaos dan 0,0095 % untuk pin, dengan demikian ELM lebih bagus dibandingkan dengan metode konvensional *Moving Average* yang mempunyai tingkat kesalahan sebesar 19,19 % untuk kaos dan 55,43 % untuk pin serta *Exponential Smoothing* yang mempunyai tingkat kesalahan sebesar 32,93 % untuk kaos dan 111,39 % untuk pin(Agustina dkk, 2009).

Tinjauan kedua dilakukan berdasarkan penelitian sebelumnya mengenai *time series prediction*. Penelitian ini disimpulkan bahwa ELM merupakan metode yang menjanjikan untuk permasalahan *time series prediction* (Singh & Balasundaram, 2007).

Tinjauan ketiga mengenai peramalan jumlah kunjungan pasien yang diimplementasikan menggunakan metode ELM. Penelitian ini dilakukan pada Rumah Sakit Wahidin Sudiro Husodo Mojokerto Bagian Poli Gigi. Evaluasi pada penelitian ini dilakukan beberapa kali menggunakan 116 data *testing* melalui uji coba berdasarkan fungsi aktivasi dan jumlah *hidden layer*. Sehingga didapatkan hasil peramalan dengan akurasi yang optimal dan nilai *error* rendah yaitu sebesar 0,027 pada fungsi aktivasi *sigmoid biner* dan jumlah *hidden layer* sebanyak 7 unit serta *epoch* 500 (Fardani dkk, 2015).

Tinjauan keempat mengenai peramalan permintaan kebutuhan *reseller* yang diimplementasikan menggunakan ELM dalam konteks *Intelligent Warehouse Management System* (IWMS). Penelitian ini dilakukan pada perusahaan untuk mengoptimalisasi manajemen rantai pasok (*supply chain*). Model peramalan kebutuhan *reseller* menggunakan ELM dapat dikembangkan lebih lanjut sesuai dengan kebutuhan. Secara umum, penggunaan ELM cukup efektif dalam hal pendekatan hasil peramalan terhadap target (Atmojo dkk, 2013).

Tinjauan kelima melakukan penelitian mengenai kinerja metode *Extreme Learning Machine* (ELM) pada sistem peramalan. Dari hasil penelitian dapat disimpulkan bahwa metode ELM memiliki tingkat kesalahan lebih kecil daripada

metode *Backpropagation*. Metode ELM menghasilkan rata-rata nilai MSE sebesar 1,1 %, sedangkan proses pembelajaran menggunakan metode *Backpropagation* menghasilkan rata-rata nilai MSE sebesar 3,1933 % (Khotimah dkk, 2010).

Pada penelitian ini, akan menggunakan metode ELM untuk memprediksi jumlah permintaan koran. Perbandingan objek dan metode penelitian yang telah dilakukan terhadap tinjauan pustaka dari penelitian sebelumnya ditunjukkan oleh Tabel 2.1 berikut.

Tabel 2.1 Tinjauan Pustaka

No.	Judul	Objek	Metode	Hasil
1.	Penerapan Metode *Extreme Learning Machine* untuk Peramalan Permintaan	Kaos, Pin	*Extreme Learning Machine*(ELM)	Menghasilkan output peramalan dengan tingkat kesalahan yang kecil yaitu 0,0042 % untuk kaos dan 0,0095 % untuk pin, lebih baik dibandingkan dengan metode konvensional *Moving Average* dan *Exponential Smoothing*
2.	*Application of Extreme Learning Machine Method for Time Series Analysis*	*Time series prediction*	*Extreme Learning Machine*(ELM)	Metode ELM merupakan metode yang menjanjikan untuk permasalahan *time series prediction*
3.	Sistem Pendukung Keputusan Peramalan Jumlah Kunjungan Pasien Menggunakan Metode *Extreme Learning Machine* (Studi Kasus : Poli Gigi RSU Dr. Wahidin Sudiro Husodo Mojokerto)	Kunjungan Pasien	*Extreme Learning Machine*(ELM)	Hasil peramalan dengan akurasi yang optimal dan nilai *error* rendah yaitu sebesar 0,027 pada fungsi aktivasi *sigmoid* dan jumlah *hidden layer* sebanyak 7 unit serta *Epoch* 500
4.	Pengembangan Model Peramalan Permintaan Kebutuhan *Reseller*	Kebutuhan reseller	*Extreme Learning Machine*(ELM)	Metode ELM cukup efektif dalam hal pendekatan hasil prediksi atau

	Menggunakan *Extreme Learning Machine* Dalam Konteks *Intelligent Warehouse Management System* (IWMS)			peramalan terhadap target
5.	Kinerja Metode *Extreme Learning Maching*(ELM) Pada Sistem Peramalan	Data *time series* yang atributnya *dependent*	*Extreme Learning Machine*(ELM)	Metode ELM menghasilkan rata-rata nilai MSE sebesar 1,1 %, sedangkan proses pembelajaran menggunakan metode *Backpropagation* menghasilkan rata-rata nilai MSE sebesar 3,1933 %

2.2 Prediksi

Prediksi adalah suatu proses memperkirakan tentang sesuatu yang mungkin terjadi di masa yang akan datang berdasarkan informasi masa lampau dan informasi sekarang yang dimiliki, supaya kesalahannya (selisih antara yang akan terjadi dengan hasil prediksi atau perkiraan) dapat diperkecil. Prediksi tidak harus memberikan jawaban pasti tentang kejadian yang akan terjadi, melainkan berusaha untuk mencari kemungkinan jawaban kejadian sedekat mungkin dengan yang akan terjadi (Jaya, 2006).

Berdasarkan teknik yang digunakan prediksi dapat dibagi menjadi dua bagian yaitu prediksi kualitatif dan prediksi kuantitatif (Jaya, 2006).

1. Prediksi kualitatif didasarkan atas data kualitatif pada masa lalu. Metode kualitatif digunakan jika data masa lalu dari variabel yang akan diprediksi tidak ada, tidak cukup atau kurang dipercaya. Hasil prediksi yang dibuat sangat tergantung pada individu yang menyusunnya. Metode kualitatif ini disebut juga *judgemental*, *subjective*, dan *intuitive*.
2. Prediksi kuantitatif didasarkan atas data kuantitatif pada masa lalu. Hasil prediksi yang dibuat tergantung pada metode yang digunakan dalam prediksi tersebut. Dengan metode yang berbeda, maka akan diperoleh hasil prediksi yang berbeda. Metode yang baik adalah yang memberikan nilai-nilai penyimpangan atau perbedaan yang kemungkinan kecil.

Pada penelitian ini menggunakan teknik prediksi kuantitatif. Karena prediksi ini didasarkan pada data kuantitatif pada masa lalu.

2.3 Permintaan

Menurut Gilarso (2004), dalam ilmu ekonomi dikenal istilah permintaan (*demand*) yang mempunyai arti tertentu. Permintaan yaitu selalu menunjuk pada hubungan tertentu antara jumlah barang yang akan dibeli dan harga barang tersebut. Permintaan juga bisa diartikan dengan jumlah dari barang yang akan dan mampu dibeli dengan berbagai kemungkinan harga, selama waktu tertentu, dengan anggapan hal-hal lainnya tetap sama.

Permintaan jumlah koran Jawa Pos Radar Madura mengalami fluktuasi yang diikuti dengan jumlah penjualannya. Untuk jumlah permintaan Jawa Pos Radar Madura selama tahun 2015 disajikan pada Tabel 2.2 berikut.

Tabel 2.2 Jumlah Permintaan Jawa Pos Radar Madura Tahun 2015

Bulan	Volume Permintaan
Januari	248.299
Februari	232.503
Maret	261.558
April	243.896
Mei	260.833
Juni	242.462
Juli	220.928
Agustus	247.328
September	235.938
Oktober	248.759
November	241.883
Desember	315.994

Sumber: Jawa Pos Radar Madura (2016)

Menurut Danniel, permintaan dipengaruhi beberapa faktor. Faktor yang mempengaruhi permintaan antara lain adalah harga barang, harga barang komplemen atau substitusinya, jumlah penduduk, selera, dan tingkat pendapatan (Danniel, 2004).

2.4 Koran

Menurut Jhon Tabbel dalam bukunya yang berjudul Karier Jurnalistik, pengertian koran adalah rangkuman semua isi berita yang ditulis melalui media cetak sebagai sarana komunikasi massa, dimana koran ini diperuntukkan untuk umum yang menyangkut kepentingan umum, serta berita yang disajikan dalam koran tersusun dalam alinea kalimat yang mencetak pada kertas(Tabbel, 2003).

Radar Madura adalah salah satu perusahaan koran harian yang berpusat di Bangkalan, Madura, Jawa Timur. Radar Madura merupakan bagian dari Jawa Pos Group, Jawa Pos Group ini sendiri tersebar pada seluruh daerah di Indonesia. Jawa Pos Radar Madura merupakan koran harian terbesar di Madura dan merupakan koran harian dengan tingkat oplah terbesar di Madura, pada tahun 2015 rata-rata mencapai 244.946 eksemplar per bulannya (Wijono, 2016).

Contoh halaman pertama koran harian Jawa Pos Radar Madura dapat dilihat pada Gambar 2.1.


RADAR MADURA

29

Tuan Rumah Kurang Beruntung

46 Hari Mangkir, Terancam Pecat

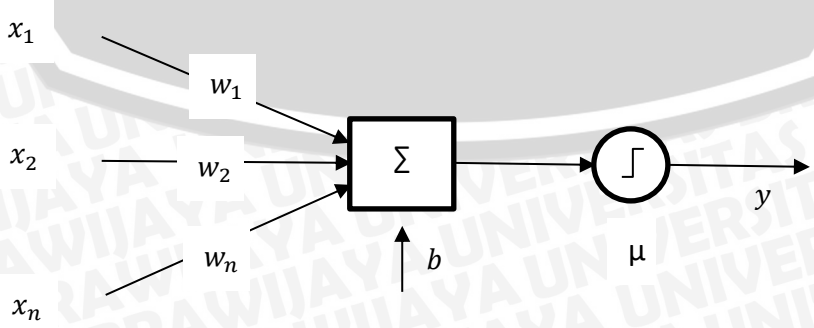
Obrak PKL demi Adipura

Menurun 8,84 Persen, dari 22.333 Hektare Tersisa 20.358 Hektare

Gambar 2.1 Koran Radar Madura

2.5 Jaringan Syaraf Tiruan (JST)

Jaringan syaraf tiruan terdiri atas beberapa *node* atau *neuron* dan terdapat *relasi* (hubungan) antara *neuron-neuron* tersebut yang selanjutnya akan mentranformasikan informasi yang akan diterima melalui *neuron-neuron* lainnya. Struktur *node* atau *neuron* pada JST ditunjukkan pada Gambar 2.2 berikut ini.



Gambar 2.2 Struktur *Neuron* Jaringan Syaraf Tiruan

Struktur *neuron* mempunyai n sinyal masukan, yaitu $x_1, x_2, ..., x_n$ dengan $x \in \{0,1\}$. Masing-masing sinyal tersebut kemudian akan dimodifikasi oleh bobot $w_1, w_2, ..., w_3$ sehingga sinyal yang diterima oleh *neuron* adalah $x_i{}^1 = x_i w_i,\ i = 1, 2, \dots, n$. Selanjutnya *neuron* menghitung hasil penjumlahan semua sinyal *input*. Fungsi aktivasi yang terdapat dalam model JST umumnya berupa fungsi non-linier. Tingkat aktivasi diwujudkan dalam nilai ambang (*threshold*). Apabila *input* tersebut melebihi suatu nilai ambang tertentu, maka *neuron* tersebut diaktifkan dan mengirimkan *output* (Sinuhaji, 2009).

Berikut ini beberapa fungsi aktivasi yang sering digunakan dalam JST menggunakan metode ELM (Singh & Balasundaram, 2007) yaitu:

1. Fungsi aktivasi *sigmoid*

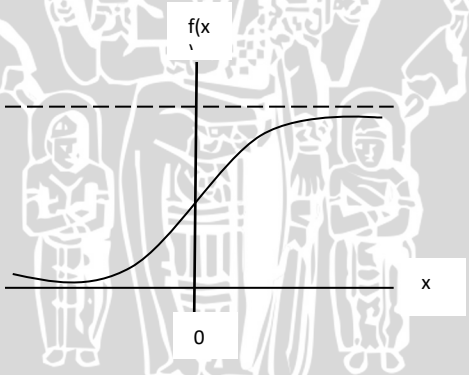
   Fungsi aktivasi *sigmoid* memiliki interval *output* 0 sampai 1. Fungsi aktivasi *sigmoid* dapat dirumuskan seperti Persamaan 2.1. Dan ilustrasi fungsi aktivasi *sigmoid* ditunjukkan pada Gambar 2.3.

   $$f(x) = \frac{1}{1+e^{-x}} \quad (2.1)$$

   Keterangan:

   $f(x)$ = Fungsi aktivasi *sigmoid*

   e^{-x} = Eksponensial pangkat minus data ke-x



Gambar 2.3 Ilustrasi Fungsi Aktivasi *Sigmoid*

2. Fungsi aktivasi s*in*

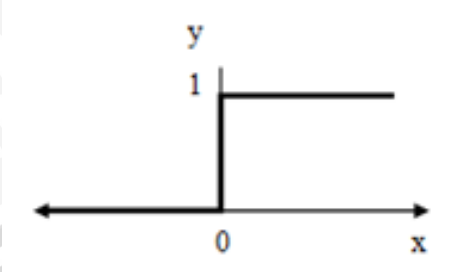
   Fungsi aktivasi *sin* memiliki nilai pada rentang -1 sampai 1. Secara matematis, fungsi aktivasi *sin* dituliskan seperti Persamaan 2.2.

   $$f(x) = \sin x \quad (2.2)$$

3. Fungsi aktivasi *hardlim*

   Fungsi aktivasi *hardlim* ini biasanya digunakan oleh jaringan lapisan tunggal untuk menkonversi nilai *input* yang bernilai kontinu menjadi suatu nilai *output* biner (0 atau 1). Secara matematis, fungsi aktivasi *hardlim* dituliskan seperti Persamaan 2.3. Ilustrasi fungsi aktivasi *hardlim* ditunjukkan pada Gambar 2.4.

$$y = \begin{cases} 0, x \le 0 \\ 1, x > 0 \end{cases} \quad (2.3)$$



Gambar 2.4 Ilustrasi Fungsi Aktivasi *Hardlim*

2.6 Metode *Extreme Learning Machine* (ELM)

Extreme Learning Machine (ELM) merupakan metode pembelajaran baru dari JST. Metode ini diperkenalkan oleh Huang (2004). ELM adalah jaringan syaraf tiruan *feedforward* sederhana dengan satu *hidden layer* yang lebih dikenal dengan istilah *Single Hidden Layer Feedforward Neural Networks* (SLFNs) (Sun et al, 2008).

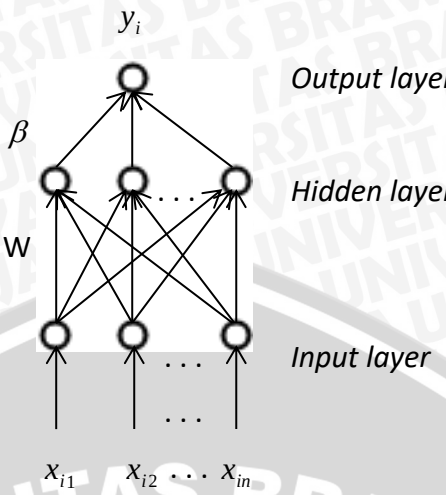
Metode ELM ini dibuat untuk mengurangi kelemahan-kelemahan JST *feedforward* terutama mengenai *learning speed*. Huang et al mengemukakan alasan mengapa JST *feedforward* memiliki *learning speed* rendah, yaitu:

1. Menggunakan *slow gradient based learning algorithm* untuk melakukan *training*.
2. Semua parameter pada jaringan ditentukan secara *iterative* dengan menggunakan metode pembelajaran tersebut.

Secara umum, semua parameter dari jaringan *feedforward* ditentukan secara manual. Parameter yang dimaksud yaitu *input weight* dan *bias*. Parameter tersebut saling berhubungan antara *layer* yang satu dengan yang lain, sehingga membutuhkan *learning speed* yang lama (Huang et al, 2005).

Sedangkan pada ELM parameter-parameter seperti *input weight* dan *hidden bias* dipilih secara *random* dari nilai tertentu, sehingga ELM memiliki *learning speed* yang cepat dan mampu menghasilkan *good generalization performance*. Dengan *random* pada *range* tertentu, bisa menghindari hasil prediksi yg tidak stabil.

Secara umum model jaringan syaraf tiruan yang menggunakan ELM sebagai metode pembelajarannya menurut Huang, dkk. dapat dilihat pada Gambar 2.2.



Gambar 2.5 Struktur ELM

Metode ELM mempunyai model matematis yang berbeda dari jaringan syaraf tiruan *feedforward*. Model matematis dari ELM lebih sederhana dan efektif.

Langkah-langkah perhitungan dengan metode ELM ini dibedakan menjadi 2 proses, yaitu proses *training* dan proses *testing*.

2.6.1 Proses *Training*

Sebelum digunakan untuk proses prediksi, ELM harus melalui proses *training* terlebih dahulu. Tujuan dari proses ini adalah mendapatkan *output weight* dengan tingkat kesalahan yang rendah. Langkah-langkah proses *training* metode ELM yaitu sebagai berikut:

1. Inisialisasi *input weight* dan *bias* dengan bilangan acak yang kecil dengan *range* -1 hingga 1(Liang dkk, 2006).
2. Menghitung semua keluaran di *hidden layer* dengan menggunakan fungsi aktivasi. Perhitungan keluaran *hidden layer* ditunjukkan pada Persamaan 2.4. Setelah keluaran *hidden layer* (H_{init}) didapatkan, kemudian H_{init} dihitung dengan menggunakan fungsi aktivasi yang ditunjukkan pada Persamaan 2.1 atau Persamaan 2.2.

$$H_{init\ ij} = \left(\sum_{k=1}^{n} w_{jk} . x_{ik} \right) + b_j \qquad (2.4)$$

Keterangan:

H_{init} = Matriks keluaran *hidden layer*.

i = [1, 2, ..., N], dimana N adalah jumlah data.

j = [1, 2, ..., Ñ], dimana Ñ adalah jumlah hidden neuron.

n = Jumlah *input neuron*.

w = Bobot *input*.

x = *Input*.

b = *Bias*.

3. Menghitung *output weight* dari *hidden layer* ke *output layer*. Untuk menghitung *output weight*, pertama harus menghitung Matriks *Moore-Penrose Generalized Invers* dari hasil keluaran *hidden layer* dengan fungsi aktivasi. Setelah itu baru dihitung *output weight* yang ditunjukkan pada Persamaan 2.5.

$$\beta = H^{+}T \quad (2.5)$$

Keterangan:

β = Matriks *Output weight* dari *hidden layer* ke *output layer*.

H^{+} = Matriks *Moore-Penrose Generalized Invers* dari matriks H .

T = Matriks Target

2.6.2 Proses *Testing*

Proses *testing* dilakukan berdasarkan *input weight*, *bias*, dan *output weight* yang sesuai dari perhitungan *training*. Proses *training* bertujuan untuk mengembangkan model ELM, sedangkan proses *testing* bertujuan untuk mengevaluasi kemampuan ELM sebagai metode untuk memprediksi. Langkah-langkah proses *testing* yaitu sebagai berikut:

1. Inisialisasi *input weight* dan *bias* yang didapatkan pada perhitungan *training*.
2. Menghitung semua keluaran di *hidden layer* dengan menggunakan fungsi aktivasi ($H(x)$) menggunakan Persamaan 2.4.
3. Menggunakan hasil *output weight* dari *hidden layer* ke *output layer* dari hasil proses *training*. Menghitung keluaran di *output layer* yang merupakan *output* hasil prediksi yang ditunjukkan pada Persamaan 2.6.

$$y = H\beta \quad (2.6)$$

Keterangan:

y = *Output* hasil prediksi.

β = *Output weight*.

H = Keluaran *Hidden layer* dengan fungsi aktivasi.

4. Menghitung nilai *error* semua *output layer* dengan menggunakan Persamaan 2.9.

2.7 Normalisasi Data

Pada metode ELM, umumnya fungsi aktivasi yang digunakan adalah fungsi *sigmoid* dan *sin* (Singh & Balasundaram, 2007). Fungsi tersebut akan membawa

nilai *input* dengan *range* yang tak terbatas ke nilai *output* yang terbatas, yaitu dalam *range* 0 sampai 1. Supaya dapat membawa *range* nilai *output* ke dalam range *input*, maka data *input* harus dilakukan normalisasi data ke dalam *range* 0 sampai 1 (Chamidah, 2012). Metode normalisasi yang digunakan adalah *Min-Max Normalization* seperti yang ditunjukkan pada Persamaan 2.7, metode ini mengubah data ke *range* baru lain yaitu antara 0 sampai 1.

$$x' = \frac{x - \min}{\max - \min} \tag{2.7}$$

Keterangan:

x' = nilai data setelah dinormalisasi

x = nilai data sebelum dinormalisasi

$\min$ = nilai minimum pada data set

$\max$ = nilai maksimal pada data set

2.8 Denormalisasi Data

Denormalisasi data digunakan untuk mendapatkan nilai sebenarnya dari *output* berdasarkan hasil *output* prediksi. Proses denormalisasi data dijabarkan pada Persamaan 2.8.

$$x = x'(\max - \min) + \min \tag{2.8}$$

Keterangan:

x' = nilai data sebelum didenormalisasi

x = nilai data sesudah didenormalisasi

$\min$ = nilai minimum pada data set

$\max$ = nilai maksimal pada data set

2.9 *Mean Square Error* (MSE)

Mean Square Error (MSE) adalah metode untuk mengevaluasi hasil prediksi. Masing-masing kesalahan atau sisa dikuadratkan. Pendekatan ini mengatur kesalahan peramalan yang besar karena kesalahan-kesalahan itu dikuadratkan. Time series dengan skala yang berbeda tidak dapat menggunakan MSE sebagai metode evaluasi peramalan (Hyndman & Koehler, 2005).

$$MSE = \frac{\sum_{i=1}^{n} e_i^2}{n} = \frac{\sum_{i=1}^{n} (y_i - t_i)^2}{n} \tag{2.9}$$

Keterangan:

n = Jumlah data

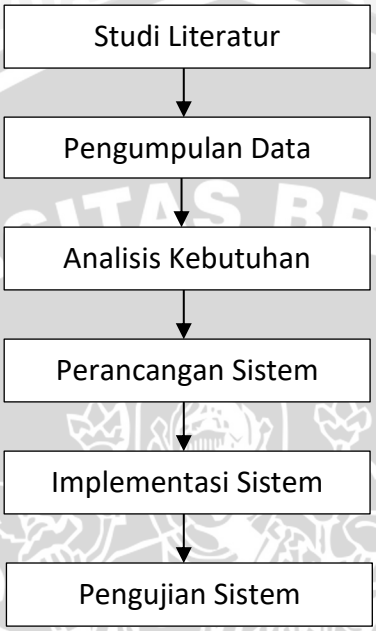
e = *Error*

y_i = Nilai *output* (prediksi)

t_i = Nilai aktual

BAB 3 METODOLOGI

Dalam bab ini akan dijelaskan mengenai metodologi yang digunakan untuk memprediksi jumlah permintaan koran. Untuk dapat memberikan kemudahan dalam menjelaskan metodologi yang digunakan, maka penulis menggunakan diagram alir seperti yang tertera pada Gambar 3.1.



Gambar 3.1 Diagram Alir

3.1 Studi Literatur

Studi literatur ini digunakan untuk menggali informasi dan mengkaji dari berbagai literatur yang mendukung penelitian penulis. Literatur berasal dari buku, jurnal, artikel dan dokumentasi. Pokok bahasan literatur meliputi:

1. Jumlah permintaan koran
2. Metode *Extreme Learning Machine*(ELM)
3. MSE untuk menghitung tingkat *error*

Sumber yang diperoleh untuk melakukan studi literatur ini berupa jurnal, laporan ilmiah, wawancara, observasi, dan paper.

3.2 Pengumpulan Data

Teknik pengumpulan data dilakukan dengan observasi ke Radar Madura, narasumber dari bagian sirkulasi dan juga dari literatur buku-buku yang terkait. Data yang digunakan dalam penelitian ini berupa jumlah permintaan koran harian selama satu tahun. Data-data pada penelitian ini digunakan untuk menghitung prediksi jumlah permintaan koran menggunakan metode *Extreme*

Learning Machine yang kemudian akan dihitung tingkat *error* dari hasil prediksi tersebut.

3.3 Analisis Lingkungan Implementasi

Analisis lingkungan implementasi bertujuan untuk menganalisis dan mendapatkan semua kebutuhan yang diperlukan dalam pembuatan sistem prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine*. Analisis lingkungan disesuaikan dengan kebutuhan sistem. Sistem dapat menampilkan data permintaan koran yang kemudian dihitung dengan metode *Extreme Learning Machine* untuk memprediksi jumlah permintaan koran. Sistem akan menampilkan hasil prediksi jumlah permintaan koran. Kemudian sistem juga menampilkan tingkat *error* dari prediksi tersebut.

3.4 Perancangan Proses

Perancangan proses dibangun berdasarkan hasil analisis kebutuhan dan pengambilan data yang telah dilakukan pada prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine*.

Pada perancangan sistem dijelaskan tentang rancangan langkah kerja dari sistem secara menyeluruh untuk mempermudah implementasi, pengujian, dan analisis. Langkah-langkah yang dilakukan dalam perancangan sistem ini disesuaikan dengan perancangan metode *Extreme Learning Machine* (ELM). Proses sistem adalah perhitungan metode ELM untuk memprediksi jumlah produksi surat kabar. Keluaran atau output berupa nilai prediksi dan *error*/nilai kesalahan.

3.5 Pengujian dan Analisis

Pengujian dan analisis dilakukan untuk mengetahui apakah sistem berjalan dengan baik dan sesuai dengan spesifikasi yang telah ditetapkan. Pengujian sistem dilakukan dengan 5 cara, yaitu:

1. Pengujian jumlah *neuron* pada *hidden layer*

   Pengujian jumlah *neuron* pada *hidden layer* digunakan agar menghasilkan *output* prediksi yang stabil. Jumlah neuron pada hidden layer yang diuji yaitu 1 hingga 7 hidden neuron.

2. Pengujian fungsi aktivasi

   Pengujian fungsi aktivasi dilakukan dengan uji coba menggunakan fungsi aktivasi *sigmoid* dan *sin* karena kedua fungsi tersebut yang paling sering digunakan pada permasalahan *forecasting* menggunakan metode ELM.

3. Pengujian perbandingan jumlah data *training* dan data *testing*

   Pengujian perbandingan jumlah data *training* dan data *testing* ini dilakukan untuk mengetahui pengaruh dari perbandingan jumlah data *training* dengan data *testing* terhadap nilai MSE yang dihasilkan. Perbandingan jumlah data

training dan data *testing* yang diuji yaitu 80%:20%, 70%:30%, 60%:40%, 50%:50%, dan 40%:60%(keterangan: data *training* : data *testing*)

4. Pengujian variasi fitur data

   Pengujian variasi fitur data digunakan untuk mengetahui pola data terbaik yang cocok dan menghasilkan nilai MSE yang minimal. Variasi jumlah fitur data yang diuji yaitu 2 hingga 7 fitur. Yang dimaksudkan 2 hingga 7 disini adalah memprediksi jumlah permintaan koran yang dilihat dari 2 hingga 7 hari sebelumnya.

5. Pengujian berdasarkan hari yang sama

   Pengujian berdasarkan hari digunakan untuk mengetahui pengaruh hari yang sama terhadap nilai MSE yang dihasilkan. Hari yang diuji yaitu senin, selasa, rabu, kamis, jumat, sabtu, dan minggu.

Dari kelima pengujian tersebut, nantinya didapatkan hasil parameter terbaik pada setiap pengujian.

BAB 4 PERANCANGAN

Bab ini menjelaskan formulasi permasalahan, siklus algoritma *Extreme Learning Machine*(ELM), perhitungan manual, serta perancangan antar muka.

4.1 Formulasi Permasalahan

Permasalahan yang akan diselesaikan adalah memprediksi jumlah permintaan koran. Hasil prediksi kemudian dievaluasi tingkat nilai *error*-nya untuk mengetahui kualitas hasil prediksi. Semakin kecil nilai *error* semakin baik pula kualitas hasil prediksi.

Solusi untuk permasalahan prediksi jumlah permintaan koran yaitu dengan membuat perhitungan atau komputasi menggunakan metode ELM. Masukan untuk komputasi ini antara lain data permintaan koran dalam format .xls dan parameter perhitungan meliputi: jumlah *neuron* pada *hidden layer*, fungsi aktivasi, dan perbandingan jumlah data training serta data testing. Selanjutnya masuk pada proses perhitungan prediksi menggunakan metode *Extreme Learning Machine*(ELM). Inisialisasi data *training* dan data *testing* harus dinormalisasi terlebih dahulu sebelum memulai perhitungan metode ELM, normalisasi yang digunakan yaitu *MinMax Normalization*. Selanjutnya masuk pada proses evaluasi menghitung nilai *error* menggunakan metode *Mean Square Error* (MSE). Keluaran dalam komputasi ini antara lain hasil prediksi dan nilai *Mean Square Error* (MSE).

Adapun sampel data permintaan koran yang digunakan pada penelitian ini berjumlah 10 data sebagai berikut: (data lengkap disertakan pada Lampiran B)

Tabel 4.1 Data Permintaan Koran

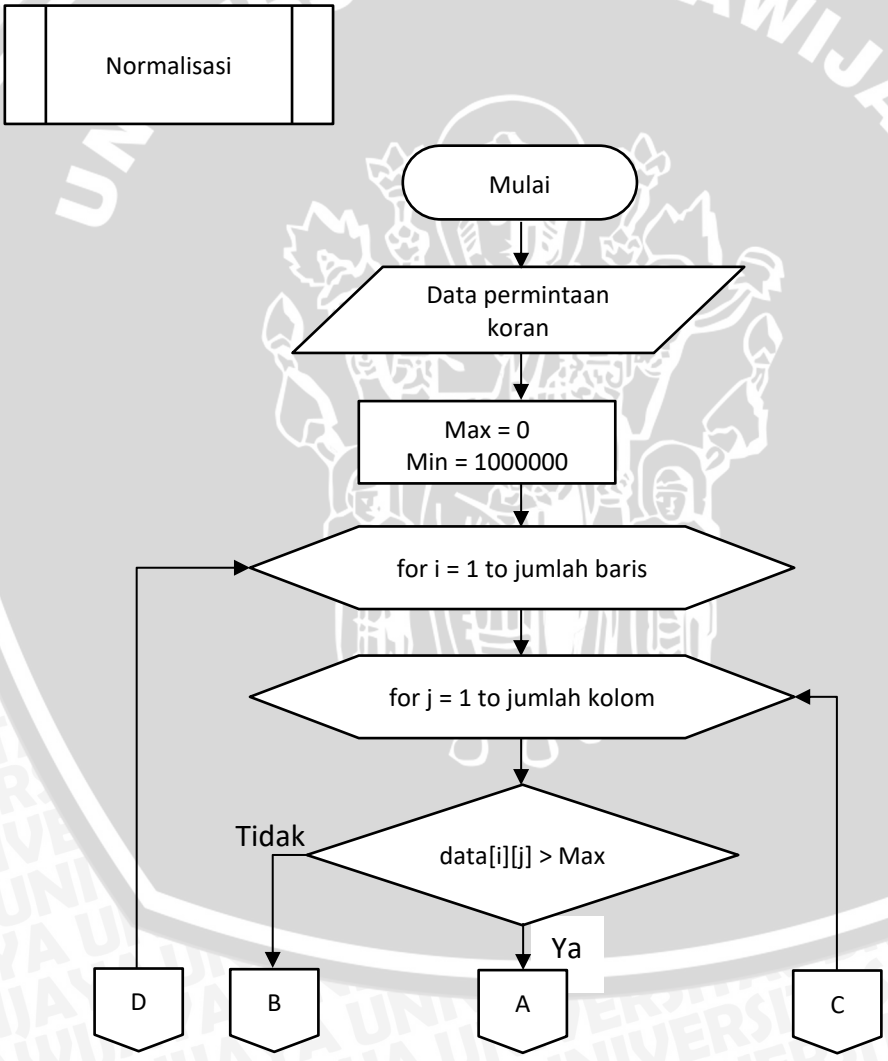
Data ke-	Tanggal	Jumlah permintaan
1	1 Januari 2015	7899
2	2 Januari 2015	7961
3	3 Januari 2015	7560
4	4 Januari 2015	8166
5	5 Januari 2015	7899
6	6 Januari 2015	7911
7	7 Januari 2015	7987
8	8 Januari 2015	7947
9	9 Januari 2015	7836
10	10 Januari 2015	8206

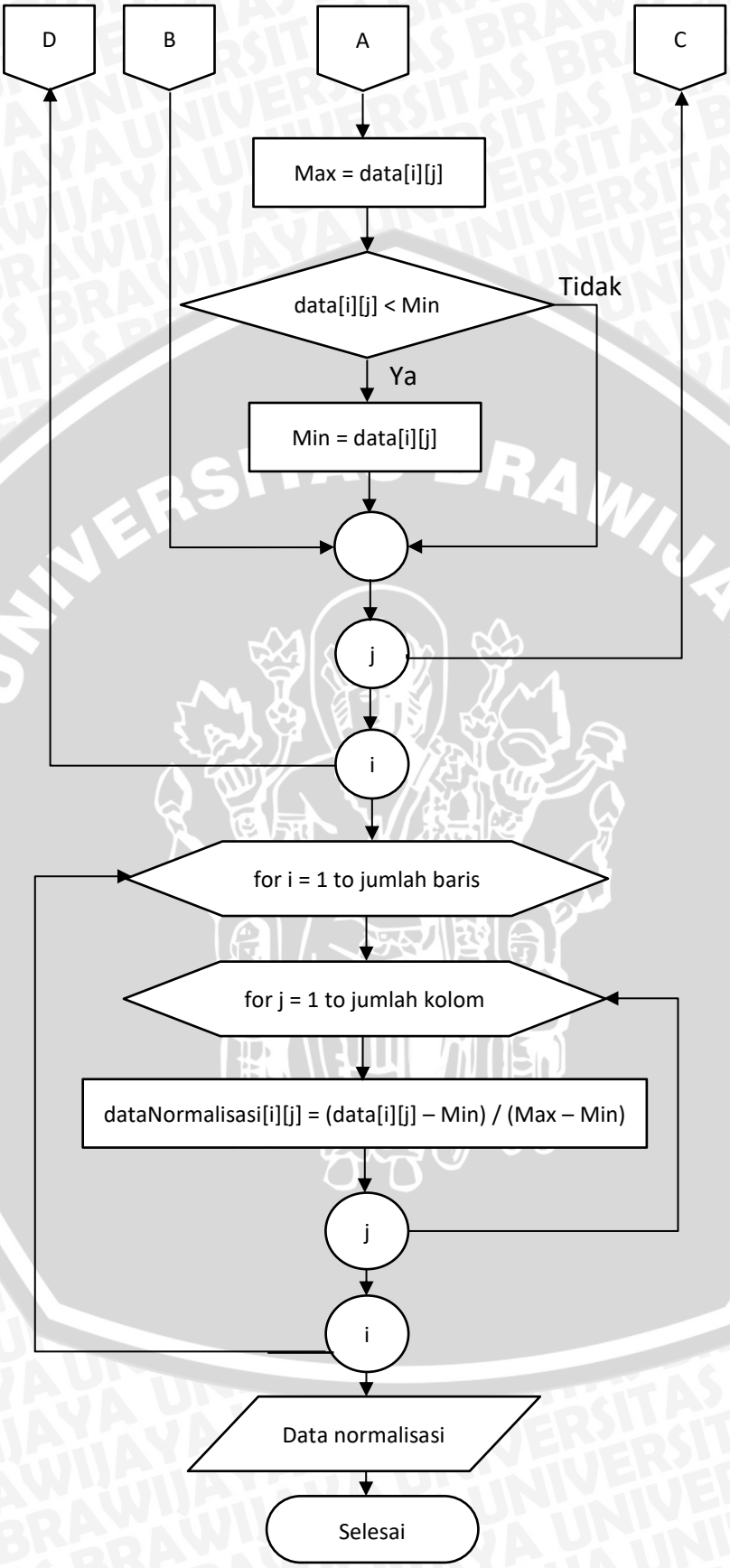
4.2 Siklus Metode *Extreme Learning Machine*(ELM)

Siklus metode ELM merupakan urutan penyelesaian masalah secara sekuensial. Pada siklus algoritma yang akan dijelaskan pada sub bab berikutnya, selain proses prediksi menggunakan metode ELM, telah ditambahkan pula proses normalisasi data menggunakan *Min-Max Normalization* dan proses menghitung *error* menggunakan metode *Mean Square Error*(MSE).

4.2.1 Proses Normalisasi Data

Normalisasi data merupakan proses transformasi data dimana nilai data diskalakan menjadi lebih kecil antara rentang 0 sampai dengan 1. Pada penelitian ini, proses normalisasi data menggunakan metode *Min-Max Normalization*. Diagram alir proses normalisasi data ditunjukkan pada Gambar 4.1 sebagai berikut:





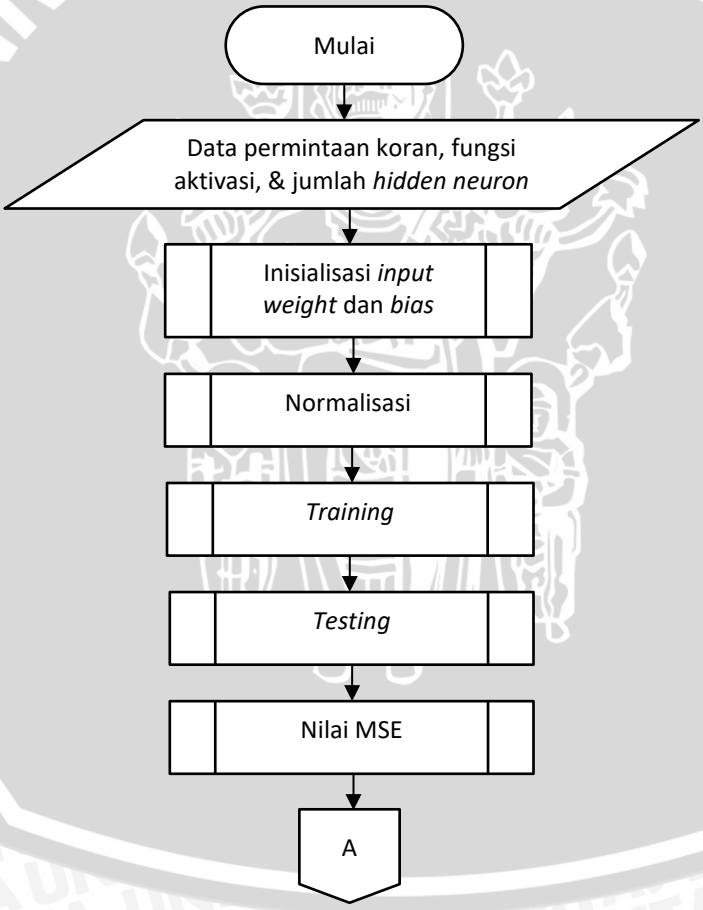
Gambar 4.1 Proses Normalisasi

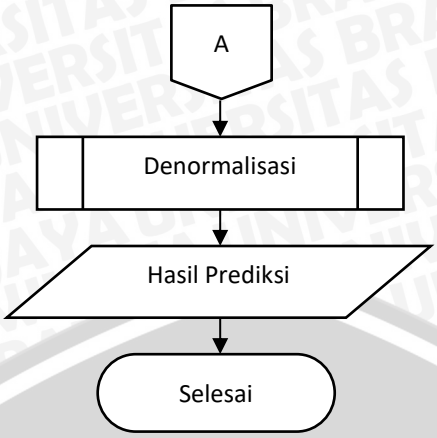
Langkah-langkah normalisasi data permintaan koran menggunakan metode *Min-Max Normalization* berdasarkan Gambar 4.1 adalah sebagai berikut:

1. Sistem menerima masukan berupa data permintaan koran
2. Mencari nilai maksimal(*Max*) dan minimal(*Min*) dari seluruh data
3. Menghitung nilai normalisasi menggunakan Persamaan 2.7 untuk setiap data
4. Keluaran sistem adalah data permintaan koran yang telah dinormalisasi

4.2.2 Proses Prediksi Jumlah Permintaan Koran Menggunakan Metode *Extreme Learning Machine* (ELM)

Prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine* bertujuan untuk memperoleh hasil prediksi yang mendekat dengan target dan memperoleh tingkat *error* yang paling kecil. Diagram alir proses prediksi jumlah permintaan koran menggunakan metode ELM ditunjukkan pada Gambar 4.2 sebagai berikut:

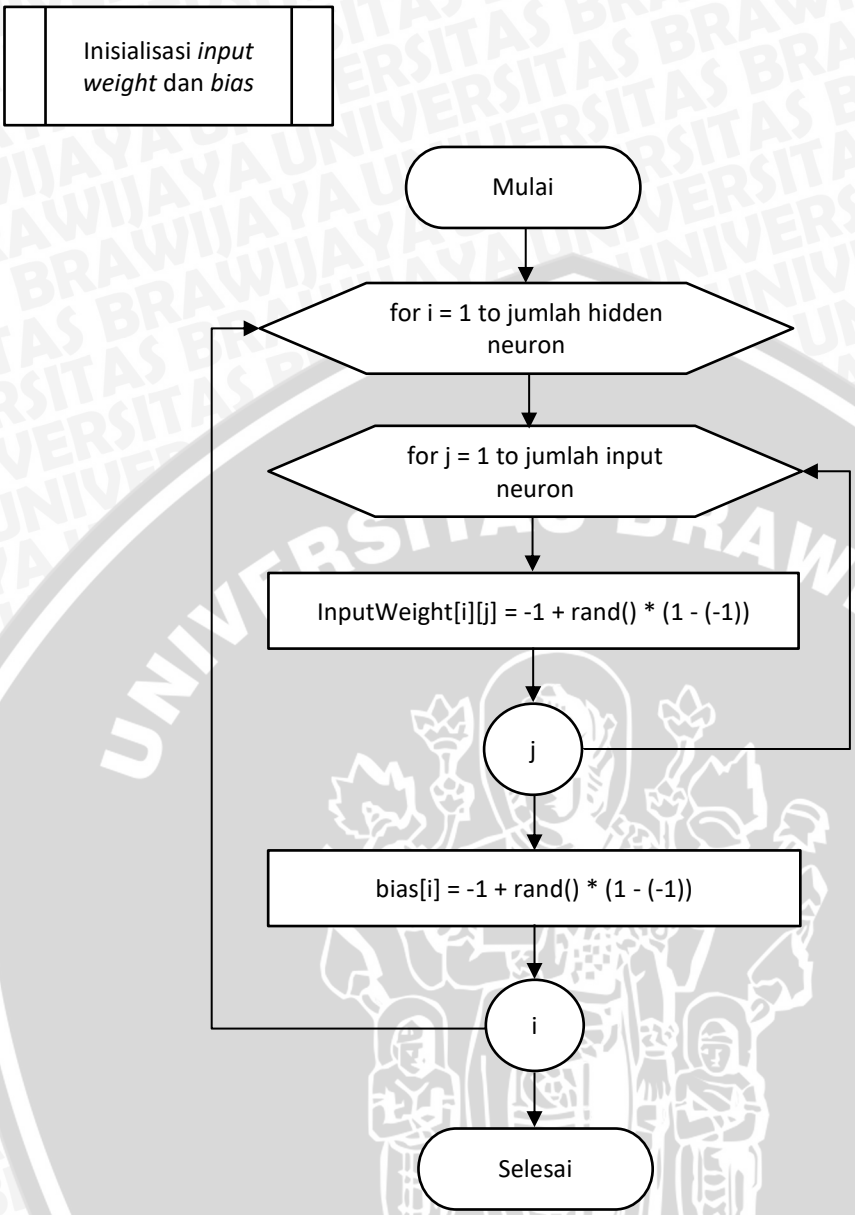




Gambar 4.2 Diagram Alir Proses Prediksi Menggunakan ELM

Langkah-langkah proses prediksi data permintaan koran menggunakan metode ELM berdasarkan Gambar 4.2 adalah sebagai berikut:

1. Sistem menerima masukan berupa data permintaan koran, jumlah *neuron* pada *hidden layer*, fungsi aktivasi yang digunakan dan perbandingan jumlah data *training* dan *testing*
2. Inisialisasi *input weight* dan *bias* ditunjukkan pada Gambar 4.3
3. Menghitung normalisasi data menggunakan Persamaan 2.7. Diagram alir untuk menghitung normalisasi data ditunjukkan pada Gambar 4.1
4. Melakukan proses *training* menggunakan data yang telah di normalisasi. Diagram alir untuk proses *training* ditunjukkan pada Gambar 4.4
5. Melakukan proses *testing* untuk proses prediksi. Diagram alir untuk proses *testing* ditunjukkan pada Gambar 4.10
6. Menghitung nilai MSE untuk mengevaluasi hasil prediksi proses *testing* menggunakan Persamaan 2.9. Diagram alir untuk proses menghitung nilai MSE ditunjukkan pada Gambar 4.14
7. Menghitung denormalisasi untuk hasil akhir proses *testing* menggunakan Persamaan 2.8. Diagram alir untuk proses denormalisasi ditunjukkan pada Gambar 4.13
8. Keluaran sistem adalah hasil prediksi setelah didenormalisasi



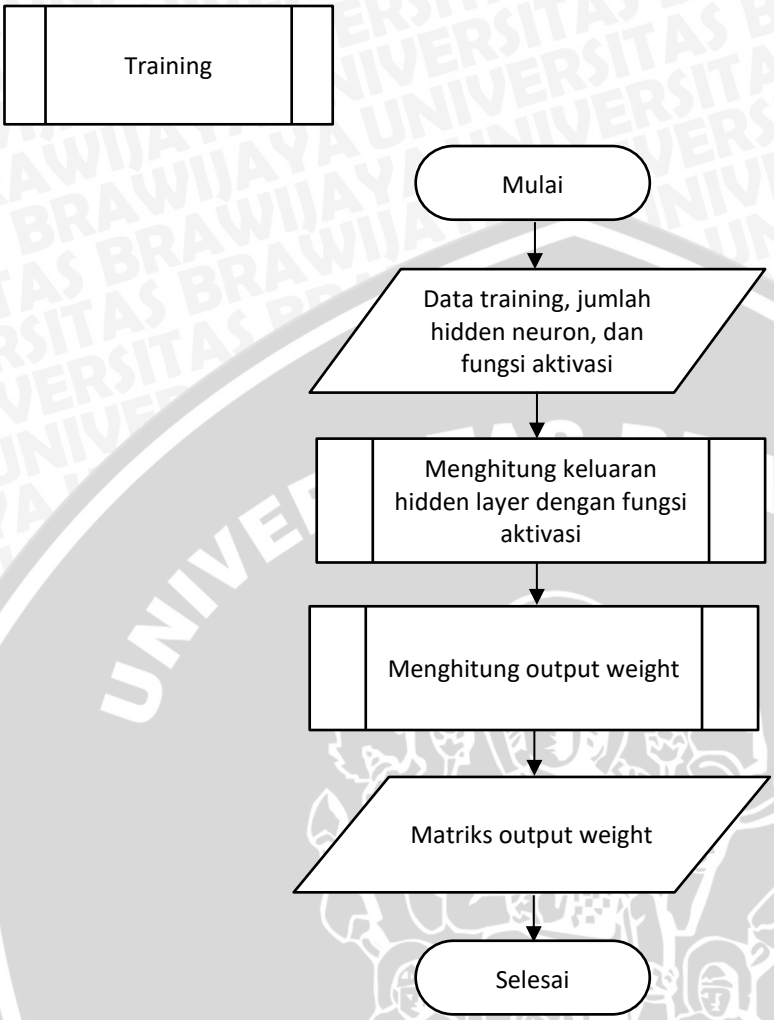
Gambar 4.3 Diagram Alir Proses Inisialisasi *Input Weight* dan *Bias*

Langkah-langkah proses inisialisasi *input weight* dan *bias* berdasarkan pada Gambar 4.3 adalah sebagai berikut:

1. Melakukan perulangan sebanyak jumlah *hidden neuron* dan jumlah *input neuron*
2. Menghitung *input weight* dan *bias* secara random dengan *range* antara -1 dengan 1

4.2.2.1 Proses Training

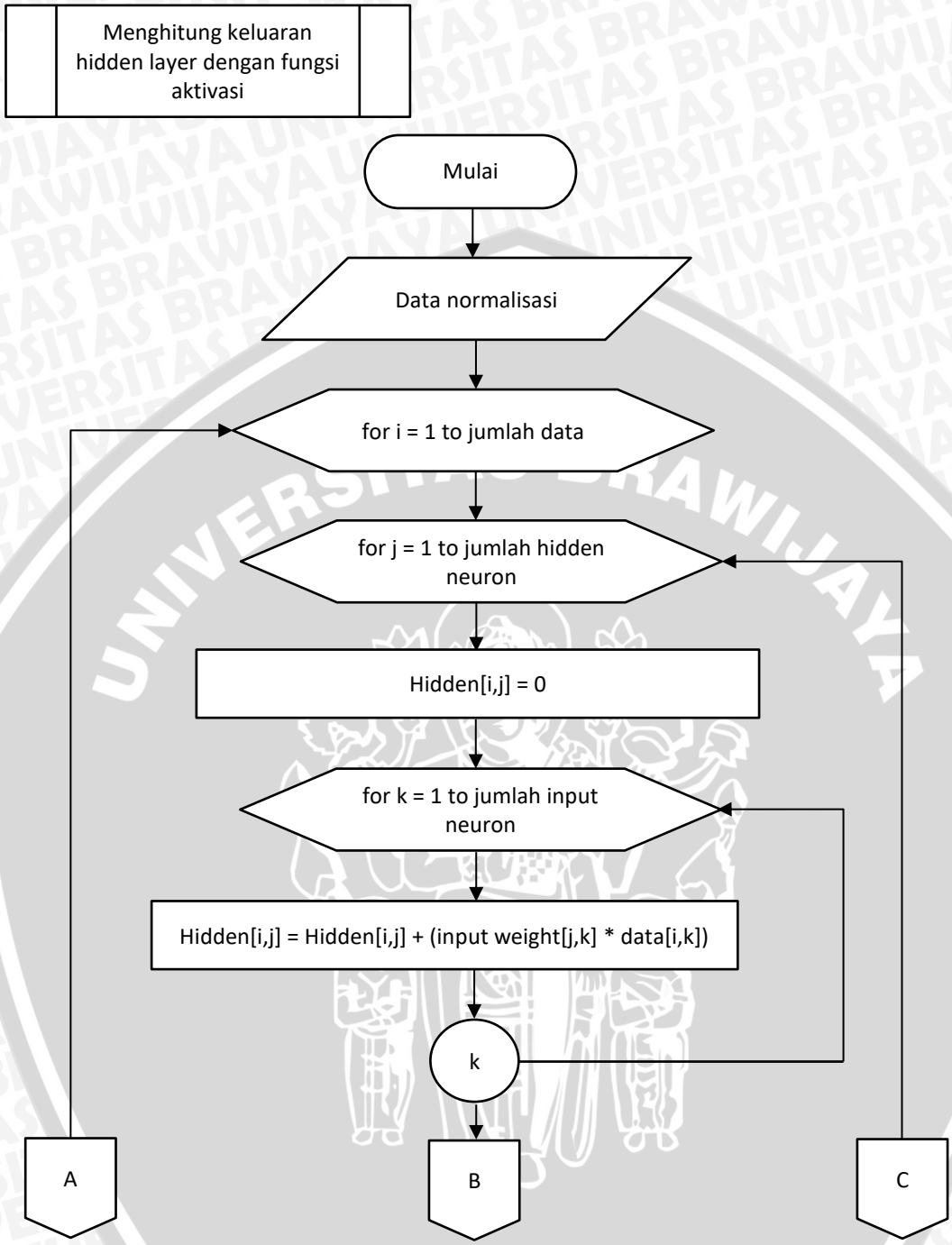
Proses *training* dilakukan untuk memperoleh *input weight*, *bias*, dan *output weight* dengan tingkat kesalahan yang rendah untuk digunakan pada proses *testing*. Diagram alir proses *training* dengan metode ELM ditunjukkan pada Gambar 4.4 sebagai berikut:

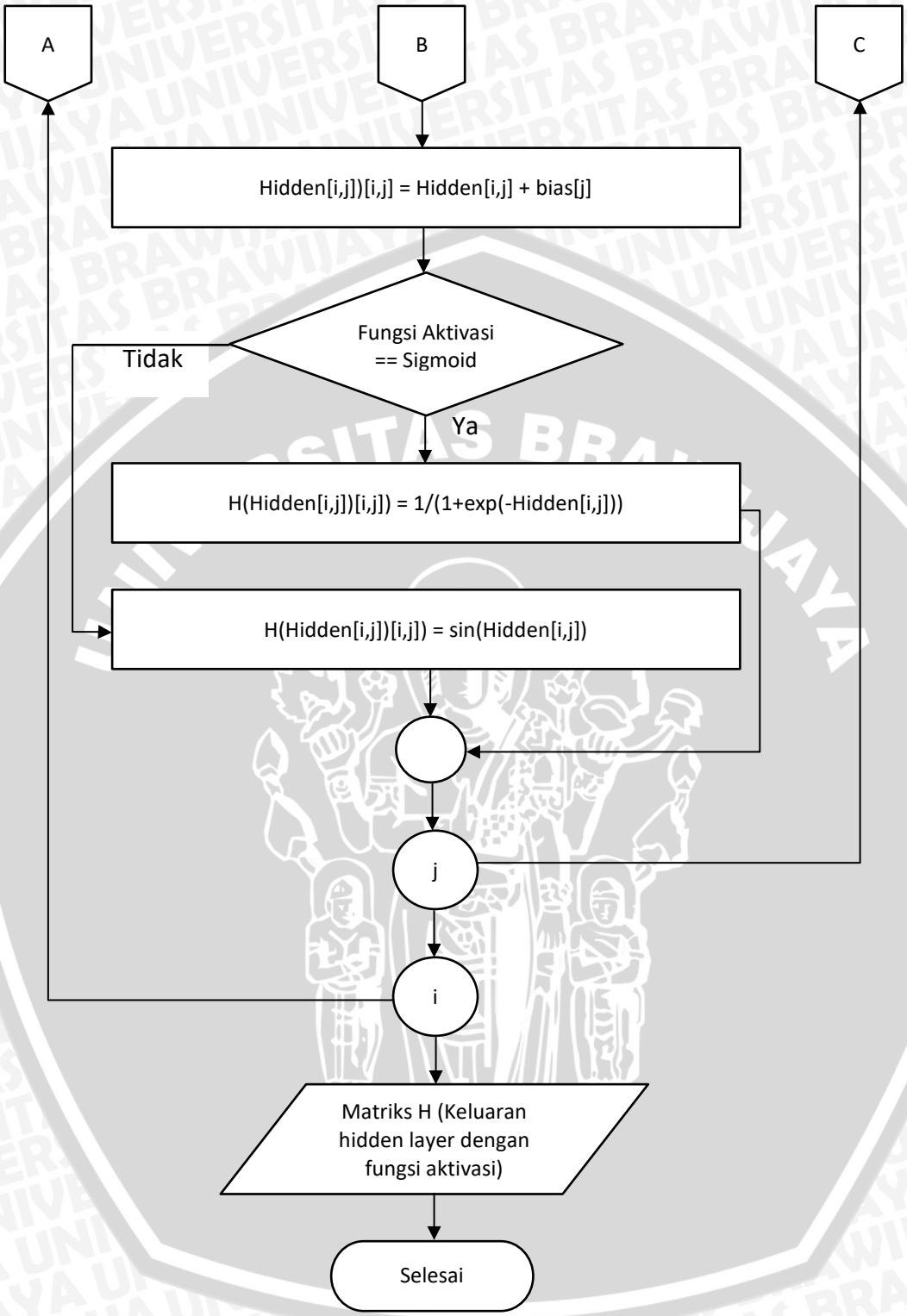


Gambar 4.4 Diagram Alir Proses *Training*

Langkah-langkah diagram alir proses *training* metode ELM berdasarkan Gambar 4.4 adalah sebagai berikut:

1. Sistem menerima masukan yaitu data *training*, jumlah *hidden neuron*, dan fungsi aktivasi
2. Menghitung keluaran *hidden layer* dengan fungsi aktivasi menggunakan Persamaan 2.4. Diagram alir untuk proses menghitung keluaran *hidden layer* dengan fungsi aktivasi ditunjukkan pada Gambar 4.5
3. Menghitung *output weight* menggunakan Persamaan 2.5. Diagram alir untuk proses menghitung *output weight* ditunjukkan pada Gambar 4.6
4. Keluaran sistem yaitu matriks *output weight*

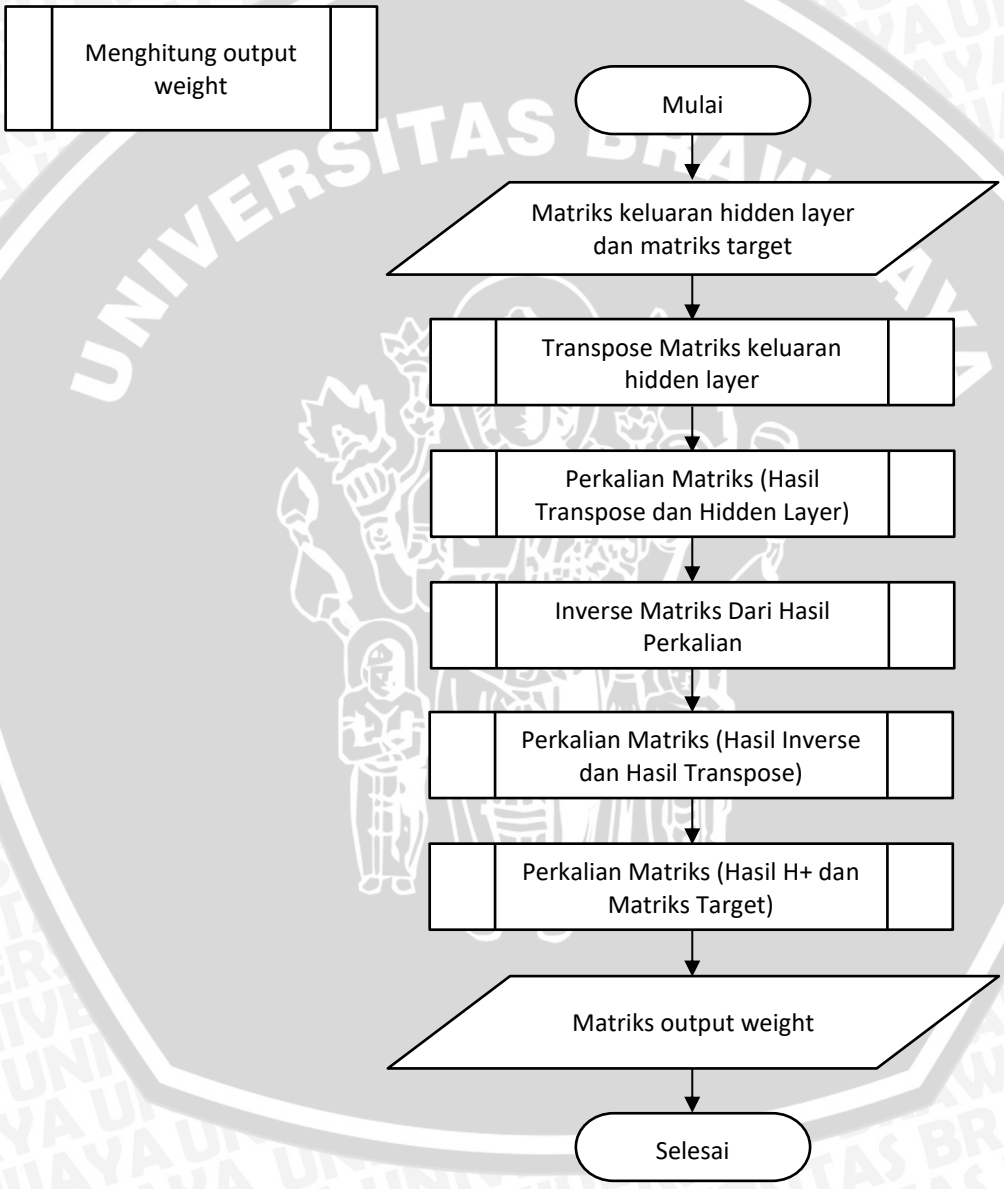

Menghitung keluaran hidden layer dengan fungsi aktivasi
Mulai
Data normalisasi
for i = 1 to jumlah data
for j = 1 to jumlah hidden neuron
Hidden[i,j] = 0
for k = 1 to jumlah input neuron
Hidden[i,j] = Hidden[i,j] + (input weight[j,k] * data[i,k])
k
A
B
C



Gambar 4.5 Diagram Alir Proses Menghitung Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

Langkah-langkah diagram alir proses menghitung keluaran *hidden layer* dengan fungsi aktivasi berdasarkan Gambar 4.5 adalah sebagai berikut:

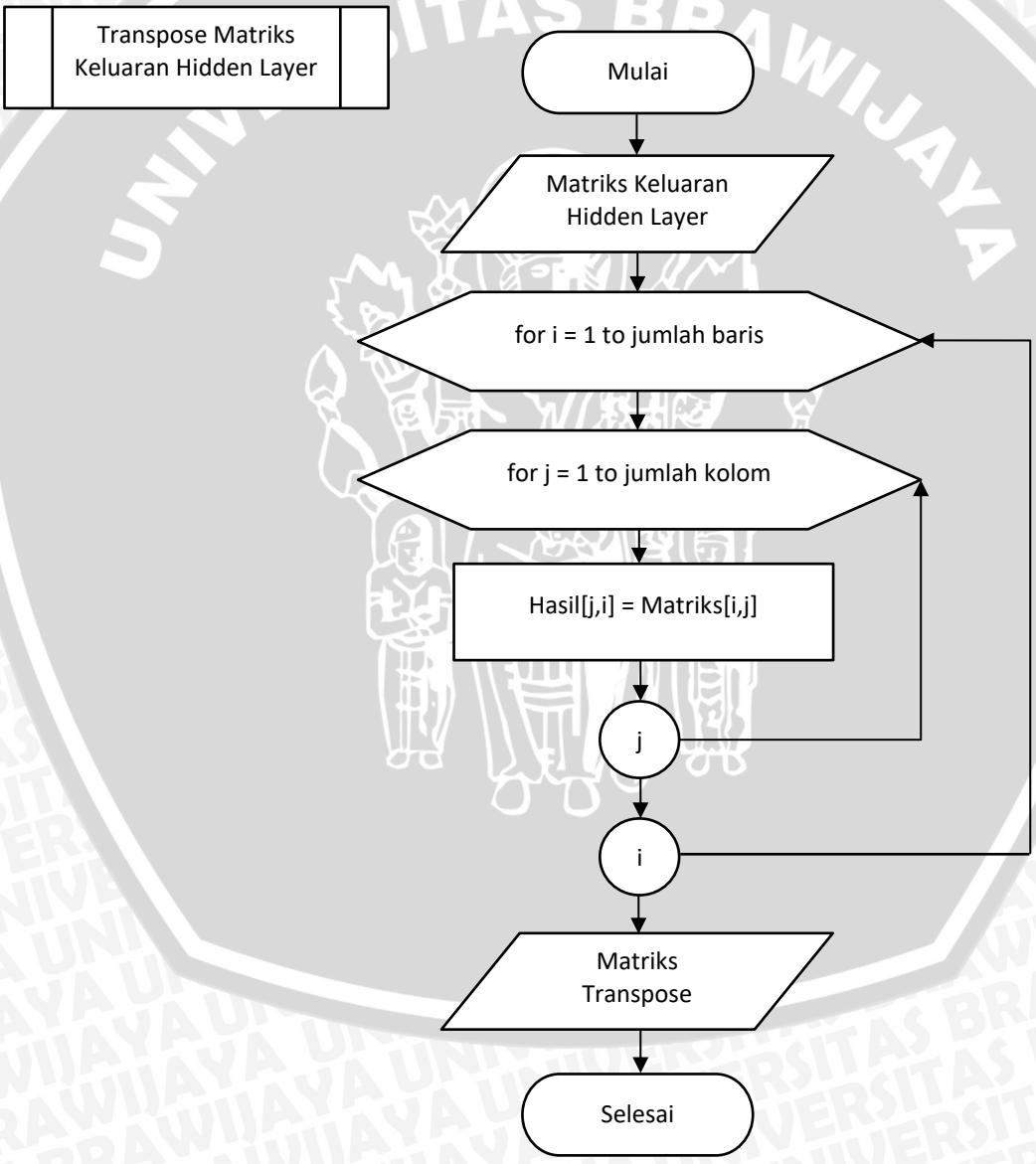
1. Sistem menerima masukan yaitu data yang telah dinormalisasi apabila pada proses *training* menggunakan data *training*, apabila pada proses *testing* menggunakan data *testing*. Sehingga jumlah data menyesuaikan pada saat apa proses ini dipanggil
2. Menghitung keluaran *hidden layer* menggunakan Persamaan 2.4
3. Menghitung keluaran *hidden layer* dengan fungsi aktivasi menggunakan Persamaan 2.1 atau Persamaan 2.2
4. Keluaran sistem yaitu matriks keluaran *hidden layer* dengan fungsi aktivasi



Gambar 4.6 Diagram Alir Menghitung *Output Weight*

Langkah-langkah diagram alir proses menghitung *output weight* berdasarkan Gambar 4.6 adalah sebagai berikut:

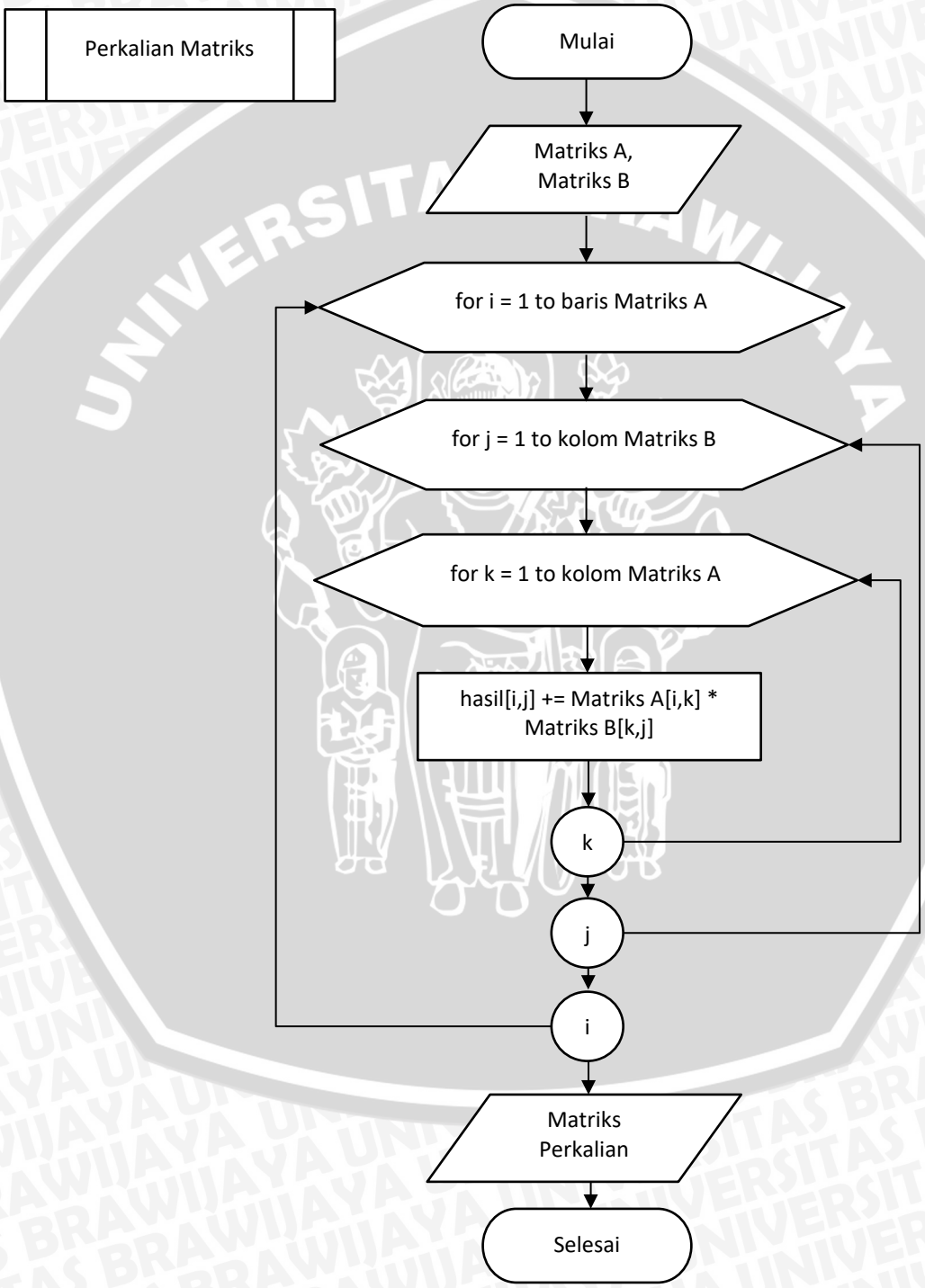
1. Sistem menerima masukan yaitu matriks keluaran *hidden layer* dan matriks target
2. Menghitung matriks *Moore-Penrose Generalized Invers* dari matriks keluaran *hidden layer* dengan melakukan *transpose* pada matriks keluaran *hidden layer*, kemudian melakukan perkalian matriks hasil *transpose* dengan matriks keluaran *hidden layer*. Hasil dari perkalian matriks tersebut dilakukan *invers*, kemudian melakukan perkalian matriks hasil *invers* dengan matriks hasil *transpose* menjadi matriks H^+. Terakhir melakukan perkalian matriks H^+ dengan matriks target untuk menghasilkan matriks *output weight*
3. Menghitung *output weight* menggunakan Persamaan 2.5
4. Keluaran sistem yaitu matriks *output weight*



Gambar 4.7 Diagram Alir *Transpose* Matriks Keluaran *Hidden Layer*

Langkah-langkah diagram alir proses *transpose* matriks keluaran *hidden layer* berdasarkan Gambar 4.7 adalah sebagai berikut:

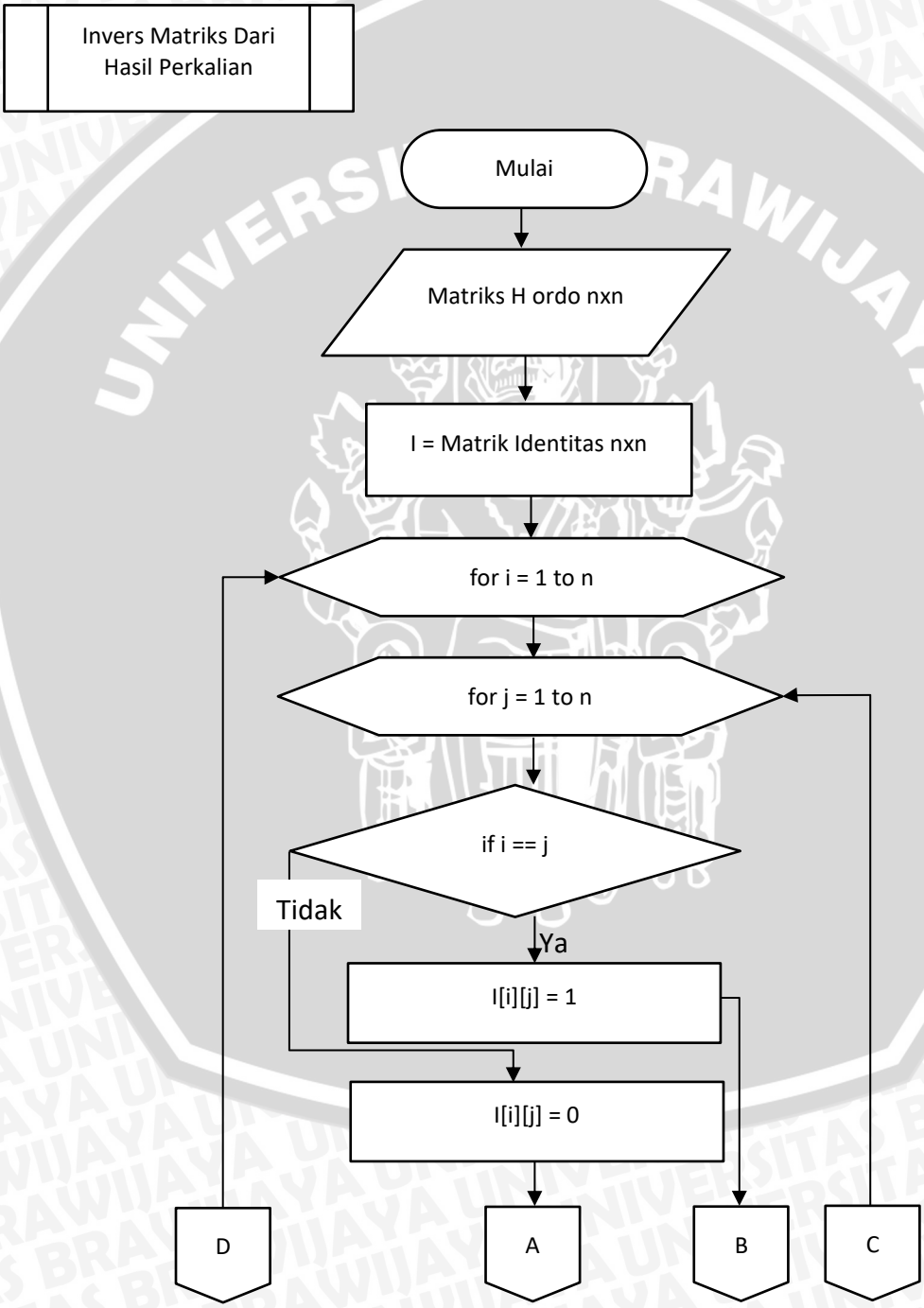
1. Sistem menerima masukan yaitu matriks keluaran *hidden layer*
2. Proses *transpose* matriks mengubah ordo matriks yang awalnya [baris, kolom] menjadi [kolom, baris]
3. Keluaran sistem yaitu matriks hasil *transpose*

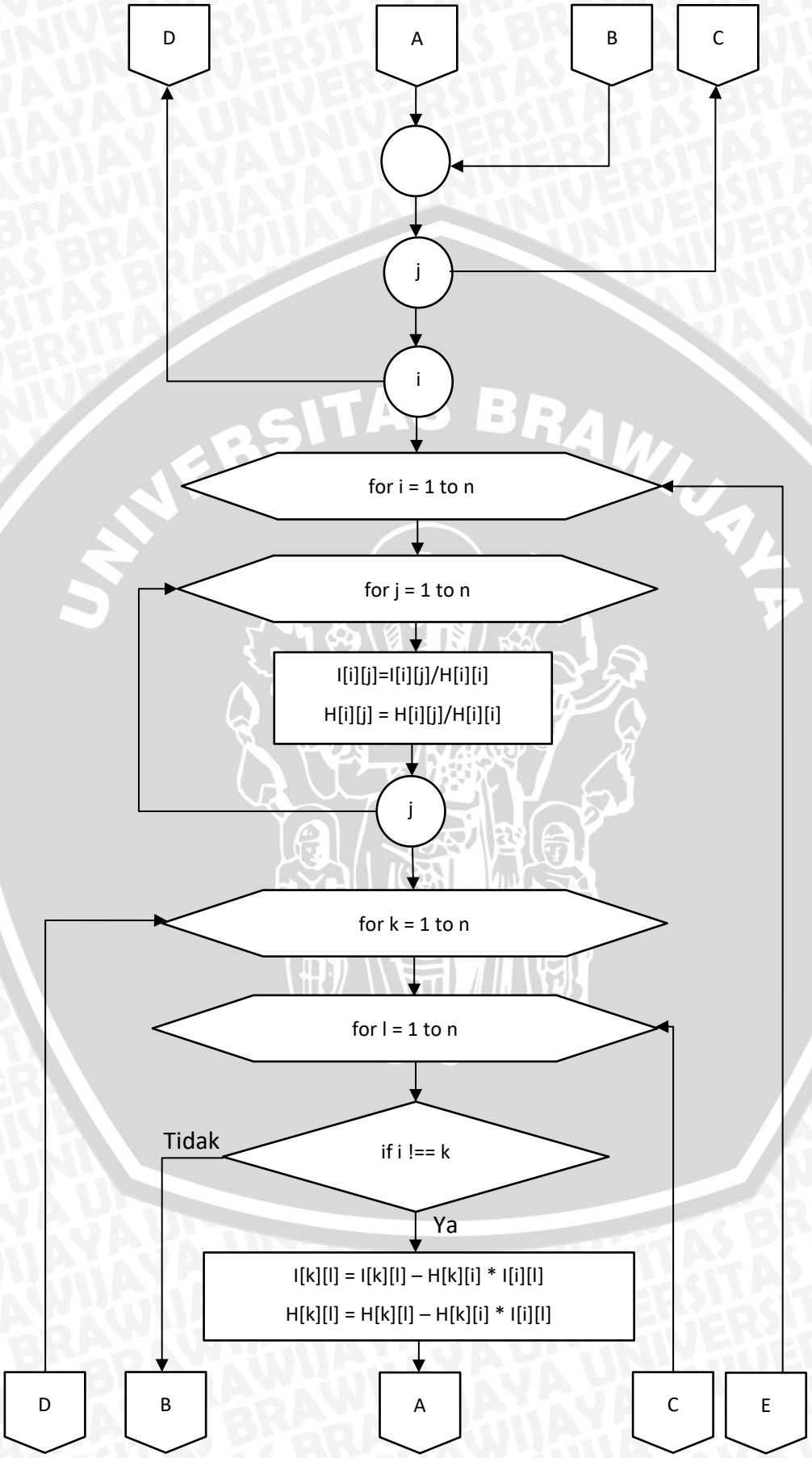


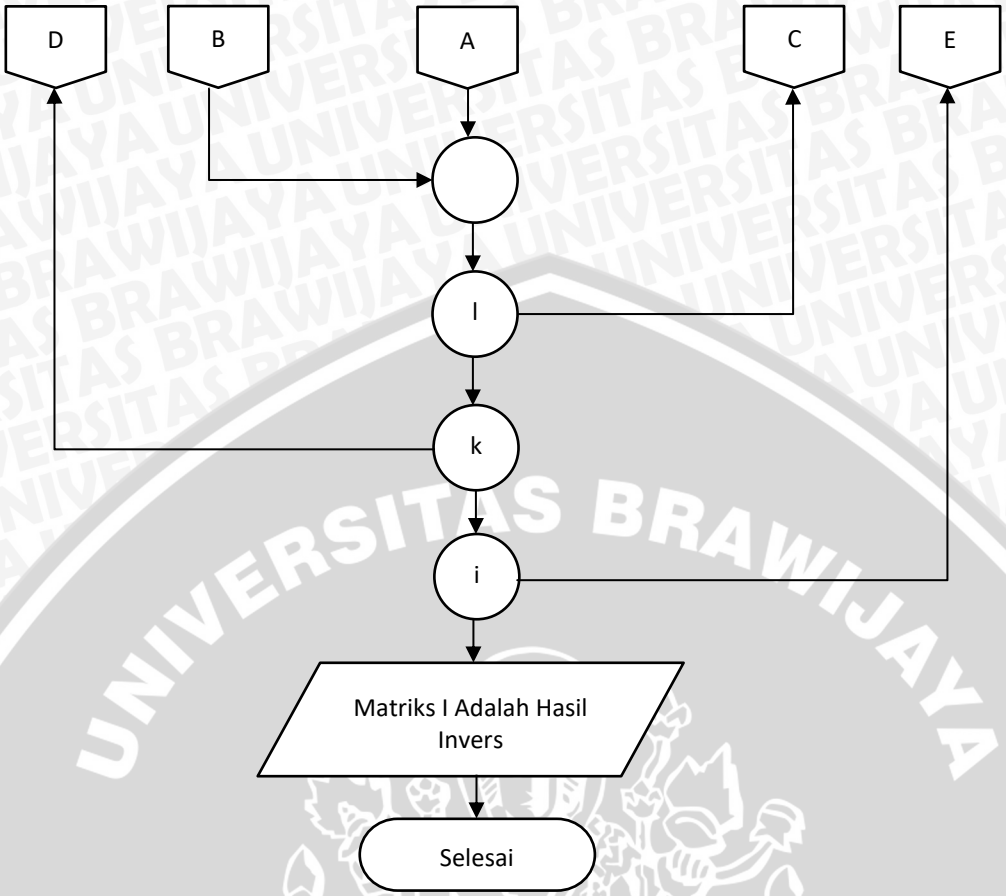
Gambar 4.8 Diagram Alir Perkalian Matriks

Langkah-langkah diagram alir proses perkalian matriks berdasarkan Gambar 4.8 adalah sebagai berikut:

1. Sistem menerima masukan yaitu 2 matriks yang akan dikalikan, missal: Matriks A dan Matriks B
2. Proses perkalian matriks dilakukan dengan cara mengalikan elemen kolom dari matriks A dengan elemen baris dari matriks B
3. Keluaran sistem yaitu matriks hasil perkalian




D
A
B
C
j
i
for i = 1 to n
for j = 1 to n
I[i][j]=I[i][j]/H[i][i]
H[i][j] = H[i][j]/H[i][i]
j
for k = 1 to n
for l = 1 to n
Tidak
if i !== k
Ya
I[k][l] = I[k][l] – H[k][i] * I[i][l]
H[k][l] = H[k][l] – H[k][i] * I[i][l]
D
B
A
C
E



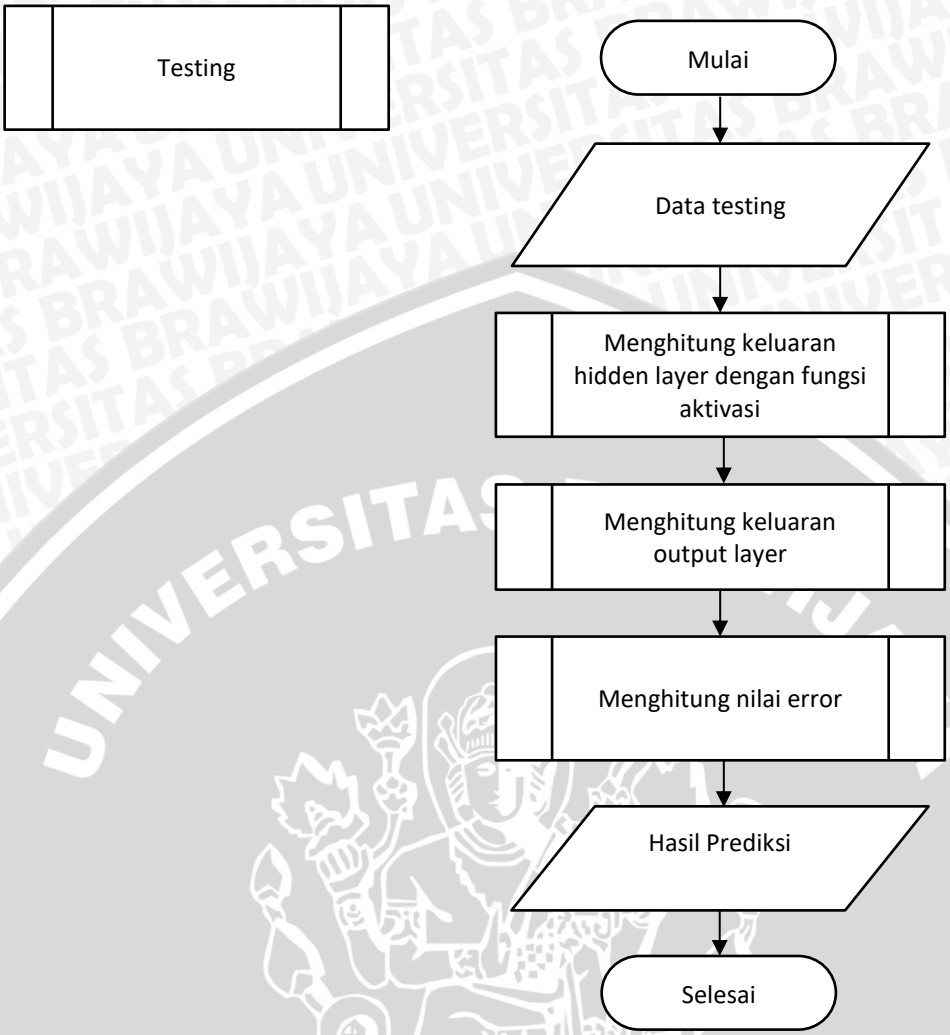
Gambar 4.9 Diagram Alir *Invers* Matriks Dari Hasil Perkalian

Langkah-langkah diagram alir proses *invers* matriks dari hasil perkalian berdasarkan Gambar 4.9 adalah sebagai berikut:

1. Sistem menerima masukan yaitu matriks hasil perkalian dari matriks *transpose* keluaran *hidden layer* dengan matriks keluaran *hidden layer*
2. Inisialisasi matriks identitas dengan ordo sama dengan matriks hasil perkalian. Pada perhitungan *invers,* matriks identitas digunakan dalam Operasi Baris Elementer (OBE)
3. Proses *invers* matriks dilakukan dengan cara mengubah matriks hasil perkalian (H) menjadi matriks identitas, sehingga matriks identitas (I) menjadi matriks baru
4. Keluaran sistem yaitu matriks identitas yang menjadi matriks baru ini lah yang merupakan *invers* dari matriks hasil perkalian

4.2.2.2 Proses Testing

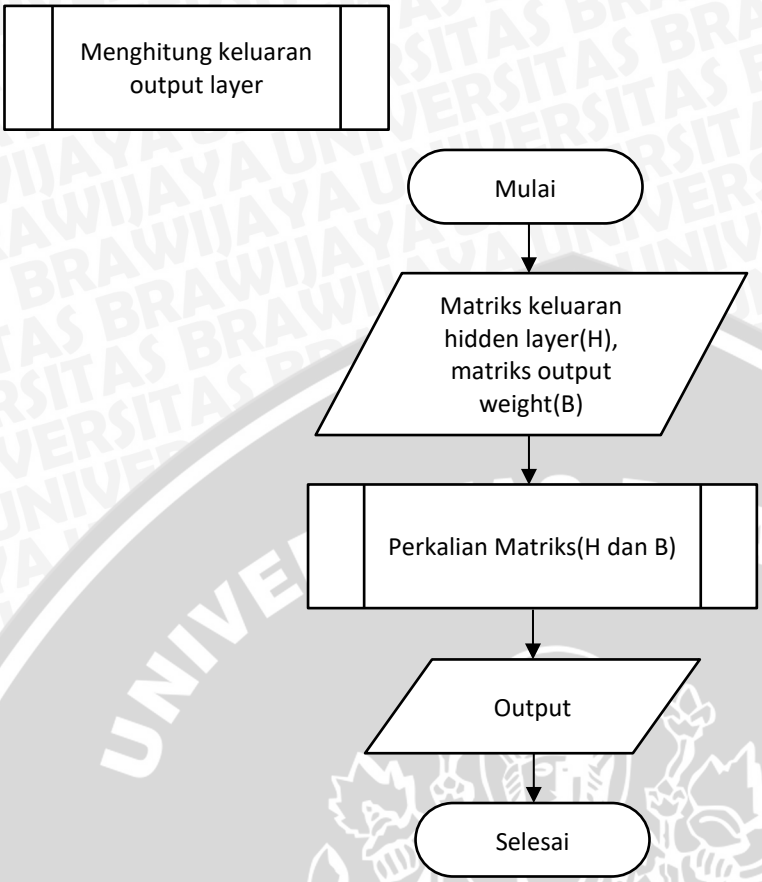
Proses *testing* dilakukan berdasarkan *input weight*, *bias*, dan *output weight* yang sesuai dari perhitungan *training*. Proses *testing* dilakukan sesuai urutan proses *training*. Diagram alir proses *testing* dengan metode ELM ditunjukkan pada Gambar 4.10 sebagai berikut:



Gambar 4.10 Diagram Alir Proses *Testing*

Langkah-langkah diagram alir proses *testing* metode ELM berdasarkan Gambar 4.10 adalah sebagai berikut:

1. Sistem menerima masukan yaitu data *testing*. Menggunakan nilai *input weight*, *bias*, serta *output weight* dari proses *training*
2. Menghitung keluaran *hidden layer* dengan fungsi aktivasi menggunakan Persamaan 2.4. Diagram alir untuk proses menghitung keluaran *hidden layer* dengan fungsi aktivasi ditunjukkan pada Gambar 4.5
3. Menghitung keluaran *output layer* menggunakan Persamaan 2.6. Diagram alir untuk proses menghitung keluaran *output layer* ditunjukkan pada Gambar 4.11
4. Menghitung nilai *error* menggunakan Persamaan 2.7. Diagram alir proses menghitung nilai *error* ditunjukkan pada Gambar 4.12
5. Keluaran sistem yaitu hasil prediksi dan nilai *error* dari *output layer* dengan target



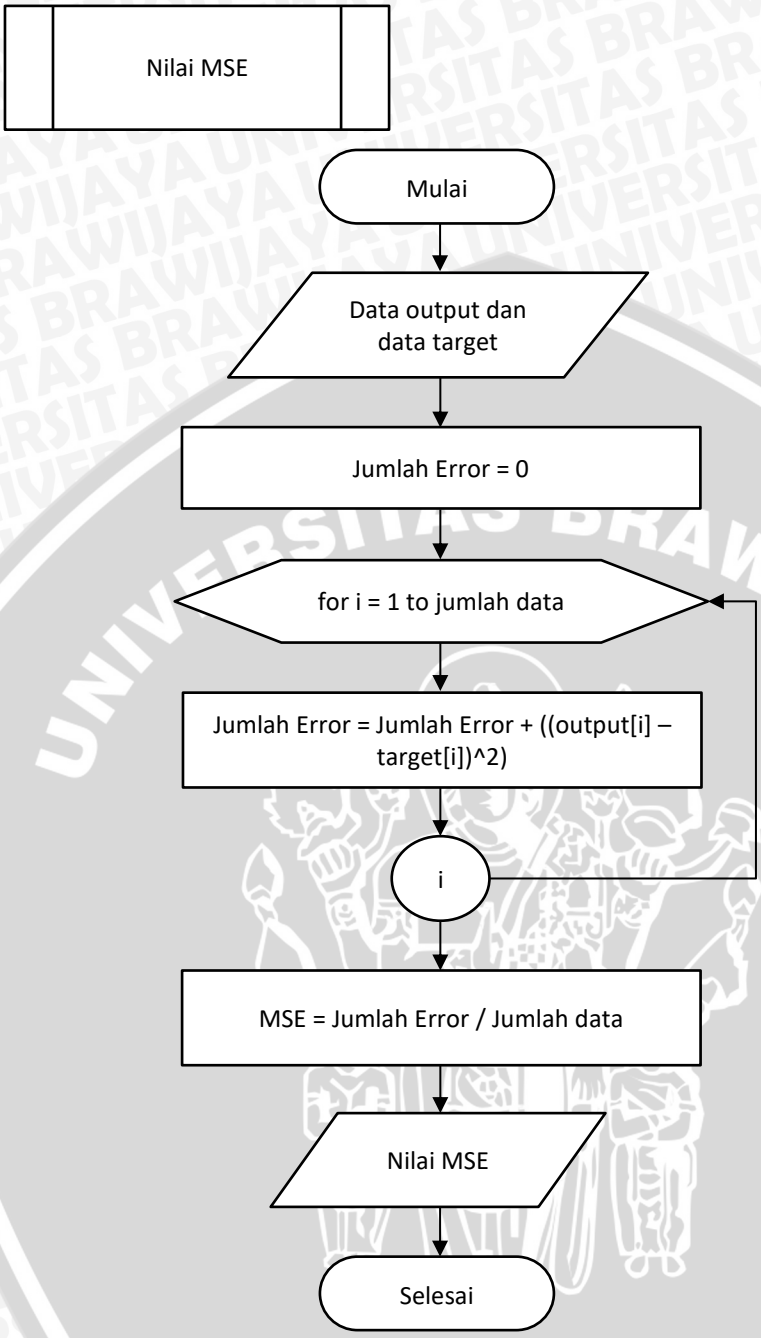
Gambar 4.11 Diagram Alir Menghitung Keluaran *Output Layer*

Langkah-langkah diagram alir proses menghitung keluaran di *output layer* berdasarkan Gambar 4.11 adalah sebagai berikut:

1. Sistem menerima masukan yaitu matriks keluaran di *hidden layer* dan matriks *output weight* dari proses *training*
2. Menghitung keluaran di *output layer* dengan menggunakan Persamaan 2.6 yaitu dengan perkalian matriks keluaran *hidden layer* dengan matriks *output weight* dari proses *training*
3. Keluaran sistem yaitu keluaran di *output layer* yang merupakan output hasil proses *testing*

4.2.3 Proses Hitung Nilai *Mean Square Error* (MSE)

Menghitung *error rate* menggunakan *Mean Square Error* (MSE) bertujuan untuk mengevaluasi metode prediksi yang digunakan yaitu ELM. Diagram alir proses menghitung *error rate* menggunakan MSE ditunjukkan pada Gambar 4.12 sebagai berikut:



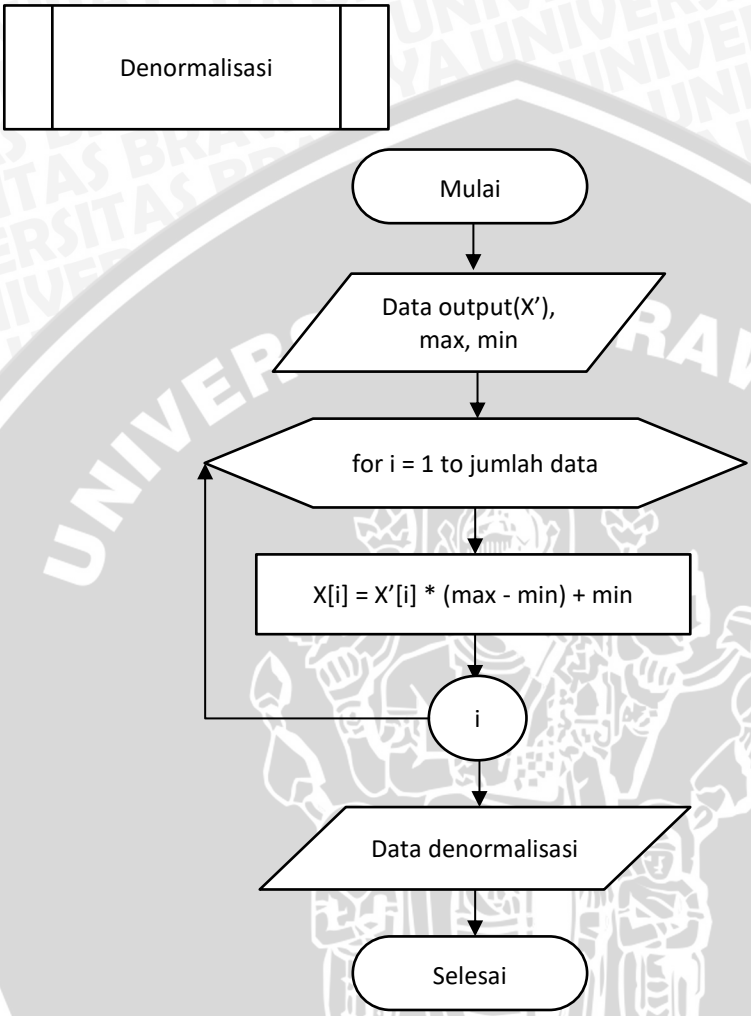
Gambar 4.12 Diagram Alir Proses Hitung Tingkat *Error* Menggunakan MSE

Langkah-langkah proses menghitung tingkat *error* menggunakan MSE berdasarkan Gambar 4.12 adalah sebagai berikut:

1. Sistem menerima masukan berupa data *output*, data target, dan jumlah data
2. Menghitung hasil MSE akhir dengan menggunakan Persamaan 2.9
3. Keluaran sistem berupa nilai MSE yang menunjukkan tingkat *error* dari hasil perhitungan ELM

4.2.4 Proses Denormalisasi

Denormalisasi data digunakan untuk mendapatkan nilai sebenarnya dari *output* berdasarkan hasil *output* prediksi. Diagram alir proses denormalisasi data ditunjukkan pada Gambar 4.13 sebagai berikut.



Gambar 4.13 Diagram Alir Proses Denormalisasi Data

Langkah-langkah diagram alir proses denormalisasi data berdasarkan Gambar 4.13 adalah sebagai berikut:

1. Sistem menerima masukan yaitu matriks keluaran di *output layer*, nilai maksimal, dan nilai minimal
2. Menghitung nilai denormalisasi dengan menggunakan Persamaan 2.8 untuk setiap data
3. Keluaran sistem yaitu data yang telah di denormalisasi

4.3 Perhitungan Manual

Perhitungan manual pada penelitian ini menggunakan sampel data sebanyak 10 *record* dengan perbandingan data *training* dan data *testing* yaitu 80%:20%

sehingga didapat 8 data *training* dan 2 data *testing*. Data yang digunakan ditunjukkan pada tabel 4.2 berikut ini:

Tabel 4.2 Data *Training* dan Data *Testing*

Data ke-	X1	X2	X3	X4	T
1	7899	7961	7560	8166	7899
2	7961	7560	8166	7899	7911
3	7560	8166	7899	7911	7987
4	8166	7899	7911	7987	7947
5	7899	7911	7987	7947	7836
6	7911	7987	7947	7836	8206
7	7987	7947	7836	8206	8052
8	7947	7836	8206	8052	8072
9	7836	8206	8052	8072	7906
10	8206	8052	8072	7906	7990

Pada contoh perhitungan manual ini menggunakan fungsi aktivasi *sigmoid* dan jumlah *neuron* pada *hidden layer* sebanyak 2. Selanjutnya akan dibahas perhitungan manual menggunakan metode ELM.

4.3.1 Perhitungan Normalisasi Data

Langkah-langkah normalisasi data menggunakan *Min-Max Normalization* dihitung berdasarkan Persamaan 2.7 adalah sebagai berikut:

Langkah 1: menghitung nilai maksimal dan minimal dari semua data. Nilai maksimal dan minimal ditunjukkan pada Tabel 4.3 sebagai berikut:

Tabel 4.3 Nilai Maksimal dan Minimal

Max	8206
Min	7560

Langkah 2: menghitung nilai normalisasi data menggunakan *Min-Max Normalization*. Berikut contoh perhitungan nilai normalisasi:

$$x'_{1,1} = \frac{x_{1,1} - \min}{\max - \min} = \frac{7899 - 7560}{8206 - 7560} = 0.524768$$

Hasil perhitungan normalisasi secara keseluruhan ditunjukkan pada Tabel 4.4 sebagai berikut:

Tabel 4.4 Normalisasi Data

Data ke-	X1	X2	X3	X4	T
1	0.524768	0.620743	0	0.93808	0.524768
2	0.620743	0	0.93808	0.524768	0.543344

Data ke-	X1	X2	X3	X4	T
3	0	0.93808	0.524768	0.543344	0.660991
4	0.93808	0.524768	0.543344	0.660991	0.599071
5	0.524768	0.543344	0.660991	0.599071	0.427245
6	0.543344	0.660991	0.599071	0.427245	1
7	0.660991	0.599071	0.427245	1	0.76161
8	0.599071	0.427245	1	0.76161	0.79257
9	0.427245	1	0.76161	0.79257	0.535604
10	1	0.76161	0.79257	0.535604	0.665635

4.3.2 Inisialisasi *Input Weight* dan *Bias*

Jumlah *neuron* pada *hidden layer* yang digunakan dalam perhitungan manual ini sebanyak 2. Untuk menentukan *input weight* dan *bias* didapatkan secara random dalam bentuk matriks dengan ordo sesuai dengan banyaknya *input neuron* dan *hidden neuron*. Dalam perhitungan manual ini, jumlah *input neuron* sebanyak 4 sehingga *input weight* memiliki ordo 2x4. Nilai *input weight* dan *bias* memiliki *range* [-1, 1]. Matriks *input weight* ditunjukkan pada Tabel 4.5.

Tabel 4.5 Matriks Nilai *Input Weight*

w	1	2	3	4
1	0.970588	-0.41618	-0.77892	-0.20598
2	-0.2936	0.274177	0.770641	0.717264

Sedangkan jumlah *bias* sama dengan jumlah *neuron* pada *hidden layer* yaitu sebanyak 2. Matriks *bias* ditunjukkan pada Tabel 4.6.

Tabel 4.6 Matriks Nilai *Bias*

1	2
0.300383	-0.81216

4.3.3 Perhitungan Proses *Training*

Pada proses *training* data yang digunakan sebanyak 8, jumlah *neuron* pada *hidden layer* yaitu 2, dan menggunakan fungsi aktivasi *sigmoid*. Langkah-langkah perhitungan manual proses *training* metode ELM adalah sebagai berikut:

Langkah 1: Menghitung keluaran di *hidden layer* dengan fungsi aktivasi. Perhitungan keluaran *hidden layer* ditunjukkan pada Persamaan 2.4. Kemudian setelah dilakukan perhitungan dilanjutkan dengan menghitung fungsi aktivasi. Fungsi aktivasi yang digunakan adalah fungsi aktivasi *sigmoid*. Berikut ini contoh perhitungan keluaran *hidden layer*:

$$H_{init\ 1,1} = \left(\sum_{k=1}^{4} w_{1,k} . x_{1,k} \right) + b_1$$

$$H_{init\ 1,1} = ((0.970588 * 0.524768) + (-0.41618 * 0.620743) + (-0.77892 * 0) + (-0.20598 * 0.93808)) + 0.300383 = 0.358151$$

Hasil dari keluaran *hidden layer* secara keseluruhan ditunjukkan pada Tabel 4.7 sebagai berikut:

Tabel 4.7 Matriks Keluaran *Hidden Layer*

H_{init}	1	2
1	0.358151	-0.12319
2	0.06409	0.104911
3	-0.6107	0.239172
4	0.433104	-0.05087
5	-0.05467	0.121821
6	-0.00198	-0.02234
7	0.153844	0.204542
8	-0.23177	0.446011

Berikut ini contoh perhitungan keluaran *hidden layer* dengan fungsi aktivasi.

$$H(x)_{1,1} = \frac{1}{1+e^{-x}} = \frac{1}{1+e^{-0.358151}} = 0.588593$$

Hasil keluaran *hidden layer* dengan fungsi aktivasi secara keseluruhan ditunjukkan pada Tabel 4.8 sebagai berikut:

Tabel 4.8 Matriks Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

$H(x)$	1	2
1	0.588593	0.469242
2	0.516017	0.526204
3	0.3519	0.559509
4	0.606615	0.487284
5	0.486337	0.530418
6	0.499506	0.494415
7	0.538385	0.550958
8	0.442315	0.60969

Langkah 2: Menghitung *output weight* dari *hidden layer* ke *output layer*. Untuk menghitung *output weight*, pertama harus menghitung Matriks *Moore-Penrose Generalized Invers* dari hasil keluaran *hidden layer* dengan fungsi aktivasi. Setelah itu baru dihitung *output weight* seperti yang ditunjukkan pada Persamaan 2.5.

Langkah 2.1: Menghitung matriks *Moore-Penrose Generalized Invers* dari hasil keluaran *hidden layer* dengan fungsi aktivasi yaitu $H^{+} = (H^{T}H)^{-1}H^{T}$. Pertama mencari matriks *transpose* dari Matriks H. Hasil *transpose* ditunjukkan pada Tabel 4.9.

Tabel 4.9 *Transpose* Matriks Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

$H(x)$	1	2	3	...	7	8
1	0.588593	0.516017	0.3519	...	0.538385	0.442315
2	0.469242	0.526204	0.559509	...	0.550958	0.60969

Kemudian menghitung perkalian matriks dari hasil *transpose* dengan keluaran *hidden layer* dengan fungsi aktivasi. Berikut ini contoh perhitungan perkalian matriks.

$$(H^{T}H)_{1,1} = (0.588593 * 0.588593) + (0.516017 * 0.516017) + (0.3519 * 0.3159) +$$

$$... + (0.442315 * 0.442315) = 2.076061$$

Hasil perkalian matriks secara keseluruhan ditunjukkan pada Tabel 4.10 sebagai berikut.

Tabel 4.10 Perkalian Matriks Hasil *Transpose* Dengan Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

$H^{T}H$	1	2
1	2.076061	2.111436
2	2.111436	2.248642

Kemudian menghitung *Invers* matriks hasil perhitungan $H^{T}H$. Berikut ini contoh perhitungan *invers* matriks menggunakan perhitungan Operasi Baris Elementari (OBE).

1. Bentuk matriks identitas (I), Baris 1($R1$), dan Baris 2($R2$)

$$[H \mid I] = \begin{bmatrix} 2.076061 & 2.111436 \\ 2.111436 & 2.248642 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

2. $R1 : 2.076061 \rightarrow R1$

$$[H \mid I] = \begin{bmatrix} 1 & 1.017039 \\ 2.111436 & 2.248642 \end{bmatrix} \begin{bmatrix} 0.481681 & 0 \\ 0 & 1 \end{bmatrix}$$

3. $R2 - 2.111436 * RI \rightarrow R2$

$$[H \mid I] = \begin{bmatrix} 1 & 1.017039 \\ 0 & 0.101228 \end{bmatrix} \begin{bmatrix} 0.481681 & 0 \\ -1.01704 & 1 \end{bmatrix}$$

4. $R2 : 0.101228 \rightarrow R2$

$$[H \mid I] = \begin{bmatrix} 1 & 1.017039 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0.481681 & 0 \\ -10.047 & 9.87866 \end{bmatrix}$$

5. $R1 - 1.017039 * R2 \rightarrow R1$

$$[H \mid I] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 10.69986 & -10.074 \\ -10.047 & 9.87866 \end{bmatrix}$$

Hasil *invers* matriks secara keseluruhan ditunjukkan pada Tabel 4.11 sebagai berikut.

Tabel 4.11 *Invers* Matriks

$(H^T H)^{-1}$	1	2
1	10.69986	-10.047
2	-10.047	9.87866

Terakhir menghitung matriks *Moore-Penrose Generalized Invers* (H^+) dengan cara perkalian matriks antara *invers* matriks dengan *transpose* keluaran *hidden layer*. Berikut ini contoh perhitungan perkalian matriks antara *invers* matriks dengan *transpose* keluaran *hidden layer*.

$$H^+ = (H^T H)^{-1} H^T$$

$$H^+_{1,1} = (10.69986 * 0.588593) + (-10.047 * 0.469242) = 1.58339$$

Hasil matriks *Moore-Penrose Generalized Invers* secara keseluruhan ditunjukkan pada Tabel 4.12 sebagai berikut.

Tabel 4.12 Matriks *Moore-Penrose Generalized Invers*

H^+	1	2	3	...	7	8
1	1.58339	0.234549	-1.8561	...	0.225182	-1.39284
2	-1.2781	0.013772	1.991667	...	0.033576	1.578987

Langkah 2.2: Menghitung *output weight* dengan cara perkalian matriks *Moore-Penrose Generalized Invers* dengan matriks target. Berikut ini contoh perhitungan perkalian matriks *Moore-Penrose Generalized Invers* dengan matriks target.

$$\beta = H^+ T$$

$$\beta_1 = (1.58339 * 0.52322) + (0.234549 * 0.543344) + (-1.8561 * 0.660991) + \ldots +$$
$$(-1.39284 * 0.79257) = 0.075807$$

Hasil akhir *output weight* secara keseluruhan ditunjukkan pada Tabel 4.13 sebagai berikut.

Tabel 4.13 Nilai *Output Weight*

β
0.075807
1.181595

4.3.4 Perhitungan Proses *Testing*

Perhitungan proses *testing* data yang digunakan sebanyak 2, jumlah *neuron* pada *hidden layer* yaitu 2, dan menggunakan fungsi aktivasi *sigmoid*. Langkah-langkah perhitungan manual proses *testing* metode ELM adalah sebagai berikut:

Langkah 1: Menghitung keluaran di *hidden layer* dengan fungsi aktivasi. Perhitungan keluaran *hidden layer* ditunjukkan pada Persamaan 2.4. Kemudian setelah dilakukan perhitungan dilanjutkan dengan menghitung fungsi aktivasi. Berikut ini contoh perhitungan keluaran *hidden layer*.

$$H_{init\ 1,1} = \left(\sum_{k=1}^{4} w_{1,k} . x_{1,k} \right) + b_1$$

$$H_{init\ 1,1} = ((0.970588 * 0.427245) + (-0.41618 * 1) + (-0.77892 * 0.76161) + (-0.20598 * 0.79257)) + 0.300383 = -0.4576$$

Hasil dari keluaran *hidden layer* secara keseluruhan ditunjukkan pada Tabel 4.14 sebagai berikut.

Tabel 4.14 Matriks Keluaran *Hidden Layer*

H_{init}	1	2
1	-0.4576	0.491989
2	0.226333	0.098012

Berikut ini contoh perhitungan dan hasil keluaran *hidden layer* dengan fungsi aktivasi.

$$H(x)_{1,1} = \frac{1}{1+e^{-x}} = \frac{1}{1+e^{-0.4576}} = 0.387554$$

Hasil keluaran *hidden layer* dengan fungsi aktivasi secara keseluruhan ditunjukkan pada Tabel 4.15 sebagai berikut.

Tabel 4.15 Matriks Keluaran *Hidden Layer* Dengan Fungsi Aktivasi

$H(x)$	1	2
1	0.387554	0.620575
2	0.556343	0.524483

Langkah 2: Menghitung keluaran di *output layer*. Perhitungan keluaran di *output layer* ditunjukkan pada Persamaan 2.6 dengan menggunakan *output weight* yang

telah dihitung pada proses *training*. Berikut ini contoh perhitungan keluaran *output layer*.

$$y = H\beta$$

$$y_1 = (0.387554 * 0.075807) + (0.524483 * 1.181595) = 0.762647$$

Hasil keluaran *output layer* secara keseluruhan ditunjukkan pada Tabel 4.16 sebagai berikut.

Tabel 4.16 Hasil Keluaran di *Output Layer*

y
0.762647
0.661902

4.3.5 Perhitungan Nilai MSE

Mean Square Error (MSE) adalah metode untuk mengevaluasi hasil prediksi. Untuk menghitung nilai MSE menggunakan Persamaan 2.9. Berikut ini perhitungan hasil nilai MSE:

$$MSE = \frac{\sum_{i=1}^{2}(y_i - t_i)^2}{n}$$

$$= \frac{(0.649106 - 0.535604)^2 + (0.405434 - 0.665635)^2}{2} = 0.025781$$

4.3.6 Perhitungan Denormalisasi

Perhitungan denormalisasi data yaitu menggunakan Persamaan 2.8. Nilai maksimal dan minimal sudah didapat sehingga langsung pada perhitungan denormalisasi. Berikut contoh perhitungan nilai denormalisasi:

$$x = x'(\max - \min) + \min$$

$$x_1 = 0.649106(8206 - 7560) + 7560 = 8052.67$$

Hasil perhitungan denormalisasi secara keseluruhan ditunjukkan pada Tabel 4.17 sebagai berikut:

Tabel 4.17 Hasil Denormalisasi

denormalisasi
8052.67
7987.588

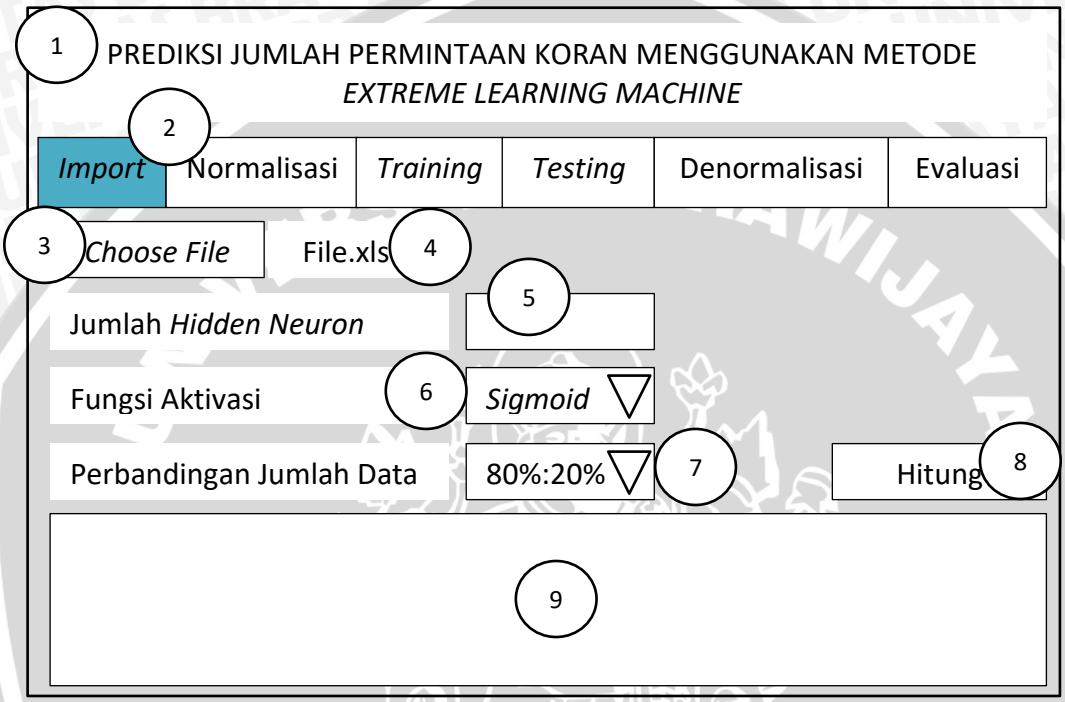
4.4 Perancangan Antar Muka

Perancangan antar muka bertujuan untuk menggambarkan keadaan sebenarnya dari implementasi komputasi metode ELM untuk prediksi jumlah

permintaan koran yang akan dibangun. Implementasi terdiri dari 6 halaman utama yaitu *import* data, normalisasi data, proses *training* metode ELM, proses *testing* metode ELM, denormalisasi data, dan evaluasi.

4.4.1 Perancangan Halaman *Import* Data

Halaman *import* data merupakan halaman untuk melakukan *import* data permintaan koran berupa *file* dengan format *file* .xls. Rancangan antar muka halaman *import* data ditunjukkan pada Gambar 4.14 sebagai berikut:



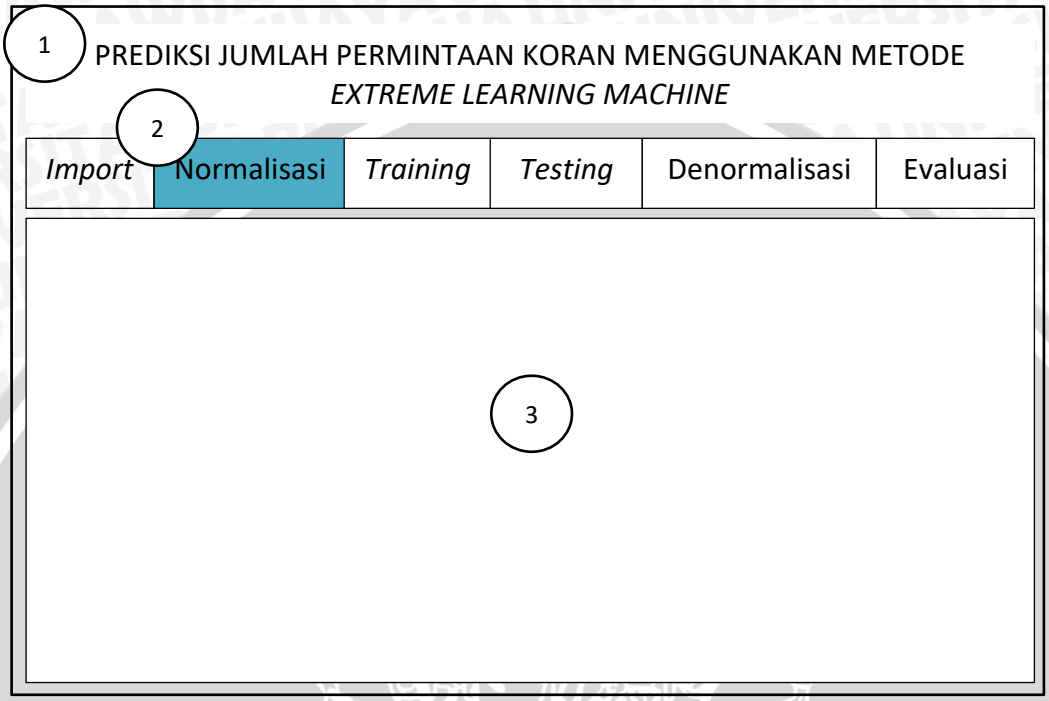
Gambar 4.14 Rancangan Halaman *Import* Data

Keterangan rancangan antar muka halaman *import* data pada Gambar 4.14 adalah sebagai berikut:

1. *Header* aplikasi
2. *Tab* menu aplikasi, *tab* menu berwarna biru menandakan bahwa menu *import* sedang aktif
3. *Button* untuk *choose file* dengan format .xls
4. *Text View* menunjukkan nama *file* yang dipilih
5. *Text Box* untuk memasukkan jumlah *neuron* pada *hidden* layer
6. *Combo Box* untuk pilihan fungsi aktivasi yang akan dipakai
7. *Combo Box* untuk pilihan perbandingan jumlah data yang akan digunakan
8. *Button* untuk memulai proses hitung
9. Data *View* untuk menampilkan data yang sudah di *import* dalam tabel

4.4.2 Perancangan Halaman Normalisasi Data

Halaman normalisasi data merupakan halaman untuk menampilkan hasil normalisasi data permintaan koran yang telah dihitung menggunakan metode *Min-Max Normalization*. Rancangan antar muka halaman normalisasi data ditunjukkan pada Gambar 4.15 sebagai berikut:



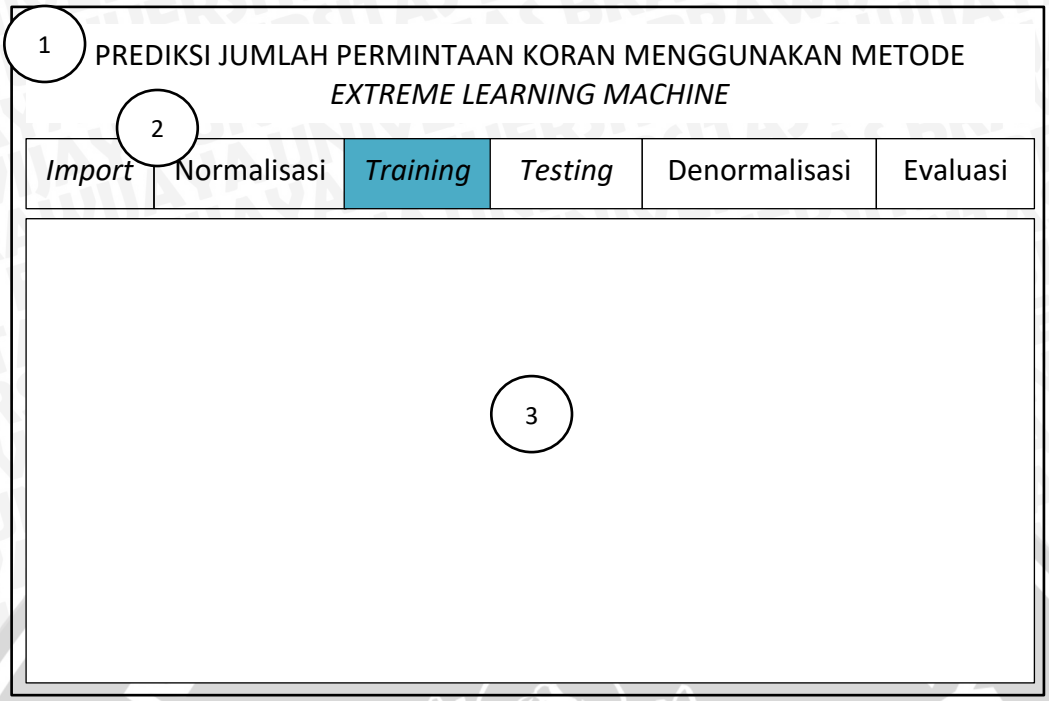
Gambar 4.15 Rancangan Halaman Normalisasi Data

Keterangan rancangan antar muka halaman normalisasi data pada Gambar 4.15 adalah sebagai berikut:

1. *Header* aplikasi
2. *Tab* menu aplikasi, *tab* menu berwarna biru menandakan bahwa menu normalisasi sedang aktif
3. Data *View* untuk menampilkan hasil normalisasi data dalam tabel

4.4.3 Perancangan Halaman *Training*

Halaman *training* merupakan halaman untuk menampilkan hasil perhitungan pada proses *training*. Rancangan antar muka halaman *training* ditunjukkan pada Gambar 4.16 sebagai berikut:



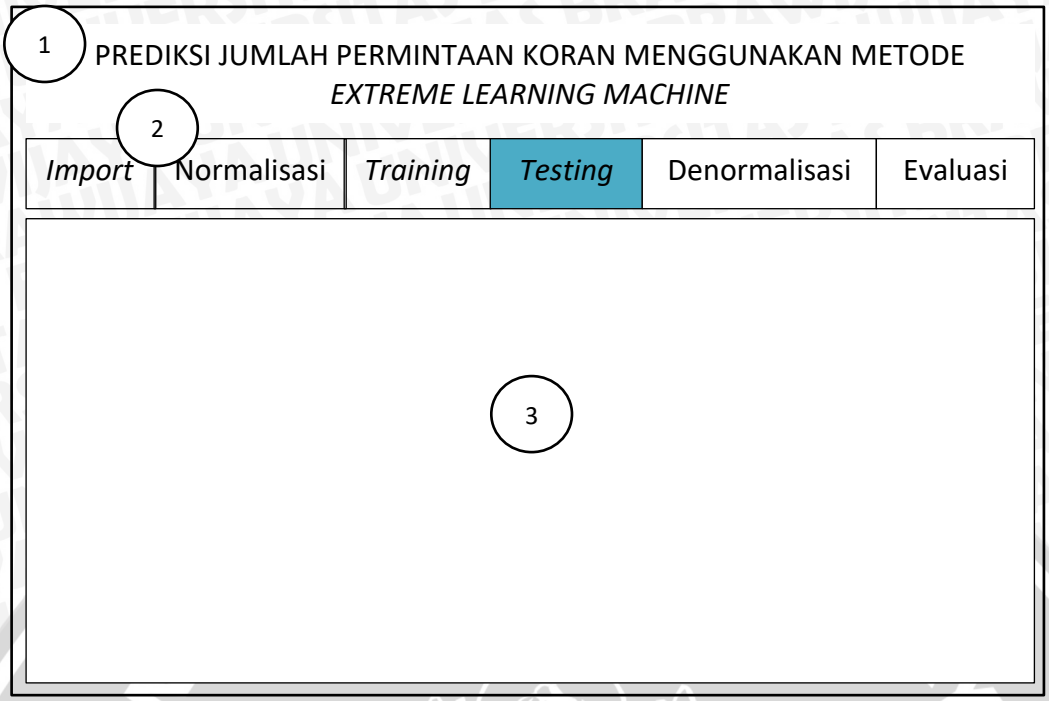
Gambar 4.16 Rancangan Halaman *Training*

Keterangan rancangan antar muka halaman *training* pada Gambar 4.16 adalah sebagai berikut:

1. *Header* aplikasi
2. *Tab* menu aplikasi, *tab* menu berwarna biru menandakan bahwa menu *training* sedang aktif
3. Data *View* untuk menampilkan hasil perhitungan proses *training* dalam tabel

4.4.4 Perancangan Halaman *Testing*

Halaman *testing* merupakan halaman untuk menampilkan hasil perhitungan pada proses *testing*. Rancangan antar muka halaman *testing* ditunjukkan pada Gambar 4.17 sebagai berikut:



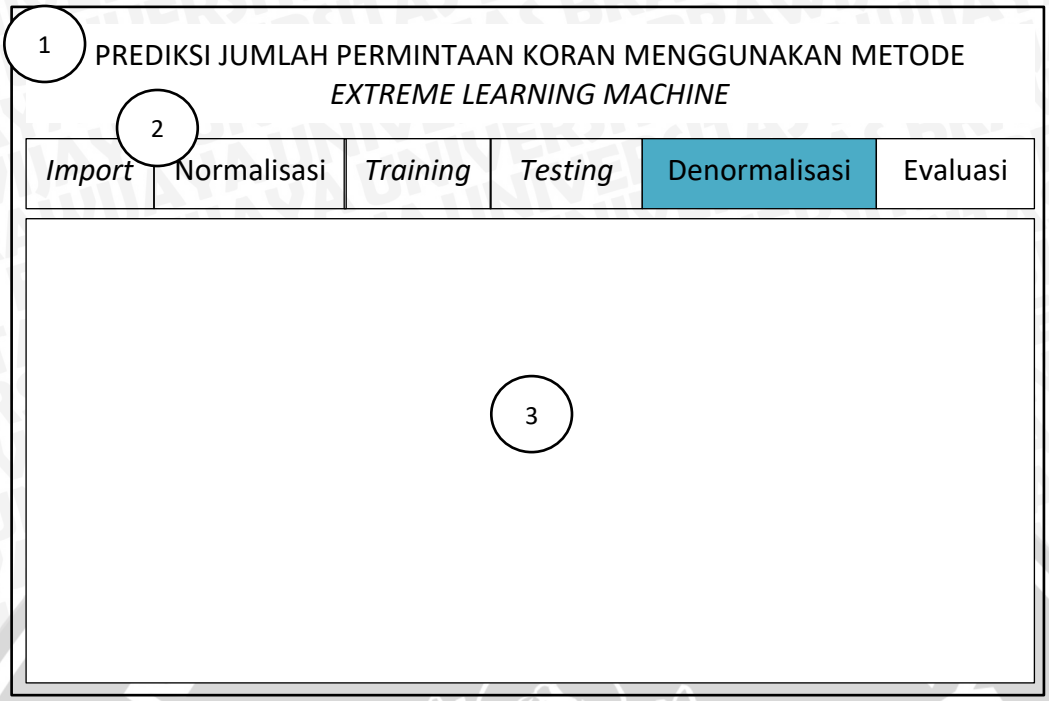
Gambar 4.17 Rancangan Halaman *Testing*

Keterangan rancangan antar muka halaman *testing* pada Gambar 4.17 adalah sebagai berikut:

1. *Header* aplikasi
2. *Tab* menu aplikasi, *tab* menu berwarna biru menandakan bahwa menu *testing* sedang aktif
3. Data *View* untuk menampilkan hasil perhitungan proses *testing* dalam tabel

4.4.5 Perancangan Halaman Denormalisasi Data

Halaman denormalisasi data merupakan halaman untuk menampilkan hasil denormalisasi data permintaan koran yang telah dihitung. Rancangan antar muka halaman denormalisasi data ditunjukkan pada Gambar 4.18 sebagai berikut:



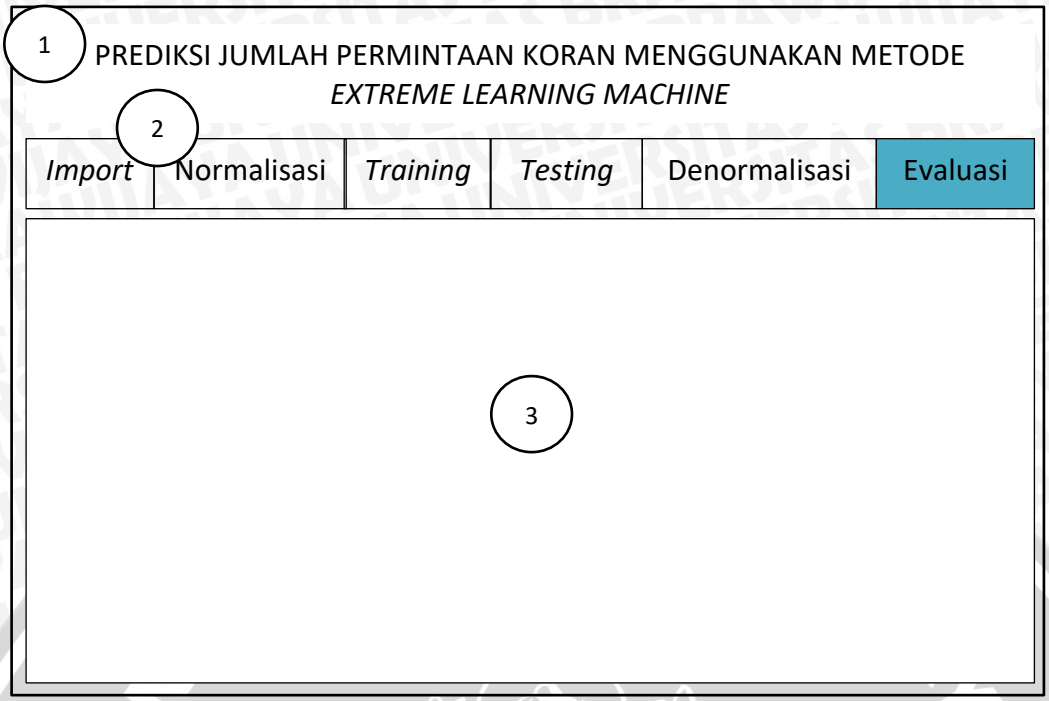
Gambar 4.18 Rancangan Halaman Denormalisasi Data

Keterangan rancangan antar muka halaman denormalisasi data pada Gambar 4.18 adalah sebagai berikut:

1. *Header* aplikasi
2. *Tab* menu aplikasi, *tab* menu berwarna biru menandakan bahwa menu denormalisasi data sedang aktif
3. Data *View* untuk menampilkan hasil perhitungan denormalisasi data dalam tabel

4.4.6 Perancangan Halaman Evaluasi

Halaman evaluasi merupakan halaman untuk menampilkan hasil evaluasi prediksi yang dihitung menggunakan metode MSE. Rancangan antar muka halaman evaluasi ditunjukkan pada Gambar 4.19 sebagai berikut:



Gambar 4.19 Rancangan Halaman Evaluasi

Keterangan rancangan antar muka halaman evaluasi pada Gambar 4.19 adalah sebagai berikut:

1. *Header* aplikasi
2. *Tab* menu aplikasi, *tab* menu berwarna biru menandakan bahwa menu evaluasi sedang aktif
3. Data *View* untuk menampilkan nilai error dalam tabel dan hasil perhitungan evaluasi MSE

4.5 Pengujian Algoritma

Pengujian pada penelitian ini terdiri dari pengujian yang terkait algoritma *Extreme Learning Machine* (ELM) yang digunakan untuk memberikan prediksi jumlah permintaan dan pengujian terkait tingkat *error*. Skenario pengujian yang dilakukan antara lain sebagai berikut:

a. Pengujian jumlah *neuron* pada *hidden layer*
b. Pengujian fungsi aktivasi
c. Pengujian jumlah data *training* dan data *testing*
d. Pengujian variasi fitur data
e. Pengujian berdasarkan hari yang sama

4.5.1 Pengujian Jumlah *Neuron* Pada *Hidden Layer*

Pengujian jumlah *neuron* pada *hidden layer* bertujuan untuk mendapatkan hasil pengenalan yang lebih baik. Jumlah *neuron* pada *hidden layer* yang diuji coba pada penelitian ini adalah 1 sampai 7 (Agustina, 2009).

Tabel 4.18 Rancangan Pengujian Jumlah *Neuron* Pada *Hidden Layer*

Percobaan ke-*i*	Nilai MSE Pada Jumlah *Hidden Neuron*						
	1	2	3	4	5	6	7
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
Rata-Rata Nilai MSE							

4.5.2 Pengujian Fungsi Aktivasi

Pengujian fungsi aktivasi dilakukan untuk mentransformasikan suatu *input* menjadi suatu *output* tertentu. Pada JST suatu informasi akan diterima oleh masukan. Masukan ini akan diproses melalui suatu fungsi. Fungsi ini akan menjumlahkan sejumlah masukan, hasil dari penjumlahan kemudian akan dibandingkan dengan nilai target (*threshold*) tertentu melalui fungsi aktivasi pada setiap *layer* (Hajar, 2013).

Ada berbagai macam fungsi aktivasi yang biasa digunakan pada JST. Untuk pengujian fungsi aktivasi ini yang akan diuji coba adalah fungsi aktivasi *sigmoid* dan fungsi aktivasi *sin* karena kedua fungsi tersebut yang paling sering digunakan pada permasalahan *forecasting* menggunakan metode ELM.

Tabel 4.19 Rancangan Pengujian Fungsi Aktivasi

Percobaan ke-*i*	Nilai MSE Pada Fungsi Aktivasi	
	Sigmoid	***Sin***
1		

Percobaan ke-*i*	Nilai MSE Pada Fungsi Aktivasi	
	Sigmoid	***Sin***
2		
3		
4		
5		
6		
7		
8		
9		
10		
Rata-Rata Nilai MSE		

4.5.3 Pengujian Perbandingan Jumlah Data *Training* dan data *Testing*

Pengujian perbandingan jumlah data *training* dan data *testing* ini dilakukan untuk mengetahui pengaruh dari perbandingan jumlah data *training* dengan data *testing* terhadap nilai MSE yang dihasilkan. Pengujian perbandingan jumlah data *training* dan data *testing* terbaik dengan 5 jenis perbandingan data yaitu 80% : 20%, 70% : 30%, 60% : 40%, 50% : 50%, dan 40% : 60% (Data *Training* : Data *Testing*).

Tabel 4.20 Rancangan Pengujian Perbandingan Jumlah Data *Training* dan data *Testing*

Percobaan ke-*i*	Nilai MSE Pada Perbandingan Jumlah Data *Training* dan Data *Testing*				
	80%:20%	**70%:30%**	**60%:40%**	**50%:50%**	**40%:60%**
1					
2					
3					
4					
5					
6					
7					
8					

Percobaan ke-*i*	Nilai MSE Pada Perbandingan Jumlah Data *Training* dan Data *Testing*				
	80%:20%	70%:30%	60%:40%	50%:50%	40%:60%
9					
10					
Rata-Rata Nilai MSE					

4.5.4 Pengujian Variasi Fitur Data

Pengujian variasi fitur data dilakukan untuk mengetahui pengaruh dari jumlah fitur yang digunakan terhadap nilai MSE yang dihasilkan. Jumlah fitur yang diuji coba pada penelitian ini adalah 2 sampai 7. Karena menurut pakar untuk memprediksi jumlah permintaan dilakukan dengan melihat jumlah permintaan dari 4 sampai 7 hari sebelumnya.

Tabel 4.21 Rancangan Pengujian Variasi Fitur Data

Percobaan ke-*i*	Nilai MSE Pada Jumlah Fitur Data					
	2	3	4	5	6	7
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
Rata-Rata Nilai MSE						

4.5.5 Pengujian Berdasarkan Hari yang Sama

Pengujian berdasarkan hari yang sama dilakukan untuk mengetahui pengaruh dari hari yang digunakan terhadap nilai MSE yang dihasilkan. Hari yang diuji yaitu senin, selasa, rabu, kamis, jumat, sabtu, dan minggu.

Tabel 4.22 Rancangan Pengujian Berdasarkan Hari yang Sama

Percobaan ke-*i*	Nilai MSE Pada Hari						
	Senin	Selasa	Rabu	Kamis	Jumat	Sabtu	Minggu
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
Rata-Rata Nilai MSE							

BAB 5 IMPLEMENTASI

Pada bab ini membahas tentang implementasi dan antarmuka dari perancangan komputasi yang telah dibuat untuk prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine*.

5.1 Implementasi Sistem

Berdasarkan perancangan yang telah dibahas pada Bab 4, maka selanjutnya dibahas mengenai implementasi sistem sesuai dengan perancangan yang telah dibuat. Sistem diimplementasikan menggunakan bahasa pemrograman Java dengan editor NetBeans IDE 7.4.

5.1.1 Implementasi Normalisasi Data

Proses ini diawali dengan melakukan normalisasi terhadap seluruh data yang akan diolah untuk standarisasi data. Langkah awal adalah mencari nilai *maximum* dan *minimum* data. Proses normalisasi menggunakan *Min-Max Normalization*. Proses normalisasi data dapat dilihat pada Kode Program 5.1 Implementasi Normalisasi Data.

Kode Program 5.1 Implementasi Normalisasi Data

```
1     void normalisasi() {
2         for (int i = 0; i < jumlahData; i++) {
3             for (int j = 0; j < jumlahInputLayer; j++) {
4                 if (isiData[i][j] > max) {
5                     max = isiData[i][j];
6                 }
7                 if (isiData[i][j] < min) {
8                     min = isiData[i][j];
9                 }
10            }
11        }
12        for (int i = 0; i < jumlahData; i++) {
13            for (int j = 0; j < jumlahInputLayer; j++) {
14                isiData[i][j] = (isiData[i][j] - min) / (max
15                - min);
16            }
17            target[i] = (target[i] - min) / (max - min);
18        }
19    }
```

Penjelasan Kode Program 5.1 tentang implementasi normalisasi data adalah sebagai berikut:

Baris 2 s.d 11 : Proses mencari nilai maksimal dan nilai minimal dari data.
Baris 12 s.d 19 : Proses menghitung nilai normalisasi setiap data menggunakan Persamaan 2.7

5.1.2 Implementasi Inisialisasi *Input Weight* dan *Bias*

Implementasi inisialisasi *input weight* dan *bias* merupakan proses untuk menginisialisasi *input weight* dan *bias* yang selanjutnya digunakan untuk menghitung keluaran *hidden layer*. Untuk menentukan *input weight* dan *bias* didapatkan secara random dalam bentuk matriks dengan ordo sesuai dengan banyaknya *input neuron* dan *hidden neuron*. Implementasi inisialisasi *input weight* dan *bias* ditunjukkan pada Kode Program 5.2 sebagai berikut.

Kode Program 5.2 Implementasi Inisialisasi *Input Weight* dan *Bias*

```
1     void inisialisasi() {
2         for (int i = 0; i < jumlahHiddenLayer; i++) {
3             for (int j = 0; j < jumlahInputLayer; j++) {
4                 double w = -1 + random.nextDouble() * (1 -
5                 (-1));
6                 inputWeight[i][j] =
7                 Double.valueOf(df.format(w));
8             }
9             double b = -1 + random.nextDouble() * (1 - (
10            1));
11            bias[i] = Double.valueOf(df.format(b));
12        }
13    }
```

Penjelasan Kode Program 5.2 tentang implementasi normalisasi data adalah sebagai berikut:

Baris 2 s.d 13 : Proses inisialisasi *input weight* dan *bias* secara random dengan *range* -1 hingga 1.

5.1.3 Implementasi Hitung Keluaran *Hidden Layer* dengan Fungsi Aktivasi

Proses menghitung keluaran *hidden layer* dengan fungsi aktivasi dilakukan berdasarkan perhitungan yang telah dijelaskan pada Bab 4. Perhitungan keluaran *hidden layer* dengan fungsi aktivasi pada proses *training* sama dengan yang dilakukan pada proses testing. Proses mencari keluaran *hidden layer* dengan fungsi aktivasi dapat dilihat pada Kode Program 5.3 sebagai berikut.

Kode Program 5.3 Implementasi Hitung Keluaran *Hidden Layer* dengan Fungsi Aktivasi

```
1     void keluaranHiddenLayer(int jmlhdata, double[][] data)
2     {
3         outputHidden = new
4         double[jmlhdata][jumlahHiddenLayer];
5         for (int i = 0; i < jmlhdata; i++) {
6             for (int j = 0; j < jumlahHiddenLayer; j++) {
7                 outputHidden[i][j] = 0;
8                 for (int k = 0; k < jumlahInputLayer; k++) {
9                     outputHidden[i][j] = outputHidden[i][j]
10                    + (inputWeight[j][k] * data[i][k]);
```

```
11                  }
12                  outputHidden[i][j] = outputHidden[i][j] +
13                  (bias[j]);
14                  if (fungsiAktivasi == "Sigmoid") {
15                      outputHidden[i][j] = 1 / (1 + Math.exp(
16                      outputHidden[i][j]));
17                  } else {
18                      outputHidden[i][j] =
19                      Math.sin(outputHidden[i][j]);
20                  }
21              }
22          }
23      }
```

Penjelasan Kode Program 5.3 tentang proses menghitung keluaran *hidden layer* dengan fungsi aktivasi adalah sebagai berikut:

Baris 8 s.d 13 : Proses menghitung nilai keluaran *hidden layer* menggunakan Persamaan 2.4.

Baris 14 s.d 20 : Proses menghitung nilai keluaran *hidden layer* dengan fungsi aktivasi menggunakan Persamaan 2.1 atau Persamaan 2.2

5.1.4 Implementasi Hitung *Output Weight*

Proses menghitung *output weight* dilakukan berdasarkan perhitungan yang telah dijelaskan pada Bab 4. Untuk menghitung *output weight*, pertama harus menghitung Matriks *Moore-Penrose Generalized Invers* dari hasil keluaran *hidden layer* dengan fungsi aktivasi. Proses mencari *output weight* dapat dilihat pada Kode Program 5.4 sebagai berikut.

Kode Program 5.4 Implementasi Hitung *Output Weight*

```
1       double[][] perkalianMatrik(double[][] matrikA,
2       double[][] matrikB) {
3           double hasil[][] = new
4           double[matrikA.length][matrikB[0].length];
5           for (int i = 0; i < hasil.length; i++) {
6               for (int j = 0; j < hasil[0].length; j++) {
7                   for (int k = 0; k < matrikA[0].length; k++)
8                   {
9                       hasil[i][j] += matrikA[i][k] *
10                      matrikB[k][j];
11                  }
12              }
13          }
14          return hasil;
15      }
16
```

```
17      void outputWeight() {
18          double[][] transpose = new
19          double[jumlahHiddenLayer][jumlahTraining];

20          for (int i = 0; i < jumlahTraining; i++) {
21              for (int j = 0; j < jumlahHiddenLayer; j++) {
22                  transpose[j][i] = outputHidden[i][j];
23              }
24          }
25          double[][] transposeKaliHidden =
26          perkalianMatrik(transpose, outputHidden);
27          double[][] invers = new
29          double[transpose.length]
30          [transposeKaliHidden[0].length];
31          for (int k = 0; k < transpose.length; k++) {
32              for (int l = 0; l <
33              transposeKaliHidden[0].length; l++) {
34                  if (k == l) {
35                      invers[k][l] = 1;
36                  } else {
37                      invers[k][l] = 0;
38                  }
39              }
40          }
41
42          for (int k = 0; k < transpose.length; k++) {
43              double t = transposeKaliHidden[k][k];
44              for (int n = 0; n <
45              transposeKaliHidden[0].length; n++) {
46                  invers[k][n] = invers[k][n] / t;
47                  transposeKaliHidden[k][n] =
48                  transposeKaliHidden[k][n] / t;
49              }
50              for (int l = 0; l <
51              transposeKaliHidden[0].length; l++) {
52                  double c = transposeKaliHidden[l][k];
53                  for (int m = 0; m <
54                  transposeKaliHidden[0].length; m++) {
55                      if (k != l) {
56                          invers[l][m] = invers[l][m] - c *
57                          invers[k][m];
58                          transposeKaliHidden[l][m] =
59                          transposeKaliHidden[l][m] - c *
60                          transposeKaliHidden[k][m];
61                      }
62                  }
63              }
64          }
65
66          double[][] hPlus = perkalianMatrik(invers,
```

```
67          transpose);
68
69          outputWeight = perkalianMatrik(hPlus,
70          targetTraining);
71          for (int i = 0; i < hPlus.length; i++) {
72              for (int j = 0; j < outputWeight[0].length; j++)
73              {
74                  outputW.addRow(new
75                  Object[]{outputWeight[i][j]});
76              }
77          }
78      }
```

Penjelasan Kode Program 5.4 tentang proses menghitung *output weight* adalah sebagai berikut:

Baris 1 s.d 15 : *Method* untuk proses menghitung perkalian matriks.

Baris 18 s.d 25 : Proses menghitung *transpose* matriks dari nilai keluaran *hidden layer* dengan fungsi aktivasi.

Baris 26 s.d 27 : Proses menghitung perkalian matriks hasil transpose dengan keluaran *hidden layer* menggunakan *method* perkalian yang sudah dibuat.

Baris 29 s.d 40 : Proses inisialisasi matriks identitas dengan nama variabel invers.

Baris 42 s.d 64 : Proses mengubah matriks hasil perkalian menjadi matriks identitas, sehingga matriks dengan nama variabel invers menjadi matriks baru hasil invers.

Baris 66 s.d 67 : Proses menghitung H^+ dari perkalian matriks hasil *invers* dengan hasil *transpose* menggunakan *method* perkalian yang sudah dibuat.

Baris 69 s.d 70 : Proses menghitung *output weight* dari perkalian matriks H^+ dengan matriks target menggunakan *method* perkalian yang sudah dibuat.

Baris 71 s.d 77 : Proses menampilkan hasil *output weight* ke tabel.

5.1.5 Implementasi Hitung Keluaran di *Output Layer*

Proses menghitung keluaran di *output layer* dilakukan berdasarkan perhitungan yang telah dijelaskan pada Bab 4. Untuk menghitung *output layer* menggunakan *output weight* yang telah dihitung sebelumnya. Proses menghitung keluaran di *output layer* dapat dilihat pada Kode Program 5.5 sebagai berikut.

Kode Program 5.5 Implementasi Hitung Keluaran di *Output Layer*

```
1       void outputLayer() {
2           output = perkalianMatrik(outputHidden,
3           outputWeight);
4           for (int i = 0; i < jumlahTesting; i++) {
5               outputLayer.addRow(new Object[]{i + 1,
6               output[i][0]});
```

```
7           }
8       }
```

Penjelasan Kode Program 5.5 tentang menghitung keluaran di *output layer* adalah sebagai berikut:

Baris 2 s.d 3 : Proses menghitung nilai keluaran di *output layer* menggunakan Persamaan 2.6.

Baris 4 s.d 7 : Proses menampilkan hasil perhitungan keluaran di *output layer*.

5.1.6 Implementasi Hitung *Mean Square Error* (MSE)

Proses menghitung *Mean Square Error* (MSE) menggunakan nilai target dan hasil perhitungan nilai *output*. Proses menghitung MSE dapat dilihat pada Kode Program 5.8 Implementasi Hitung MSE.

Kode Program 5.8 Implementasi Hitung MSE

```
1       void mse() {
2           nilaiMse = 0;
3           for (int i = 0; i < jumlahTesting; i++) {
4               nilaiMse = nilaiMse + Math.pow((output[i][0] -
5               targetTesting[i]), 2);
6           }
7           nilaiMse = nilaiMse / jumlahTesting;
8       }
```

Penjelasan Kode Program 5.8 tentang menghitung nilai MSE adalah pada baris 2 s.d 7 merupakan proses menghitung MSE menggunakan Persamaan 2.9.

5.1.7 Implementasi Denormalisasi Data

Proses denormalisasi menggunakan nilai minimal dan maksimal yang telah didapat pada proses normalisasi data. Proses denormalisasi data dapat dilihat pada Kode Program 5.7 Implementasi Denormalisasi Data.

Kode 5.7 Implementasi Denormalisasi Data

```
1       void denormalisasi(){
2           for(int i=0; i<jumlahTesting;i++){
3               output[i]=output[i]*(max-min)+min;
4               denormalisasi.addRow(new
5               Object[]{i+1,Math.round(output[i])});
6           }
7       }
```

Penjelasan Kode Program 5.7 tentang menghitung denormalisasi data adalah sebagai berikut:

Baris 3 : Proses menghitung denormalisasi data menggunakan Persamaan 2.8

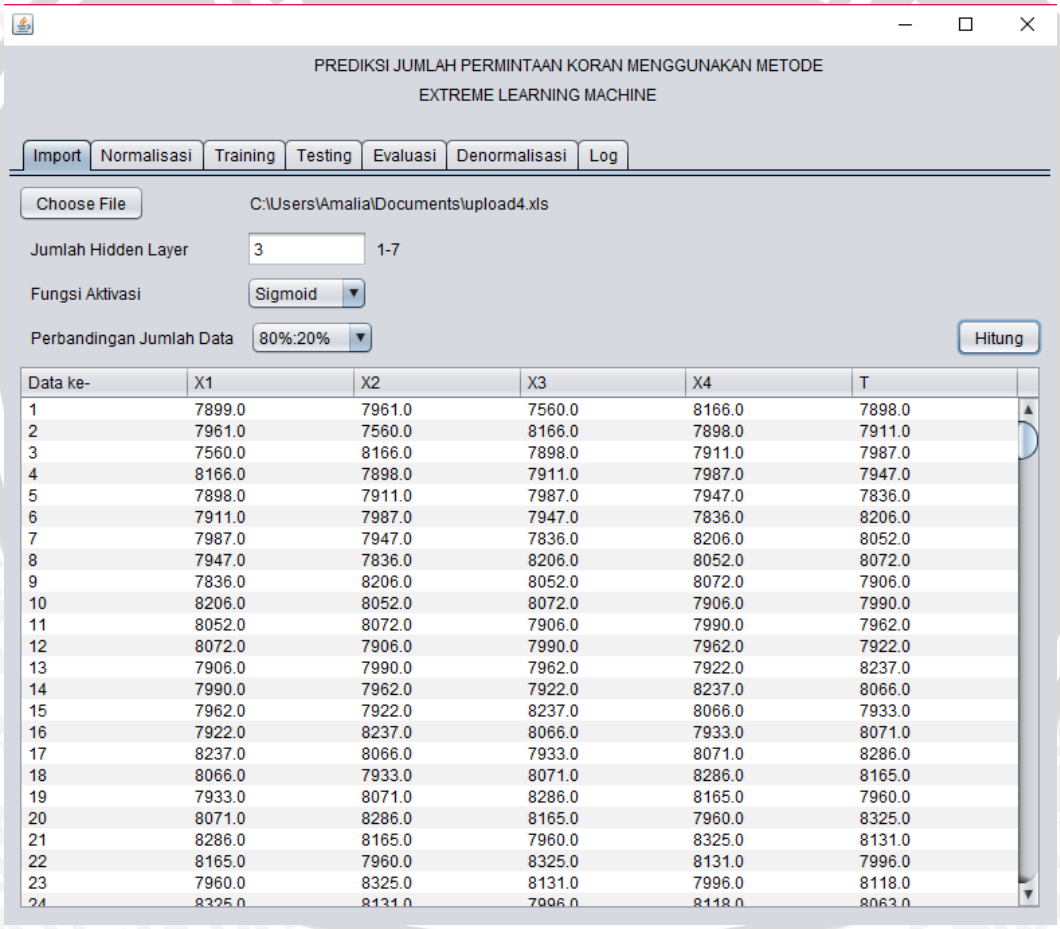
Baris 4 s.d 5 : Proses menampilkan hasil perhitungan denormalisasi data.

5.2 Implementasi Antarmuka

Implementasi antarmuka merupakan hasil perancangan antarmuka yang telah dibuat sebelumnya. Antarmuka komputasi prediksi jumlah permintaan koran ini terdiri dari 6 halaman utama meliputi halaman *import* data, halaman normalisasi data, halaman proses *training*, halaman proses *testing*, halaman denormalisasi, dan halaman evaluasi.

5.2.1 Implementasi Halaman *Import* Data

Implementasi halaman *import* data merupakan halaman utama pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menerima masukan dataset jumlah permintaan koran dan beberapa masukan. Halaman ini juga berfungsi untuk menampilkan hasil *load* data pada *datagridview* dalam bentuk tabel. Implementasi halaman *import* data ditunjukkan pada Gambar 5.1 sebagai berikut:



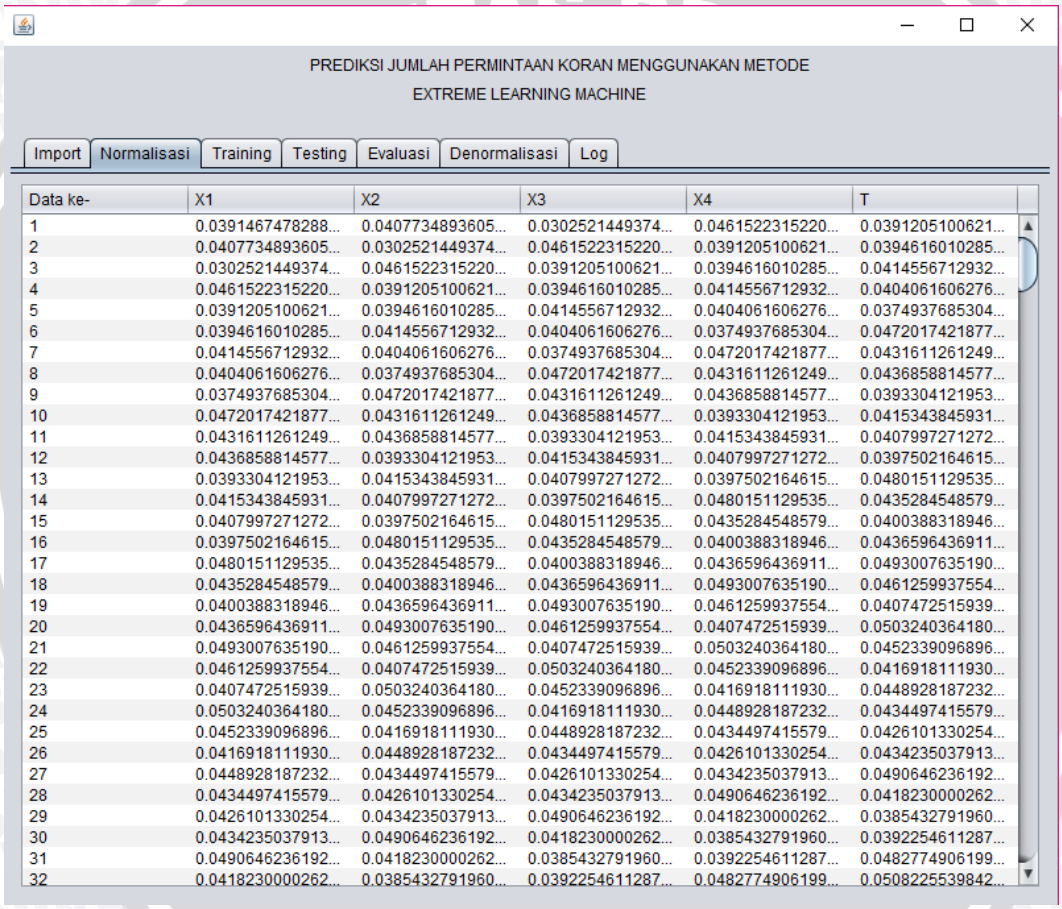
Gambar 5.1 Implementasi Halaman *Import* Data

Gambar 5.1 menjelaskan hasil implementasi halaman *import* data. Proses *import* data diawali dengan pengguna menekan tombol *choose file* bertujuan untuk memasukkan file berisi data jumlah permintaan koran dengan format .xls dari komputer pengguna. Lokasi *file* ditampilkan pada *textbox* pertama. Kemudian pengguna memasukan *input* berupa jumlah *neuron* pada *hidden layer*,

fungsi aktivasi yang digunakan, dan perbandingan jumlah data *training* dan data *testing*. Langkah selanjutnya, pengguna menekan tombol hitung bertujuan untuk me-*copy* data dari *file* ke dalam *array* (kebutuhan komputasi) dan langsung dihitung proses komputasi prediksi jumlah permintaan koran. Data yang berhasil diproses oleh sistem ditampilkan dalam bentuk tabel pada *datagridview*.

5.2.2 Implementasi Halaman Normalisasi Data

Implementasi halaman normalisasi data merupakan halaman kedua pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menampilkan hasil proses normalisasi data pada *datagridview* dalam bentuk tabel. Implementasi halaman normalisasi data ditunjukkan pada Gambar 5.2 sebagai berikut:

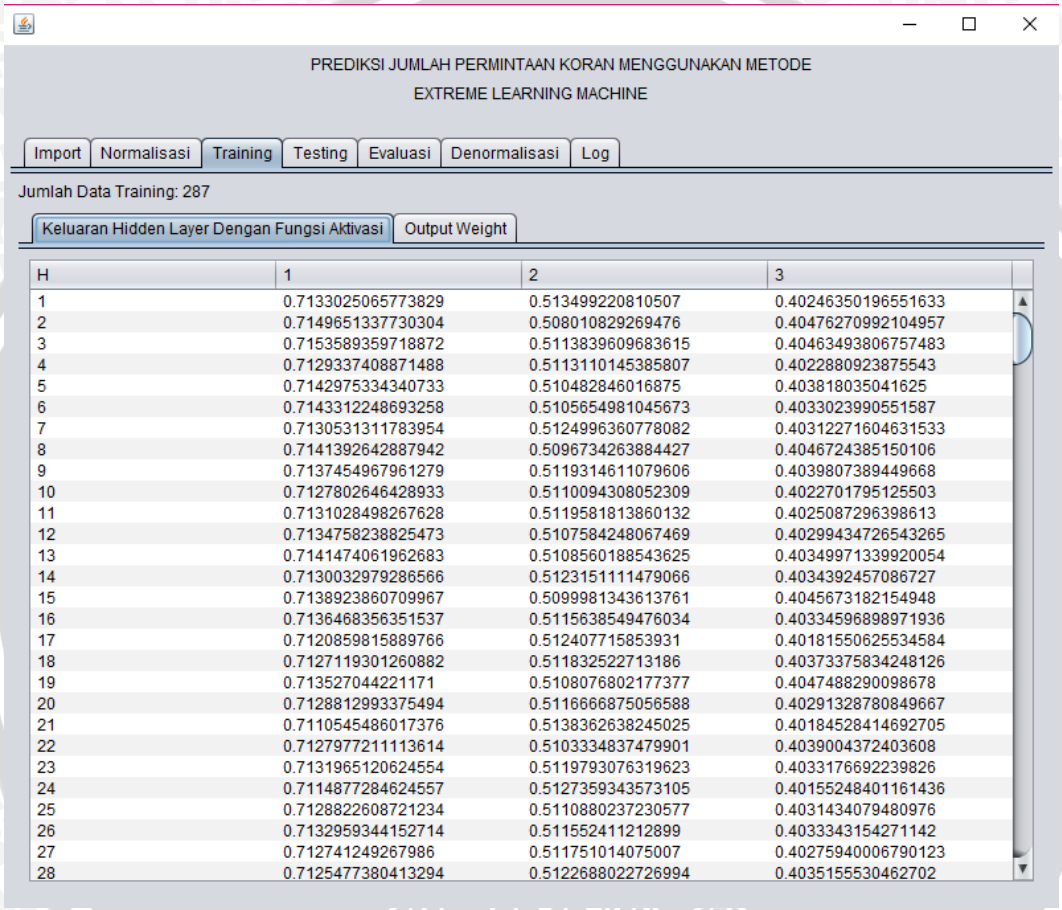


Gambar 5.2 Implementasi Halaman Normalisasi Data

Gambar 5.2 menjelaskan hasil implementasi halaman normalisasi data. Proses normalisasi data menggunakan metode *Min-Max Normalization* telah dilakukan perhitungan sejak tombol hitung pada halaman utama ditekan. Hasil normalisasi data jumlah permintaan koran ditampilkan dalam bentuk tabel pada *datagridview*.

5.2.3 Implementasi Halaman Proses *Training*

Implementasi halaman proses *training* merupakan halaman ketiga pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menampilkan hasil proses *training* berupa hasil keluaran pada *hidden layer* dengan fungsi aktivasi pada data *training* dan hasil *output weight*. Halaman ini akan menampilkan hasilnya pada *datagridview* dalam bentuk tabel. Implementasi halaman proses *training* ditunjukkan pada Gambar 5.3 sebagai berikut:



H	1	2	3
1	0.7133025065773829	0.513499220810507	0.40246350196551633
2	0.7149651337730304	0.508010829269476	0.40476270992104957
3	0.7153589359718872	0.5113839609683615	0.40463493806757483
4	0.7129337408871488	0.5113110145385807	0.4022880923875543
5	0.7142975334340733	0.510482846016875	0.403818035041625
6	0.7143312248693258	0.5105654981045673	0.4033023990551587
7	0.7130531311783954	0.5124996360778082	0.40312271604631533
8	0.7141392642887942	0.5096734263884427	0.4046724385150106
9	0.7137454967961279	0.5119314611079606	0.4039807389449668
10	0.7127802646428933	0.5110094308052309	0.4022701795125503
11	0.7131028498267628	0.5119581813860132	0.4025087296398613
12	0.7134758238825473	0.5107584248067469	0.40299434726543265
13	0.7141474061962683	0.5108560188543625	0.40349971339920054
14	0.7130032979286566	0.5123151111479066	0.4034392457086727
15	0.7138923860709967	0.5099981343613761	0.4045673182154948
16	0.7136468356351537	0.5115638549476034	0.40334596898971936
17	0.7120859815889766	0.512407715853931	0.40181550625534584
18	0.7127119301260882	0.511832522713186	0.40373375834248126
19	0.713527044221171	0.5108076802177377	0.4047488290098678
20	0.7128812993375494	0.5116666875056588	0.40291328780849667
21	0.7110545486017376	0.5138362638245025	0.40184528414692705
22	0.7127977211113614	0.5103334837479901	0.4039004372403608
23	0.7131965120624554	0.5119793076319623	0.40331766922239826
24	0.7114877284624557	0.5127359343573105	0.40155248401161436
25	0.7128822608721234	0.5110880237230577	0.4031434079480976
26	0.7132959344152714	0.511552411212899	0.4033343154271142
27	0.712741249267986	0.511751014075007	0.40275940006790123
28	0.7125477380413294	0.5122688022726994	0.4035155530462702

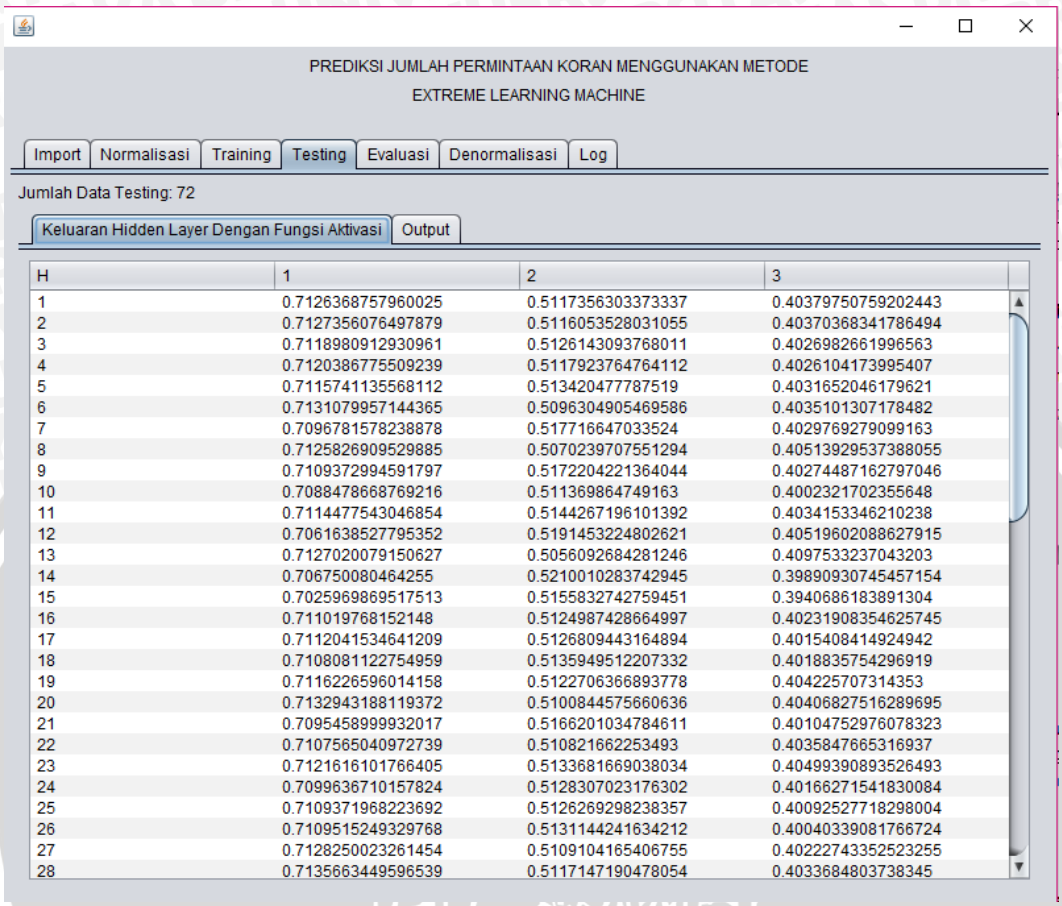
Gambar 5.3 Implementasi Halaman Proses *Training*

Gambar 5.3 menjelaskan hasil implementasi halaman proses *training*. Proses *training* telah dilakukan sejak tombol hitung pada halaman utama ditekan. Hasil perhitungan proses *training* yaitu keluaran pada *hidden layer* dengan fungsi aktivasi dan *output weight* yang telah diproses oleh sistem ditampilkan dalam bentuk tabel pada *datagridview*.

5.2.4 Implementasi Halaman Proses *Testing*

Implementasi halaman proses *testing* merupakan halaman keempat pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menampilkan hasil proses *testing* berupa hasil keluaran pada *hidden layer* dengan fungsi aktivasi pada data *testing* dan hasil keluaran pada *output layer*.

Halaman ini akan menampilkan hasilnya pada *datagridview* dalam bentuk tabel. Implementasi halaman proses *testing* ditunjukkan pada Gambar 5.4 sebagai berikut:



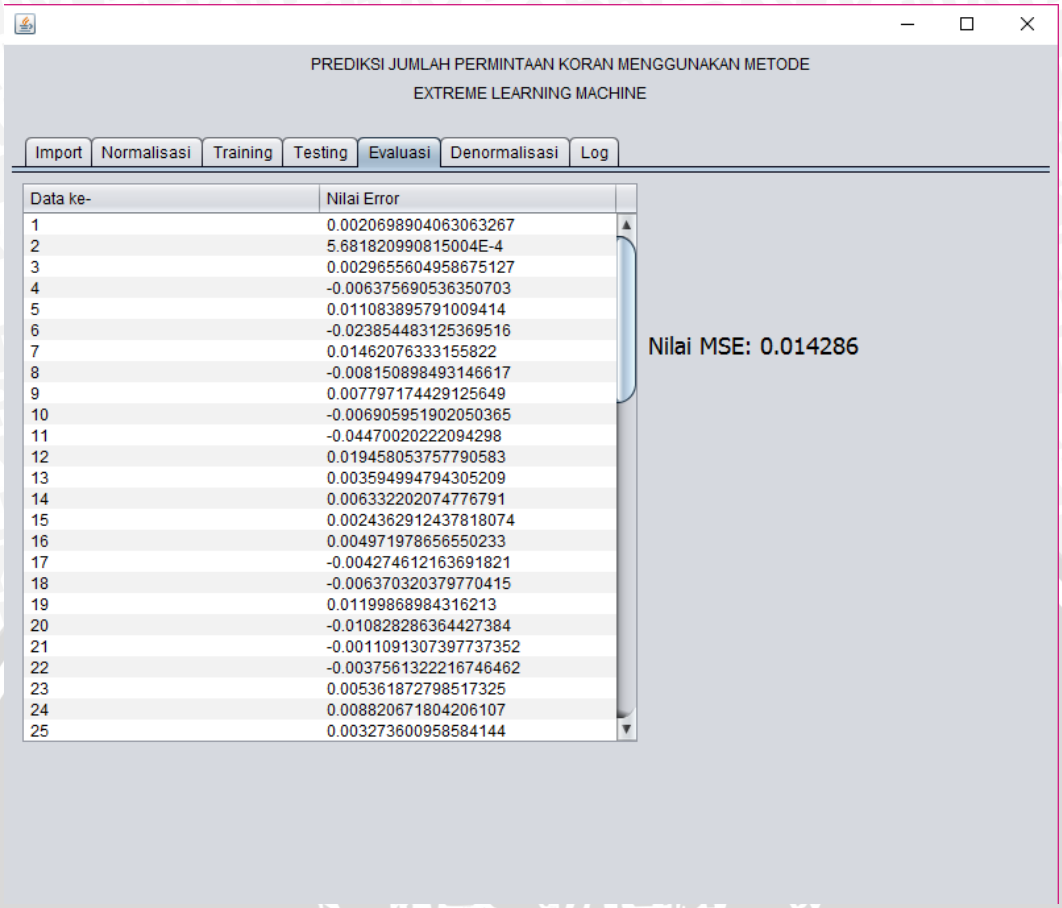
H	1	2	3
1	0.7126368757960025	0.5117356303373337	0.40379750759202443
2	0.7127356076497879	0.5116053528031055	0.40370368341786494
3	0.7118980912930961	0.5126143093768011	0.4026982661996563
4	0.7120386775509239	0.5117923764764112	0.4026104173995407
5	0.7115741135568112	0.513420477787519	0.4031652046179621
6	0.7131079957144365	0.5096304905469586	0.4035101307178482
7	0.7096781578238878	0.517716647033524	0.4029769279099163
8	0.7125826909529885	0.5070239707551294	0.40513929537388055
9	0.7109372994591797	0.5172204221364044	0.40274487162797046
10	0.7088478668769216	0.511369864749163	0.4002321702355648
11	0.7114477543046854	0.5144267196101392	0.4034153346210238
12	0.7061638527795352	0.5191453224802621	0.40519602088627915
13	0.7127020079150627	0.5056092684281246	0.4097533237043203
14	0.706750080464255	0.5210010283742945	0.39890930745457154
15	0.7025969869517513	0.5155832742759451	0.3940686183891304
16	0.711019768152148	0.5124987428664997	0.40231908354625745
17	0.7112041534641209	0.5126809443164894	0.4015408414924942
18	0.7108081122754959	0.5135949512207332	0.4018835754296919
19	0.7116226596014158	0.5122706366893778	0.404225707314353
20	0.7132943188119372	0.5100844575660636	0.40406827516289695
21	0.7095458999932017	0.5166201034784611	0.40104752976078323
22	0.7107565040972739	0.510821662253493	0.4035847665316937
23	0.7121616101766405	0.5133681669038034	0.40499390893526493
24	0.7099636710157824	0.5128307023176302	0.40166271541830084
25	0.7109371968223692	0.5126269298238357	0.40092527718298004
26	0.7109515249329768	0.5131144241634212	0.40040339081766724
27	0.7128250023261454	0.5109104165406755	0.40222743352523255
28	0.7135663449596539	0.5117147190478054	0.4033684803738345

Gambar 5.4 Implementasi Halaman Proses *Testing*

Gambar 5.4 menjelaskan hasil implementasi halaman proses *testing*. Proses *testing* telah dilakukan sejak tombol hitung pada halaman utama ditekan. Hasil perhitungan proses *testing* yaitu keluaran pada *hidden layer* dengan fungsi aktivasi pada data *testing* dan keluaran pada *output layer* yang telah diproses oleh sistem ditampilkan dalam bentuk tabel pada *datagridview*.

5.2.5 Implementasi Halaman Evaluasi

Implementasi halaman evaluasi merupakan halaman keenam pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menampilkan hasil evaluasi berupa hasil nilai *error* pada *output layer* dan hasil tingkat error dengan menggunakan metode *Mean Square Error* (MSE). Halaman ini akan menampilkan hasilnya pada *datagridview* dalam bentuk tabel. Implementasi halaman evaluasi ditunjukkan pada Gambar 5.5 sebagai berikut:

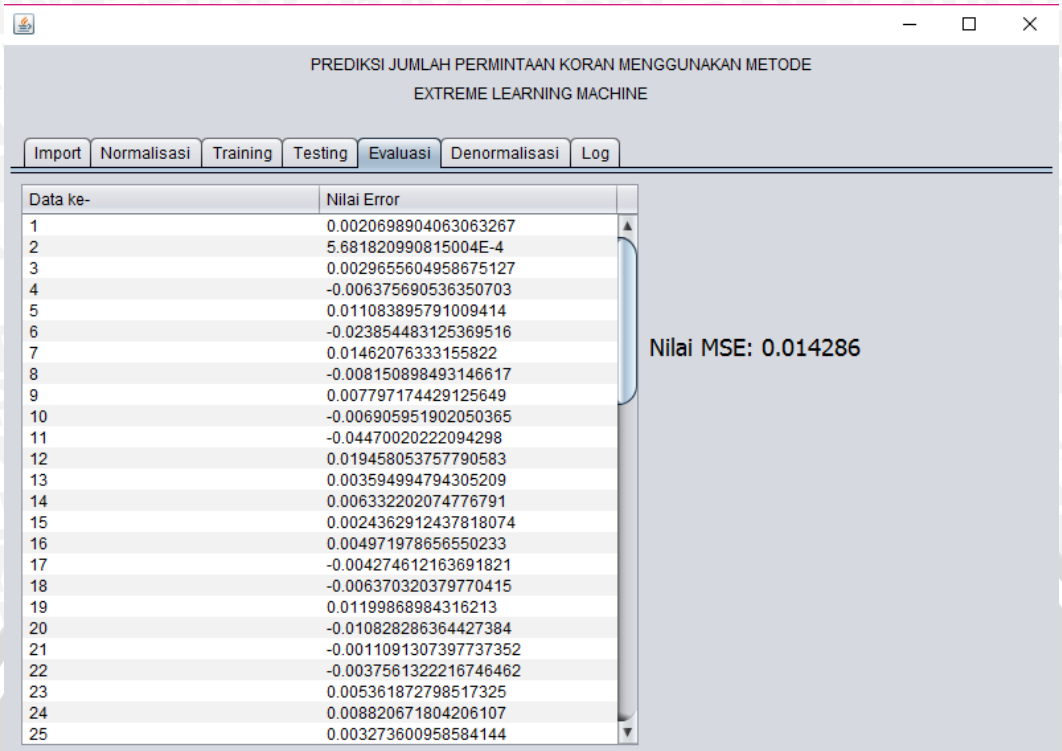


Gambar 5.5 Implementasi Halaman Evaluasi

Gambar 5.5 menjelaskan hasil implementasi halaman evaluasi. Proses evaluasi telah dilakukan sejak tombol hitung pada halaman utama ditekan. Hasil perhitungan evaluasi yaitu nilai *error* setiap *output layer* yang telah diproses oleh sistem ditampilkan dalam bentuk tabel pada *datagridview* dan nilai evaluasi MSE yang ditampilkan pada *textbox*.

5.2.6 Implementasi Halaman Denormalisasi Data

Implementasi halaman denormalisasi data merupakan halaman kelima pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menampilkan hasil proses denormalisasi data pada *datagridview* dalam bentuk tabel. Implementasi halaman denormalisasi data ditunjukkan pada Gambar 5.6 sebagai berikut:

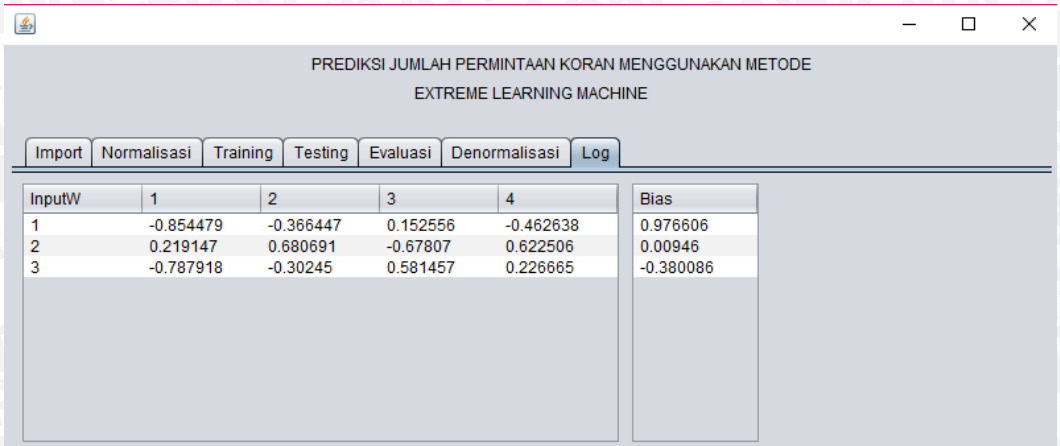


Gambar 5.6 Implementasi Halaman Denormalisasi Data

Gambar 5.6 menjelaskan hasil implementasi halaman denormalisasi data. Proses denormalsasi data telah dilakukan perhitungan sejak tombol hitung pada halaman utama ditekan. Hasil denormalisasi data jumlah permintaan koran ditampilkan dalam bentuk tabel pada *datagridview*.

5.2.7 Implementasi Halaman Log

Implementasi halaman log merupakan halaman ketujuh pada komputasi prediksi jumlah permintaan koran. Halaman ini berfungsi untuk menampilkan hasil nilai *input weight* dan *bias* yang dihitung secara *random*. Halaman ini akan menampilkan hasilnya pada *datagridview* dalam bentuk tabel. Implementasi halaman log ditunjukkan pada Gambar 5.7 sebagai berikut:



Gambar 5.7 Implementasi Halaman Log

Gambar 5.7 menjelaskan hasil implementasi halaman log. Proses inisialisasi *input weight* dan *bias* telah dilakukan sejak tombol hitung pada halaman utama ditekan. Hasil inisialisasi ditampilkan dalam bentuk tabel pada *datagridview*.

BAB 6 PENGUJIAN DAN PEMBAHASAN

Bab ini menjelaskan hasil pengujian dan pembahasan dari hasil komputasi prediksi jumlah permintaan koran menggunakan metode ELM. Pengujian yang dilakukan meliputi: pengujian jumlah *neuron* pada *hidden layer*, pengujian fungsi aktivasi, pengujian perbandingan jumlah data *training* dan data *testing*, serta pengujian variasi fitur data.

6.1 Pengujian Jumlah *Neuron* Pada *Hidden Layer*

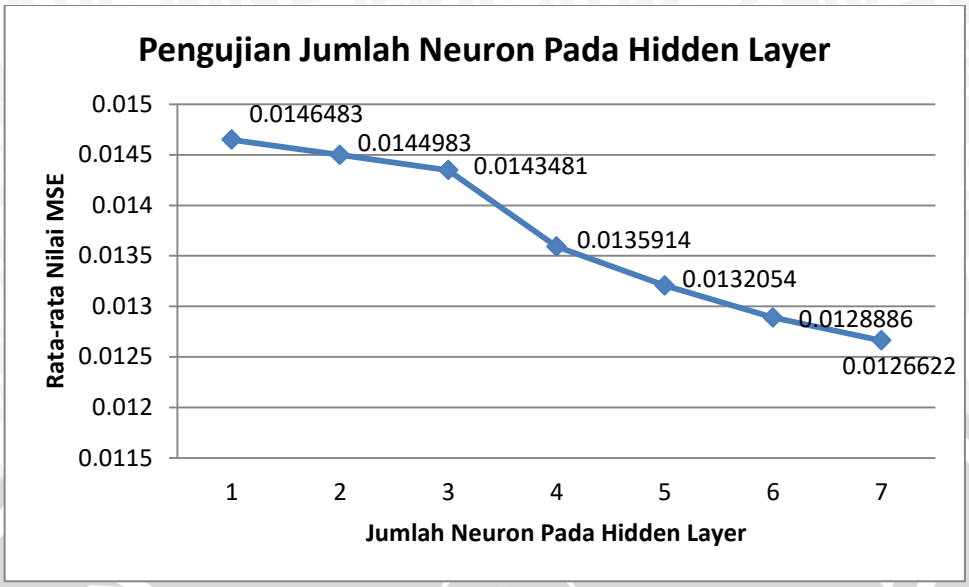
Pengujian jumlah *neuron* pada *hidden layer* bertujuan untuk mendapatkan hasil pengenalan yang lebih baik. Pengujian jumlah *neuron* pada *hidden layer* dievaluasi berdasarkan penentuan rata-rata nilai MSE. Komputasi ELM akan menghasilkan nilai berbeda setiap kali program dijalankan karena nilai *input weight* dan *bias* yang diinisialisasi secara random. Dengan demikian untuk memperoleh rata-rata nilai secara keseluruhan, dilakukan percobaan sebanyak 10 kali setiap jumlah *neuron* pada *hidden layer* yang diuji.

Banyaknya jumlah *neuron* pada *hidden layer* yang diuji coba adalah dari angka 1 sampai dengan 7. Inisialisasi parameter yang digunakan pada pengujian ini adalah menggunakan fungsi aktivasi sigmoid, perbandingan data *training* dan data *testing* sebesar 80%:20%, serta jumlah fitur data 4. Hasil pengujian jumlah *neuron* pada *hidden layer* dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Uji Coba Jumlah *Neuron* Pada *Hidden Layer*

Percobaan ke-*i*	Nilai MSE Pada Jumlah *Neuron* Pada *Hidden Layer*						
	1	2	3	4	5	6	7
1	0.014309	0.014611	0.014523	0.012705	0.013046	0.014534	0.014068
2	0.014128	0.01388	0.014782	0.013969	0.0129	0.013294	0.012551
3	0.014153	0.015661	0.013828	0.013441	0.013795	0.01259	0.013764
4	0.014257	0.015081	0.014198	0.012619	0.014384	0.012721	0.010409
5	0.014138	0.01427	0.014804	0.014402	0.01266	0.011377	0.013118
6	0.014214	0.014195	0.014703	0.01405	0.013309	0.013661	0.011452
7	0.01423	0.014133	0.013783	0.013729	0.012743	0.012247	0.013537
8	0.014397	0.014803	0.013842	0.014653	0.012828	0.012289	0.012765
9	0.014009	0.014253	0.014515	0.012769	0.013812	0.012975	0.012369
10	0.018648	0.014096	0.014503	0.013577	0.012577	0.013198	0.012589
Rata-Rata Nilai MSE	0.014648	0.014498	0.014348	0.013591	0.013205	0.012889	0.012662

Grafik hasil pengujian jumlah *neuron* pada *hidden layer* ditunjukkan pada Gambar 6.1 sebagai berikut:



Gambar 6.1 Grafik Hasil Pengujian Jumlah *Neuron* Pada *Hidden Layer*

Hidden layer pada *Extreme Learning Machine* terdiri dari *neuron-neuron* yang merupakan unit yang melakukan proses perhitungan atau mengolah masukan menjadi keluaran. Selain itu, *hidden layer* memiliki parameter yang menghubungkan antar *layer* dengan nilai yang berbeda. Hal ini memungkinkan adanya perbedaan hasil yang dicapai oleh setiap unit *neuron.*

Berdasarkan grafik pada Gambar 6.1, semakin banyak jumlah *neuron* pada *hidden layer*, semakin kecil rata-rata nilai MSE yang didapatkan. Dari hasil pengujian yang telah dilakukan, didapatkan rata-rata nilai MSE sebesar 0.0126622 dengan jumlah *neuron* pada *hidden layer* sebanyak 7 (Hasil *input weight* dan *bias* pada jumlah *neuron* 7 ada pada Lampiran C.1). Jumlah *hidden neuron* yang semakin banyak akan membentuk banyak penghubung (*connector*) dengan *input layer* dan *output layer.* Kondisi ini memungkinkan unit pemroses pada sistem yang melakukan proses pembobotan untuk mengenali data memiliki kemampuan yang baik dengan semakin banyaknya pertimbangan keputusan yang dilakukan oleh *hidden neuron*.

6.2 Pengujian Fungsi Aktivasi

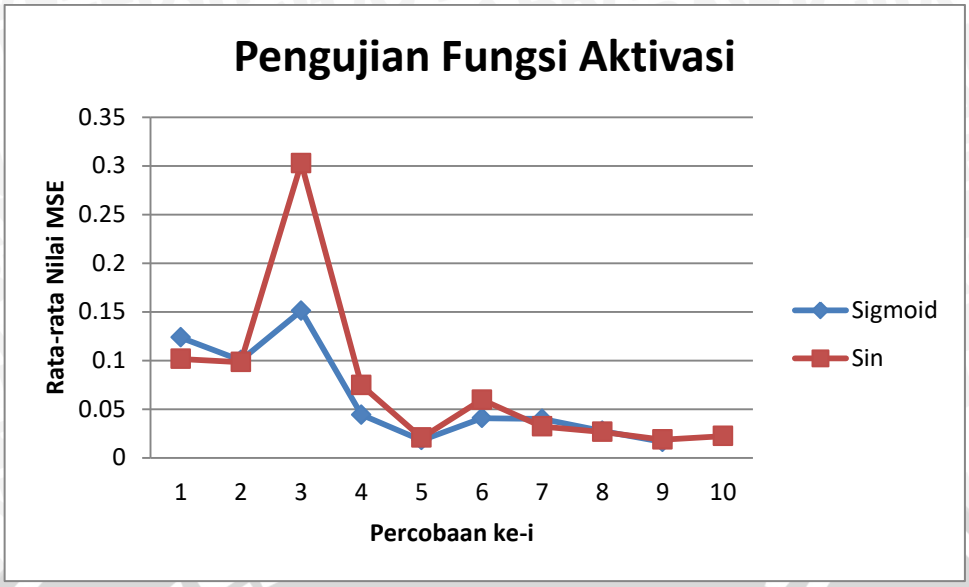
Pengujian fungsi aktivasi dilakukan untuk mentransformasikan suatu *input* menjadi suatu *output* tertentu. Pengujian fungsi aktivasi dievaluasi berdasarkan penentuan rata-rata nilai MSE. Komputasi ELM akan menghasilkan nilai berbeda setiap kali program dijalankan karena nilai *input weight* dan *bias* yang diinisialisasi secara random. Dengan demikian untuk memperoleh rata-rata nilai secara keseluruhan, dilakukan percobaan sebanyak 10 kali setiap fungsi aktivasi yang diuji.

Pengujian fungsi aktivasi dilakukan dengan uji coba menggunakan fungsi aktivasi *sigmoid* dan *sin* karena kedua fungsi tersebut yang paling sering digunakan pada permasalahan *forecasting* menggunakan metode ELM. Inisialisasi parameter yang digunakan pada pengujian ini adalah jumlah *hidden neuron* sebanyak 7, perbandingan data *training* dan data *testing* sebesar 80%:20%, serta jumlah fitur data 4. Hasil pengujian fungsi aktivasi dapat dilihat pada Tabel 6.2.

Tabel 6.2 Hasil Uji Coba Fungsi Aktivasi

Percobaan ke-*i*	**Nilai MSE Pada Fungsi Aktivasi**	
	Sigmoid	***Sin***
1	0.123702	0.101582
2	0.100221	0.09832
3	0.151109	0.302804
4	0.044093	0.074545
5	0.017981	0.02071
6	0.040795	0.059564
7	0.039839	0.032225
8	0.02753	0.026565
9	0.016131	0.018742
10	0.020481	0.02232
Rata-Rata Nilai MSE	0.054712	0.075738

Grafik hasil pengujian fungsi aktivasi ditunjukkan pada Gambar 6.2 sebagai berikut:



Gambar 6.2 Grafik Hasil Pengujian Fungsi Aktivasi

Berdasarkan Tabel 6.12, rata-rata nilai MSE terkecil yang didapatkan adalah menggunakan fungsi aktivasi *sigmoid*. Dari hasil pengujian yang telah dilakukan, didapatkan rata-rata nilai MSE sebesar 0.054712 menggunakan fungsi aktivasi *sigmoid* (Hasil *input weight* dan *bias* menggunakan fungsi aktivasi *sigmoid* ada pada Lampiran C.2). Hal ini menunjukkan bahwa dengan fungsi aktivasi *sigmoid* hasil prediksi yang didapat memiliki akurasi lebih baik. Berdasarkan grafik pada Gambar 6.2 juga menunjukkan nilai MSE yang cenderung rendah menggunakan fungsi aktivasi *sigmoid*. Karena menunjukkan hasil yang baik, maka untuk uji coba selanjutnya digunakan fungsi aktivasi *sigmoid*.

6.3 Pengujian Perbandingan Jumlah Data *Training* dan Data *Testing*

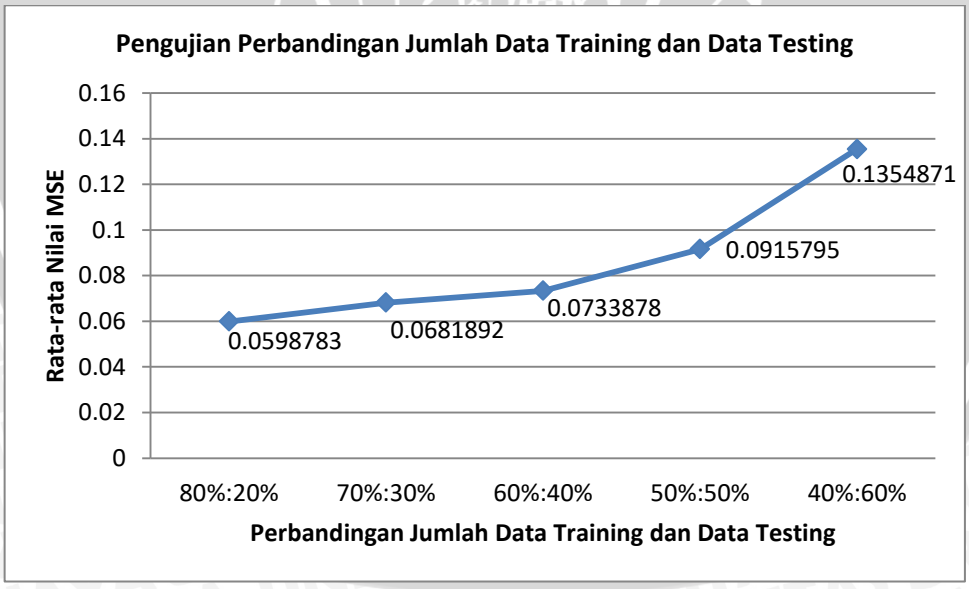
Pengujian perbandingan jumlah data *training* dan data *testing* ini dilakukan untuk mengetahui pengaruh dari perbandingan jumlah data *training* dengan data *testing* terhadap nilai MSE yang dihasilkan. Pengujian perbandingan jumlah data *training* dan data *testing* dievaluasi berdasarkan penentuan rata-rata nilai MSE. Komputasi ELM akan menghasilkan nilai berbeda setiap kali program dijalankan karena nilai *input weight* dan *bias* yang diinisialisasi secara random. Dengan demikian untuk memperoleh rata-rata nilai secara keseluruhan, dilakukan percobaan sebanyak 10 kali setiap perbandingan jumlah data yang diuji.

Pengujian perbandingan jumlah data *training* dan data *testing* terbaik dengan 5 jenis perbandingan data yaitu 80%:20%, 70%:30%, 60%:40%, 50%:50%, dan 40%:60%. Inisialisasi parameter yang digunakan pada pengujian ini adalah jumlah *hidden neuron* sebanyak 6 dan menggunakan fungsi aktivasi *sigmoid* serta jumlah fitur data 4. Hasil pengujian perbandingan jumlah data dapat dilihat pada Tabel 6.3.

Tabel 6.3 Hasil Uji Coba Perbandingan Jumlah Data *Training* dan Data *Testing*

Percobaan ke-*i*	Nilai MSE Pada Perbandingan Jumlah Data *Training* dan Data *Testing*				
	80%:20%	70%:30%	60%:40%	50%:50%	40%:60%
1	0.041529	0.059341	0.023321	0.161008	0.145111
2	0.033553	0.248393	0.074869	0.035493	0.605171
3	0.035794	0.018031	0.229965	0.191193	0.025854
4	0.061311	0.044461	0.133442	0.064577	0.151275
5	0.054461	0.111673	0.052044	0.335473	0.058001
6	0.082165	0.033803	0.014424	0.035395	0.1261
7	0.039163	0.051001	0.010225	0.015056	0.052478
8	0.03869	0.041799	0.120913	0.013938	0.099239
9	0.080258	0.036318	0.030266	0.014615	0.029094
10	0.131859	0.037072	0.044409	0.049047	0.062548
Rata-Rata Nilai MSE	0.059878	0.068189	0.073388	0.09158	0.135487

Grafik hasil pengujian perbandingan jumlah data *training* dan data *testing* ditunjukkan pada Gambar 6.3 sebagai berikut:



Gambar 6.3 Grafik Hasil Pengujian Perbandingan Jumlah Data *Training* dan Data *Testing*

Data *training* merupakan data yang digunakan untuk proses pembelajaran pada metode ELM. Data *training* pada proses *training* memberikan nilai *input weight, bias* dan nilai *output weight* yang akan dilanjutkan pada proses *testing*.

Data *testing* merupakan data uji yang akan digunakan untuk menghitung hasil ramalan dan kesalahan ramalan berdasarkan nilai MSE.

Berdasarkan grafik pada Gambar 6.3, nilai MSE terkecil didapatkan pada perbandingan jumlah data *training* sebanyak 80% dan data *testing* sebanyak 20% yaitu rata-rata nilai *error* sebesar 0.0598783 (Hasil *input weight* dan *bias* pada perbandingan jumlah data *training* 80% dan data *testing* 20% ada pada Lampiran C.3). Banyaknya jumlah data *training* dan data *testing* pada ELM ini berpengaruh terhadap nilai *error* yang dihasilkan. Hal ini disebabkan karena metode ELM merupakan metode pelatihan. Sehingga makin banyak data latih akan semakin baik hasil prediksi yang diperoleh.

6.4 Pengujian Variasi Fitur Data

Pengujian variasi fitur data dilakukan untuk mengetahui pengaruh dari jumlah fitur yang digunakan terhadap nilai MSE yang dihasilkan. Pengujian variasi fitur data dievaluasi berdasarkan penentuan rata-rata nilai MSE. Komputasi ELM akan menghasilkan nilai berbeda setiap kali program dijalankan karena nilai *input weight* dan *bias* yang diinisialisasi secara random. Dengan demikian untuk memperoleh rata-rata nilai secara keseluruhan, dilakukan percobaan sebanyak 10 kali setiap perbandingan jumlah data yang diuji.

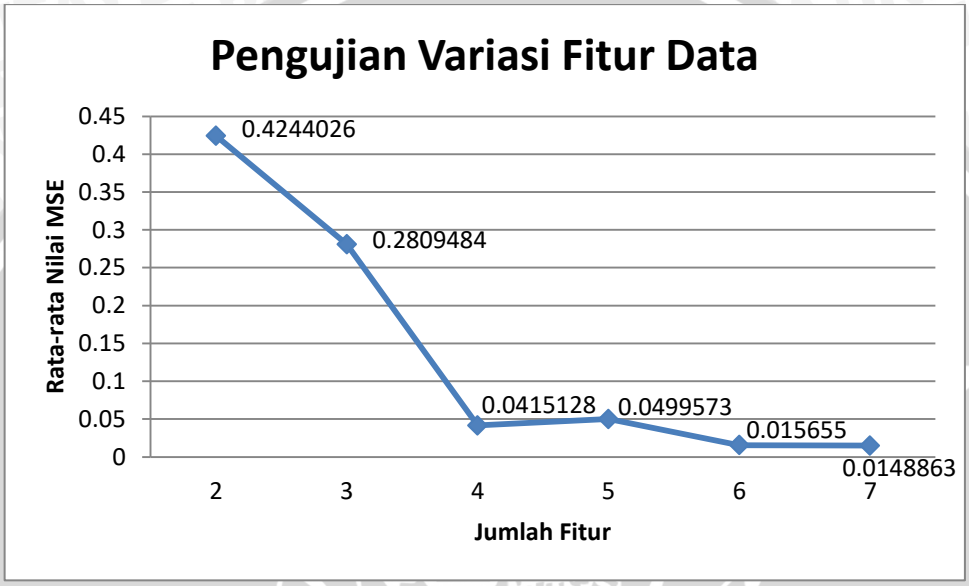
Pengujian variasi fitur data menggunakan jumlah fitur sebanyak 2, 3, 4, 5, 6, dan 7 fitur. Jumlah fitur pada pengujian ini maksudnya adalah perhitungan prediksi jumlah permintaan koran berdasarkan 2 hari sebelumnya, 3 hari sebelumnya, 4 hari sebelumnya, dan seterusnya. Inisialisasi parameter yang digunakan pada pengujian ini adalah jumlah *hidden neuron* sebanyak 6, perbandingan jumlah data *training* 80% dan data *testing* 20% serta menggunakan fungsi aktivasi *sigmoid*. Hasil pengujian variasi fitur data dapat dilihat pada Tabel 6.4.

Tabel 6.4 Hasil Uji Coba Variasi Fitur Data

Percobaan ke-*i*	Nilai MSE Pada Jumlah Fitur Data					
	2	**3**	**4**	**5**	**6**	**7**
1	0.440908	0.084786	0.200093	0.10484	0.01169	0.049821
2	0.278517	0.03393	0.310882	0.042751	0.009911	0.009423
3	0.308784	0.335672	0.153577	0.146066	0.009297	0.009687
4	0.130515	0.039892	0.149184	0.046249	0.010014	0.019943
5	0.299318	0.845882	0.076005	0.021101	0.009985	0.010669
6	0.41397	0.237771	0.065426	0.013655	0.009096	0.009659
7	0.893882	0.186579	0.039604	0.01696	0.009394	0.00938
8	0.764112	0.21372	0.056016	0.053048	0.043059	0.009637
9	0.00341	0.307721	0.254537	0.036953	0.00981	0.01057

Percobaan ke-*i*	Nilai MSE Pada Jumlah Fitur Data					
	2	**3**	**4**	**5**	**6**	**7**
10	0.71061	0.523531	0.101374	0.01795	0.034294	0.010074
Rata-Rata Nilai MSE	0.424403	0.280948	0.14067	0.049957	0.015655	0.014886

Grafik hasil pengujian variasi fitur data ditunjukkan pada Gambar 6.4 sebagai berikut:



Gambar 6.4 Grafik Hasil Pengujian Variasi Fitur Data

Jumlah fitur data menggunakan 2 sampai 7 karena untuk memprediksi jumlah permintaan data digunakan data historis mulai dari 2 hingga 7 hari sebelumnya. Variasi jumlah fitur bertujuan untuk mendapatkan jumlah yang tepat untuk mendapatkan hasil pengenalan yang lebih baik.

Berdasarkan grafik pada Gambar 6.4, nilai rata-rata MSE paling kecil didapatkan pada jumlah fitur sebanyak 7 sebesar 0.0148863 (Hasil *input weight* dan *bias* pada jumlah fitur 7 ada pada Lampiran C.4). Semakin banyak jumlah fitur belum tentu menghasilkan prediksi yang baik, karena bergantung juga dengan objek dan fitur yang digunakan untuk memprediksi.

6.5 Pengujian Berdasarkan Hari yang Sama

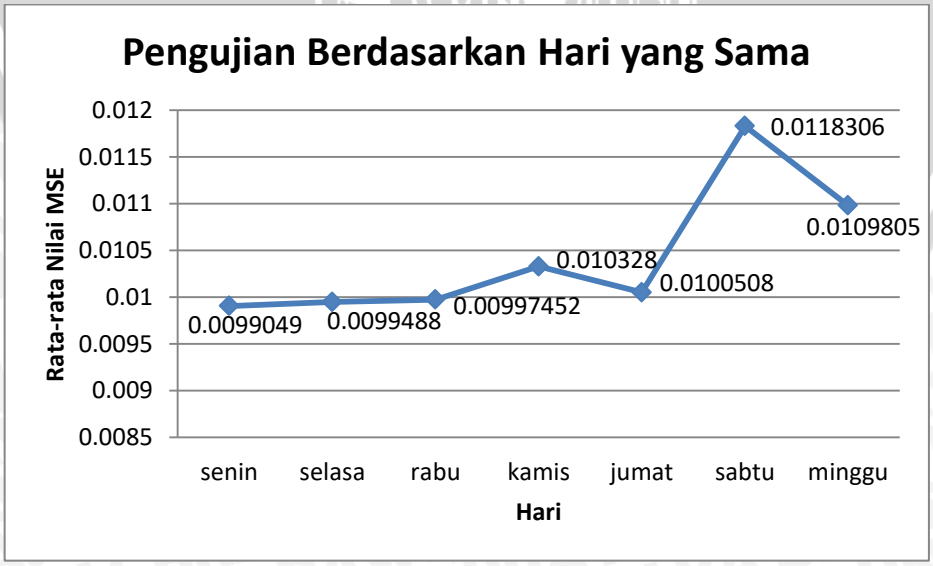
Pengujian berdasarkan hari dilakukan untuk mengetahui pengaruh hari yang digunakan terhadap nilai MSE yang dihasilkan. Pengujian berdasarkan hari dievaluasi berdasarkan penentuan rata-rata nilai MSE. Komputasi ELM akan menghasilkan nilai berbeda setiap kali program dijalankan karena nilai *input weight* dan *bias* yang diinisialisasi secara random. Dengan demikian untuk memperoleh rata-rata nilai secara keseluruhan, dilakukan percobaan sebanyak 10 kali setiap perbandingan jumlah data yang diuji.

Hari yang diuji yaitu senin, selasa, rabu, kamis, jumat, sabtu, dan minggu. Inisialisasi parameter yang digunakan pada pengujian ini adalah jumlah *hidden neuron* sebanyak 6, jumlah fitur 7, perbandingan jumlah data *training* 80% dan data *testing* 20% serta menggunakan fungsi aktivasi *sigmoid* (Contoh dataset pada Lampiran D). Hasil pengujian berdasarkan hari yang sama dapat dilihat pada Tabel 6.5.

Tabel 6.5 Hasil Uji Coba Berdasarkan Hari yang Sama

Percobaan ke-*i*	Nilai MSE Pada Hari						
	Senin	Selasa	Rabu	Kamis	Jumat	Sabtu	Minggu
1	0.010724	0.009572	0.001138	0.009593	0.010184	0.011595	0.01138
2	0.009653	0.010333	0.011143	0.010857	0.009287	0.011593	0.019916
3	0.009989	0.010608	0.010643	0.010901	0.010449	0.013215	0.010282
4	0.009857	0.010282	0.010783	0.012854	0.010104	0.011895	0.009509
5	0.00942	0.009406	0.010932	0.010589	0.01029	0.011767	0.010407
6	0.010046	0.009825	0.011083	0.00982	0.009233	0.01089	0.009107
7	0.009648	0.009971	0.011386	0.00938	0.010244	0.01112	0.00987
8	0.009893	0.009457	0.010901	0.009959	0.010862	0.013471	0.009351
9	0.010508	0.009926	0.010865	0.009801	0.009996	0.010802	0.009876
10	0.009311	0.010108	0.010871	0.009526	0.009859	0.011958	0.010107
Rata-Rata Nilai MSE	0.009905	0.009949	0.009975	0.010328	0.010051	0.011831	0.010981

Grafik hasil pengujian berdasarkan hari yang sama ditunjukkan pada Gambar 6.5 sebagai berikut:



Gambar 6.5 Grafik Hasil Pengujian Berdasarkan Hari yang Sama

Berdasarkan grafik pada Gambar 6.5, nilai rata-rata MSE paling kecil didapatkan pada hari Senin sebesar 0.0099049 (Hasil *input weight* dan *bias* pada hari Senin ada pada Lampiran C.5). Pengujian ini menunjukkan bahwa pada hari Senin jumlah permintaan cenderung stabil, bisa dilihat dari nilai rata-rata MSE yang kecil.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil pengujian dan pembahasan dari prediksi jumlah permintaan koran menggunakan metode *Extreme Learning Machine* (ELM) maka diperoleh kesimpulan sebagai berikut:

1. Metode ELM dapat digunakan untuk memprediksi jumlah permintaan koran yang menghasilkan nilai *error* kecil dengan melibatkan jumlah *neuron* pada *hidden layer*. *Hidden layer* pada ELM terdiri dari *neuron-neuron* yang berfungsi sebagai unit pemrosesan yang menghubungkan *input layer* dengan *output layer*. Banyaknya *hidden neuron* pada bagian *hidden layer* mempengaruhi hasil perhitungan dari suatu permasalahan dimana *hidden neuron* mengolah semua masukan yang nantinya menjadi keluaran.
2. Perbandingan jumlah data *training* dan data *testing* serta penambahan jumlah *neuron* pada *hidden layer* berpengaruh terhadap *output* prediksi yang dihasilkan. Tingkat kesalahan yang rendah berdasarkan pengujian yang dilakukan dengan nilai MSE yaitu sebesar 0.009311 menggunakan fungsi aktivasi *sigmoid*, menggunakan 7 fitur, perbandingan jumlah data *training* dan data *testing* yaitu 80%:20% serta jumlah *hidden neuron* sebanyak 7.

7.2 Saran

Penelitian tentang prediksi menggunakan metode *Extreme Learning Machine* (ELM) dapat dikembangkan dengan beberapa saran sebagai berikut:

1. Untuk penelitian selanjutnya, peneliti dapat menambahkan fitur lain yang merupakan faktor-faktor dalam prediksi jumlah permintaan koran. Hal ini bertujuan agar hasil prediksi lebih obyektif.
2. Pengembangan metode atau penggabungan metode lain dengan ELM seperti *Optimally Pruned Extreme Learning Machine* (OPELM) dapat digunakan pada penelitian selanjutnya untuk meningkatkan hasil prediksi.
3. Pada penelitian selanjutnya juga diharapkan peneliti dapat menambahkan untuk prediksi sampai beberapa hari ke depan, tidak hanya satu hari saja.

DAFTAR PUSTAKA

Agustina, Irwin D., Anggraeni, W., & Mukhlason, Ahmad, 2009. *Penerapan Metode Extreme Learning Machine untuk Peramalan Permintaan*. Skripsi S1. Institut Teknologi Sepuluh Nopember Surabaya. Tersedia di <http://digilib.its.ac.id/public/ITS-Undergraduate-9832-Paper.pdf> [Diakses 28 Januari 2016].

Atmojo, Tri B., Pulungan, R., & Syahputra, Hermawan, 2013. Pengembangan Model Peramalan Permintaan Kebutuhan Reseller Menggunakan Extreme Learning Machine Dalam Konteks Intelligent Warehouse Management System(IWMS). *Seminar Nasional Informatika(Semnaif 2013)*, Vol 1, p.258.

Chamidah, N., Wiharto, Salamah, Umi, 2012. Pengaruh Normalisasi Data Pada Jaringan Syaraf Tiruan Backpropagasi Gradient Descent Adaptive Gain(BPGDAG) untuk Klasifikasi. *Jurnal ITSmart*, Vol 1, p.28.

Danniel, Moehar. 2004. Pengantar Ekonomi Pertanian. Jakarta: Bumi Aksara.

Fardani, Delia P., Wuryanto, E., & Werdiningsih, Indah, 2015. Sistem Pendukung Keputusan Peramalan Jumlah Kunjungan Pasien Menggunakan Metode Extreme Learning Machine(Studi Kasus: Poli Gigi RSU Dr. Wahidin Sudiro Husodo Mojokerto). *Journal of Information Systems Engineering and Business Intelligence*, Vol 1, p.33.

Gilarso, T. 2004. Pengantar Ilmu Ekonomi Mikro. Yogyakarta: Kanisius.

Gunawan, Hayun. 2003. *Kajian Layout Cover Koran Pikiran Rakyat*. Skripsi S1. Universitas Komputer Indonesia. Tersedia di <http://elib.unikom.ac.id/files/disk1/563/jbptunikompp-gdl-neraagiant-28138-6-bab2-nera.pdf> [Diakses 3 Februari 2016].

Hajar, Ibnu. 2013. Fungsi Aktivasi Jaringan Syaraf Tiruan. Tersedia di <http://ibnu-h--fst10.web.unair.ac.id/artikel_detail-76049-Umum-Fungsi%20Aktivasi%20Jaringan%20Saraf%20Tiruan.html> [Diakses 17 Juni 2016].

Huang, G.B., Zhu, Q.Y., dan Siew, C.K. 2004. *Extreme Learning Machine : A New Learning Scheme of Feedforward neural Networks*. Proceedings of International Joint Conference on Neural Networks. Budapest, Hungary, 25-29 Juli 2004. Singapura: Nanyang Avenue.

Huang, G.B., Zhu, Q.Y., dan Siew, C.K. 2005. Extreme Learning Machine : Theory and applications. Elsevier science : Neurocomputing, 70(2006), p.489-501.

Hyndman, Rob J., Koehler, Anne B. 2005. Another look at measures of forecast accuracy. International Journal of Forecasting, 22, p.679-688.

Jawa Pos Radar Madura PT., 2016. *Profil Redaksi*. [Online] Radar Madura. Tersedia di <http://radarmadura.jawapos.com/profil> [Diakses 14 Mei 2016].

Jaya, Desi Indra. 2006. *Perancangan dan Pembuatan Perangkat Lunak Berbasis Jaringan Saraf Tiruan Untuk Memprediksi Harga Saham Dengan Menggunakan Backpropagation*. Skripsi S1. Universitas Kristen Petra. Tersedia di <http://dewey.petra.ac.id/catalog/ft_viewer.php?fname=jiunkpe/s1/info/2005/jiunkpe-ns-s1-2005-26401114-4881-backpropagation-chapter2.pdf> [Diakses 3 Februari 2016].

JPNN. 2015. *Inilah Empat Koran dengan Pembaca Terbanyak*. [Online] JPNN.com. Tersedia di <http://www.jpnn.com/read/2015/04/16/298324/Inilah-Empat-Koran-dengan-Pembaca-Terbanyak-> [Diakses 10 Februari 2016].

Khotimah, Bain K., Sari, Eka Mala R., & Yulianarta, Handry. 2010. Kinerja Metode Extreme Learning Machine(ELM) Pada Sistem Peramalan. *Jurnal Simantec*, Vol 1, p.186.

Liang, Nan-Ying, Huang, Guang-Bin, Saratchandran, P., & Sundararajan, N., 2006. *A Fast and Accurate Online Sequential Learning Algorithm for Feedforward Networks*. IEEE Transactions On Neural Networks, Vol 17(6), p.1411-1423.

Singh, R. & Balasundaram, S., 2007. *Application of Extreme Learning Machine Method for Time Series Analysis*. International Journal of Intelligent Technology, 2(4), p.256.

Sinuhaji, Ferdinand. 2009. *Jaringan Syaraf Tiruan Untuk Prediksi Keputusan Medis Pada Penyakit Asma*. Skripsi S1, Universitas Sumatera Utara. Medan.

Sun, Z.L., Choi, T.M., Au, K.F., dan Yu, Y. 2008. Sales Forecasting using Extreme Learning Machine with Application in Fashion Retailing. Elsevier Decision Support Systems, 46(2008), p.411-419.

Tabbel, John. 2003. *Karier Jurnalistik*. Semarang: Dhara Prize.

Wijono, Ari. 2016. Wawancara Jumlah Permintaan Koran Radar Madura (Lampiran A). Diwawancarai oleh Amalia Ramadhanty. Lawang, 23 April 2016.

Zhang, G., Pattuwo, B.E., dan Hu, M.Y. 1998. Forecasting with Artificial Neural Networks : The State of the Art. Elsevier International Journal of Forecasting, 14(1998), p.35-62.

Zumar, Dhorifi. 2009. *Konsumen Koran Terus Menyusut*. [Online]Mars Indonesia. Tersedia di <https://marsindonesia.com/> [Diakses 10 Februari 2016].

LAMPIRAN A HASIL WAWANCARA DENGAN JAWA POS RADAR MADURA

Berikut merupakan hasil wawancara dengan Ari Wijono selaku kepala Bagian Sirkulasi Radar Madura.

1. Bagaimana proses cetak koran Jawa Pos Radar Madura?
   Jawab: Jawa Pos Radar Madura diterbitkan oleh PT Intermedia Madura Pers. Proses cetak berada di PT Temprina Media Grafika Sumengko, Gresik.
2. Bagaimana proses distribusi koran Jawa Pos Radar Madura?
   Jawab: Seluruh wilayah Madura
3. Bagaimana menentukan jumlah permintaan koran Jawa Pos Radar Madura?
   Jawab: Jawa Pos Radar Madura memiliki bagian sirkulasi yang mendata jumlah permintaan agen yang bersifat langganan dan eceran. Setiap agen memiliki jatah sesuai dengan langganan yang mereka miliki. Agen langganan boleh memiiliki koran eceran dengan harga berbeda. Bisa juga dikira-kira dari jumlah penjualan hari sebelumnya dan minggu lalu.
4. Apakah jumlah produksi bergantung pada jumlah permintaan?
   Jawab: Permintaan langganan bersifat tetap, dan akan naik jika terdapat tambahan pelanggan. Sedangkan permintaan eceran itu fluktuatif. Sehingga sulit untuk memprediksi permintaan eceran.
5. Dari mana saja pemasukan dari Jawa Pos Radar Madura?
   Jawab: Dari penjualan koran, iklan-iklan dan koran digital Jawa Pos Radar Madura yang tersedia di situs www.radarmadura.co.id.
6. Faktor yang mempengaruhi jumlah produksi?
   Jawab: Potensi wilayah, jumlah penduduk, niat baca, produk pesaing yang sama juga mempengaruhi jumlah produksi.
7. Apa saja faktor saat mengalami kerugian?
   Jawab: Kerugian terjadi karena beberapa faktor antara lain, harga cetak lebih mahal dari harga jual, kiriman ada yang rusak seperti sobek atau basah dan keterlambatan pengiriman kerena beberapa faktor seperti mesin trouble, jalan macet, mobil angkut mogok dan terjadi kecelakaan.
8. Apa yang selama ini dilakukan Jawa Pos Radar Madura untuk mengatasi kerugian?
   Jawab: Pihak koran Radar Madura memilih dari jumlah distribusi yang memiliki jaringan yang luas, wilayah edar yang potensial, dan distribusi memiliki tenaga penjual yang handal.
9. Berapa batas tolerir jumlah kelebihan produksi?
   Jawab : ± 5%

LAMPIRAN B DATA PERMINTAAN KORAN RADAR MADURA TAHUN 2015

Data ke-	Tanggal	Jumlah Permintaan
1	01/01/15	7899
2	02/01/15	7961
3	03/01/15	7560
4	04/01/15	8166
5	05/01/15	7898
6	06/01/15	7911
7	07/01/15	7987
8	08/01/15	7947
9	09/01/15	7836
10	10/01/15	8206
11	11/01/15	8052
12	12/01/15	8072
13	13/01/15	7906
14	14/01/15	7990
15	15/01/15	7962
16	16/01/15	7922
17	17/01/15	8237
18	18/01/15	8066
19	19/01/15	7933
20	20/01/15	8071
21	21/01/15	8286
22	22/01/15	8165
23	23/01/15	7960
24	24/01/15	8325
25	25/01/15	8131
26	26/01/15	7996
27	27/01/15	8118
28	28/01/15	8063
29	29/01/15	8031
30	30/01/15	8062
31	31/01/15	8277
32	01/02/15	8001
33	02/02/15	7876
34	03/02/15	7902
35	04/02/15	8247
36	05/02/15	8344
37	06/02/15	8387
38	07/02/15	8463

39	08/02/15	8049
40	09/02/15	8521
41	10/02/15	8277
42	11/02/15	8155
43	12/02/15	8360
44	13/02/15	8255
45	14/02/15	8171
46	15/02/15	8002
47	16/02/15	7869
48	17/02/15	8363
49	18/02/15	8097
50	19/02/15	8220
51	20/02/15	7912
52	21/02/15	7948
53	22/02/15	7804
54	23/02/15	8132
55	24/02/15	8371
56	25/02/15	8539
57	26/02/15	8471
58	27/02/15	8584
59	28/02/15	8609
60	01/03/15	8438
61	02/03/15	8821
62	03/03/15	10190
63	04/03/15	8548
64	05/03/15	8631
65	06/03/15	8586
66	07/03/15	8811
67	08/03/15	8350
68	09/03/15	8350
69	10/03/15	8608
70	11/03/15	8766
71	12/03/15	9167
72	13/03/15	9506
73	14/03/15	8731
74	15/03/15	8613
75	16/03/15	8593
76	17/03/15	8578
77	18/03/15	8361
78	19/03/15	8495
79	20/03/15	8472
80	21/03/15	8447
81	22/03/15	8359

82	23/03/15	8477
83	24/03/15	8418
84	25/03/15	8604
85	26/03/15	9007
86	27/03/15	8627
87	28/03/15	8947
88	29/03/15	8504
89	30/03/15	8583
90	31/03/15	8864
91	01/04/15	8718
92	02/04/15	8262
93	03/04/15	8456
94	04/04/15	8504
95	05/04/15	8400
96	06/04/15	8648
97	07/04/15	8296
98	08/04/15	8615
99	09/04/15	8357
100	10/04/15	8326
101	11/04/15	8646
102	12/04/15	8277
103	13/04/15	8698
104	14/04/15	8648
105	15/04/15	8542
106	16/04/15	8357
107	17/04/15	8247
108	18/04/15	8493
109	19/04/15	8285
110	20/04/15	8361
111	21/04/15	8591
112	22/04/15	8828
113	23/04/15	8453
114	24/04/15	8491
115	25/04/15	8536
116	26/04/15	8242
117	27/04/15	8404
118	28/04/15	8563
119	29/04/15	8692
120	30/04/15	8976
121	01/05/15	8488
122	02/05/15	8439
123	03/05/15	8181
124	04/05/15	8747

125	05/05/15	8785
126	06/05/15	9867
127	07/05/15	9333
128	08/05/15	9769
129	09/05/15	12476
130	10/05/15	9465
131	11/05/15	8296
132	12/05/15	8225
133	13/05/15	8580
134	14/05/15	8189
135	15/05/15	8300
136	16/05/15	8314
137	17/05/15	8361
138	18/05/15	8726
139	19/05/15	9068
140	20/05/15	8543
141	21/05/15	8677
142	22/05/15	8377
143	23/05/15	8447
144	24/05/15	8369
145	25/05/15	8706
146	26/05/15	8149
147	27/05/15	8582
148	28/05/15	8698
149	29/05/15	8885
150	30/05/15	8375
151	31/05/15	8193
152	01/06/15	8374
153	02/06/15	8106
154	03/06/15	8250
155	04/06/15	8404
156	05/06/15	8709
157	06/06/15	8410
158	07/06/15	8118
159	08/06/15	9185
160	09/06/15	8452
161	10/06/15	8671
162	11/06/15	8830
163	12/06/15	8462
164	13/06/15	9307
165	14/06/15	8020
166	15/06/15	10086
167	16/06/15	8469

168	17/06/15	8432
169	18/06/15	8265
170	19/06/15	8028
171	20/06/15	8162
172	21/06/15	8071
173	22/06/15	8150
174	23/06/15	8329
175	24/06/15	8239
176	25/06/15	8412
177	26/06/15	7964
178	27/06/15	8176
179	28/06/15	7982
180	29/06/15	8000
181	30/06/15	8256
182	01/07/15	7708
183	02/07/15	7681
184	03/07/15	7604
185	04/07/15	7837
186	05/07/15	7514
187	06/07/15	7707
188	07/07/15	7675
189	08/07/15	7892
190	09/07/15	8287
191	10/07/15	8208
192	11/07/15	7822
193	12/07/15	7575
194	13/07/15	7654
195	14/07/15	7655
196	15/07/15	7567
197	16/07/15	7358
198	19/07/15	6790
199	20/07/15	7245
200	21/07/15	7293
201	22/07/15	7495
202	23/07/15	7481
203	24/07/15	6824
204	25/07/15	7789
205	26/07/15	7452
206	27/07/15	7660
207	28/07/15	8423
208	29/07/15	7795
209	30/07/15	8541
210	31/07/15	7636

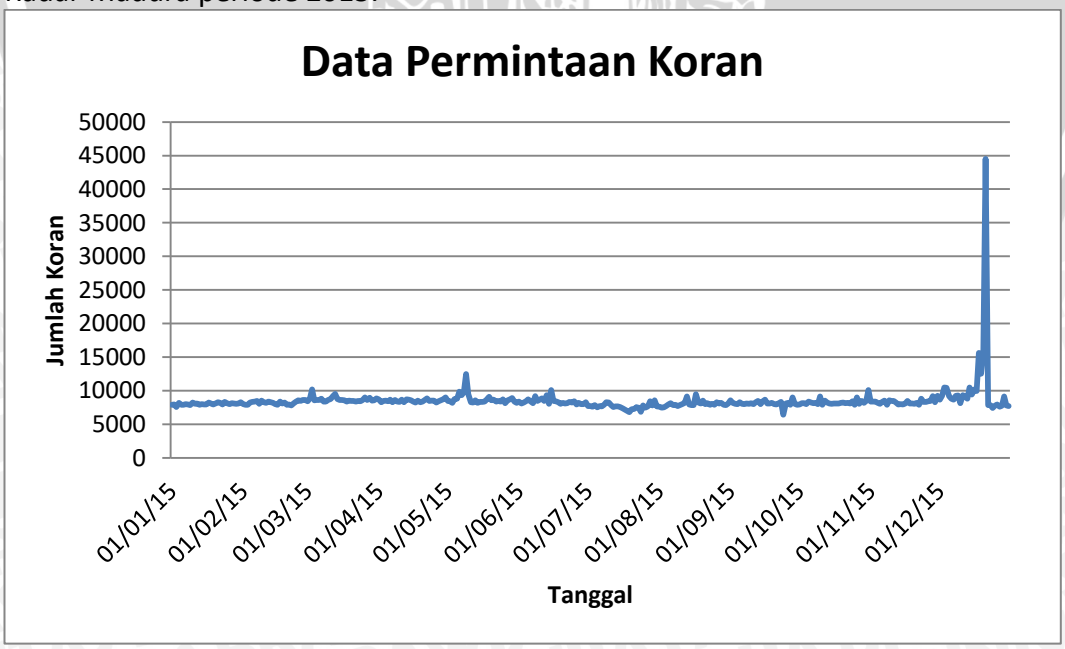
211	01/08/15	7625
212	02/08/15	7469
213	03/08/15	7530
214	04/08/15	7704
215	05/08/15	7925
216	06/08/15	8113
217	07/08/15	7845
218	08/08/15	7908
219	09/08/15	7683
220	10/08/15	7828
221	11/08/15	7995
222	12/08/15	8195
223	13/08/15	9122
224	14/08/15	7939
225	15/08/15	7851
226	16/08/15	7834
227	17/08/15	9485
228	18/08/15	8226
229	19/08/15	8093
230	20/08/15	8504
231	21/08/15	7987
232	22/08/15	8128
233	23/08/15	7910
234	24/08/15	8031
235	25/08/15	7887
236	26/08/15	8267
237	27/08/15	8101
238	28/08/15	8201
239	29/08/15	7893
240	30/08/15	7822
241	31/08/15	8089
242	01/09/15	8576
243	02/09/15	8161
244	03/09/15	8048
245	04/09/15	7998
246	05/09/15	8266
247	06/09/15	8019
248	07/09/15	7993
249	08/09/15	8095
250	09/09/15	8043
251	10/09/15	8128
252	11/09/15	7987
253	12/09/15	8252

254	13/09/15	8475
255	14/09/15	7987
256	15/09/15	8439
257	16/09/15	8637
258	17/09/15	8101
259	18/09/15	8069
260	19/09/15	8192
261	20/09/15	7970
262	21/09/15	7931
263	22/09/15	8132
264	23/09/15	8337
265	24/09/15	6407
266	25/09/15	8069
267	26/09/15	8155
268	27/09/15	7889
269	28/09/15	8975
270	29/09/15	8020
271	30/09/15	7901
272	01/10/15	7920
273	02/10/15	8122
274	03/10/15	8140
275	04/10/15	7981
276	05/10/15	8369
277	06/10/15	8231
278	07/10/15	8145
279	08/10/15	8158
280	09/10/15	8008
281	10/10/15	9132
282	11/10/15	7895
283	12/10/15	8499
284	13/10/15	8229
285	14/10/15	8076
286	15/10/15	8049
287	16/10/15	8092
288	17/10/15	8072
289	18/10/15	8061
290	19/10/15	8181
291	20/10/15	8240
292	21/10/15	8145
293	22/10/15	8182
294	23/10/15	8104
295	24/10/15	8411
296	25/10/15	7889

297	26/10/15	8994
298	27/10/15	8040
299	28/10/15	8475
300	29/10/15	8169
301	30/10/15	8445
302	31/10/15	10084
303	01/11/15	8358
304	02/11/15	8361
305	03/11/15	8346
306	04/11/15	8178
307	05/11/15	8055
308	06/11/15	8334
309	07/11/15	8507
310	08/11/15	7898
311	09/11/15	8549
312	10/11/15	8442
313	11/11/15	8452
314	12/11/15	8219
315	13/11/15	7942
316	14/11/15	8011
317	15/11/15	7931
318	16/11/15	8063
319	17/11/15	8460
320	18/11/15	8084
321	19/11/15	8079
322	20/11/15	8017
323	21/11/15	8163
324	22/11/15	7905
325	23/11/15	8790
326	24/11/15	8332
327	25/11/15	8325
328	26/11/15	8408
329	27/11/15	8478
330	28/11/15	9194
331	29/11/15	8282
332	30/11/15	9244
333	01/12/15	8632
334	02/12/15	9225
335	03/12/15	10452
336	04/12/15	10414
337	05/12/15	9190
338	06/12/15	8798
339	07/12/15	8648

340	08/12/15	9213
341	09/12/15	9279
342	10/12/15	8142
343	11/12/15	9320
344	12/12/15	9138
345	13/12/15	8799
346	14/12/15	10467
347	15/12/15	9404
348	16/12/15	10193
349	17/12/15	9949
350	18/12/15	15594
351	19/12/15	12534
352	20/12/15	15842
353	21/12/15	44520
354	22/12/15	7849
355	23/12/15	7864
356	24/12/15	7405
357	25/12/15	7752
358	26/12/15	7957
359	27/12/15	7620
360	28/12/15	7760
361	29/12/15	9148
362	30/12/15	7810
363	31/12/15	7704

Berikut merupakan grafik data jumlah permintaan koran harian Jawa Pos Radar Madura periode 2015.



LAMPIRAN C HASIL RANDOM *INPUT WEIGHT* DAN *BIAS*

C.1 Pengujian Jumlah *Neuron* Pada *Hidden Layer*

Jumlah *Hidden Neuron* 7

- Percobaan ke-1

Input Weight				Bias
1	2	3	4	
-0.46086	-0.98854	0.477612	-0.54545	-0.52427
0.629247	0.166546	0.061604	0.02961	-0.66269
-0.62854	-0.1177	0.338459	0.384733	0.891898
-0.8982	-0.85378	-0.69124	0.357768	0.86633
-0.60211	0.942375	0.400631	-0.38509	0.998847
0.853194	0.11537	0.325776	-0.47196	-0.12156
-0.3466	-0.7229	0.376051	-0.00686	-0.12361

- Percobaan ke-2

Input Weight				Bias
1	2	3	4	
-0.85014	0.773255	-0.86651	-0.59999	0.341822
0.479742	-0.20561	0.0698	0.215183	-0.47196
-0.76969	0.937079	0.666234	0.341159	-0.08963
0.140324	-0.11583	-0.20734	-0.4861	-0.25197
0.456994	0.332164	0.199427	0.581906	-0.91392
0.168663	-0.8598	0.314691	-0.91304	0.685947
0.391544	0.724659	-0.41769	0.758261	0.039912

- Percobaan ke-3

Input Weight				Bias
1	2	3	4	
0.889428	0.834378	0.441186	0.281047	0.314432
-0.48267	0.631791	-0.41578	-0.4831	0.896129
0.723733	-0.57812	-0.57595	0.857138	-0.97284
-0.82119	-0.90546	-0.47343	0.779892	-0.19797
-0.26473	-0.18358	-0.05454	-0.48724	-0.23923
0.653562	-0.32466	-0.52644	0.730544	-0.12373
0.306695	-0.31102	-0.59997	-0.17678	-0.92625

- Percobaan ke-4

Input Weight				Bias
1	2	3	4	
-0.38499	0.8336	-0.50609	0.32985	-0.03739
-0.44627	0.824984	-0.99078	0.002549	0.037986
-0.89403	-0.69883	0.314057	0.52614	0.848512
-0.93348	0.366453	-0.93745	0.526938	0.394876

Input Weight				Bias
1	2	3	4	
-0.41683	-0.02578	0.524038	-0.11679	-0.39037
0.57299	0.344253	0.710255	0.39115	-0.22937
-0.30231	0.38779	-0.63544	0.508478	-0.58825

- Percobaan ke-5

Input Weight				Bias
1	2	3	4	
-0.8107	-0.4233	0.42877	0.293344	0.68778
0.865358	0.052476	-0.95712	-0.42114	0.373093
0.879121	0.112648	-0.97801	0.611388	-0.71324
-0.99468	0.101443	-0.94742	-0.11426	-0.07537
-0.30915	0.799155	0.067755	0.918571	-0.00135
-0.7352	-0.10785	0.696628	-0.62098	0.998887
0.164187	-0.84397	-0.38857	-0.20427	0.585572

- Percobaan ke-6

Input Weight				Bias
1	2	3	4	
0.317939	-0.21578	0.052447	-0.80481	-0.14683
-0.83895	-0.02631	-0.03135	-0.3321	0.966439
-0.33088	-0.59212	-0.87661	-0.10498	-0.55096
-0.77237	0.37985	0.331522	-0.30414	0.833429
0.082956	-0.23812	-0.4259	0.186125	-0.84743
0.663214	-0.46152	0.86081	0.786866	0.279
0.109826	0.633566	-0.03917	-0.88225	0.710482

- Percobaan ke-7

Input Weight				Bias
1	2	3	4	
0.767153	-0.32308	-0.24252	-0.51414	-0.68273
0.016434	-0.405	-0.01404	0.243852	-0.35627
0.548091	-0.65955	0.887313	-0.16842	-0.59968
-0.86681	-0.54255	0.304211	0.472471	0.566869
-0.41677	0.235856	-0.38897	-0.29447	0.68441
0.596001	-0.8965	0.59288	0.951566	-0.30147
-0.93824	-0.13395	0.058726	0.29057	0.05904

- Percobaan ke-8

Input Weight				Bias
1	2	3	4	
0.119743	-0.6166	-0.19222	0.362544	0.722579
0.471608	-0.47361	0.97298	0.047429	-0.23913
-0.74173	0.863887	0.149727	-0.83787	-0.03615

Input Weight				Bias
1	2	3	4	
-0.69222	-0.18747	0.001779	-0.80125	-0.01725
0.574336	0.038041	-0.33659	0.672439	0.865461
-0.20871	-0.70882	0.803593	-0.29895	-0.73434
0.288614	0.810096	-0.4865	-0.71012	0.333

- Percobaan ke-9

Input Weight				Bias
1	2	3	4	
-0.01712	0.144393	-0.41913	-0.3931	0.506953
0.789932	0.721377	-0.56615	-0.20205	-0.59267
-0.38887	0.047156	-0.05466	-0.6406	0.120127
0.621861	-0.17365	0.612247	0.280863	-0.19626
0.969525	-0.43818	0.631454	-0.43188	-0.25644
-0.40745	-0.87376	-0.6783	0.973447	-0.50428
0.709092	0.157434	-0.28699	0.705992	0.80294

- Percobaan ke-10

Input Weight				Bias
1	2	3	4	
-0.2158	-0.90759	0.206411	-0.31035	0.326499
0.911017	-0.09214	-0.96229	-0.07408	-0.03943
-0.67389	0.68073	0.874962	0.940597	0.459958
-0.84188	-0.12507	0.267311	-0.08369	-0.50008
-0.0662	0.079332	-0.46667	0.270905	0.405797
0.307958	-0.97976	0.985035	-0.27615	0.806669
0.462434	0.659151	0.486756	-0.77599	0.933233

C.2 Pengujian Fungsi Aktivasi

Fungsi Aktivasi *Sigmoid*

- Percobaan ke-1

Input Weight				Bias
1	2	3	4	
0.418469	0.56456	0.268537	-0.61028	-0.58265
-0.43634	0.820968	-0.2861	0.850639	0.537062
-0.66434	-0.50952	0.409599	0.105965	0.822816
-0.44705	-0.91885	-0.17648	-0.14615	-0.01496
-0.04498	-0.43115	-0.98344	-0.41335	0.191747
0.673777	-0.90701	0.264388	-0.80637	-0.14582
-0.59073	-0.32263	-0.89928	-0.5741	0.667965

- Percobaan ke-2

Input Weight				Bias
1	2	3	4	
-0.09158	0.876588	0.256038	-0.5025	-0.06733
0.453675	-0.56645	-0.06073	-0.77999	0.675508
0.738757	0.142116	0.934853	0.103338	0.649719
0.645141	-0.11383	-0.85287	-0.15607	-0.19654
0.632527	-0.56392	-0.06287	0.148101	0.961547
-0.13875	-0.45323	0.769905	-0.66009	0.303147
0.675642	-0.82135	-0.13028	-0.30041	-0.11782

- Percobaan ke-3

Input Weight				Bias
1	2	3	4	
-0.48754	0.788871	0.814221	0.736974	0.458982
0.617417	-0.16637	0.660696	-0.20168	-0.3208
0.751314	0.397045	-0.91045	-0.08292	0.388925
0.094771	-0.30395	0.672321	-0.13935	0.256248
0.081225	0.413091	0.304022	0.655641	0.105461
0.27383	0.780448	-0.26203	0.91587	0.70696
0.391567	0.981082	0.944686	-0.44988	0.862333

- Percobaan ke-4

Input Weight				Bias
1	2	3	4	
-0.76432	-0.48174	0.376711	0.109517	-0.84561
-0.37251	0.124427	-0.3557	-0.52113	0.521963
0.685908	-0.94975	-0.82511	0.963565	0.397623
-0.76705	0.188215	-0.7878	0.177535	0.655371
0.940263	0.114535	-0.56097	-0.6472	0.59017
-0.97168	0.84809	0.164069	0.561089	0.738663
0.492605	-0.67938	0.861458	-0.9912	-0.90281

- Percobaan ke-5

Input Weight				Bias
1	2	3	4	
-0.69724	0.327772	0.956892	0.543795	-0.29763
0.073609	0.061604	0.147974	0.723927	0.150307
0.273136	0.892363	0.910863	0.707971	-0.55739
-0.86628	-0.29814	0.791108	-0.91839	0.176518
0.501597	0.311457	-0.67497	0.545795	-0.90677
-0.28875	0.894018	-0.20635	-0.57507	-0.63665
-0.15746	0.238491	0.070209	0.850427	-0.08065

- Percobaan ke-6

Input Weight				Bias
1	2	3	4	
0.162961	-0.00784	0.923495	-0.51096	0.68584
-0.36604	0.160926	-0.87076	0.77539	0.813395
0.132138	0.257702	0.68587	0.416411	-0.78404
0.44152	-0.524	0.46512	0.131817	-0.59768
-0.79246	0.777976	-0.28736	-0.84505	0.924266
0.145089	-0.51848	0.203557	-0.42428	-0.29085
0.189921	-0.73221	0.677561	0.072525	0.63346

- Percobaan ke-7

Input Weight				Bias
1	2	3	4	
0.6231	0.689402	-0.45628	0.781825	0.813152
-0.15695	0.891349	-0.45715	-0.77162	0.468706
-0.00851	0.326248	0.843865	-0.25657	-0.16573
0.980828	0.27078	-0.77432	-0.50182	0.649678
0.199674	0.541779	-0.09444	0.62102	0.300054
-0.84554	0.440521	-0.65187	0.689625	0.270789
0.840324	-0.53006	-0.80214	0.635967	-0.83584

- Percobaan ke-8

Input Weight				Bias
1	2	3	4	
0.884603	-0.17759	-0.78824	-0.90196	0.711576
-0.98686	0.552491	0.372308	0.323377	0.717507
-0.93137	0.648254	-0.46351	0.482201	0.307498
-0.12709	0.521473	-0.67009	0.88236	0.806185
-0.97431	-0.3837	-0.25407	0.192128	0.09757
-0.37874	0.130972	-0.12395	-0.3403	-0.76043
-0.32967	0.062008	-0.0013	-0.18974	-0.03939

- Percobaan ke-9

Input Weight				Bias
1	2	3	4	
0.845696	0.149456	-0.93682	-0.56591	-0.96945
0.580407	-0.09485	0.372081	-0.5118	-0.01262
-0.94389	-0.76133	0.150723	-0.46984	-0.72654
0.12208	0.813181	-0.94054	0.822109	-0.23986
-0.68279	-0.16061	-0.96782	0.139367	0.41871
0.272121	-0.83519	-0.79981	-0.43528	0.673025
0.273103	-0.3028	-0.13086	0.911422	0.34108

- Percobaan ke-10

Input Weight				Bias
1	2	3	4	
-0.18783	0.20383	0.562755	0.141513	-0.27109
-0.48892	-0.88978	0.974416	0.05277	-0.80444
-0.31087	-0.26897	0.694053	-0.23206	-0.11525
-0.42612	0.633324	-0.21022	-0.82541	-0.06826
-0.152	0.608441	0.74849	-0.52102	0.603419
-0.62731	-0.35487	0.301919	0.29781	-0.61555
-0.58892	0.550768	0.099677	0.968715	-0.18153

C.3 Pengujian Perbandingan Jumlah Data *Training* dan Data *Testing*

80%:20%

- Percobaan ke-1

Input Weight				Bias
1	2	3	4	
0.512131	-0.39352	-0.91998	0.622822	-0.0099
-0.74513	-0.72005	-0.49237	-0.04692	-0.76502
-0.76097	0.221518	-0.27272	0.489448	0.045269
-0.43392	-0.9252	0.047356	-0.97046	-0.27019
0.848933	-0.26388	0.380323	0.834477	0.582671
-0.00944	-0.36526	0.062134	-0.70818	0.452997
0.312755	0.01851	-0.73975	0.122732	-0.27145

- Percobaan ke-2

Input Weight				Bias
1	2	3	4	
0.58429	-0.87611	0.440317	-0.15637	0.664072
0.181073	0.881254	-0.08168	0.96338	0.531032
0.385944	0.673884	0.207402	-0.62156	-0.73299
0.694529	-0.93922	0.640722	-0.88785	0.879203
0.415984	0.856291	0.640051	-0.10479	-0.31146
-0.68433	-0.53101	0.790996	0.290522	0.323715
-0.67799	0.192996	-0.4505	0.550041	-0.68036

- Percobaan ke-3

Input Weight				Bias
1	2	3	4	
0.841419	0.998269	0.536107	0.70062	-0.38745
0.181423	0.601162	0.562885	0.630176	0.908048
-0.37434	0.03163	0.586207	-0.77453	0.489536
-0.36174	0.970936	-0.70273	0.304073	-0.36663
-0.44441	0.839364	0.999939	-0.54105	-0.82482
-0.19872	0.320252	0.669395	-0.37285	0.575436

Input Weight				Bias
1	2	3	4	
0.81643	-0.78208	0.331296	-0.82473	-0.94417

- Percobaan ke-4

Input Weight				Bias
1	2	3	4	
0.214382	-0.71073	0.544052	0.924088	-0.98752
-0.16331	-0.56082	-0.35868	0.798736	0.698863
-0.81744	0.910283	0.732695	0.940817	-0.02414
0.475725	0.127097	-0.45475	0.450165	-0.70171
-0.84113	-0.44316	0.652701	0.532347	-0.83387
0.999081	-0.23099	0.546097	-0.28656	-0.9002
-0.46275	-0.66625	-0.70312	0.754195	0.531815

- Percobaan ke-5

Input Weight				Bias
1	2	3	4	
-0.15689	-0.18737	0.609108	-0.72307	0.97852
-0.27297	0.670162	-0.50371	-0.66226	-0.73684
-0.08692	0.713291	0.102784	0.858357	-0.97665
-0.11319	-0.07148	-0.35079	0.09348	0.801068
0.159177	-0.19534	-0.12565	-0.47659	-0.74833
-0.84478	0.007785	0.402351	0.89617	0.755569
0.559486	-0.20128	0.997858	-0.23046	0.126947

- Percobaan ke-6

Input Weight				Bias
1	2	3	4	
0.0262	0.420504	-0.61851	0.814351	-0.17714
-0.809	0.3144	-0.55342	0.083999	-0.27443
0.974946	-0.46292	0.556794	-0.43803	-0.75247
-0.2707	0.730878	0.068566	0.523723	-0.3008
0.254407	0.649839	-0.31935	-0.16394	0.863499
0.07917	-0.58891	0.411893	-0.9664	-0.23625
-0.43976	-0.59573	0.344529	-0.01735	-0.0616

- Percobaan ke-7

Input Weight				Bias
1	2	3	4	
-0.83328	-0.42806	0.365437	-0.58156	-0.58934
-0.07744	-0.17885	-0.70516	-0.03768	-0.40264
0.138724	-0.60161	0.821075	0.395848	-0.37227
0.211374	-0.02066	0.547526	0.397518	-0.92458
0.374511	-0.42979	0.6577	0.884323	-0.86721

Input Weight				Bias
1	2	3	4	
0.946967	-0.37756	-0.1373	0.617883	0.767953
0.292776	-0.32205	-0.19639	0.934814	0.216449

- Percobaan ke-8

Input Weight				Bias
1	2	3	4	
-0.31373	0.462043	0.572099	0.672319	0.115551
0.852228	0.969002	-0.846	-0.78338	-0.58275
0.804864	-0.48213	0.266277	-0.46446	-0.55192
-0.08573	0.643879	0.046686	0.574912	0.23332
-0.67088	0.208582	-0.57572	-0.3779	0.43092
-0.46568	0.416783	0.428555	0.234265	-0.96289
-0.72592	-0.21079	0.691384	0.554255	-0.25427

- Percobaan ke-9

Input Weight				Bias
1	2	3	4	
0.143305	-0.05683	0.357614	-0.68311	-0.94913
-0.27172	0.795	0.371554	-0.44954	-0.92039
0.749793	0.45695	-0.05975	0.416875	0.790623
0.171665	0.732912	-0.30769	0.995272	-0.39966
-0.66195	-0.49398	0.75965	-0.37698	0.904724
0.758127	-0.72362	-0.28252	0.800619	0.039613
-0.73004	0.543038	-0.05914	-0.72226	-0.40693

- Percobaan ke-10

Input Weight				Bias
1	2	3	4	
-0.55969	0.978225	-0.97058	-0.63139	-0.6106
-0.77214	-0.74501	-0.49528	0.93975	-0.87653
-0.593	0.61513	0.158129	0.988218	-0.7671
-0.3555	-0.26309	-0.04761	0.81767	0.82895
-0.93701	0.961374	-0.91826	0.014133	0.268358
0.991002	0.201566	0.808492	-0.93335	-0.17595
-0.76543	0.566906	-0.47437	0.040141	-0.39262

C.4 Pengujian Variasi Fitur Data

7 fitur

- Percobaan 1

Input Weight							Bias
1	2	3	4	5	6	7	
0.629266	-0.36849	0.768734	-0.02735	0.660002	0.260491	0.756161	0.394643
0.005211	0.723147	0.148444	-0.91715	-0.52135	0.6752	0.900358	0.341519

Input Weight							Bias
1	2	3	4	5	6	7	
-0.28789	0.468221	0.889343	-0.5264	-0.6261	-0.66663	0.4664	0.775813
0.771463	0.161031	-0.93081	-0.65109	-0.80717	0.411329	0.600811	-0.63527
-0.63281	-0.64351	-0.79075	0.253549	-0.23743	-0.66545	-0.65319	0.910361
-0.91081	0.985017	-0.45469	0.465654	-0.66027	-0.45966	0.212066	0.426241
-0.02659	-0.83657	-0.06677	-0.93613	0.438371	-0.95029	0.265404	0.814137

- Percobaan 2

Input Weight							Bias
1	2	3	4	5	6	7	
-0.87248	0.553236	-0.19722	-0.85456	0.075628	0.95008	0.426468	0.867613
0.231173	0.618026	0.074156	-0.82812	-0.02608	0.390228	-0.88753	-0.17106
-0.29999	0.631469	-0.54901	-0.08125	-0.24105	-0.81081	0.906042	-0.77411
-0.19761	-0.21253	0.004045	-0.99385	0.692464	-0.33941	0.408275	-0.78491
-0.48209	-0.73595	-0.54038	-0.51353	-0.02113	-0.24599	-0.059	-0.84176
0.426933	-0.4074	0.106727	-0.5341	0.184764	0.114865	0.262367	0.658113
-0.5801	-0.81613	0.737085	0.236377	0.810474	0.164554	-0.04019	-0.21941

- Percobaan 3

Input Weight							Bias
1	2	3	4	5	6	7	
0.515775	0.362463	0.532053	0.894435	-0.44742	0.802177	-0.31558	-0.31057
0.578497	-0.23866	0.310493	-0.07643	-0.70343	-0.1704	-0.10916	-0.57959
0.220588	0.4278	-0.94833	0.716673	-0.62132	0.701337	-0.75334	-0.97884
-0.31704	-0.50741	0.107228	0.515055	0.155943	-0.29922	-0.62447	-0.21647
-0.02004	-0.24627	0.016345	0.740713	0.423486	-0.85554	0.097831	0.794908
0.098781	0.026543	-0.66194	-0.80169	-0.28173	0.715752	-0.16681	-0.46501
-0.45782	-0.38855	-0.58511	-0.87653	0.048156	-0.47031	0.328731	0.744739

- Percobaan 4

Input Weight							Bias
1	2	3	4	5	6	7	
0.003727	0.731183	-0.51819	0.553978	0.085525	0.030444	-0.62274	0.293961
0.464518	-0.71108	-0.9654	3.78E-04	-0.63564	-0.67705	-0.31067	-0.72591
0.500453	0.798827	0.87164	0.892603	0.520017	0.040363	0.58328	0.042782
-0.91083	0.84076	0.723286	-0.22085	0.110558	-0.65121	-0.34258	0.446826
0.705674	-0.10952	-0.11722	-0.24459	-0.94108	-0.80905	0.108239	0.949988
0.995691	0.877096	-0.24714	-0.90239	0.254801	-0.62973	-0.14115	-0.10769
0.815898	0.319235	-0.27802	-0.42085	-0.64458	-0.42299	-0.83835	0.607443

- Percobaan 5

Input Weight							Bias
1	2	3	4	5	6	7	
-0.07001	0.391492	0.284477	-0.55443	-0.67581	0.957943	-0.16068	0.624575

Input Weight							Bias
1	2	3	4	5	6	7	
0.149726	-0.22724	-0.18162	0.525054	-0.25362	0.331203	-0.17189	-0.31848
0.209947	-0.37203	-0.06473	-0.54922	0.460805	0.647834	-0.8123	-0.22447
-0.83329	-0.42518	0.816022	-0.52822	-0.45338	-0.19842	0.241409	0.978355
-0.39248	0.987059	0.446596	-0.17106	0.204273	0.388558	-0.98898	-0.75778
0.462776	-0.51412	0.167915	0.172486	-0.43029	-0.48177	-0.38976	0.923142
-0.29253	0.970958	-0.54611	-0.0022	-0.99917	0.277451	-0.23916	-0.75178

- Percobaan 6

Input Weight							Bias
1	2	3	4	5	6	7	
-0.25779	0.042775	-0.66836	-0.42411	-0.12465	0.948917	0.095476	0.453521
0.742621	0.167692	0.358045	-0.64475	-0.561	0.585777	0.411391	-0.18102
-0.2338	-0.56258	-0.43346	-0.6885	-0.25669	0.043839	0.538952	0.327002
-0.09432	0.274384	-0.7016	0.221566	-0.7882	0.073	-0.01089	-0.8287
0.27796	-0.0654	-0.66755	-0.66121	0.873918	0.569434	-0.95612	-0.15754
0.137653	-0.84043	0.227259	-0.55436	-0.19448	-0.40988	0.453842	0.0169
0.904379	0.82882	0.290009	0.760527	0.815155	-0.76877	-0.59321	-0.02429

- Percobaan 7

Input Weight							Bias
1	2	3	4	5	6	7	
0.392762	0.964578	-0.91406	-0.32468	-0.08491	-0.02795	-0.06846	0.779964
0.034279	0.940215	0.902535	0.800114	-0.38376	-0.41421	0.401013	-0.0709
0.37999	0.243967	0.082476	-0.7082	0.585964	-0.78089	-0.70142	0.728813
0.6732	-0.35048	0.639441	0.943011	0.707855	-0.25833	0.911794	0.095216
0.78115	-0.85496	-0.92921	0.008044	-0.91741	-0.0325	-0.88241	0.021617
0.274762	-0.52788	-0.21876	-0.75778	-0.99114	0.849513	-0.07711	0.43464
-0.05447	0.626387	-0.9685	-0.86866	0.483458	0.248937	-0.07513	-0.33383

- Percobaan 8

Input Weight							Bias
1	2	3	4	5	6	7	
-0.45141	-0.82889	-0.17391	-0.21219	-0.74037	-0.61889	0.354778	-0.73228
0.061079	0.715556	0.654218	-0.51015	0.002678	0.539243	0.86941	0.393747
-0.75521	0.409518	-0.50208	0.340554	-0.41807	0.19764	-0.24558	-0.51951
-0.19779	0.805707	-0.97853	0.023227	-0.96185	0.096093	-0.29665	-0.55303
-0.08499	0.94146	-0.64389	-0.87043	-0.73465	-0.59316	0.836569	-0.85544
0.454078	0.991381	-0.60582	0.037366	-0.05374	-0.41423	-0.87586	0.28182
-0.34995	-0.65753	0.006721	-0.18592	-0.21972	0.009364	0.87667	-0.44887

- Percobaan 9

Input Weight							Bias
1	2	3	4	5	6	7	
0.513526	-0.55668	-0.90137	0.123121	0.543198	0.998997	0.468207	0.183624
0.794002	0.278853	0.915317	-0.39084	0.017533	-0.66589	0.95529	-0.34094
-0.23351	0.822337	0.515115	0.73915	-0.66723	-0.14416	-0.24386	0.797172
0.635039	-0.68395	0.821323	-0.42437	0.754398	-0.73733	0.054135	-0.04301
-0.15794	0.73227	0.168062	-0.5412	0.460831	0.69965	-0.91047	0.570858
-0.92733	0.266151	0.753959	0.073982	-0.65262	0.242791	-0.99638	0.739501
0.243004	-0.78803	0.917731	-0.11895	-0.60378	0.880105	0.399275	0.114896

- Percobaan 10

Input Weight							Bias
1	2	3	4	5	6	7	
0.261609	0.074427	0.649728	0.382323	-0.34856	0.090966	-0.718	-0.20262
0.852966	0.2664	0.253506	-0.79446	0.338319	-0.37473	0.545527	-0.5264
-0.76779	0.244914	0.571479	-0.52155	0.165782	0.58401	-0.33394	0.521926
-0.07987	0.436711	-0.31127	-0.52839	0.351151	-0.59724	0.769159	0.311735
0.487449	-0.19492	-0.47098	0.985936	-0.59373	0.839754	-0.26661	-0.57667
0.517515	-0.59051	0.257085	-0.5385	-0.77599	0.569942	-0.5052	0.01489
-0.42355	0.809551	0.901283	-0.71568	-0.49568	0.486785	-0.7838	0.063094

C.5 Pengujian Berdasarkan Hari yang Sama

Hari Senin

- Percobaan ke-1

Input Weight							Bias
1	2	3	4	5	6	7	
-0.60886	0.016015	0.269222	0.699073	-0.53901	0.487564	0.576503	-0.93739
-0.65041	-0.53176	0.342269	-0.92638	0.028578	-0.80369	0.196419	0.727586
0.996913	-0.29359	-0.00412	-0.93181	0.985954	-0.1166	0.250499	0.090311
0.488792	-0.0374	0.178712	0.940266	0.13749	0.750709	0.21845	-0.62811
0.13363	-0.08624	0.406694	-0.85408	0.431693	0.563518	-0.05853	0.600934
-0.81856	0.070743	-0.54706	0.977724	0.054598	-0.45631	0.035623	-0.75504
-0.48723	0.732347	-0.06818	-0.84748	0.81772	-0.74624	0.90639	0.029908

- Percobaan ke-2

Input Weight							Bias
1	2	3	4	5	6	7	
0.641682	0.981025	-0.18369	-0.54941	0.504621	0.198361	0.653507	-0.70067
0.946921	-0.30325	0.427558	-0.99225	-0.65984	0.369712	0.655111	0.764804
0.545265	-0.93433	0.935862	-0.63804	-0.75353	0.463542	0.120262	-8.41E-04
-0.27883	-0.43998	-0.08797	0.856061	-0.95021	-0.00783	-0.60107	-0.47595
-0.67327	-0.13405	0.968224	-0.50189	-0.69791	-0.97879	0.338469	0.843481
0.066067	-0.26418	0.02737	-0.1191	0.881669	0.756549	0.475687	0.274749

Input Weight							Bias
1	2	3	4	5	6	7	
0.078149	0.241406	0.260126	0.262344	0.036785	0.708006	0.353525	-0.07148

- Percobaan ke-3

Input Weight							Bias
1	2	3	4	5	6	7	
0.732155	0.404755	-0.77146	0.408037	-0.96375	0.656451	-0.24855	-0.31053
-0.95058	-0.31687	-0.72454	-0.70214	0.091237	0.150755	-0.88141	0.549086
-0.33395	0.284643	0.698661	0.297925	0.712171	0.761235	0.998726	0.142226
-0.24983	-0.20885	-0.25109	-0.10037	-0.83504	0.816131	0.736113	-0.16102
0.78808	-0.93507	-0.00911	0.627203	-0.80952	0.847675	0.251292	-0.75661
-0.34688	0.915813	0.066136	-0.78455	-0.34859	0.474317	-0.8241	0.450549
-0.12848	-0.49486	-0.60305	-0.34081	-0.72945	-0.05433	0.135526	0.913764

- Percobaan ke-4

Input Weight							Bias
1	2	3	4	5	6	7	
0.499156	0.963309	-0.12948	-0.76406	-0.87542	0.07676	-0.53025	0.273201
0.523358	0.197361	0.042101	0.103468	0.057432	0.79805	-0.82166	0.99767
-0.79618	-0.63483	0.896491	-0.42631	0.677219	-0.95203	0.382699	-0.80892
-0.73339	-0.00714	0.706768	0.664544	0.100886	-0.76564	-0.90833	0.415106
0.365452	-0.78808	-0.97061	-0.96859	0.11059	-0.00858	-0.43928	-0.12029
0.724239	-0.90732	0.374696	0.879539	-0.11411	-0.54118	-0.96648	0.852969
-0.62511	0.550682	0.59191	0.959197	0.975785	-0.19057	0.155105	-0.35818

- Percobaan ke-5

Input Weight							Bias
1	2	3	4	5	6	7	
-0.09537	-0.19762	-0.49234	0.244964	0.674736	0.252258	-0.15444	-0.00465
-0.87332	-0.15302	-0.1429	-0.59918	0.46661	0.237209	-0.07666	0.282072
0.588571	-0.04924	-0.95326	0.632558	0.700731	-0.91185	-0.73269	-0.70714
-0.6681	0.610499	0.787936	-0.97092	-0.83597	0.790903	-0.19346	-0.97899
0.817889	-0.02731	0.140205	0.994176	-0.28662	-0.8373	-0.87311	-0.72892
-0.56748	0.184917	0.003416	-0.12974	-0.36782	-0.8703	0.983511	-0.55202
0.485562	0.656048	-0.40215	-0.64425	0.875219	0.339021	-0.49172	-0.62202

- Percobaan ke-6

Input Weight							Bias
1	2	3	4	5	6	7	
0.117336	-0.29813	-0.02872	-0.18244	0.865938	-0.16797	-0.59873	-0.48689
0.027449	0.552588	0.072045	-0.8997	-0.98988	-0.9947	0.89522	-0.62948
0.439456	0.198892	-0.57274	-0.06627	-0.40996	0.585816	0.782231	-0.44157
-0.78445	0.053019	-0.03481	0.54005	0.203233	-0.15288	-0.70841	-0.75982
-0.46463	-0.0985	0.020858	-0.4574	-0.5902	0.792756	-0.89211	-0.61205

Input Weight							Bias
1	2	3	4	5	6	7	
-0.94713	-0.87395	0.063883	0.777515	-0.24752	0.533781	-0.68156	0.677679
-0.05445	-0.33075	0.187764	0.109828	0.813368	-0.41327	-0.42091	-0.62313

- Percobaan ke-7

Input Weight							Bias
1	2	3	4	5	6	7	
-0.23022	-0.27847	-0.10049	0.855762	-0.52845	0.318716	0.287502	-0.92335
-0.46336	-0.39403	-0.75123	0.912575	-0.97448	0.2662	-0.94713	0.619845
-0.07799	-0.16235	0.835506	-0.80235	-0.65545	-0.44606	0.251318	-0.92353
-0.97479	-0.16609	0.735271	-0.50165	0.713076	0.140744	0.955821	-0.8817
-0.40985	0.252209	0.881131	0.334184	0.53212	-0.88142	-0.97009	0.644063
0.103028	0.432107	-0.05866	-0.06713	0.54183	0.791578	-0.32594	-0.99322
0.680745	0.917934	0.231802	-0.07812	0.729203	0.761488	-0.786	0.694859

- Percobaan ke-8

Input Weight							Bias
1	2	3	4	5	6	7	
-0.83604	-0.46129	0.702352	-0.16304	0.585058	-0.2538	-0.58024	-0.18155
-0.67104	0.021867	0.536006	-0.95074	0.874308	-0.64214	0.144835	-0.74728
-0.36762	0.617387	0.400148	-0.78743	-0.4164	0.567076	0.574203	0.789527
-0.59722	0.708818	0.888771	0.832331	0.210374	0.079965	-0.12947	-0.85098
-0.10681	-0.11216	0.19482	0.057812	-0.94254	0.354362	0.507426	-0.79709
0.10875	0.468872	-0.4483	0.299207	0.69718	-0.51806	0.078975	0.260707
-0.17667	0.920314	-0.517	-0.22893	-0.73093	-0.33726	-0.12556	-0.22923

- Percobaan ke-9

Input Weight							Bias
1	2	3	4	5	6	7	
0.699377	0.18897	0.399547	0.563726	-0.22468	-0.50138	-0.99264	-0.71253
0.924938	-0.60577	-0.41725	-0.65581	0.037957	-0.16771	-0.61202	0.056257
-0.71802	-0.13146	-0.90659	0.288172	0.48176	-0.64159	-0.67182	0.067989
-0.93999	-0.97918	-0.35502	-0.65529	-0.59286	0.733384	0.067853	0.934263
0.729159	-7.92E-04	-0.01959	-0.56496	-0.0831	0.410304	0.011886	-0.56481
0.786648	-0.37009	-0.20851	0.687168	-0.83901	-0.07426	-0.03321	-0.90878
0.690841	-0.39146	0.719996	0.307862	-0.9367	0.864869	-0.83399	-0.3894

- Percobaan ke-10

Input Weight							Bias
1	2	3	4	5	6	7	
0.296859	0.420653	0.479466	-0.80655	0.194641	0.231451	0.372956	0.563985
0.707842	-0.03211	-0.36321	0.881119	0.955832	-0.55064	0.850938	0.137778
-0.04407	-0.62357	-0.90733	0.94223	0.735057	0.722091	0.48891	0.317794
-0.57297	0.964265	0.799987	-0.58082	0.203536	0.976868	0.872197	0.35476

Input Weight							Bias
1	2	3	4	5	6	7	
-0.55664	-0.15624	0.75911	-0.59254	0.413558	0.11993	-0.41278	0.863252
0.918017	-0.86371	-0.17068	-0.61565	0.862938	0.829522	0.549308	-0.59439
0.495346	-0.72474	0.395256	-0.61408	-0.81002	-0.38683	-0.3574	0.912405

LAMPIRAN D DATASET PADA HARI SENIN

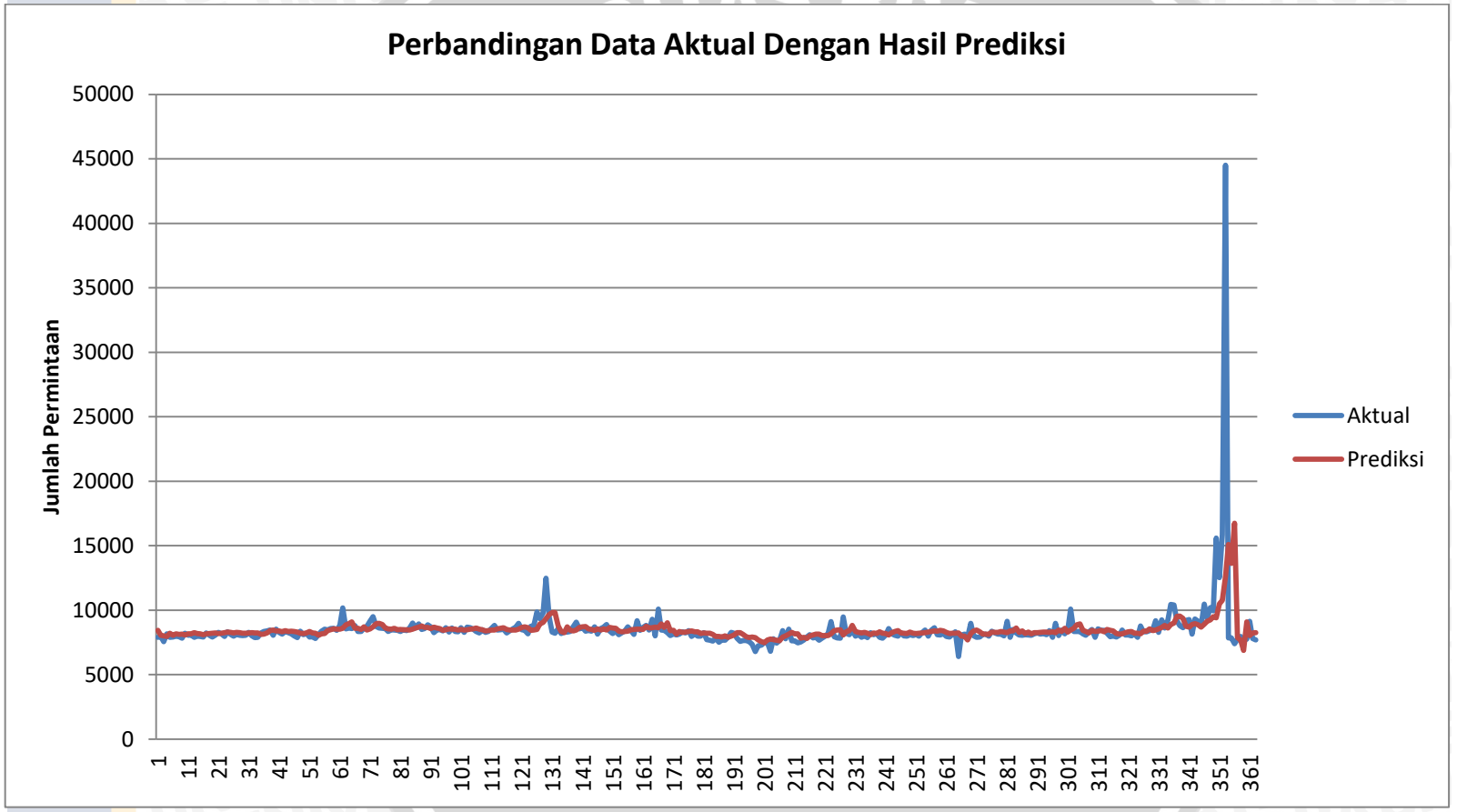
Data *Training*

Data ke-*i*	X1	X2	X3	X4	X5	X6	X7	T
1	7898	8072	7933	7996	7876	8521	7869	8132
2	8072	7933	7996	7876	8521	7869	8132	8821
3	7933	7996	7876	8521	7869	8132	8821	8350
4	7996	7876	8521	7869	8132	8821	8350	8593
5	7876	8521	7869	8132	8821	8350	8593	8477
6	8521	7869	8132	8821	8350	8593	8477	8583
7	7869	8132	8821	8350	8593	8477	8583	8648
8	8132	8821	8350	8593	8477	8583	8648	8698
9	8821	8350	8593	8477	8583	8648	8698	8361
10	8350	8593	8477	8583	8648	8698	8361	8404
11	8593	8477	8583	8648	8698	8361	8404	8747
12	8477	8583	8648	8698	8361	8404	8747	8296
13	8583	8648	8698	8361	8404	8747	8296	8726
14	8648	8698	8361	8404	8747	8296	8726	8706
15	8698	8361	8404	8747	8296	8726	8706	8374
16	8361	8404	8747	8296	8726	8706	8374	9185
17	8404	8747	8296	8726	8706	8374	9185	9086
18	8747	8296	8726	8706	8374	9185	9086	8150
19	8296	8726	8706	8374	9185	9086	8150	8000
20	8726	8706	8374	9185	9086	8150	8000	7707
21	8706	8374	9185	9086	8150	8000	7707	7654
22	8374	9185	9086	8150	8000	7707	7654	7495
23	9185	9086	8150	8000	7707	7654	7495	7795
24	9086	8150	8000	7707	7654	7495	7795	7925
25	8150	8000	7707	7654	7495	7795	7925	8195
26	8000	7707	7654	7495	7795	7925	8195	8093
27	7707	7654	7495	7795	7925	8195	8093	8267
28	7654	7495	7795	7925	8195	8093	8267	8161
29	7495	7795	7925	8195	8093	8267	8161	8043
30	7795	7925	8195	8093	8267	8161	8043	8637
31	7925	8195	8093	8267	8161	8043	8637	8337
32	8195	8093	8267	8161	8043	8637	8337	7901
33	8093	8267	8161	8043	8637	8337	7901	8145
34	8267	8161	8043	8637	8337	7901	8145	8076
35	8161	8043	8637	8337	7901	8145	8076	8145
36	8043	8637	8337	7901	8145	8076	8145	8475

Data *Testing*

Data ke-*i*	X1	X2	X3	X4	X5	X6	X7	T
1	8637	8337	7901	8145	8076	8145	8475	8178
2	8337	7901	8145	8076	8145	8475	8178	8452
3	7901	8145	8076	8145	8475	8178	8452	8084
4	8145	8076	8145	8475	8178	8452	8084	8325
5	8076	8145	8475	8178	8452	8084	8325	9225
6	8145	8475	8178	8452	8084	8325	9225	9279
7	8475	8178	8452	8084	8325	9225	9279	10193
8	8178	8452	8084	8325	9225	9279	10193	7864
9	8452	8084	8325	9225	9279	10193	7864	7810

LAMPIRAN E GRAFIK PERBANDINGAN DATA AKTUAL DENGAN HASIL PREDIKSI



Nilai MSE : 0.009533