BAB II
KAJIAN TEORI

Pada bab ini, dalam *Fuzzy* FFNNuntuk Peramalan IHSG dengan Algoritma Genetika Variasi Seleksi berisi tentang kajian teori yang akan digunakan dalam penelitian ini, diantaranya mengenai IHSG, *time series*, *neural network*, algoritma genetika, serta logika *fuzzy*.

A. Indeks Harga Saham Gabungan

Indeks Harga Saham Gabungan (IHSG) atau *composite share price index* merupakan indeks gabungan dari seluruh jenis saham yang tercatat di bursa efek (Samsul, 2006, hal. 185). IHSG diterbitkan oleh bursa efek. Tanggal 10 Agustus 1982 ditetapkan sebagai hari dasar. Hari dasar merupakan hari penetapan indeks dengan nilai dasar 100. Nilai dasar merupakan kumulatif dari perkalian harga saham dengan jumlah saham tercatat(IDX, 2015, hal. 52). IHSG dihitung setiap hari atau setiap detik selama jam perdagangan sesuai dengan kebutuhan. Rumusan untuk menghitung nilai IHSG sebagai berikut:

$$IHSG = \frac{Nilai\ Pasar}{Nilai\ Dasar} \times 100$$

Nilai pasar merupakan perkalian antara jumlah saham tercatat dengan harga terakhir. Sedangkan nilai dasar merupakan perkalian antara jumlah saham tercatat dengan harga perdana.

Contoh 2. 1 :

Terdapat 3 jenis saham yaitu saham A, saham B, dan saham C. Hari dasar ditentukan pada tanggal 5 Oktober 2004. Pada hari tersebut, saham A ditutup dengan harga Rp 2.500,-; saham B Rp 5.000,-; dan saham C Rp 1.500,-. Jumlah saham A 1.000.000 lembar, saham B 2.000.000 lembar dan saham C 3.000.000. Pada tanggal 10 Oktober 2004 harga mengalami perubahan sebagai berikut: saham A Rp 2.700.-; saham B Rp 4.800,-; dan saham C Rp 1.700,-. Perhitungan Indeks Harga Saham Gabungan pada tanggal 10 Oktober 2004 sebagai berikut:

Tabel 2. 1. Contoh data saham A, saham B, dan saham C

Keterangan	Saham A	Saham B	Saham C
Harga Pasar (Ps)	Rp 2.700,-	Rp 4.800,-	Rp 1.700,-
Harga Dasar (Pb)	Rp 2.500,-	Rp 5.000,-	Rp 1.500,-
Jumlah Lembar Saham	1.000.000	2.000.000	3.000.000

$$IHSG$$

$$= \frac{(Rp\ 2.700 \times 1.000.000) + (Rp\ 4.800 \times 2.000.000) + (Rp\ 1.700 \times 3.000.000)}{(Rp\ 2.500 \times 1.000.000) + (Rp\ 5.000 \times 2.000.000) + (Rp\ 1.500 \times 3.000.000)}$$

$$= 102{,}35$$

Dari perhitungan tersebut diperoleh IHSG sebesar 102,35.

IHSG berubah setiap hari karena perubahan harga pasar yang terjadi setiap hari dan adanya saham tambahan.Perubahan harga saham individu di pasar terjadi karena faktor permintaan dan penawaran. Terdapat berbagai variabel yang mempengaruhi permintaan dan penawan, baik yang rasional maupun yang irrasional. Pengaruh yang

sifatnya rasional mencakup kinerja perusahaan, tingkat bunga, tingkat inflasi, kurs valuta asing, atau indeks harga saham dari negara lain. Pengaruh irrasional mencakup rumor di pasar, mengikuti mimpi, bisikan teman, atau permainan harga (Samsul, 2006: 186).

Kinerja perusahaan memiliki pengertian sebagai hasil dari sebuah kegiatan manajemen di sebuah perusahaan. Hasil dari kegiatan manajemen ini kemudian dijadikan sebuah parameter atau tolok ukur untuk menilai keberhasilan manajemen sebuah perusahaan dalam hal pencapaian tujuan yang sudah ditetapkan dalam periode tertentu. Selain kinerja perusahaan, variabel yang mempengaruhi IHSG adalah tingkat bunga. Tingkat bunga menurut Boedionoyaitu sebagai harga dari penggunaan uang untuk jangka waktu tertentu. Selain itu tingkat bunga dapat diartikan sebagai harga yang harus dibayar apabila terjadi pertukaran antara satu rupiah sekarang dengan satu rupiah nanti.

Tingkat inflasi juga merupakan salah satu variabel yang mempengaruhi IHSG. Menurut kamus Bank Indonesia, tingkat inflasi merupakan tingkat perubahan harga, dua indikasi utama dalam perhitungan tingkat perubahan inflasi berupa indeks harga konsumen dan indeks harga produsen yang mengikuti perubahan harga yang dibayar oleh konsumen dan produsen. Selain itu kurs valuta asing juga mempengaruhi IHSG. Kurs valuta asing didefinisikan sebagai harga mata uang suatu negara dalam suatu negara dalam unit komoditas (seperti mata uang dapat diartikan sebagai perbandingan

nilai mata uang. Kurs menunjukkan harga suatu mata uang, jika dipertukarkan dengan mata uang lain).

Pada IHSG, nilai IHSG pada negara lain juga merupakan variabel yang mempengaruhi pergerakan IHSG itu sendiri. Indeks harga saham merupakan indikator utama yang menggambarkan pergerakan harga saham (Darmadji &Fakhruddin, 2001: 95). Indeks harga saham negara lain berarti pergerakan harga saham yang berada di negara lain. Contoh indeks dari negara lain antara lain indeks FTSE yang berasal dari Inggris, indeks NYSE yang berasal dari New York, indeks Hang Seng yang berasal dari Hongkong, dan indeks Dow Jones yang berasal dari Amerika Serikat.

B. Konsep Dasar *Time Series*

Dalam meramalkan suatu nilai di masa yang akan datang, perlu diketahui mengenai perkembangan dan sifat tentang variabel tersebut di masa lalu. Nilai suatu variabel dapat diramalkan apabila diketahui terlebih dahulu mengenai perkembangan dan sifatnya di masa yang lalu. Konsep dasar *time series* dapat digunakan untuk mengetahui perkembangan dan sifat dari suatu variabel tersebut di masa lalu.

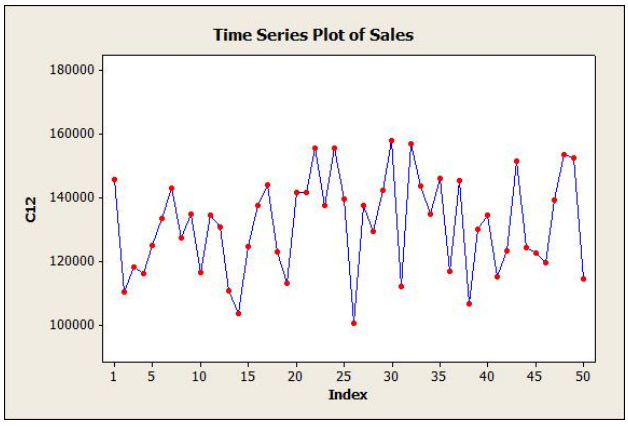
Time series adalah pengamatan beberapa variabel yang terdiri dari data yang dikumpulkan, dicatat, atau diamati secara berturut-turut dari waktu ke waktu (Hanke & Wichern, 2005: 58). Pencatatan data yang dilakukan dalam periode tertentu, misalnya harian, bulanan, dan tahunan. *Time series* mempunyai empat tipepola umum

yaitu horisontal, *trend*, *seasonal* (musiman), dan siklis. Pola data *time series* ini merupakan salah satu konsep dasar dalam analisis *time series.*

Untuk pola data *time series*, grafik yang terbentuk berasal dari data dengan plot waktu yang lama. Pada plot waktu dinyatakan dalam beberapa *trend* waktu yang lama, beberapa kebiasaan musiman, dan data sistematik lainnya. Langkah penting dalam memilih metode peramalan yang tepat yaitu mempertimbangkan tipe dari pola data. Ada empat tipe pola data *time series* yaitu horisontal, *seasonal*, siklis, dan *trend* (Makridakis, *et al*, 1998: 10)

a. Pola data Horisontal

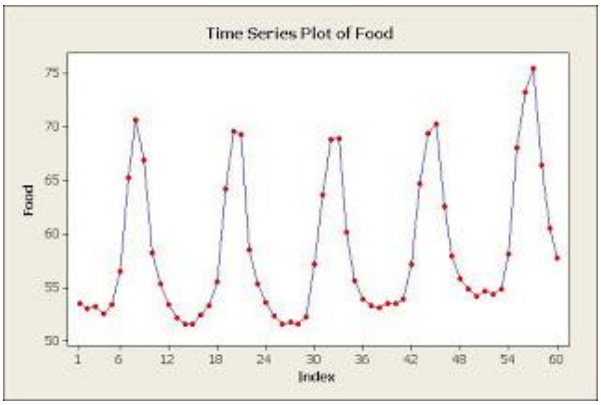
Pola data horisontal merupakan pola data ketika nilai data naik turun pada sekitar nilai rata-rata yang konstan. Suatu produk yang penjualannya tidak naik atau tidak turun selama waktu tertentu merupakan contoh dari pola data horisontal. Pola data horisontal terlihat pada Gambar 2.1.



Gambar 2. 1. Pola Data Horisontal
Sumber:http://itaresume.blogspot.co.id/2014/06/analisis-time-series-dan-forecasting.html

b. Pola data *Seasonal*(Musiman)

Pola data *seasonal* atau musiman merupakan pola data ketika ada rangkaian data yang dipengaruhi oleh faktor musiman seperti tiga bulanan, satu bulanan, atau setiap minggu. Data musiman juga sering disebut "periodik" meskipun data musiman tidak tepat berulang pada setiap periode. Pola data seasonal terlihat pada Gambar 2.2.

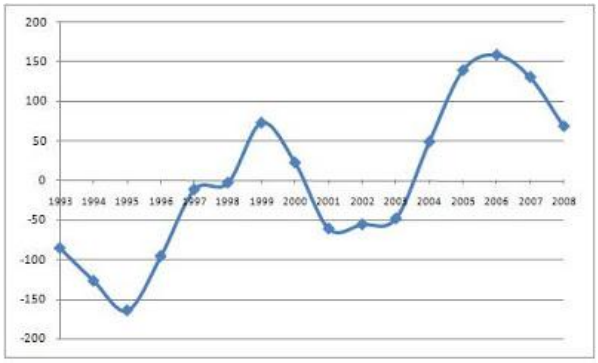


Gambar 2. 2. Pola Data Seasonal

Sumber:http://itaresume.blogspot.co.id/2014/06/analisis-time-series-dan-forecasting.html

c. Pola data Siklis

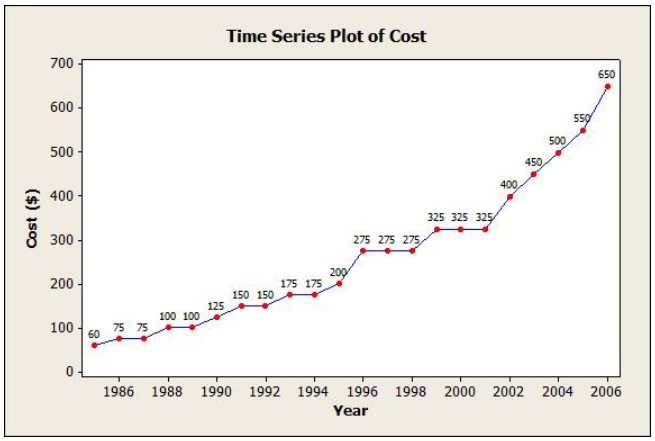
Pola data siklis merupakan pola data naik dan turun tetapi tidak pada periode yang ditetapkan. Contoh dari pola data siklis yaitu data ekonomi yang sering mengalami fluktuasi ekonomi, seperti siklus bisnis. Pola data siklis terlihat pada Gambar 2.3.



Gambar 2. 3. Pola Data Siklis
Sumber:http://junaidichaniago.blogspot.co.id/2009/05/manual-deteksi-autokorelasi-dengan.html

d. Pola data *Trend*

Pola data *trend* merupakan pola data jika data naik atau turun seraca jangka panjang. Penjualan beberapa perusahaan, GNP, dan indikator ekonomi mengkuti pola data *trend*. Pola data trend terlihat pada Gambar 2.4.



Gambar 2. 4. Pola Data Trend
Sumber:http://itaresume.blogspot.co.id/2014/06/analisis-time-series-dan-forecasting.html

Selain pola data *time series*, autokorelasi merupakan konsep dasar dalam analisis *time series.* Analisis *time series* merupakan suatu metode kuantitatif untuk menentukan pola data masa lampau yang dikumpulkan secara teratur.

1. Autokorelasi

Autokorelasi pada analisis *time series* merupakan suatu korelasi antara X_t dengan X_{t+k}. Untuk mendefinisikan autokorelasi dibutuhkan definisi autokovarians. Autokovarians dinotasikan sebagai γ_k. Sedangkan fungsi autokorelasi (*autocorrelation function*) atau ACF dinotasikan sebagai ρ_k. Autokovarians dan autokorelasi berturut-turut dirumuskan sebagai berikut (Wei, 2006: 10) :

$$\gamma_k = Cov(X_t, X_{t+k}) = E(X_t - \mu)(X_{t+k} - \mu) \tag{2.1}$$

dan autokorelasi dirumuskan sebagai

$$\rho_k = \frac{Cov(X_t, X_{t+k})}{\sqrt{Var(X_t)}\sqrt{Var(X_{t+k})}} = \frac{\gamma_k}{\gamma_0} \tag{2.2}$$

dengan $Var(X_t) = Var(X_{t+k}) = \gamma_0$ dan $\rho_0 = 1$.

Nilai estimasi untuk autokorelasi dirumuskan sebagai:

$$\hat{\rho}_k = r_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0} = \frac{\sum_{t=1}^{n-k}(X_t - \bar{X})(X_{t+k} - \bar{X})}{\sum_{t=1}^{n}(X_t - \bar{X})^2} \tag{2.3}$$

dengan :

r_k = koefisien autokorelasi untuk lag ke-*k*

X_t = pengamatan pada waktu ke-t

X_{t+k} = pengamatan pada waktu t+k

$\bar{X}$ = rata-rata pengamatan

ACF mewakili kovarian dan korelasi antara X_t dan X_{t+k} dari proses yang sama, hanya dipisahkan oleh *time* lag ke-k(Wei, 2006: 10). Jika $X_t$dan X_{t+k} independen maka $\gamma_k = Cov(X_t, X_{t+k}) = 0$ tetapi tidak berlaku sebaliknya. X_t dan X_{t+k} dikatakan tidak berkorelasi jika $\rho_k = 0$. Dua variabel dengan hubungan negatif sempurna memiliki koefisien korelasi sama dengan -1. Sebaliknya, dua variabel dengan hubungan positif sempurna memiliki koefisien korelasi sama dengan +1. Koefisien korelasi bervariasi antara -1 dan +1 (Hanke & Wichern, 2005: 35).

Langkah-langkah untuk menguji signifikansi dari autokorelasi lag ke-*k*sebagai berikut:

1) Menentukan hipotesis untuk menguji signifikansi dari autokorelasi lag ke-*k*.

H_0: $\rho_k = 0$ artinya autokorelasi pada lag ke-*k* tidak signifikan

H_1: $\rho_k \neq 0$ artinya autokorelasi pada lag ke-*k* signifikan.

2) Menghitung statistik uji yang digunakan untuk menguji tingkat signifikansi dari autokorelasi lag ke-*k*dengan menggunakanuji t yang didefinisikan sebagai:

$$t = \frac{r_k}{SE(r_k)} \tag{2.4}$$

Notasi $SE(r_k)$ merupakan standar *error*untuk autokorelasi pada lag ke-k yang dirumuskan sebagai (Hanke & Wichern, 2005: 64):

$$SE(r_k) = \sqrt{\frac{1+2\sum_{i=1}^{k-1} r_i^2}{n}} \quad (2.5)$$

dengan :

$SE(r_k)$ = standar *error*autokorelasi pada lag ke-*k*

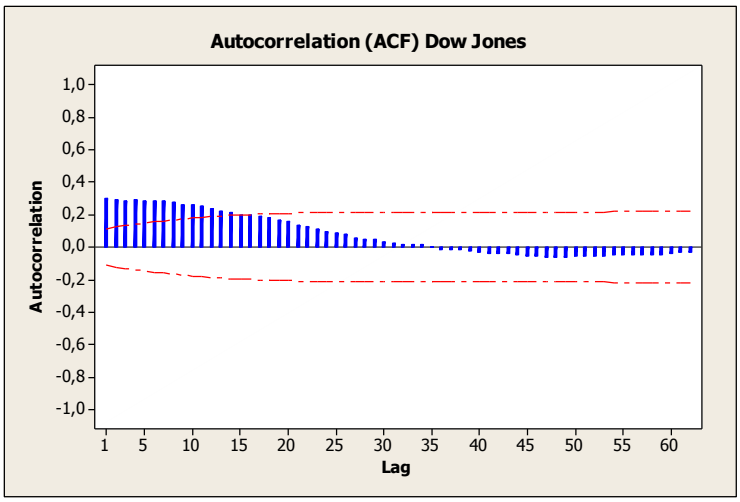
r_i = autokorelasi pada lag ke-*i*

n = banyaknya pengamatan

Autokorelasi pada lag ke-*k* dikatakan signifikan jika $|t_{hitung}| > t_{\frac{\alpha}{2},n-1}$. Dengan t_{hitung} merupakan nilai yang dihasilkan dalam uji t. Sedangkan $t_{\frac{\alpha}{2},n-1}$ merupakan nilai pada tabel t dengan $\frac{\alpha}{2}$ merupakan taraf signifikansi dan $(n-1)$ merupakan derajat bebas. Tabel t tersaji pada Lampiran 15. Signifikansi suatu autokorelasi juga dapat dilihat dengan menggunakan interval kepercayaan yang berpusat di nol dan didefinisikan sebagai berikut:

$$0 \pm t_{\frac{\alpha}{2},n-1} \times SE(r_k) \quad (2.6)$$

Autokorelasi pada lag ke-*k* dikatakan signifikan jika nilai r_k berada diluar interval kepercayaan. Pada Gambar 2.5 ditunjukkan plot autokorelasi pada *time series.*



Gambar 2. 5. Plot autokorelasi data *time series*

Besaran statistik lainnya yang digunakan dalam analisis *time series* adalah fungsi autokorelasi parsial (*Partial Autocorrelation Function*)atau PACF. PACF digunakan untuk mengukur tingkat korelasi antara X_t dan X_{t-k} apabila pengaruh dari *time* lag $1, 2, 3, ..., k-1$ dihilangkan. PACF dinotasikan dengan ϕ_{kk} dan dirumuskan sebagai berikut(Wei, 2006: 15).

$$\phi_{kk} = \frac{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_1 \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & \rho_k \end{vmatrix}}{\begin{vmatrix} 1 & \rho_1 & \rho_2 & \cdots & \rho_{k-2} & \rho_{k-1} \\ \rho_1 & 1 & \rho_1 & \cdots & \rho_{k-3} & \rho_{k-2} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ \rho_{k-1} & \rho_{k-2} & \rho_{k-3} & \cdots & \rho_1 & 1 \end{vmatrix}} \tag{2.7}$$

C. Konsep Dasar *Neural Network* dan Algoritma *Backpropagation*

Dalam meramalkan Indeks Harga Saham Gabungan, digunakan jaringan syaraf atau *neural network* untuk melakukan penyelesaian proses peramalan. *Neural network* atau jaringan syaraf merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak

manusia (Kusumadewi S. , 2003: 207). Komponen-komponen dari jaringan syaraf tiruan terdiri dari beberapa neuron dan setiap neuron ada hubungannya. Selain itu, jaringan syaraf tiruan adalah sistem pemroses informasi yang memiliki karateristik mirip dengan jaringan syaraf biologi. Jaringan syaraf tiruan ditentukan oleh tiga hal yaitu pola hubungan antar neuron (arsitektur jaringan), metode untuk menentukan bobot penghubung (metode *training/learning/*algoritma), dan fungsi aktivasi (Fausett, 1994: 3).

Neural Network terdiri atas elemen-elemen pemrosesan yang disebut neuron, unit, dan sel atau node (Fausett, 1994: 3). Setiap neuron yang berhubungan pada neuron lainnya dengan suatu *communication link* direpresentasikan dengan *weight* atau bobot. *Neural network* dapat dialpikasikan pada ragam masalah yang luas, seperti menyimpan data atau pola, klasifikasi pola (seperti huruf, angka, suara, atau tanda tangan), memetakan *perform* umum dari pola *input* ke pola *output*, dan menemukan solusi untuk membatasi optimisasi masalah.

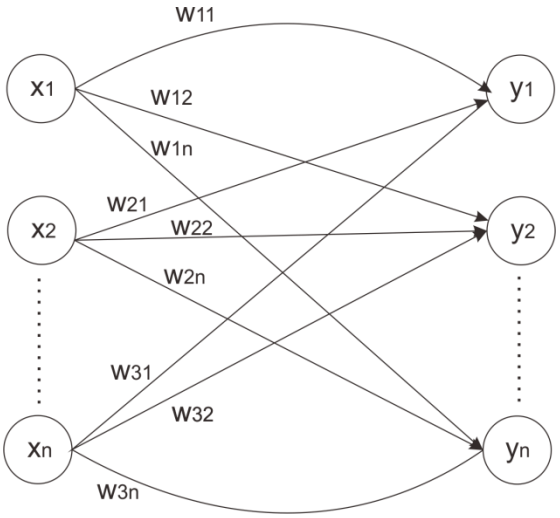
1. Arsitektur Jaringan

Secara khusus, neuron pada lapisan yang sama berjalan pada pola yang sama. Faktor kunci pada penentuan karateristik neuron adalah fungsi aktivasi dan pola dari bobot koneksi yang dikirim dan diterima oleh sinyal. Selain pada setiap lapisan (*layer*), neuron biasanya mempunyai fungsi aktivasi yang sama dan hubungan pola yang sama neuron lainnya. Susunan dari neuron di dalam lapisan serta hubungan pola di dalamnya dan antar lapisan disebut arsitektur jaringan (Fausett, 1994: 12).

Arsitektur jaringan yang sering dipakai dalam jaringan syaraf tiruan sebagai berikut.

a. Jaringan Lapisan Tunggal (*Single Layer Network*)

Dalam jaringan lapisan tunggal sekumpulan *input* neuron dihubungkan langsung dengan sekumpulan *output*nya.



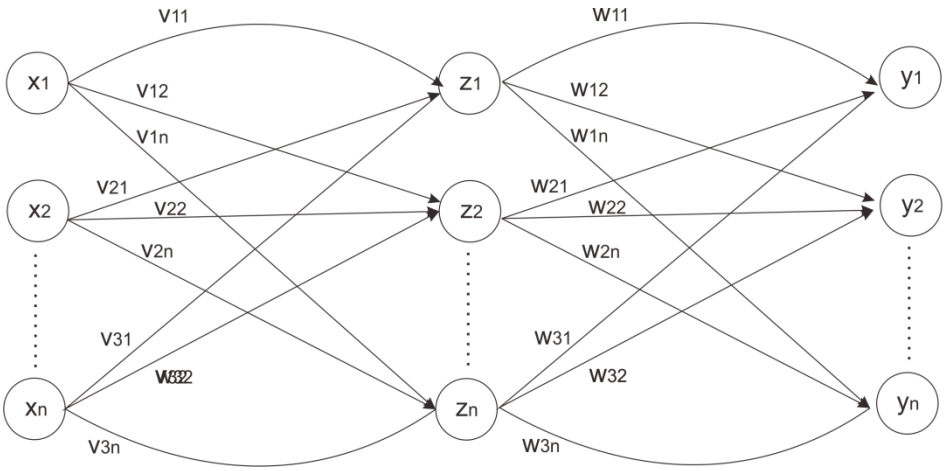
Gambar 2. 6. JaringanLapisan Tunggal

Gambar 2.6 menunjukkan arsitektur jaringan dengan n unit *input*$(X_1, X_2, \dots, X_n)$ dan m buah unit *output*$(Y_1, Y_2, \dots, Y_n)$. Semua unit *input* dihubungkan dengan semua unit *output*, meskipun dengan bobot yang berbeda-beda. Tidak ada unit *input* yang dihubungkan dengan unit *input* lainnya, demikian pula dengan unit *output*nya.

b. Jaringan Lapisan Jamak (*Multi Layer Network*)

Jaringan lapisan jamak merupakan perluasan dari jaringan lapisan tunggal. Dalam jaringan ini, selain unit *input* dan *output*, ada unit lain yang sering disebut lapisan tersembunyi. Dalam lapisan tersembunyi memungkinkan ada beberapa

lapisan. Sama seperti pada unit *input* dan *output*, unit-unit dalam satu lapisan tidak saling berhubungan.

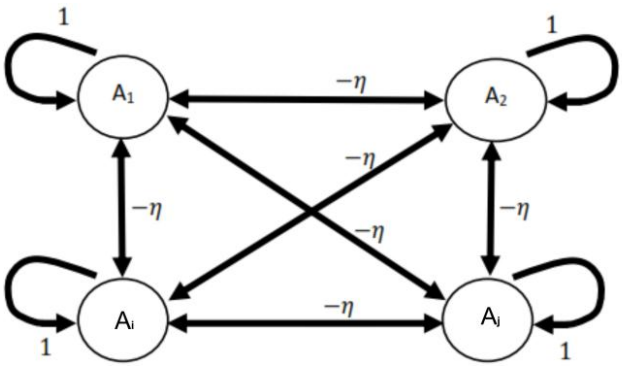


Gambar 2. 7. Jaringan Lapisan Jamak

Gambar 2.7 merupakan jaringan lapisan jamak dengan n buah *input*$(X_1, X_2, \dots, X_n)$, sebuah lapisan tersembunyi yang terdiri dari p buah unit $(z_1, z_2, \dots, z_n)$dan m buah unit *output*$(Y_1, Y_2, \dots, Y_n)$.

c. Jaringan *Reccurent*

Model jaringan *reccurent* mirip dengan jaringan lapisan tunggal ataupun jamak. Hanya saja, ada neuron *output* yang memberikan sinyal pada unit *input* (*feedback loop*). Jaringan *reccurent* ditunjukkan pada Gambar 2.8.



Gambar 2. 8. Jaringan *Reccurent*

2. Metode Pembelajaran

Selain arsitektur, metode pengaturan pada nilai bobot adalah penting untuk membedakan karateristik pada perbedaan *neural network.* Pembelajaran dalam *neural network* dibedakan menjadi 2 yaitu *supervised* (terawasi) dan *unsupervised* (tidak terwasi) (Fausett, 1994: 15).

a. Pembelajaran Terawasi (*Supervised Learning*)

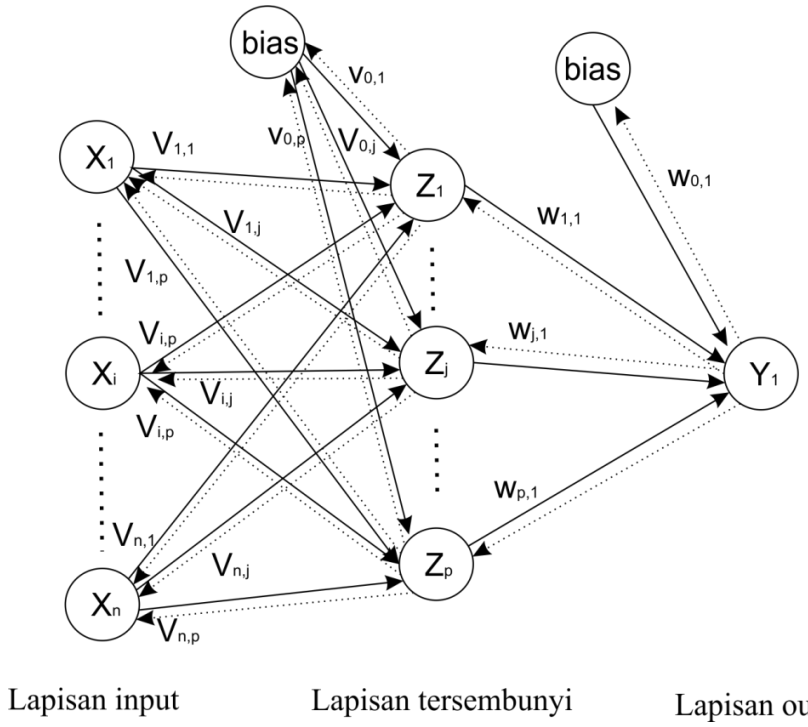
Pembelajaran terawasi adalah metode pembelajaran pada *neural network* dimana *output* yang diharapkan (target) telah diketahui sebelumnya. Tujuan dari pembelajaran terawasi adalah untuk memperbaiki bobotdari *error* atau selisih nilai target dengan *output* yang dihasilkan pada model *neural network.*Jaringan lapisan tunggal seperti klasifikasi pola dan jaringan hubungan pola merupakan contoh pembelajaran terawasi. Selain itu, jaringan lapisan jamak seperti *backpropagation* juga merupakan contoh pembelajaran terawasi. Setiap algoritma pembelajaran dapat dideskripsikan dengan rinci serta deskripsi pada jaringan yang digunakan.

b. Pembelajaran Tidak Terawasi (*Unsupervised Learning*)

Pembelajaran tidak terawasi atau *unsupervised training* merupakan sebuah pembelajaran yang tidak memerlukan *output* yang diharapkan (target). Pada pembelajaran tidak terawasi ini, tidak dapat ditentukan hasil *output*nya. Selama proses pembelajaran, nilai bobot disusun dalam suatu interval tertentu sesuai dengan nilai *input* yang diberikan.*Self organizing, self organizing maps kohonen,* dan *adaptive resonance theory* merupakan contoh dari pembelajaran tidak terawasi.

Dalam penelitian ini, algoritma *backpropagation* digunakan pada arsitektur jaringan pada jaringan *feed forward neural network* dengan banyak lapisan. *Backpropagation* merupakan algoritma pembelajaran yang terawasi (*supervised*) dan biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyinya. Algoritma *backpropagation* menggunakan *erroroutput* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*) (Kusumadewi S. , 2004: 93).

Arsitektur jaringan dalam algoritma *backpropagation* tersaji pada Gambar 2.9. Pada Gambar 2.9 jaringan terdiri atas n unit (neuron) pada lapisan *input* yaitu$x_1, x_2, \dots, x_n$; $p$unit (neuron) pada lapisan tersembunyi yaitu $z_1, z_2, \dots, z_p$; serta 1 neuron pada lapisan *output* yaitu y_1. Dalam arsitektur jaringan tersebut juga terdapat bias untuk menghubungkan antar lapisan (Kusumadewi S. , 2004: 94).

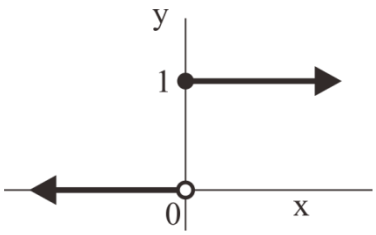


Gambar 2. 9. Arsitektur jaringan *backpropagation*

Pada algoritma *backpropagation* digunakan fungsi aktivasi untuk menghitung sinyal *output*. Ada beberapa fungsi aktivasi yang digunakan dalam jaringan syaraf tiruan, tetapi pada algoritma *backpropagation* hanya menggunakan fungsi aktivasi yang dapat didiferensialkan. Berikut merupakan fungsi aktivasi dalam jaringan syaraf tiruan (Kusumadewi S. , 2004: 51).

a. Fungsi Undak Biner (*Hard Limit*)

Jaringan dengan lapisan tunggal sering menggunakan fungsi undak biner untuk mengkonversikan *input* dari suatu variabel yang bernilai kontinu ke suatu *output* biner (0 atau 1). Gambar 2.10 merupakan fungsi aktivasi undak biner.



Gambar 2. 10. Fungsi Aktivasi Undak Biner

Fungsi undak biner (*hard limit*) dirumuskan sebagai:

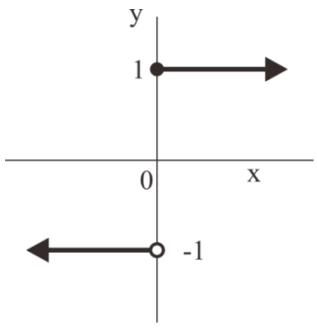
$$y = \begin{cases} 0, & jika\ x < 0 \\ 1, & jika\ x \geq 0 \end{cases} \tag{2.8}$$

Pada MATLAB, fungsi aktivasi undak biner dikenal dengan nama *hardlim*. Syntax untuk fungsi tersebut sebagai berikut.

$$Y = \texttt{hardlim(a)}$$

b. Fungsi Bipolar (*Symetric Hard Limit*)

Fungsi bipolar hampir sama dengan fungsi undak biner, hanya saja *output* yang dihasilkan berupa 1 atau -1. Gambar 2.11 merupakan fungsi aktivasi bipolar.



Gambar 2. 11. Fungsi Aktivasi Bipolar

Fungsi *Symetric Hard Limit* dirumuskan sebagai:

$$y = \begin{cases} 1, & jika\ x \geq 0 \\ -1, & jika\ x < 0 \end{cases} \tag{2.9}$$

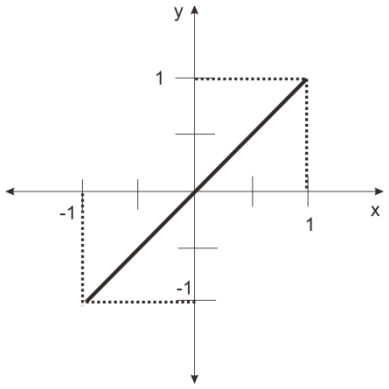
Pada MATLAB, fungsi aktivasi bipolar dikenal dengan nama *hardlims*. Syntax untuk fungsi tersebut sebagai berikut.

$$Y = \texttt{hardlims(a)}$$

c. Fungsi Linear (Identitas)

Fungsi linear memiliki nilai *output* yang sama dengan nilai *input*nya. Fungsi linear dirumuskan sebagai:

$$y = x \tag{2.10}$$



Gambar 2. 12. Fungsi Linear

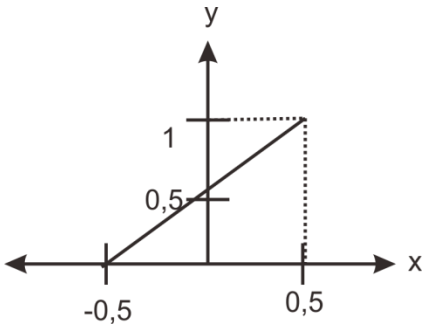
Gambar 2.12 merupakan representasi fungsi linear. Pada MATLAB, fungsi aktivasi identitas dikenal dengan nama *purelin*. Syntax untuk fungsi tersebut sebagai berikut.

$$Y = \texttt{purelin(a)}$$

d. Fungsi *Saturating Linear*

Fungsi ini akan bernilai 0 jika *input*nya kurang dari $-{}^{1}/_{2}$ dan akan bernilai 1 jika *input*nya lebih dari ${}^{1}/_{2}$. Sedangkan jika nilai *input* terletak antara $-{}^{1}/_{2}$ dan ${}^{1}/_{2}$

maka *output*nya akan bernilai sama dengan nilai *input* ditambah $^1/_2$. Gambar 2.13 merupakan fungsi *saturating linear.*



Gambar 2. 13. Fungsi*Saturating Linear*

Fungsi *saturating linear* dirumuskan sebagai :

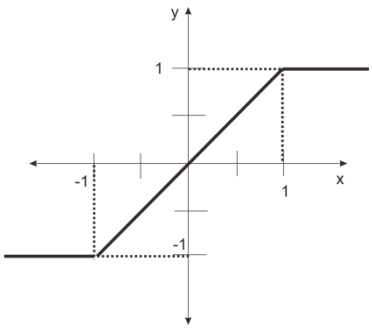
$$y = \begin{cases} 1; & jika\ x \geq 0{,}5 \\ x + 0{,}5; & jika -0{,}5 \leq x \leq 0{,}5 \\ 0; & jika\ x \leq -0{,}5 \end{cases} \quad (2.11)$$

Pada MATLAB, fungsi aktivasi *saturating linear* dikenal dengan nama *satlin*. Syntax untuk fungsi tersebut sebagai berikut :

$$Y = \texttt{satlin(a)}$$

e. Fungsi *Symetric Saturating Linear*

Fungsi ini akan bernilai -1 jika *input*nya kurang dari -1 dan akan bernilai 1 jika *input*nya lebih dari 1. Sedangkan jika nilai *input*terletak antara -1 dan 1, maka *output*nya akan bernilai sama dengan nilai *input*nya. Gambar 2.14 merupakan fungsi *symetric saturating linear.*



Gambar 2. 14. Fungsi*Symetric Saturating Linear*

Fungsi *symetric saturating linear* dirumuskan sebagai:

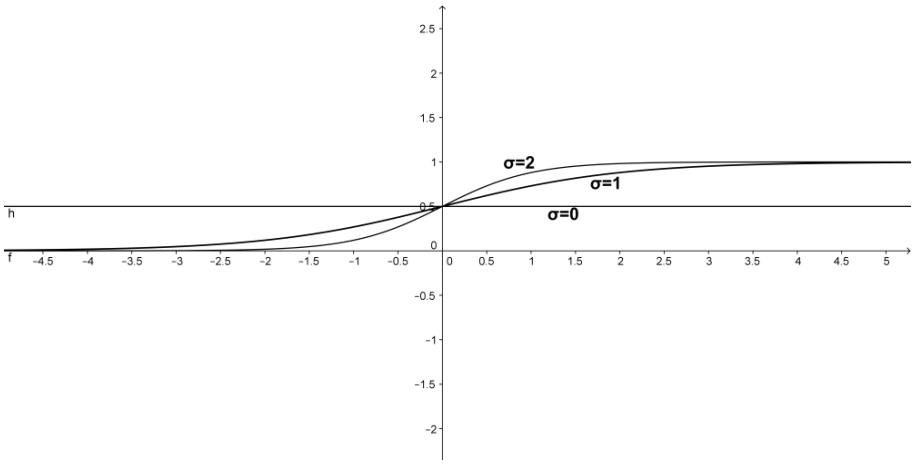
$$y = \begin{cases} 1; & jika \geq 1 \\ x; jika -1 \leq x \leq 1 \\ 0; & jika\ x \leq -1 \end{cases} \quad (2.12)$$

Pada MATLAB, fungsi aktivasi *symetric saturating linear* dikenal dengan nama *satlins*. Syntax untuk fungsi tersebut sebagai berikut:

$$Y = \texttt{satlins(a)}$$

f. Fungsi Sigmoid Biner

Fungsi sigmoid biner digunakan untuk jaringan syaraf yang dilatih dengan menggunakan metode *backpropagation*. Fungsi sigmoid biner memiliki nilai pada range 0 sampai 1. Oleh karena itu, fungsi ini sering digunakan untuk jaringan syaraf yang membutuhkan nilai *output* yang terletak pada interval 0 sampai 1. Namun, fungsi ini bisa juga digunakan oleh jaringan syaraf yang nilai *output*nya 0 atau 1. Gambar 2.15 merupakan fungsi sigmoid biner.



Gambar 2. 15. Fungsi Sigmoid Biner dengan $\boldsymbol{\sigma = 0}$,$\boldsymbol{\sigma = 1}$, dan $\boldsymbol{\sigma = 2}$

Fungsi sigmoid biner dirumuskan sebagai:

$$y = f(x) = \frac{1}{1+e^{-\sigma x}} \tag{2.13}$$

Turunan terhadap x untuk fungsi sigmoid biner sebagai berikut:

$$f'(x) = \frac{\sigma e^{-\sigma x}}{(1+e^{-\sigma x})^2}$$

$$f'(x) = \frac{\sigma e^{-\sigma x}}{(1+e^{-\sigma x})} \cdot \frac{1}{(1+e^{-\sigma x})}$$

$$f'(x) = \sigma\left[\frac{1+e^{-\sigma x}}{(1+e^{-\sigma x})} - \frac{1}{(1+e^{-\sigma x})}\right] \cdot \frac{1}{(1+e^{-\sigma x})}$$

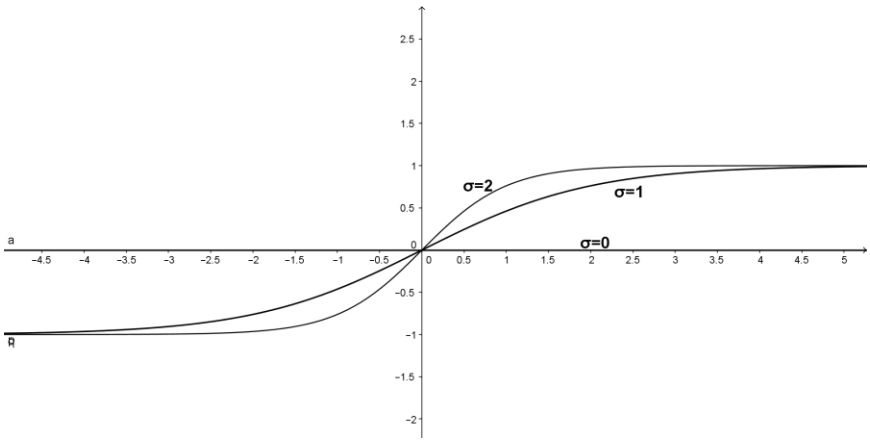
$$f'(x) = \sigma[1-f(x)]f(x)$$

Pada MATLAB, fungsi aktivasi sigmoid biner dikenal dengan nama *logsig*. Syntax untuk fungsi tersebut sebagai berikut:

$$Y = \texttt{logsig(a)}$$

g. Fungsi Sigmoid Bipolar

Fungsi sigmoid bipolar hampir sama dengan fungsi sigmoid biner, hanya saja *output* dari fungsi ini memiliki range antara 1 sampai -1.



Gambar 2. 16. Fungsi sigmoid bipolar $\boldsymbol{\sigma = 0}$,$\boldsymbol{\sigma = 1}$, dan $\boldsymbol{\sigma = 2}$

$$y = f(x) = \frac{1-e^{-\sigma x}}{1+e^{-\sigma x}} \tag{2.14}$$

Turunan terhadap x untuk fungsi sigmoid bipolar sebagai berikut:

$$f'(x) = \frac{2\sigma e^{-\sigma x}}{(1+e^{-\sigma x})^2}$$

$$f'(x) = \frac{\sigma}{2} \cdot \frac{2}{(1+e^{-\sigma x})} \cdot \frac{2e^{-\sigma x}}{(1+e^{-\sigma x})}$$

$$f'(x) = \frac{\sigma}{2} \cdot \left[1 + \frac{1-e^{-\sigma x}}{1+e^{-\sigma x}}\right]\left[1 - \frac{1-e^{-\sigma x}}{1+e^{-\sigma x}}\right]$$

$$f'(x) = \frac{\sigma}{2}[1+f(x)][1-f(x)]$$

Fungsi ini sangat dekat dengan fungsi *hyperbolic tangent.* Keduanya memiliki range antara -1 sampai 1. Untuk fungsi *hyperbolic tangent,* dirumuskan sebagai :

$$y = f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2.15}$$

$$y = f(x) = \frac{1-e^{-2x}}{1+e^{-2x}} \quad (2.16)$$

Pada MATLAB, fungsi aktivasi sigmoid bipolar dikenal dengan nama *tansig*. Syntax untuk fungsi tersebut adalah :

$$Y = \texttt{tansig(a)}$$

Pada algoritma *backpropagtion,* tidak semua fungsi aktivasi dapat digunakan. Fungsi aktivasi yang digunakan dalam *backpropagation* harus memenuhi beberapa syarat antara lain yaitu kontinu, terdiferensial dengan mudah, dan merupakan fungsi yang tidak turun (Siang, 2009: 99). Dalam fungsi aktivasi di atas, fungsi yang memenuhi ketiga syarat tersebut yaitu fungsi sigmoid biner, sigmoid bipolar, dan identitas.

Setelah menentukan fungsi aktivasi kemudian dilakukan pemrosesan algoritma *backpropagation.* Algotitma *backpropagation* terdiri atas dua proses, yaitu *feed forward*dan *backpropagation*(Siang, 2009: 102).Langkah awal dalam proses algoritma *backpropagation* yaitu inisialisasi bobot dengan bilangan acak kecil serta ditetapkan maksimum epoh, target *error*, dan *learning rate.* Selama nilai epoh kurang dari maksimum epoh (epoh < maksimum epoh) dan MSE lebih dari target *error* (MSE > target *error*) maka dilakukan proses *feed forward*dan *backpropagation.*

Langkah pertama dalam proses *feed forward*(propagasi maju)yaitu setiap unit masukan ($X_i, i = 1,2, \dots, n$) menerima sinyal dan meneruskannya ke unit tersembunyi di atasnya. Setelah itu dihitung keluaran di unit tersembunyi($z_j, j = 1,2, \dots, p$) yaitu sebagai berikut:

$$z\_net_j = v_{jo} + \sum_{i=1}^{n} x_i v_{ji} \quad (2.17)$$

Setelah menghitung keluaran pada unit tersembunyi, kemudian menghitung sinyal *output* menggunakan fungsi aktivasi.

$$z_j = f(z\_net_j) \quad (2.18)$$

Langkah selanjutnya setelah didapatkan sinyal *output* yaitu mengirim sinyal *output* yang diperoleh ke semua unit pada layer atasnya yaitu unit-unit *output*. Setelah sinyal *output* dikirim, kemudian dihitung semua keluaran jaringan di unit keluaran $(y_k, k = 1,2, \dots, m)$.

$$y\_net_k = w_{ko} + \sum_{i=1}^{n} z_j w_{kj} \quad (2.19)$$

Setelah itu, dengan menggunakan fungsi aktivasi akan dihitung sinyal *output*.

$$y_k = f(y\_net_k) \quad (2.20)$$

Langkah tersebut di atas merupakan langkah dalam proses *feed forward*. Setelah langkah *feed forward*, kemudian dilakukan langkah *backpropagation* (propagasi mundur). Langkah awal dalam proses *backpropagation* yaitu menghitung faktor δunit keluaran berdasarkan kesalahan di setiap unit keluaran $(y_k, k = 1,2, \dots, m)$ yang didefinisikan sebagai berikut.

$$\delta_k = (t_k - y_k) f'(y\_net_k) \quad (2.21)$$

δ_k merupakan unit kesalahan yang akan dipakai dalam perubahan bobot lapisantersembunyi. Setelah itu, kemudian dihitung suku perubahan bobot w_{kj} (yang akan dipakai untuk merubah bobot w_{kj}) dengan laju percepatan α

$$\Delta w_{kj} = \alpha . \delta_k . z_j \quad (2.22)$$

Setelah didaptkan suku perubahan bobot, kemudian dihitung faktor bias koreksi (digunakan untuk merubah nilai w_{0k})

$$\Delta w_{0k} = \alpha . \delta_k \tag{2.23}$$

Langkah awal dalam proses *backpropagation* tersebut juga dilakukan sebanyak jumlah lapisan tersembunyi. Langkah selanjutnya yaitu menghitung faktor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi $(z_j, j = 1,2, \dots, p)$

$$\delta\_net_j = \sum_{k=1}^{m} \delta_k . w_{kj} \tag{2.24}$$

Kemudian nilai δ_net_j dikalikan dengan turunan dari fungsi aktivasinya untuk menghitung informasi *error*.

$$\delta_j = \delta\_net_j \, f'(z\_net_j) \tag{2.25}$$

Setelah itu menghitung suku perubahan bobot v_{ji}yang digunakan untuk merubah bobot v_{ji}.

$$\Delta v_{ji} = \alpha . \delta_j . x_i \tag{2.26}$$

Hitung juga faktor bias koreksi yang digunakan untuk merubahv_{0j}.

$$\Delta v_{0j} = \alpha . \delta_j \tag{2.27}$$

Langkah selanjutnya yaitu merubah bobot dan bias. Langkah awal yaitu setiap unit keluaran $(y_k, k = 1,2, \dots, m)$ memperbaiki bobot dan biasnya.

$$w_{kj}(baru) = w_{kj}(lama) + \Delta w_{kj} \tag{2.28}$$

Selain unit keluaran, unit tersembunyi $(z_j, j = 1,2, \dots, p)$juga memperbaiki nilai bobot dan nilai biasnya.

$$v_{ji}(baru) = v_{ji}(lama) + \Delta v_{ji} \tag{2.29}$$

Setelah proses *feed forward*dan *backpropagation* memenuhi kondisi yang ditentukan yaitu epoh < maksimum epoh dan MSE > target *error*, maka pembelajaran dengan algoritma *backpropagation* berhenti.

Setelah pembelajaran dengan algoritma *backpropagation* berhenti, maka algoritma *backpropagation* tersebut dapat digunakan sebagai pengujian data *testing*. Namun, dalam penentuan *output* jaringan dalam aplikasi jaringan ini hanya terdiri dari proses *feed forward* yaitu pada langkah (2.17) sampai dengan (2.20). Sebelum melakukan pembelajaran, maka terlebih dahulu mengatur parameter-parameter yang akan digunakan dalam proses pembelajaran. Dalam pemilihan parameter pembelajaran untuk algoritma *backpropagation* berkaitan dengan inisialisasi bobot, kecepatan parameter pembelajaran, dan momentum.

1) Inisialisasi bobot awal

Pemilihan bobot awal sangat mempengaruhi jaringan syaraf dalam mencapai minimum global (atau hanya lokal saja) terhadap nilai *error*, serta cepat tidaknya proses pembelajaran menuju kekonvergenan. Apabila nilai bobot awal terlalu besar, maka *input* ke setiap lapisan tersembunyi atau lapisan *output* akan jatuh pada daerah dimana turunan fungsi sigmoidnya akan sangat kecil. Biasanya bobot awal diinisialisasi secara random dengan nilai antara -0,5 sampai 0,5 (atau -1 sampai 1) atau pada interval lainnya (Kusumadewi S. , 2004: 97). Perintah inisialisasi bobot dalam Matlab sebagai berikut.

```
net = train(P,T)
```

2) Parameter Pembelajaran

Pemilihan parameter pembelajaran adalah proses yang paling penting dalam melakukan pembelajaran. Dalam pemilihan parameter ditentukan terlebih dahulu algoritma pembelajaran yang akan digunakan. Dalam penulisan tugas akhir ini, penulis menggunakan algoritma pembelajaran *Gradient Descent* dengan *Momentum* dan *Adaptive Laearning Rate* (*traingdx*). Adapun parameter-parameter yang digunakan dalam algoritma pembelajaran*traingdx* antara lain (Kusumadewi S. , 2004: 151) :

a) Maksimum Epoh

Maksimum epoh adalah jumlah epoh maksimum yang boleh dilakukan selama proses pembelajaran. Iterasi akan dihentikan apabila nilai epoh melebihi maksimum epoh. Perintah dalam Matlab untuk maksimum epoh sebagai berikut:

`net.trainParam.epochs = MaxEpoh`

Nilai default untuk maksimum epoh adalah 10.

b) Kinerja Tujuan

Kinerja tujuan adalah target nilai fungsi kinerja. Iterasi akan dihentikan apabila nilai fungsi kinerja kurang dari atau sama dengan kinerja tujuan. Perintah dalam Matlab untuk kinerja tujuan sebagai berikut:

`net.trainParam.goal = Target`*`Error`*

Nilai default untuk kinerja tujuan adalah 0.

c) *Learning Rate*

Laering rate adalah laju pembelajaran. Perintah dalam Matlab untuk *learning rate* sebagai berikut:

```
net.trainParam.lr = LearningRate
```

Nilai default untuk *learning rate* adalah 0,01.

d) Rasio untuk menaikkan *Learning Rate*

Rasio ini berguna sebagai faktor pengali untuk menaikkan *learning rate* apabila *learning rate* yang ada terlalu rendah untuk mencapai kekonvergenan. Perintah dalam Matlab untuk parameter ini sebagai berikut:

```
net.trainParam.lr_inc = IncLearningRate
```

Nilai default untuk kinerja tujuan adalah 1,05.

e) Rasio untuk menurunkan *Learning Rate*

Rasio ini berguna sebagai faktor pengali untuk menurunkan *learning rate* apabila *learning rate* yang ada terlalu tinggi dan menuju ke ketidakstabilan. Perintah dalam Matlab untuk rasio ini sebagai berikut:

```
net.trainParam.lr_decc = DecLearningRate
```

Nilai default untuk kinerja tujuan adalah 0,7.

f) Maksimum kenaikan kinerja

Maksimum kenaikan kinerja adalah nilai maksimum kenaikan *error* yang diijinkan, antara *error* saat ini dengan *error* sebelumnya. Perintah dalam Matlab untuk maksimum kenaikan kinerja sebagai berikut:

```
net.trainParam.max_perf_inc = MaxPerfInc
```

Nilai default untuk kinerja tujuan adalah 1,04.

g) Jumlah epoh yang akan ditunjukkan kemajuannya

Menunjukkan berapa jumlah epoh berselang yang akan ditunjukkan kemajuannya. Perintah dalam Matlab untuk parameter ini sebagai berikut:

```
net.trainParam.show = EpohShow
```

Nilai default untuk jumlah epoh yang ditunjukkan adalah 25.

h) Maksimum kegagalan

Maksimum kegagalan diperlukan apabila pada algoritma disertai dengan validitas (optional). Maksimum kegagalan adalah ketidakvalitan terbesar yang diperbolehkan. Apabila gradien pada iterasi ke-*k* lebih besar daripada gradien iterasi ke-(*k*-1), maka kegagalanya akan bertambah 1. Iterasi akan dihentikan apabila jumlah kegagalan lebih dari maksimum kegagalan. Perintah dalam Matlab untuk maksimum kegagalan sebagai berikut:

```
net.trainParam.max_fail = MaxFail
```

Nilai default untuk maksimum kegagalan adalah 5.

i) Gradien Minimum

Gradien minimum adalah akar dari jumlah kuadrat semua gradien (bobot *input*, bobot lapisan, bobot bias) terkecil yang diperbolehkan. Iterasi akan dihentikan apabila nilai akar jumlah kuadrat semua gradien ini kurang dari gradien minimum. Perintah dalam matlab untuk gradien minimum sebagai berikut:

```
net.trainParam.min_grad = MinGradien
```

Nilai default untuk kinerja tujuan adalah 10^{-10}.

j) Waktu maksimum untuk pembelajaran

Menunjukkan waktu maksimum yang diijinkan untuk melakkan pembelajaran. Iterasi akan dihentikan apabila waktu pembelajaran melebihi waktu maksimum. Perintah dalam Matlab untuk parameter ini sebagai berikut:

```
net.trainParam.time = MaxTime
```

Nilai default untuk waktu maksimum adalah tak terbatas (inf).

k) Momentum

Momentum adalah suatu konstanta yang mempengaruhi perubahan bobot. Momentum ini bernilai antara 0 sampai 1. Apabila nilai momentum sama dengan 0, maka perubahan bobot hanya akan dipengaruhi oleh gradiennya. Namun apabila nilai momentum sama dengan 1, maka perubahan bobot akan sama dengan perubahan bobot sebelumnya. Perintah dalam Matlab untuk momentum sebagai berikut:

```
net.trainParam.mc = Momentum
```

Nilai default untuk kinerja tujuan adalah 0,9.

3. Pembentukan Model *Feed Forward Neural Network*

Membentuk jaringan *Feed Forward Neural Network*(FFNN)menggunakan Algoritma *Backpropagation* memerlukan beberapa langkah. Berikut ini merupakan langkah-langkah dalam pembentukan jaringan FFNNdengan Algoritma *Backpropagation*:

a. Menentukan *Input*

Penentuan *input* merupakan langkah pertama untuk membentuk jaringan FFNN. Penentuan *input* ini berdasarkan pada lag-lag yang signifikan pada plot fungsi autokorelasi (ACF) dan plot fungsi autokorelasi parsial (PACF). Lag-lag disebut signifikan jika pada plot ACF maupun PACF lag-lag melewati garis merah.

b. Pembagian Data

Dalam pembagian data, data dibagi menjadi dua yaitu data *training* dan data *testing.* Data *training* digunakan untuk mencari model terbaik, sedangkan data *testing* digunakan untuk menguji ketepatan model hasil data*training.* Komposisi pembagian data yang sering digunakan yaitu 80% data *training* dan 20% data *testing*, 75% data *training* dan 25% data *testing*, atau 50% data *training* dan 50% data *testing*.

c. Menentukan Model *Feed Forward Neural Network*

Sebuah jaringan dibentuk dengan menentukan *input* terlebih dahulu. Penentuan *input* berdasarkan plot ACF dan plot PACF. Setelah *input* sudah diketahui, maka banyaknya neuron pada lapisan tersembunyi dapat ditentukan. Arsitektur jaringan yang sering digunakan dengan algoritma *backpropagation* adalah jaringan *feed forward* dengan banyak lapisan. Untuk membangun suatu jaringan *feed forward*digunakan perintah *newff* (Kusumadewi S. , 2004: 112). Perintah *newff* dalam MatlabR2013a sebagai berikut:

```
net = newff(PR, [S1 S2 ... SN1],{TF1 TF2 ...TFN1},
BTF,BLF,PF)
```

dengan:

PR : matriks berukuran $Rx2$ yang berisi nilai minimum dan maksimum, dengan R adalah jumlah variabel *input*.

S_i : jumlah neuron pada lapisan ke-i, dengan $i = 1, 2, \dots, N_1$.

TF_i : fungsi aktivasi pada lapisan ke-i, dengan $i = 1, 2, \dots, N_1$; (default: *tansig*).

BTF : fungsi pembelajaran jaringan (default : *trainlm*).

BLF : fungsi pembelajaran untuk bobot (default : *learngdm*).

PF : fungsi kinerja (default : *mse*).

Fungsi aktivasi TF_i harus merupakan fungsi yang dapat didiferensialkan, seperti `tansig`, `logsig`, atau `purelin`. Fungsi pembelajaran BTF dapat digunakan fungsi-fungsi pembelajaran untuk *backpropagation*, seperti `trainlm`, `trainbfg`, atau `trainrp`, `traingd`. Fungsi kinerja bisa digunakan fungsi kinerja yang dapat didiferensialkan, seperti `mse` atau `msereg`.

Dalam proses penentuan model FFNN dengan algoritma *bacpropagation* terdapat proses penentuan banyaknya neuron pada lapisan tersembunyi. Penentuan model FFNN akan dinilai keakuratannya dengan menentukan neuron yang terbaik pada lapisan tersembunyi. Untuk menentukan nilai keakuratan neuron yang terbaik pada lapisan tersembunyi digunakan *Mean Squared Error* (MSE), *Mean Absolute Percentage Error* (MAPE), dan*Mean Absolute Deviation* (MAD). MSE, MAPE, dan

MAD merupakan beberapa cara untuk mengevaluasi teknik peramalan (Hanke & Wichern, 2005: 79).

1) *Mean Squared Error*(MSE)

Mean Squared Error(MSE) merupakan salah satu metode untuk mengevaluasi teknik peramalan.Setiap kesalahan dikuadratkan kemudian dijumlahkan dan dibagi dengan nilai observasi. MSE ini merupakan rata-rata kuadrat kesalahan.

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(Y_t - \hat{Y}_t)^2 \quad (2.30)$$

2) *Mean Absolute Percentage Error*(MAPE)

Mean Absolute Percentage Error(MAPE)merupakan perhitungan dengan mencari nilai kesalahan absolut pada setiap periode kemudian dibagi dengan nilai aktual pada periode tersebut dan kemudian dicari persentase kesalahan rata-rata absolut. MAPE memberikan indikasi besarnya kesalahan peramalan dibandingkan dengan nilai aktual.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|Y_t - \hat{Y}_t|}{Y_t} \times 100\% \quad (2.31)$$

3) *Mean Absolute Deviation* (MAD)

Mean Absolute Deviation (MAD) merupakan pengukuran ketelitian peramalan oleh rata-rata dari besarnya kesalahan peramalan (nilai absolut setiap peramalan)

$$MAD = \frac{1}{n}\sum_{t=1}^{n}\left|Y_t - \hat{Y}_t\right| \quad (2.32)$$

dengan :

Y_t = nilai aktual pada periode ke-*t*

$\hat{Y}_t$ = nilai peramalan pada periode ke-*t*

n = banyaknya data

Setelah menentukan banyaknya neuron pada lapisan tersembunyi, langkah selanjutnya yaitu menentukan *input* yang optimal.Penentuan *input* yang optimal untuk model FFNN ini dilakukan dengan cara mengeliminasi pada setiap *input*. Penentuan *input* yang optimal ini bertujuan untuk membentuk jaringan FFNN yang sederhana. Penentuan *input* yang optimal ini dilihat dari nilai MAPE yang paling kecil.

D. Algoritma Genetika

Pada dasarnya, dengan menggunakan FFNN peramalan IHSG sudah dapat digunakan, tetapi dalam penelitian ini digunakan algoritma genetika untuk mengoptimasi hasil yang diperolehdari FFNNagar memperoleh nilai MAPE yang kecil. Model yang dibangun pada jaringan FFNN digunakan dalam algoritma genetika. Algoritma genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup. Algoritma genetika pertama kali dikembangkan oleh John Holland dari Universitas Michigan pada tahun 1975. Algoritma genetika adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom (Kusumadewi S. , 2003: 279).

Algoritma genetika mempunyai beberapa kelebihan, antara lain yaitu dapat diterapkan pada masalah optimasi diskrit dan kontinu; tidak memerlukan perhitungan

derivatif (turunan); pencarian solusi dilakukan secara simultan, dapat terhindar dari solusi optimum lokal yaitu nilai optimal yang dapat dicapai oleh sebuah algoritma berada dalam rentang nilai tertentu; menghasilkan beberapa solusi, tidak hanya solusi tunggal; dan menggunakan aturan probabilistik (peluang), tidak deterministik (Wati, 2011: 11)

Istilah-istilah yang terdapat dalam algoritma genetika antara lain *genotype, allele*, kromosom, individu, populasi, induk, *crossover*, *offspring*, dan generasi (Arkeman, dkk, 2012: 18). *Genotype* (gen) adalah sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Sedangkan *allele* yaitu nilai dari gen dapat berupa bilangan biner,bilangan real, dan bilangan bulat.Kromosom yaitu gabungan gen-gen yang membentuk nilai tertentu. Suatu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat dinamakan individu. Sedangkan populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi. Kromosom yang akan dikenai operator genetik (*crossover*) dinamakan induk. *Crossover* atau pindah silang yaitu operator dari algoritma genetika yang melibatkan dua induk untuk membentuk kromosom baru.*Offspring* (anak) yaitu generasi berikutnya yang terbentuk dari gabungan 2 kromosom generasi sekarang yang bertindak sebagai induk dengan operator *crossover.*Generasi menyatakan satu siklus proses evolusi atau satu iterasi di dalam algoritma genetika.

1. Komponen-komponen Utama Algoritma Genetika

Dalam algoritma genetika terdapat beberapa komponen utama. Komponen-komponen tersebut yaitu (Suyanto, 2005: 6):

a. Skema Pengkodean

Terdapat tiga skema yang umum digunakan dalam pengkodean algoritma genetika, yaitu :

1) *Real-number encoding.* Pada skema ini, nilai gen berada dalam interval [0,R], dimana R adalah bilangan real positif.

   Contoh 2. 2.bilangan real: -1,405; -3,545; 1,394; 2,545

2) *Discrete decimal encoding.* Dalam skema ini, setiap gen bisa bernilai salah satu bilangan bulat dalam interval [0,9].

   Contoh 2. 3. Bilangan desimal: 0,2390; 0,0131; 1,0000; 0,1265.

3) *Binary encoding.* Pada skema ini, setiap gen hanya bisa bernilai 0 atau 1.

   Contoh 2. 4. Bilangan biner: 0011; 1010; 1001; 0101.

b. Membangkitkan Populasi Awal (*Spanning*)

Membangkitkan populasi awal adalah proses membangkitkan sejumlah individu secara acak atau melalui prosedur tertentu. Ukuran untuk populasi tergantung pada masalah yang akan diselesaikan dan jenis operator genetika yang akan diimplementasikan (Kusumadewi S. , 2003: 281). Teknik dalam pembangkitan populasi awal ini ada beberapa cara, antara lain:

1) Random Generator, yaitu cara membangkitkan populasi awal dengan melibatkan pembangkitan bilangan random untuk nilai gen sesuai dengan representasi kromosom yang digunakan.
2) Pendekatan tertentu, yaitu cara membangkitkan populasi awal dengan memasukkan nilai tertentu ke dalam gen dari populasi awal yang dibentuk.
3) Permutasi Gen, cara ini sering digunakan dalam permasalahan kombinatorial seperti *Travelling Salesman Problem* (TSP).

c. Nilai *Fitness*

Nilai *fitness* adalah nilai yang menyatakan baik tidaknya suatu solusi (individu). Nilai *fitness* ini yang dijadikan acuan dalam mencapai nilai optimal dalam algoritma genetika. Algoritma genetika bertujuan mencari individu dengan nilai *fitness* yang paling tinggi. Pada masalah optimasi, fungsi *fitness* yang digunakan adalah

$$f = \frac{1}{x} \tag{2.33}$$

dengan x merupakan nilai dari individu, yang artinya semakin kecil nilai x, maka semakin besar nilai *fitness*nya. Jika nilai *fitness* semua individu mempunyai selisih yang kecil atau hampir sama menyebabkan evolusi mencapai ke optimum lokal.Untuk itu, diperlukan mekanisme yang disebut *linear fitness ranking.Linear fitness ranking* ini bertujuan untuk melakukan penskalaan nilai-nilai *fitness.* Kecenderungan untuk konvergen pada optimum lokal dapat dikurangi dengan persamaan berikut(Suyanto, 2005: 11).

$$f(i) = f_{max} - (f_{max} - f_{min}) \left(\frac{R(i)-1}{N-1}\right) \quad (2.34)$$

dengan $f(i)$ merupakan nilai *fitness* baru individu ke-i, f_{max} merupakan nilai *fitness* tertinggi, f_{min} merupakan nilai *fitness* terendah, dan $R(i)$ menyatakan ranking individu ke-I, dan N merupakan jumlah individu dalam populasi.

d. Seleksi

Seleksi akan menentukan individu-individu mana saja yang akan dipilih untuk dilakukan rekombinasi dan bagaimana *offspring* terbentuk dari individu-individu tersebut. Langkah pertama yang dilakukan yaitu pencarian nilai *fitness.* Masing-masing individu akan menerima probabilitas yeng tergantung pada nilai obyektif dirinya dengan nilai obyektif pada semua individu. Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling fit. Beberapa metode dari seleksi orang tua antara lain (Kusumadewi S. , 2003: 282):

1) *Rank-based fitness assignment*

Pada seleksi ini, populasi diurutkan menurut nilai objektifnya. Nilai *fitness* dari tiap-tiap inividu hanya tergantung pada posisi individu tersebut dalam urutan.

Contoh 2. 5:

Fitness	0,0345	0,0014	0,0023	0,0225	0,009	0,0535

Fitness setelah diurutkan

Fitness	0,0014	0,0023	0,009	0,0225	0,0345	0,0535
Individu	2	3	5	4	1	6
Fitness baru	1	2	3	4	5	6
Probabilitas	0,0476	0,0952	0,1429	0,1905	0,2381	0,2857
Prob Kom	0,0476	0,1429	0,2857	0,4762	0,7143	1,0000

Pembangkitan nilai acak dari 0 sampai 1, diperoleh nilai acak 0,2417. Nilai acak 0,2417 menempati posisi probabilitas ke 3, maka individu yang terpilih adalah individu ke 5.

2) *Roulette Wheel Selection*

Metode seleksi ini merupakan metode yang paling sederhana. Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*nya. Sebuah bilangan random dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan random tersebut akan terseleksi. Proses ini diulang hingga diperoleh sejumlah individu yang diharapkan.

Contoh 2. 6:

Diperoleh nilai *fitness* sebagai berikut:

Fitness	0,0345	0,0014	0,0023	0,0225	0,0090	0,0535
Probabilitas	0,2800	0,0113	0,0186	0,1826	0,0730	0,4342
Prob Komulatif	0,2800	0,2914	0,3101	0,4927	0,5657	1,0000

Pembangkitan nilai acak dari 0 sampai 1, diperoleh nilai acak 0,3231. Nilai acak 0,3231 menempati posisi individu ke 4, maka individu yang terpilih adalah individu ke 4.

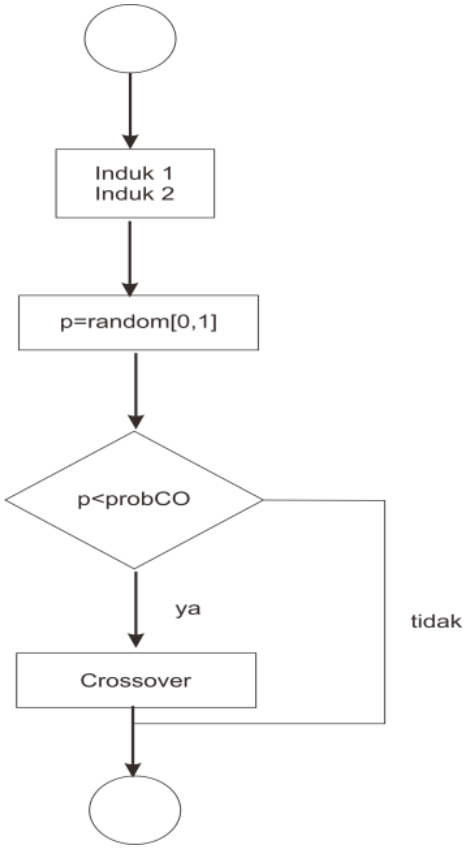
3) *Tournament Selection* (Seleksi dengan Turnamen)

Pada metode ini, akan ditetapkan suatu nilai *tour* untuk individu-individu yang dipilih secara random dari populasi. Individu-individu yang terbaik dalam kelompok

ini akan diseleksi sebagai induk. Individu terbaik yaitu individu yang mempunyai nilai *fitness* terbaik (*fitness* paling besar).

e. Pindah Silang

Pindah silang atau *crossover* merupakan salah satu komponen penting dalam algoritma genetika. Sebuah kromosom yang mengarah pada solusi yang bagus bisa diperoleh dari proses memindah-pindah dua buah kromosom.



Gambar 2. 17. Diagram Alir Proses *Crossover*

Terdapat beberapa cara dalam melakukan proses pindah silang, yaitu(Arkeman, dkk, 2012: 22):

1) Pindah silang satu titik dan banyak titik

Pindah silang satu titik dan banyak titik biasanya digunakan untuk representasi kromosom dalam biner.

Gambar 2. 18. Ilustrasi *crossover* satu titik

Gambar 2. 19. Ilustrasi *crossover* dua titik

2) Pindah silang aritmatika

Pada pindah silang aritmatika digunakan untuk representasi kromosom berupa bilangn real (Syarif, 2014: 39). Pindah silang ini dilakukan dengan menentukan nilai *r* sebagai bilangan random lebih dari 0 dan kurang dari 1. Selain itu juga ditentukan posisi dari gen yang dilakukan pindah silang menggunakan bilangan random. Nilai baru dari gen pada anak dirumuskan dalam persamaan (2.34) dan (2.35).

$$x_1'(k) = r.x_1(k) + (1-r).x_2(k) \quad (2.35)$$

$$x_2'(k) = r.x_2(k) + (1-r).x_1(k) \quad (2.36)$$

dengan:

x_1'= nilai gen baru pada anak 1

x_2'= nilai gen baru pada anak 2

x_1= nilai gen pada induk 1 yang akan dipindah silangkan

x_2= nilai gen pada induk 2 yang akan dipindah silangkan

r= bilangan random[0 1]

k= posisi gen yang akan dilakukan pindah silang

Contoh 2. 7:

Induk 1 = 1,2 0,2 1,5 1,2 0,2

Induk 2 = 2,0 1,0 1,0 0,2 2,5

Pada kedua induk tersebut akan dilakukan pindah silang pada gen ke 2 yaitu 0,2 pada induk 1 dan 1,0 pada induk 2 dengan nilai r = 0,4, maka nilai baru dari gen tersebut:

$$x_1'(k) = r.x_1(k) + (1-r).x_2(k) = (0{,}4)(0{,}2) + (0{,}6)(1{,}0) = 0{,}68$$

$$x_2'(k) = r.x_2(k) + (1-r).x_1(k) = (0{,}4)(1{,}0) + (0{,}6)(0{,}2) = 0{,}52$$

Dari nilai yang diperoleh tersebut, maka dihasilkan induk baru yaitu:

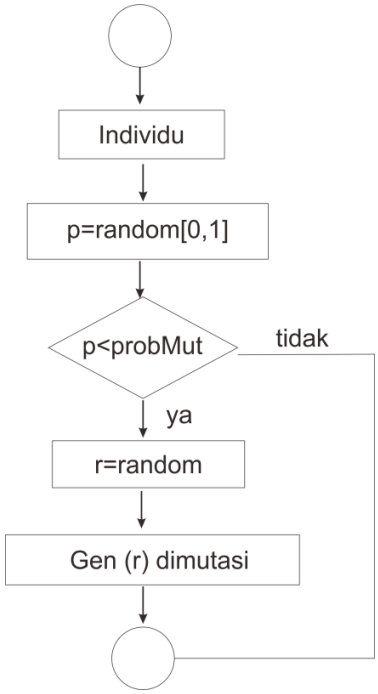
Induk 1 = 1,2 0,68 1,5 1,2 0,2

Induk 2 = 2,0 0,52 1,0 0,2 2,5

f. Mutasi

Mutasi ini berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi (Kusumadewi S. , 2003: 296). Kromosom anak dimutasi dengan menambahkan nilai random yang sangat kecil dengan probabilitas yang rendah. Peluang mutasi (p_m) didefinisikan sebagai presentasi dari jumlah total gen pada

populasi yang mengalami mutasi. Peluang mutasi yang baik berada pada interval 0 sampai dengan 0,3.



Gambar 2. 20. Diagram Alir Proses Mutasi

Pada mutasi ini ada 2 macam mutasi yaitu :

1) Mutasi Biner

Cara sederhana untuk mendapatkan mutasi biner adalah dengan mengganti satu atau beberapa nilai gen dari kromosom. Pada kromosom biner, mutasi dilakuka dengan mengubah gen biner 0 menjadi 1 dan 1 menjadi 0.

2) Mutasi Real

Pada mutasi real ada dua macam mutasi yang dapat dilakukan yaitu *random mutation* dan *shift mutation*(Adhy & Kushartantya, 2010, hal. 35).

a) *Random mutation* adalah mengganti gen dengan yang termutasi dengan nilai acak.

$$a(k) = random \tag{2.37}$$

Contoh 2. 8:

Individu

1,2	0,2	1,5	1,2	0,2

Dibangkitkan nilai p random [0,1], misalnya 0,2664 dan nilai probabilitas mutasi (p_m) 0,3542 maka nilai p kurang dari nilai p_m. Selanjutnya akan dilakukan mutasi yaitu dengan memilih posisi gen secara acak, misalnya gen yang terpilih gen nomor 2, maka nilai 0,2 diganti dengan nilai random. Misalnya nilai random yang muncul 0,6543, maka nilai 0,2 diganti dengan 0,6. Individu baru yang diperoleh.

1,2	0,6	1,5	1,2	0,2

b) *Shift mutation* adalah menggeser nilai gen termutasi sebesar ε, dimana ε adalah bilangan kecil yang ditentukan.

$$a(k) = a(k) + \varepsilon \tag{2.38}$$

Contoh 2. 9:

Individu

1,2	0,2	1,5	1,2	0,2

Dibangkitkan nilai p random [0,1], misalnya 0,2664 dan nilai probabilitas mutasi (p_m) 0,3542 maka nilai p kurang dari nilai p_m. Selanjutnya akan dilakukan mutasi yaitu dengan memilih posisi gen secara acak, misalnya gen yang terpilih gen nomor

2, maka nilai 0,2 diganti dengan menggeser nilai gen yang termutasi tersebut sebesar epsilon. Misalnya nilai epsilon yang ditentukan 0,1 kemudian nilai 0,2 ditambah dengan 0,1. Maka, nilai 0,2 diganti dengan 0,3. Individu baru yang diperoleh.

1,2	0,3	1,5	1,2	0,2

g. Elitisme

Pada proses seleksi yang dilakukan secara random, tidak ada jaminan bahwa suatu individu bernilai *fitness* tertinggi akan selalu terpilih. Jika individu bernilai *fitness* tertinggi terpilih, mungkin saja individu tersebut akan rusak (nilai *fitness*nya menurun) karena proses pindah silang. Untuk menjaga agar individu bernilai *fitness* tertinggi tidak hilang selama evolusi, maka perlu dibuat satu atau beberapa kopinya. Proses ini dikenal sebagai *elitisme* (Suyanto, 2005: 14)

h. Pembentukan Populasi Baru

Proses pembentukan populasi baru bertujuan untuk membentuk populasi baru yang berbeda dengan populasi awal. Pembentukan populasi baru ini didasarkan pada hasil proses mutasi ditambah dengan individu yang terbaik setelah dipertahankan dalam proses elitisme.

2. Algoritma Genetika Sederhana

Misalkan P(generasi) adalah populasi dari satu generasi, maka langkah-langkah algoritma genetika secara sederhana antara lain langkah awal yaitu menetapkan generasi awal (generasi = 0). Langkah selanjutnya yaitu inisialisasi populasi awal secara acak (P(generasi)). Kemudian mengevaluasi nilai *fitness* pada setiap individu

dalam populasi (P(generasi)). Setelah itu dilakukan langkah-langkah selanjutnya sampai generasi mencapai maksimum generasi.

Langkah-langkah selanjutnya tersebut yaitu generasi ditambah 1 (menambah generasi). Kemudian dilakukan operator seleksi populasi untuk mendapatkan induk (P'(generasi)). Setelah itu dilakukan *crossover* pada induk yang terseleksi tersebut (P'(generasi)). Kemudian dilakukan mutasi pada P'(generasi). Setelah itu individu yang diperoleh dalam P'(generasi) dievaluasi nilai *fitness*nya. Setelah dilakukan evaluasi nilai *fitness* kemudian dibentuk populasi baru yaitu P(generasi) sama dengan {P(generasi-1) yang *survive*,P'(generasi) (Kusumadewi S. , 2003, 297).

E. Logika *Fuzzy*

Dalam melakukan peramalan IHSG menggunakan FFNN dengan algoritma genetika digunakan logika *fuzzy* untuk nilai *input*nya. Logika *fuzzy* pertama kali diperkenalkan oleh Prof. Lotfi A. Zadeh pada tahun 1965. Logika *fuzzy* akhir-akhir ini banyak digunakan dalam berbagai bidang karena konsep logika *fuzzy* mudah dimengerti. Selain itu logika *fuzzy* sangat fleksibel, artinya mampu beradaptasi dengan perubahan-perubahan dan ketidakpastian yang menyertai permasalan. Logika *fuzzy* juga mempunyai toleransi terhadap data yang tidak tepat, logika *fuzzy* mampu memodelkan fungsi-fungsi nonlinear yang sangat kompleks. Logika *fuzzy* juga dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung, logika *fuzzy* dapat bekerja sama dengan teknik-teknik kendali secara

konvensional. Serta logika *fuzzy* didasarkan pada bahasa alami(Kusumadewi & Purnomo, 2013: 1-2).

Logika *fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang *input* menuju ke ruang *output.*Beberapa proses logika *fuzzy* seperti himpunan *fuzzy* dan fungsi keanggotaan.

1. Himpunan *Fuzzy*

Pada himpunan tegas (*crisp*), nilai-nilai keanggotaan suatu elemen$x$ dalam himpunan A, yang sering ditulis dengan $\mu_A(x)$, memiliki dua kemungkinan yaitu (Kusumadewi & Purnomo, 2013: 3):

a. satu (1), yang berarti suatu elemen menjadi anggota dalam suatu himpunan

b. nol (0), yang berarti bahwa suatu elemen tidak menjadi anggota dalam suatu himpunan.

Misalnya jika diketahui $S = \{1,2,3,4\}$ adalah semesta pembicaraan, $A = \{1,2\}$ dan $B = \{3,4\}$. Dapat dikatakan bahwa nilai keanggotaan 2 pada himpunan A, $\mu_A(2) = 1$ karena $2 \in A$. Nilai keanggotaan 3 pada himpunan A, $\mu_A(3) = 0$ karena $3 \notin A$.

Teori himpunan *fuzzy* merupakan perluasan teori himpunan klasik (*crisp*). Pada teori himpunan *fuzzy* dikatakan bahwa sebuah himpunan *fuzzy* dalam semesta pembicaraan S didefinisikan sebagaisuatu fungsi keanggotaan $\mu_A(x)$ yang berada dalam interval $[0\ 1]$(Wang, 1997: 21).

Himpunan *fuzzy* A dalam X dapat direpresentasikan sebagai himpunan pasangan berurutan(Zimmermann, 1996: 12):

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

dengan $\mu_A(x)$ merupakan derajat keanggotaan x di A yang memetakan X ke ruang keanggotaan M yang terletak pada interval [0 1].

Himpunan *fuzzy* memiliki 2 atribut yaitu (Kusumadewi & Purnomo, 2013: 6):

a. Linguistik, yaitu penanaman suatu grup yang mewakili suatu keadaan atau kondisi tertentu dengan menggunakan bahasa alami seperti RENDAH, SEDANG, TINGGI.

b. Numeris, yaitu suatu nilai yang menunjukkan ukuran dari suatu variabel, seperti 40, 25, 50, dsb.

Ada beberapa hal yang perlu diketahui dalam memahami sistem *fuzzy* yaitu:

a. Variabel *fuzzy*

Variabel *fuzzy* merupakan variabel yang akan dibahas dalam suatusistem *fuzzy*.

Contoh 2.10. Variabel *fuzzy* dalam tugas akhir ini yaituIHSG, Tingkat Bunga, Tingkat Inflasi, dan Nilai Tukar.

b. Himpunan *fuzzy*

Himpunan *fuzzy* merupakan suatu grup yang mewakili suatu kondisi atau keadaan tertentu dalam suatu variabel *fuzzy*.

Contoh 2.11. Variabel IHSG terbagi menjadi 1 himpunan *fuzzy* yaitu A_1.

c. Semesta pembicaraan (Himpunan Universal)

Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variabel *fuzzy*. Semesta pembicaraan merupakan himpunan bilangan real yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai semesta pembicaraan dapat berupa nilai positif atau negatif. Adakalanya nilai semesta pembicaraan ini tidak dibatasi batas atasnya.

Contoh 2.12. Semesta pembicaraan untuk variabel IHSG [1.240,85 5.516,80].

d. Domain

Domain himpunan *fuzzy* adalah keseluruhan nilai yang diizinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *fuzzy*.

Contoh 2.13. Domain himpunan *fuzzy* A_1 yaitu [1.140,85 5.616,8].

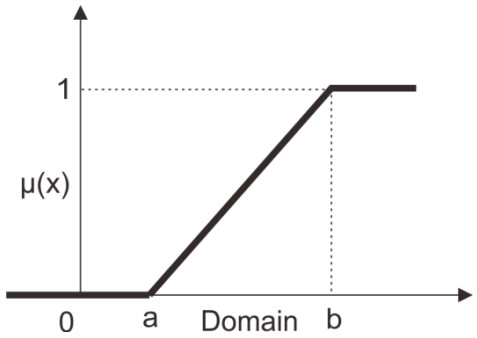
2. Fungsi Keanggotaan

Fungsi keanggotaan (*membership function*) adalah suatu kurva yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaannya (derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Adapun beberapa fungsi yang bisa digunakan (Kusumadewi & Purnomo, 2013: 8):

a. Representasi Linear

Pada representasi linear, pemetaan *input* ke derajat keanggotaannya digambarkan sebagai suatu garis lurus. Bentuk ini paling sederhana dan menjadi pilihan yang baik

untuk mendekati suatu konsep yang kurang jelas. Ada 2 keadaan himpunan *fuzzy* yang linear. Pertama, kenaikan himpunan dimulai pada nilai domain yang memiliki derajat keanggotaan nol (0) bergerak ke kanan menuju nilai domain yang memiliki derajat keanggotaan lebih tinggi. Gambar 2.21 merupakan representasi linear naik.



Gambar 2. 21. Representasi Linear Naik

Fungsi keanggotaannya yaitu :

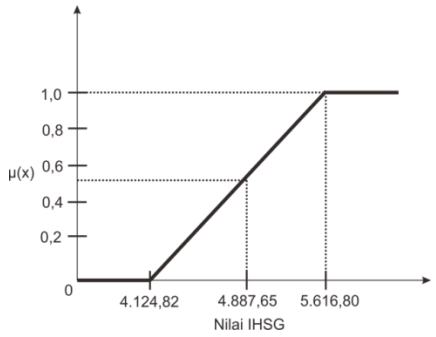
$$\mu(x) = \begin{cases} 0; & x \leq a \\ \frac{x-a}{b-a}; & a \leq x \leq b \\ 1; & x \geq b \end{cases} \tag{2.39}$$

Contoh 2. 14. Salah satu himpunan *fuzzy* nilai IHSG adalah TINGGI dengan himpunan universal $\boldsymbol{U = [1.140,85\ 5.616,80]}$ yang mempunyai fungsi keanggotaan:

$$\mu_{TINGGI}(x) = \begin{cases} 0; & x \leq 4.124,82 \\ \frac{x - 4.124,82}{1.491,98}; & 4.124,82 \leq x \leq 5.616,80 \\ 1; & x \geq 5.616,80 \end{cases}$$

Berdasarkan fungsi keanggotaan tersebut, sebagai contoh menentukan derajat keanggotaan nilai IHSG sebesar 4.877,65 sehingga dapat dilakukan perhitungan:

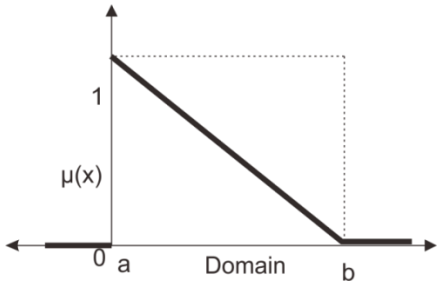
$$\mu_{PANAS}(4.877,65) = \frac{4.877,65 - 4.124,82}{1.491,98} = 0,5045$$



Gambar 2. 22. Grafik fungsi keanggotaan himpunan *fuzzy* IHSG TINGGI

Gambar 2.22 merupakan representasi linear naik himpunan *fuzzy* IHSG TINGGI. Dapat diperoleh kesimpulan bahwa derajat keanggotaan nilai IHSG sebesar 4.877,65 adalah 0,5045 pada himpunan *fuzzy* TINGGI. Sehingga nilai IHSG sebesar 4.877,65 merupakan anggota himpunan *fuzzy* dengan nilai keanggotaan sebesar 0,5045.

Kedua, garis lurus dimulai dari nilai domain dengan derajat keanggotaan tertinggi pada sisi kiri, kemudian bergerak menurun ke nilai domain yang memiliki derajat keanggotaan lebih rendah. Gambar 2.23 merupakan representasi linear turun.



Gambar 2. 23. Representasi Linear Turun

Fungsi keanggotaannya yaitu:

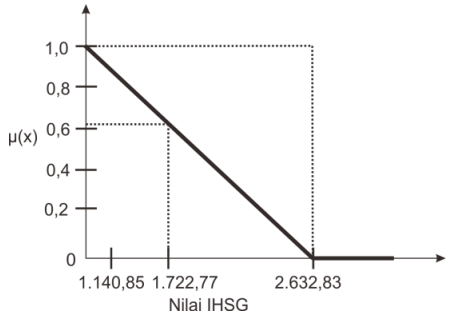
$$\mu(x) = \begin{cases} \frac{b-x}{b-a}; & a \leq x \leq b \\ 0; & x \geq b \end{cases} \quad (2.40)$$

Contoh 2. 15. Salah satu himpunan *fuzzy* nilai IHSG adalah RENDAH dengan himpunan universal $\boldsymbol{U} = [\mathbf{1.140,85}\ \mathbf{5.616,80}]$ yang mempunyai fungsi keanggotaan:

$$\mu_{DINGIN}(x) = \begin{cases} \dfrac{2.632,83 - x}{1.491,98}; & 1.140,85 \leq x \leq 2.632,83 \\ 0; & x \geq 2.632,83 \end{cases}$$

Berdasarkan fungsi keanggotaan tersebut, sebagai contoh menentukan derajat keanggotaan nilai IHSG sebesar 1.722,77 sehingga dapat dilakukan perhitungan:

$$\mu_{PANAS}(1.722,77) = \frac{2.632,83 - 1.722,77}{1.491,98} = 0,6099$$

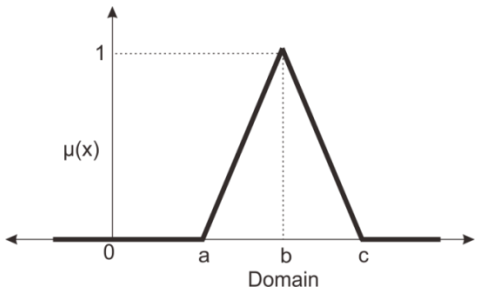


Gambar 2. 24. Grafik fungsi keanggotaan himpunan *fuzzy* IHSG RENDAH

Gambar 2.24 merupakan representasi linear turun himpunan *fuzzy* IHSG RENDAH. Dapat diperoleh kesimpulan bahwa derajat keanggotaan nilai IHSG sebesar 1.722,77 adalah 0,6099 pada himpunan *fuzzy* RENDAH. Sehingga nilai IHSG sebesar 1.722,77 merupakan anggota himpunan *fuzzy* dengan nilai keanggotaan sebesar 0,6099.

b. Representasi Kurva Segitiga

Kurva segitiga pada dasarnya merupakan gabungan antara 2 garis (linear). Gambar 2.25 merupakan representasi kurva segitiga.



Gambar 2. 25. Representasi Kurva Segitiga

Fungsi keanggotaan dari representasi kurva segitiga yaitu:

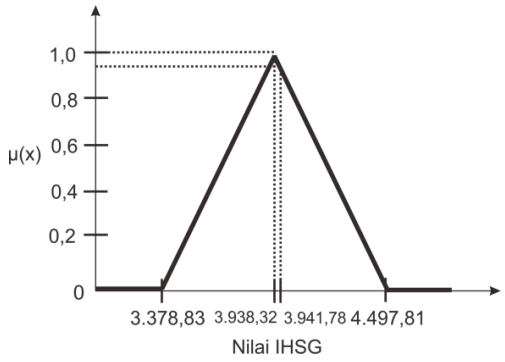
$$\mu(x) = \begin{cases} 0; & x \leq a \; atau \; x \geq c \\ \frac{x-a}{b-a}; & a \leq x \leq b \\ \frac{c-x}{c-b}; & b \leq x \leq c \end{cases} \tag{2.41}$$

Contoh 2. 16. Salah satu himpunan *fuzzy* nilai IHSG adalah SEDANG dengan himpunan universal $\boldsymbol{U = [1.140,85\ 5.616,80]}$ yang mempunyai fungsi keanggotaan:

$$\mu_{SEDANG}(x) = \begin{cases} 0; & x \leq 3.378{,}83 \; atau \; x \geq 4.497{,}81 \\ \frac{x - 3.378{,}83}{559{,}49}; & 3.378{,}83 \leq x \leq 3.938{,}32 \\ \frac{4.497{,}81 - x}{559{,}49}; & 3.938{,}32 \leq x \leq 4.497{,}81 \end{cases}$$

Berdasarkan fungsi keanggotaan tersebut, sebagai contoh menentukan derajat keanggotaan nilai IHSG sebesar 3.941,78 sehingga dapat dilakukan perhitungan:

$$\mu_{SEDANG}(3.941{,}78) = \frac{4.497{,}81 - 3.941{,}78}{559{,}49} = 0{,}9938$$

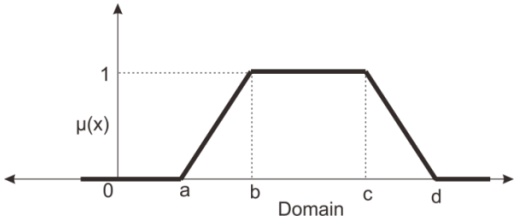


Gambar 2. 26. Grafik fungsi keanggotaan himpunan *fuzzy* IHSG SEDANG

Gambar 2.26 merupakan representasi kurva segitiga himpunan *fuzzy* IHSG SEDANG. Dapat diperoleh kesimpulan bahwa derajat keanggotaan nilai IHSG sebesar 3.941,78 adalah 0,9938 pada himpunan *fuzzy* SEDANG. Sehingga nilai IHSG sebesar 3.941,78 merupakan anggota himpunan *fuzzy* dengan nilai keanggotaan sebesar 0,9938.

c. Representasi Kurva Trapesium

Kurva trapesium pada dasarnya seperti bentuk segitiga, hanya saja ada beberapa titik yang memiliki nilai keanggotaan 1. Gambar 2.27 merupakan representasi kurva trapesium.



Gambar 2. 27. Representasi Kurva Trapesium

Fungsi keanggotaan dari kurva trapesium yaitu:

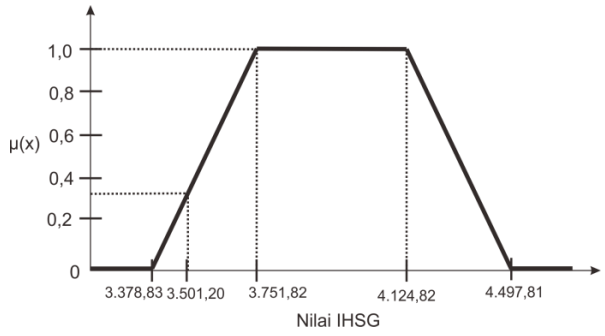
$$\mu(x) = \begin{cases} 0; & x \leq a \; atau \; x \geq d \\ \frac{x-a}{b-a}; & a \leq x \leq b \\ 1 & b \leq x \leq c \\ \frac{d-x}{d-c} & c \leq x \leq d \end{cases} \tag{2.42}$$

Contoh 2. 17.Salah satu himpunan *fuzzy* nilai IHSG adalah SEDANG dengan himpunan universal $\boldsymbol{U} = [\mathbf{1.140,85\ 5.616,80}]$ yang mempunyai fungsi keanggotaan:

$$\mu_{SEDANG}(x) = \begin{cases} 0; & x \leq 3.378,83 \; atau \; x \geq 4.497,81 \\ \frac{x - 3.378,83}{372,99}; & 3.378,83 \leq x \leq 3.751,82 \\ 1 & 3.751,82 \leq x \leq 4.124,82 \\ \frac{4.497,81 - x}{372,99} & 4.124,82 \leq x \leq 4.497,81 \end{cases}$$

Berdasarkan fungsi keanggotaan tersebut, sebagai contoh menentukan derajat keanggotaan nilai IHSG sebesar 3.501,20 sehingga dapat dilakukan perhitungan:

$$\mu_{SEDANG}(3.501,20) = \frac{3.501,20 - 3.378,83}{372,99} = 0,3281$$



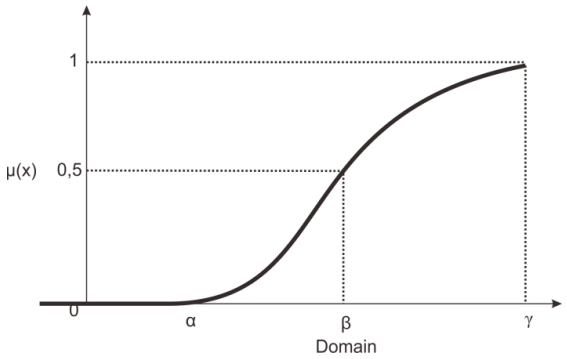
Gambar 2. 28. Grafik fungsi keanggotaan himpunan *fuzzy* IHSG SEDANG

Gambar 2.28 merupakan representasi kurva trapesium himpunan *fuzzy* IHSG SEDANG. Dapat diperoleh kesimpulan bahwa derajat keanggotaan nilai IHSG

sebesar 3.501,20 adalah 0,3281 pada himpunan *fuzzy* SEDANG. Sehingga nilai IHSG sebesar 3.501,20 merupakan anggota himpunan *fuzzy* dengan nilai keanggotaan sebesar 0,3281.

d. Representasi Kurva-S

Kurva pertumbuhan dan penyusutan merupakan kurva-S atau sigomoid yang berhubungan dengan kenaikan dan permukaan secara tak linear. Kurva-S untuk pertumbuhan akan bergerak dari sisi paling kiri (nilai keanggotaan=0) ke sisi paling kanan (nilai keanggotaan=1). Sedangkan, kurva-S penyusutan akan bergerak dari sisi paling kiri (nilai keanggotaan=1) ke sisi paling kanan (nilai keanggotaan=0). Gambar 2.29 merupakan representasi kurva-S pertumbuhan.



Gambar 2. 29. Representasi Kurva-S Pertumbuhan

Fungsi keanggotaan untuk kurva-S pertumbuhan adalah:

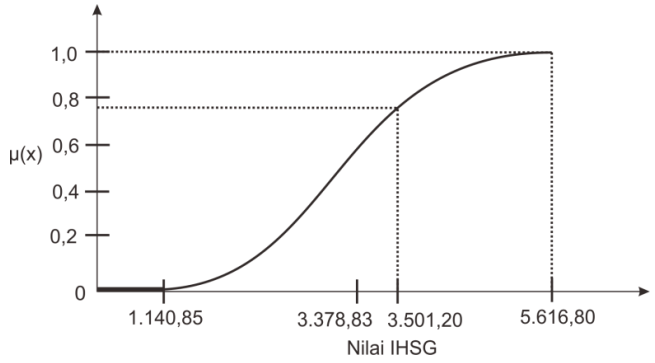
$$S(x;\alpha,\beta,\gamma) = \begin{cases} 0 & x \le \alpha \\ 2\left[\frac{(x-\alpha)}{(\gamma-\alpha)}\right]^2 & \alpha \le x \le \beta \\ 1-2\left[\frac{(\gamma-x)}{(\gamma-\alpha)}\right]^2 & \beta \le x \le \gamma \\ 1 & x \ge \gamma \end{cases} \quad (2.43)$$

Contoh 2. 18. Salah satu himpunan *fuzzy* nilai IHSG adalah TINGGI dengan himpunan universal $\boldsymbol{U} = [\mathbf{1.140,85\ 5.616,80}]$ yang mempunyai fungsi keanggotaan:

$$S(x;\alpha,\beta,\gamma) = \begin{cases} 0 & x \leq 1.140{,}85 \\ 2\left[\dfrac{(x-1.140{,}85)}{4.475{,}95}\right]^2 & 1.140{,}85 \leq x \leq 3.378{,}83 \\ 1-2\left[\dfrac{(5.616{,}8-x)}{4.475{,}95}\right]^2 & 3.378{,}83 \leq x \leq 5.616{,}8 \\ 1 & x \geq 5.616{,}8 \end{cases}$$

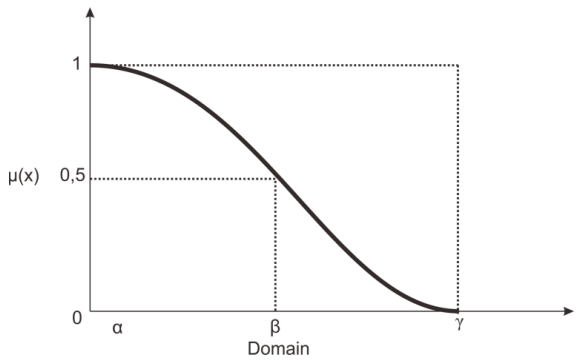
Berdasarkan fungsi keanggotaan tersebut, sebagai contoh menentukan derajat keanggotaan nilai IHSG sebesar 3.501,20 sehingga dapat dilakukan perhitungan:

$$\mu_{TINGGI}(3.501{,}20) = 1-2\left[\frac{(5.616{,}8-3.501{,}20)}{4.475{,}95}\right]^2 = 0{,}7765$$



Gambar 2. 30. Grafik fungsi keanggotaan himpunan *fuzzy* IHSG TINGGI

Gambar 2.30 merupakan representasi kurva-S pertumbuhan himpunan *fuzzy* IHSG TINGGI. Dapat diperoleh kesimpulan bahwa derajat keanggotaan nilai IHSG sebesar 3.501,20 adalah 0,7765 pada himpunan *fuzzy* TINGGI. Sehingga nilai IHSG sebesar 3.501,20 merupakan anggota himpunan *fuzzy* dengan nilai keanggotaan sebesar 0,7765.



Gambar 2. 31. Representasi Kurva-S Penyusutan

Gambar 2.31 merupakan representasi kurva-S penyusutan. Sedangkan untuk fungsi keanggotaan untuk kurva-S penyusutan adalah:

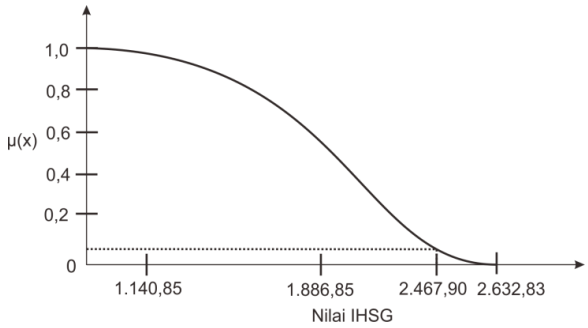
$$S(x;\alpha,\beta,\gamma)=\begin{cases}1 & x\le\alpha\\ 1-2\left[\frac{(x-\alpha)}{(\gamma-\alpha)}\right]^2 & \alpha\le x\le\beta\\ 2\left[\frac{(\gamma-x)}{(\gamma-\alpha)}\right]^2 & \beta\le x\le\gamma\\ 0 & x\ge\gamma\end{cases} \tag{2.44}$$

Contoh 2. 19. Salah satu himpunan *fuzzy* nilai IHSG adalah RENDAH dengan himpunan universal U=[1.140,85 5.616,80] yang mempunyai fungsi keanggotaan:

$$S(x;\alpha,\beta,\gamma)=\begin{cases}1 & x\le 1.140{,}85\\ 1-2\left[\frac{(x-1.140{,}85)}{1.491{,}98}\right]^2 & 1.140{,}85\le x\le 1.886{,}85\\ 2\left[\frac{(2.632{,}83-x)}{1.491{,}98}\right]^2 & 1.886{,}85\le x\le 2.632{,}83\\ 0 & x\ge 2.632{,}83\end{cases}$$

Berdasarkan fungsi keanggotaan tersebut, sebagai contoh menentukan derajat keanggotaan nilai IHSG sebesar 2.467,90 sehingga dapat dilakukan perhitungan:

$$\mu_{RENDAH}(2.467{,}90)=2\left[\frac{(2.632{,}83-2.467{,}90)}{1.491{,}98}\right]^2=0{,}0122$$

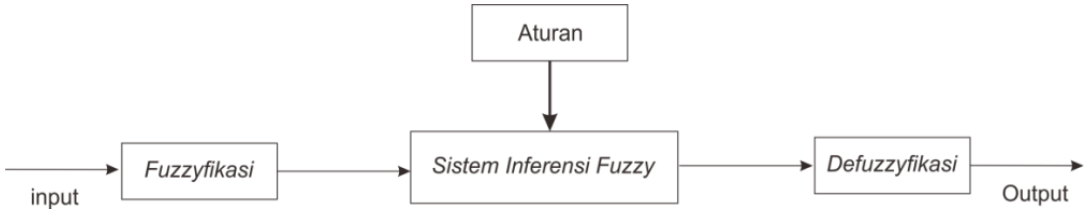


Gambar 2. 32. Grafik fungsi keanggotaan himpunan *fuzzy* IHSG RENDAH

Gambar 2.32 merupakan representasi kurva-S penyusutan himpunan *fuzzy* IHSG TINGGI. Dapat diperoleh kesimpulan bahwa derajat keanggotaan nilai IHSG sebesar 2.467,90 adalah 0,0122 pada himpunan *fuzzy* RENDAH. Sehingga nilai IHSG sebesar 2.467,90 merupakan anggota himpunan *fuzzy* dengan nilai keanggotaan sebesar 0,0122.

3. Susunan Sistem *Fuzzy*

Susunan sistem *fuzzy* terlihat pada gambar 2.33.



Gambar 2. 33. Susunan Sistem *Fuzzy*

Ada tiga tahapan dalam sistem *fuzzy* yaitu (Wang, 1997, hal. 7):

a. *Fuzzyfikasi*

*Fuzzyfikasi*merupakan transformasi variabel nilai real ke dalam himpunan *fuzzy.Fuzzyfikasi*ini berdasarkan dengan fungsi keanggotaan yang digunakan dalam himpunan *fuzzy.*

b. Inferensi *Fuzzy*

Sistem inferensi *fuzzy* adalah melakukan penalaran menggunakan *fuzzy input* dan aturan *fuzzy* yang telah ditentukan sehingga menghasilkan *fuzzy output*.

c. De*fuzzyfikasi*

De*fuzzyfikasi*merupakan transformasi himpunan *fuzzy* ke dalam variabel nilai real untuk menjadi *output*. Proses de*fuzzyfikasi* ini juga menggunakan fungsi keanggotaan yang sama dengan proses *fuzzyfikasi*.