BAB II

TINJAUAN PUSTAKA

2.1 Peramalan (*Forecasting*)

2.1.1 Pengertian Peramalan

Peramalan adalah metode untuk memperkirakan suatu nilai di masa depan dengan menggunakan data masa lalu. Peramalan juga dapat diartikan sebagai seni dan ilmu untuk memperkirakan kejadian pada masa yang akan datang, sedangkan aktivitas peramalan merupakan suatu fungsi bisnis yang berusaha memperkirakan penjualan dan penggunaan suatu produk sehingga produk-produk itu dapat dibuat dalam kuantitas yang tepat (Gaspersz, 2002:71).

Menurut Handoko (1999) Peramalan adalah suatu usaha untuk meramalkan keadaan di masa mendatang melalui pengujian keadaan di masa lalu. Jadi, dapat disimpulkan bahwa peramalan merupakan seni dan ilmu yang digunakan untuk memprediksi nilai di masa yang akan datang berdasarkan data pada masa lampau dengan menggunakan perhitungan matematis.

2.1.2 Jenis-Jenis Peramalan

Peramalan biasanya diklasifikasikan berdasarkan horizon waktu masa depan yang dicakupnya. Menurut Taylor (2004) dalam hubungannya dengan horizon waktu peramalan terbagi atas beberapa kategori, yaitu:

1) Peramalan jangka panjang, umumnya peramalan dilakukan untuk meramalkan 2 sampai 10 tahun yang akan datang. Peramalan ini digunakan untuk perencanaan produk dan perencanaan sumber daya.

2) Peramalan jangka menengah, umumnya peramalan dilakukan untuk meramalkan 1 sampai 24 bulan yang akan datang. Peramalan ini lebih mengkhusus dibandingkan peramalan jangka panjang, biasanya digunakan untuk menentukan aliran kas, perencanaan produksi, dan penentuan anggaran.
3) Peramalan jangka pendek umumnya peramalan dilakukan untuk meramalkan 1 sampai 5 minggu ke depan. Peramalan ini biasanya digunakan untuk mengambil keputusan dalam hal perlu tidaknya lembur, penjadwalan kerja, dan lain-lain keputusan kontrol jangka pendek.

Menurut Maulidah, (2012: 4) jika dilihat dari sifat ramalan yang telah disusun, maka peramalan dapat dibedakan atas dua macam, yaitu:

1) Peramalan kualitatif, yaitu peramalan yang didasarkan atas data kualitatif masa lalu. Hasil peramalan yang ada tergantung pada orang yang menyusunnya, karena peramalan tersebut sangat ditentukan oleh pemikiran yang bersifat intuisi, *judgement* (pendapat) dan pengetahuan serta pengalaman dari penyusunnya. Metode kualitatif dibagi menjadi dua metode, yaitu:
   a) Metode eksploratif

      Pada metode ini dimulai dengan masa lalu dan masa kini sebagai awal dan bergerak ke arah masa depan secara heuristik, sering kali dengan melihat semua kemungkinan yang ada.
   b) Metode normatif

      Pada metode ini dimulai dengan menetapkan sasaran tujuan yang akan datang, kemudian bekerja mundur untuk melihat apakah hal ini dapat dicapai berdasarkan kendala, sumber daya dan teknologi yang tersedia.

2) Peramalan kuantitatif, yaitu peramalan yang didasarkan atas data kuantitatif pada masa lalu. Hasil peramalan yang dibuat tergantung pada metode yang digunakan dalam peramalan tersebut. Metode yang baik adalah metode yang memberikan nilai-nilai perbedaan atau penyimpangan yang mungkin.

   Metode peramalan kuantitatif terbagi atas dua jenis model peramalan yang utama, yaitu:

   a) Model deret berkala (*time series*), merupakan metode peramalan yang didasarkan atas penggunaan analisa pola hubungan antara variabel yang akan diperkirakan dengan variabel waktu, yang merupakan deret waktu.

   b) Model kausal, yaitu metode peramalan yang didasarkan atas penggunaan analisa pola hubungan antara variabel lain yang mempengaruhinya, yang bukan waktu yang disebut metode korelasi atau sebab akibat. Model kausal terdiri dari:

      1. Metode regresi dan korelasi
      2. Metode ekonometri
      3. Metode *input* dan *output*

2.1.3 Langkah-Langkah Peramalan

Menurut Gaspersz (2002) terdapat 9 langkah yang harus diperhatikan untuk menjamin efektivitas dan efisiensi dari sistem peramalan, yaitu:

1. Menentukan tujuan dari peramalan
2. Memilih item *independent demand* yang akan diramalkan
3. Menentukan horison waktu dari peramalan (jangka pendek, menengah, atau panjang)

4. Memilih model-model peramalan
5. Memperoleh data yang dibutuhkan untuk melakukan peramalan
6. Validasi model peramalan
7. Membuat peramalan
8. Implementasi hasil-hasil peramalan
9. Memantau keandalan hasil peramalan

2.2 Artifical Neural Network (ANN)

Artifical Neural Network atau jaringan saraf tiruan merupakan suatu konsep rekayasa pengetahuan dalam bidang kecerdasan buatan yang didesain dengan mengadopsi system saraf manusia, yang pemrosesan utamanya ada di otak. Bagian terkecil dari otak manusia adalah sel saraf yang disebut unit dasar pemroses informasi atau neuron. Ada sekitar 10 miliar neuron dalam otak manusia dan sekitar 60 triliun koneksi (disebut sinaps [*synapse*]) antar neuron dalam otak manusia. Dengan menggunakan neuron-neuron tersebut secara simultan, otak manusia dapat memproses informasi secara paralel dan cepat, bahkan lebih cepat dari komputer tercepat saat ini.

Sebuah neuron terdiri atas elemen-elemen berikut: badan sel (disebut soma), sejumlah serat yang menyalurkan informasi ke neuron (disebut dendrit), dan sebuah serat tunggal yang keluar dari neuron (disebut akson). Setiap siyal luar yang diterima oleh dendrit akan melewati sinaps untuk diteruskan ke neuron, kemudian diproses di dalam soma. Setelah selesai akan dikeluarkan melalui akson untuk diproses kembali oleh neuron yang lain ataupun keluar sebagai sinyal akhir hasil proses di otak.

Dengan analogi system kerja otak manusia tersebut, ANN terdiri atas sebuah unit pemrosesan yang disebut neuron (akson kalua dalam otak manusia) yang berisi penambah (*adder*) dan fungsi aktivasi, sejumlah bobot (sinaps dalam otak manusia), sejumlah vector masukan (dendrit dalam otak manusia). Fungsi aktivasi berguna untuk mengatur keluaran yang diberikan oleh neuron. Fungsi aktivasi menggunakan sebuah nilai ambang batas (*threshold*) untuk membatasi nilai keluaran agar selalu dalam batas yang ditetapkan.

Seperti halnya manusia yang otaknya selalu belajar dari lingkungan sehingga dapat mengelolalingkungan dengan baik berdasarkan pengalaman yang sudah didapatkan. ANN yang dalam *data mining* dianggap sebagai model yang digunakan untuk proses prediksi, membutuhkan proses pelatihan agar dapat melakukan prediksi kelas suatu data uji baru yang ditemukan. Proses pelatihan dalam ANN dapat menggunakan algoritma-algoritma seperti *Perceptron, Backpropagation, Self-Organizing Map* (SOM), Delta, *Associate Memory, Learning Vector Quantization,* dan sebagainya.

Beberapa istilah dalam jaringan saraf tiruan yang sering ditemui antara lain:

1. Neuron atau node atau unit

   Sel saraf yang merupakan elemen pengolahan jaringan saraf tiruan. Setiap neuron menerima data *input*, memroses *input* tersebut (melakukan sejumlah perkalian dengan melibatkan summation function dan fungsi aktivasi), dan mengirimkan hasilnya berupa sebuah *output*.

2. Jaringan

   Kumpulan neuron yang saling terhubung dan membentuk lapisan.

3. *Input* atau masukan

   Berkoresponden dengan sebuah atribut tunggal dari sebuah pola atau data lain dari dunia latar. Sinyal-sinyal *input* ini kemudian diteruskan ke lapisan selanjutnya.

4. *Output* atau keluaran

   Solusi atau hasil pemahaman jaringan terhadap data *input*. Tujuan pembangunan jaringan saraf tiruan sendiri adalah untuk mengetahui nilai *output*.

5. Lapisan tersembunyi (*Hidden layer*)

   Lapisan yang tidak secara langsung berinteraksi dengan dunia luar. Lapisan ini memperluas kemampuan jaringan saraf tiruan dalam menghadapi masalah-masalah yang kompleks.

6. Bobot

   Bobot dalam jaringan saraf tiruan merupakan nilai matematis dari koneksi, yang mentransfer data dari satu lapisan ke lapisan lainnya. Bobot ini digunakan untuk mengatur jaringan sehingga jaringan saraf tiruan bisa menghasilkan *output* yang diinginkan sekaligus bertujuan membuat jaringan tersebut belajar.

7. *Summation function*

   Fungsi yang digunakan untuk mencari rata-rata bobot dari semua elemen *input*.

8. Fungsi aktivasi atau fungsi transfer

Fungsi yang meggambarkan hubungan antara tingkat aktivasi internal (*summation function*) yang mungkin berbentuk linier atau nonlinier.

9. Paradigma pembelajaran

   Cara pembelajaran atau pelatihan jaringan saraf tiruan yaitu apakah terawasi, tidak terawasi, atau merupakan gabungan keduanya (*hybrid*).

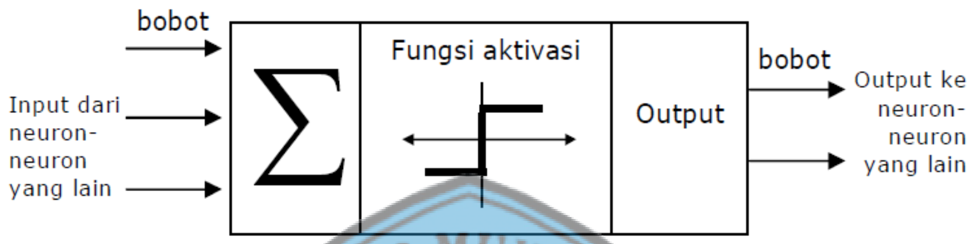
10. Aturan pembelajaran

    Aturan kerja secara umum dari teknik/algoritma jaringan saraf tiruan.

2.2.1 Komponen Jaringan Saraf Tiruan

Ada beberapa tipe jaringan saraf, namun hampir semuanya memiliki komponen-komponen yang sama. Seperti otak manusia, jaringan saraf tiruan juga terdiri dari beberapa neuron dan ada hubungan antara neuron-neuron tersebut. Neuron-neuron tersebut akan menstransformasikan informasi (*input*) yang diterima melalui sambungan keluarnya menuju neuron yang lainnya. Pada jaringan saraf tiruan, hubungan ini dikenal dengan nama bobot. Informasi disimpan pada suatu nilai tertentu pada bobot tersebut (Huda, 2014: 32).

Jika dilihat, neuron-neuron buatan tersebut bekerja dengan cara yang sama dengan neuron-neuron biologis. *Input* yang datang akan diproses oleh suatu fungsi perambatan dengan menjumlahkan nilai-nilai dari semua bobot tersebut. Hasil dari penjumlahan tersebut kemudian akan dibandingkan dengan suatu nilai ambang (*threshold*) melalui suatu fungsi aktivasi pada setiap neuron. Apabila *input* tersebut melewati suatu nilai ambang tertentu, maka neuron tersebut akan diaktifkan. Sebaliknya, jika *input* tidak terlewati suatu nilai ambang tertentu maka neuron tidak akan diaktifkan. Apabila neuron tersebut tidak diaktifkan maka neuron tersebut akan mengirimkan *output* melalui bobot-bobot *output*nya ke semua neuron yang berhubungan dengannya, begitu seterusnya (Kusumadewi dan Hartati, 2010: 70). Cara kerja neuron tersebut dapat digambarkan seperti pada Gambar 2.1.



Gambar 2.1 Struktur Neuron Jaringan

Pada jaringan saraf, neuron-neuron berada dalam lapisan-lapisan (layer) yang disebut lapisan neuron. Menurut Puspitaningrum (2006: 9), lapisan-lapisan penyusun jaringan saraf tiruan dibagi menjadi tiga, yaitu:

1) Lapisan *input*

   Neuron-neuron yang berada di dalam lapisan *input* disebut neuron-neuron *input*. Neuron-neuron ini menerima *input* dari luar. *Input* yang dimasukkan merupakan penggambaran dari suatu masalah.

2) Lapisan tersembunyi

   Neuron-neuron di dalam lapisan tersembunyi disebut neuron-neuron tersembunyi. *Output* dari lapisan ini tidak secara langsung dapat diamati.

3) Lapisan *output*

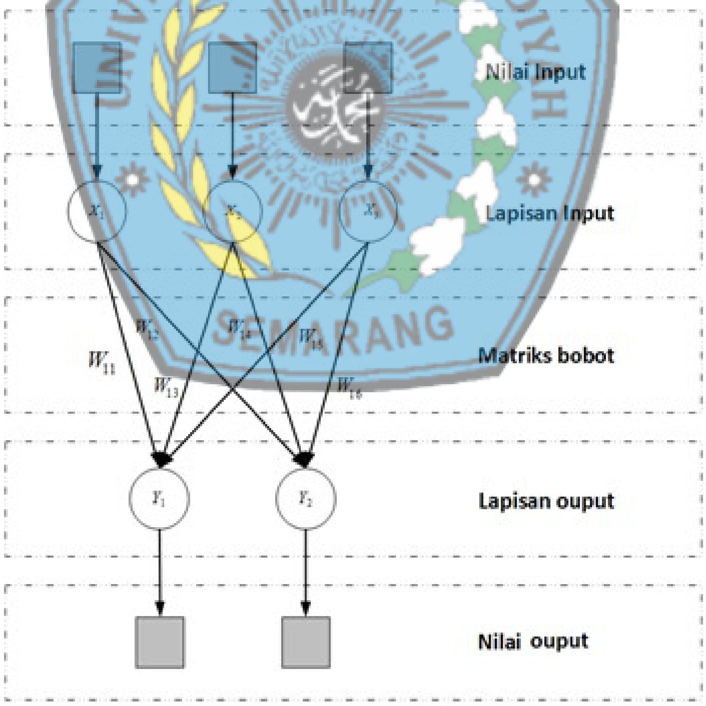
   Neuron-neuron pada lapisan *output* disebut neuron-neuron *output*. Keluaran atau *output* dari lapisan ini merupakan *output* jaringan saraf tiruan terhadap suatu permasalahan.

2.2.2 Arsitektur Jaringan Saraf Tiruan

Dalam jaringan saraf tiruan juga terdapat arsitektur jaringan. Arsitektur jaringan saraf terdiri atas 3 macam, yaitu:

1. Jaringan saraf dengan lapisan tunggal (*single layer net*)

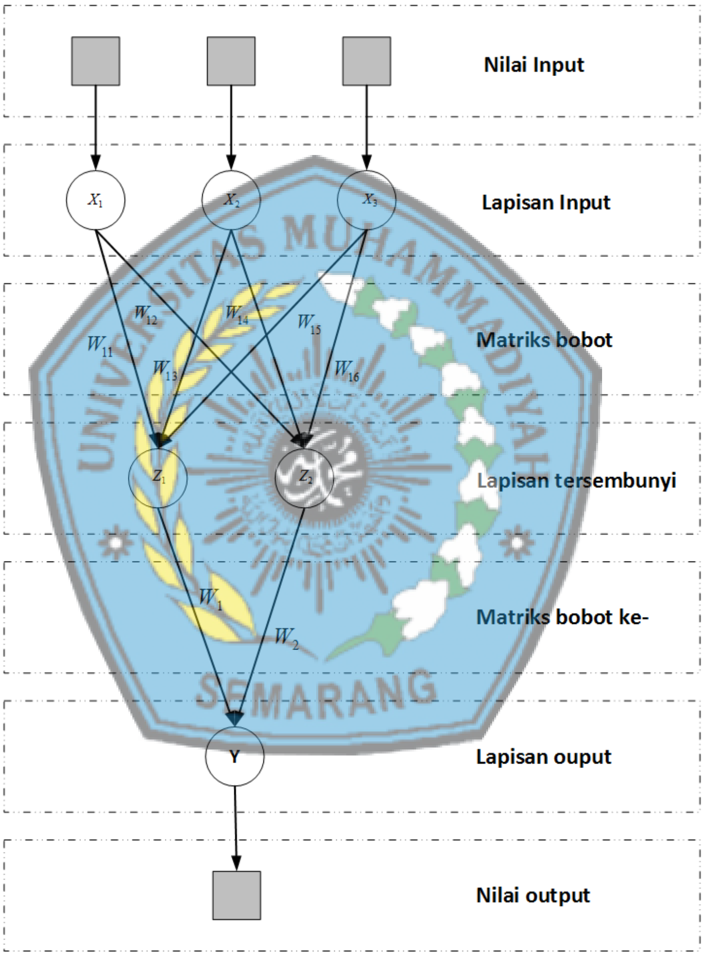
   Jaringan dengan lapisan tunggal hanya memiliki satu lapis dengan bobot-bobot terhubung. Jaringan ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi. Cara kerja jaringan saraf dengan lapisan tunggal digambarkan pada Gambar 2.2.



Gambar 2.2 Jaringan Saraf Tiruan dengan Lapisan Tunggal

2. Jaringan saraf dengan banyak lapisan (*multilayer net*)

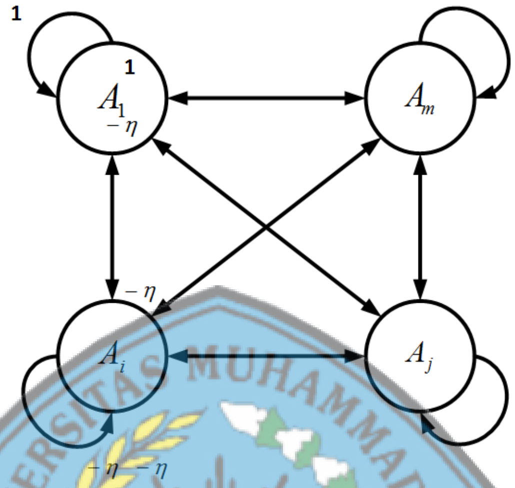
Jaringan saraf dengan banyak lapisan memiliki satu atau lebih lapisan yang terletak diantara lapisan *input* dan lapisan *output* (memiliki satu atau lebih lapisan tersembunyi). Jaringan ini dapat menyelesaikan permasalahan yang lebih sulit daripada jaringan dengan lapisan tunggal. Cara kerja jaringan saraf dengan banyak lapisan digambarkan pada Gambar 2.3.



Gambar 2.3 Jaringan Saraf Tiruan dengan Banyak Lapisan

3. Jaringan saraf dengan lapisan kompetitif (*competitive layer net*)

   Pada jaringan ini, sekumpulan neuron bersaing untuk mendapatkan hak menjadi aktif. Pada umumnya hubungan antar neuron pada lapisan kompetitif tidak diperlihatkan pada diagram arsitektur. berikut salah satu contoh arsitektur jaringan dengan lapis kompetitif yang memiliki bobot $-\eta$. Arsitektur ini memiliki bentuk yang berbeda, dimana antar neuron dapat saling dihubungkan. Hal ini dapat dilihat pada Gambar 2.4



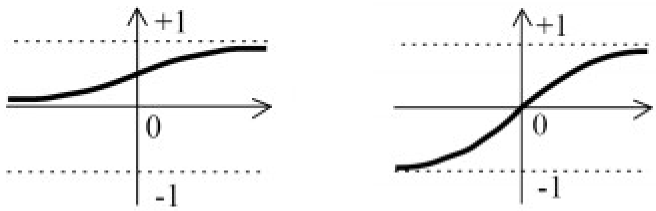
Gambar 2.4 Jaringan Saraf dengan Lapisan Kompetitif

2.2.3 Fungsi Aktivasi

Fungsi aktivasi adalah salah satu parameter yang terpenting dalam jaringan saraf tiruan. Pemilihan fungsi aktivasi dapat berpengaruh pada performa jaringan saraf tiruan. Ada beberapa fungsi aktivasi yang sering digunakan pada jaringan saraf tiruan (Siang, 2005: 26), yaitu:

1) Fungsi *Sigmoid*

   Terdapat 2 buah fungsi sigmoid yaitu *sigmoid biner* (*logsig*) dan *sigmoid bipolar* (*tansig*) yang akan diperjelas dengan gambar 2.10 terkait grafik fungsi *sigmoid biner* dan fungsi *sigmoid bipolar*.



Gambar 2.5 Grafik Fungsi Sigmoid Biner (a) dan Fungsi Sigmoid Bipolar (b)

(Hermawan, 2006:52)

Sigmoid biner memiliki nilai interval (0,1) dan memiliki bentuk fungsi:

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

dengan turunan

$$f'(x) = f(x)\left(1 - f(x)\right) \tag{2}$$

Sedangkan pada sigmoid bipolar bentuk fungsinya mirip dengan fungsi sigmoid biner, tetapi dengan nilai interval (-1,1)

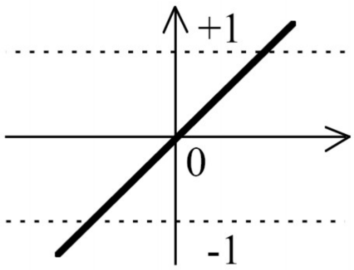
$$f(x) = \frac{2}{1+e^{-x}} - 1 \tag{3}$$

dengan turunan

$$f(x) = \frac{\left(1+f(x)\right)\left(1-f(x)\right)}{2} \tag{4}$$

2) Fungsi Identitas

Fungsi identitas memiliki nilai keluaran sembarang bilangan riil, (bukan hanya pada interval [0,1] atau [-1,1]). Di mana, $f(x)=x$, untuk semua x. Berikut ditampilkan pada gambar 2.6 grafik fungsi identitas.



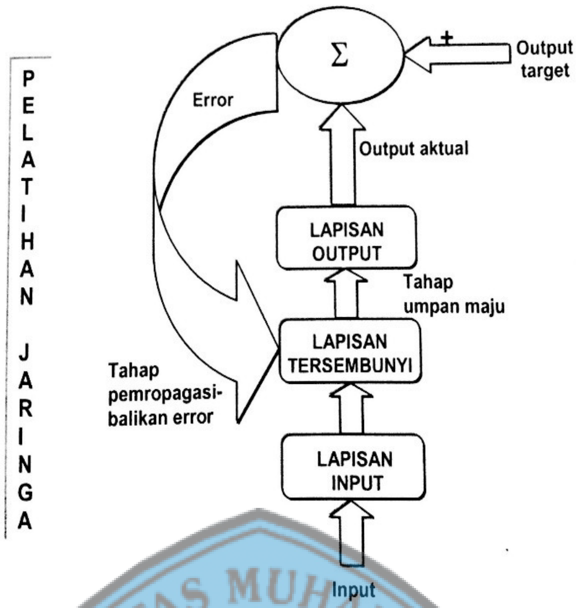
Gambar 2.6 Grafik Fungsi Identitas (Hermawan, 2006:54)

2.2.4 *Backpropagation*

Algoritma *backpropagation* merupakan bagian dari algoritma pembelajaran terawasi yang biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyi. Algoritma ini menggunakan *error* keluaran untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error* ini tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu. Saat perambatan maju, neuron-neuron diaktifkan dengan menggunakan fungsi aktivasi yang dapat didiferensiasikan seperti *sigmoid.*

Pelatihan pada *backpropagation* terdiri dari 3 fase (Siang, 2005: 100-101), yaitu (1) Propogasi maju, (2) Propogasi mundur, (3) Perubahan bobot. Untuk lebih jelasnya, algoritma pelatihan *backpropagation* dapat dilihat pada gambar 2.12 di bawah ini.

Gambar 2.7 Alur Kerja *Backpropagation* (Puspitaningrum, 2006: 127)



Penjabaran dari ketiga fase algoritma pelatihan *backpropagation* tersebut adalah sebagai berikut (Siang, 2005: 102-103):

a) Langkah 0: Inisialisasi semua bobot dengan bilangan acak kecil.

b) Langkah 1: Jika kondisi penghentian belum terpenuhi, lakukan langkah 2 – 9

c) Langkah 2: untuk setiap pasang data pelatihan, lakukan langkah 3 – 8

Fase 1: Propogasi maju

d) Langkah 3: Tiap unit masukan menerima sinyal dan meneruskannya ke unit tersembunyi di atasnya

e) Langkah 4: Hitung semua keluaran di unit tersembunyi $z_j \quad (j = 1, 2, ..., p)$

$$z_{in_j} = v_{j0} + \sum_{i=1}^{n} x_i v_{ji} \tag{5}$$

$$z_j = f\left(z_{in_j}\right) = \frac{1}{1 + e^{-z_{net\,j}}} \tag{6}$$

f) Langkah 5: Hitung semua keluaran jaringan di unit $y_k \quad (k = 1, 2, ..., m)$

$$y_{in_k} = w_{k0} + \sum_{j=1}^{p} z_j w_{kj} \tag{7}$$

$$y_k = f\left(y_{in_k}\right) = \frac{1}{1+e^{-y_{in_k}}} \tag{8}$$

Fase II: Propogasi mundur

g) Langkah 6: Hitung faktor δ unit keluaran berdasarkan kesalahan disetiap unit keluaran y_k $(k = 1, 2, ..., m)$

$$\delta_k = \left(t_k - y_k\right) f'\left(y_{in_k}\right) = \left(t_k - y_k\right) y_k \left(1 - y_k\right) \tag{9}$$

δ_k merupakan unit kesalahan yang akan dipakai dalam perubahan bobot layer di bawahnya.

Hitung suku perubahan bobot w_{kj} (yang akan dipakai nanti untuk mengubah bobot w_{kj}) dengan laju kecepatan α .

$$\Delta w_{kj} = \alpha\ \delta_k\ z_j\ ;\ k = 1, 2, ..., m\ ;\ j = 0, 1, ..., p \tag{10}$$

h) Langkah 7: Hitung faktor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi z_j $(j = 1, 2, ..., p)$

$$\delta_{in_j} = \sum_{k=1}^{m} \delta_k\ w_{kj} \tag{11}$$

Faktor δ unit tersembunyi:

$$\delta_k = \delta_{in_j} f'\left(z_{in_j}\right) = \delta_{in_j}\ z_j \left(1 - z_j\right) \tag{12}$$

Hitung suku perubahan bobot v_{kj} (yang akan dipakai nanti untuk mengubah bobot v_{kj}) dengan laju kecepatan α.

$$\Delta v_{ji} = \alpha\ \delta_j\ x_i\ ; j = 1,2,...,m\ ; i = 0,1,...p \quad (13)$$

Fase III: Perubahan bobot

i) Langkah 8: Hitung semua perubahan bobot

Perubahan bobot garis yang menuju ke unit keluaran:

$$w_{kj}(baru) = w_{kj}(lama) + \Delta w_{kj} \quad (14)$$

$k = (1,2,...,m\ ;\ j = 0,1,...,p)$

Perubahan bobot garis yang menuju ke unit tersembunyi:

$$v_{ji}(baru) = v_{ji}(lama) + \Delta v_{ji} \quad (15)$$

$j = (1,2,...,m\ ;\ \mathrm{i} = 0,1,...,p)$

Ketiga fase tersebut diulang-ulang hingga kondisi penghentian dipenuhi. Umumnya kondisi penghentian yang sering dipakai adalah jumlah iterasi atau kesalahan. Iterasi akan dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan, atau jika kesalahan yang terjadi sudah lebih kecil dari batas toleransi yang diizinkan.

Algoritma di atas merupakan algoritma standar *backpropagation* di mana menggunakan fungsi aktivasi *sigmoid biner* dengan arsitektur 1 lapis dengan layar tersembunyi.

2.2.5 Fungsi Pelatihan pada *Backpropagation*

Pelatihan standar *backpropagation* merupakan metode yang paling sederhana dalam proses pengaturan bobot. Dalam standar *backpropagation*, bobot dimodifikasi pada arah penurunan tercepat. Pengaturan bobot selalu dilakukan dalam arah negatif (gradien negatif). Meskipun penurunan fungsi berjalan cepat, tapi tidak menjamin akan konvergen dengan cepat (Siang, 2005). Oleh sebab itu digunakan beberapa perbaikan metode pada pengaturan bobot untuk mempercepat proses pelatihan. Salah satu metode tersebut adalah perbaikan dengan teknik optimasi numeris. Ada 3 buah algoritma yang terdapat pada metode ini yaitu algoritma *conjugate gradient*, *quasi newton* dan *levenberg marquardt*. Pada algoritma-algoritma yang menggunakan *conjugate gradient*, pengaturan bobot tidak selalu dalam arah menurun, tapi disesuaikan dengan arah konjugasinya. Pada metode *quasi newton* merupakan salah satu alternatif *conjugate gradient* yang bisa mendapatkan nilai optimum lebih cepat. Namun metode ini sangat kompleks, memerlukan waktu lama dan memori yang cukup besar karena pada setiap iterasinya harus menghitung turunan kedua.

Seperti halnya metode *quasi* newton, algoritma *levenberg-marquadrt* dirancang dengan menggunakan turunan kedua tanpa harus menghitung matriks *Hessian*. matriks *Hessian* dapat didekati sebagai:

$$H = J' \times J \qquad (16)$$

Dan gradien dihitung sebagai:

$$gW = J' \times e \qquad (17)$$

Dengan J adalah matriks *Jacobian* yang berisi turunan pertama dari *error* jaringan terhadap bobot, dan e adalah suatu vektor yang berisi *error* jaringan.

Matriks dapat dihitung dengan teknik *backpropagation* standar, yang tentu saja lebih sederhana dibanding menggunakan matriks *Hessian*. Algoritma *Levenberg-marquardt* menggunakan pendekatan untuk menghitung matriks *Hessian*, melalui perbaikan metode Newton:

$$W_{k+1} = W_k - \left[J' \times J + \mu \times I \right]^{-1} \times J' \times e \qquad (18)$$

Apabila μ bernilai 0, maka pendekatan ini akan sama seperti metode *quasi newton*. Namun apabila μ terlalu besar, maka pendekatan ini akan sama halnya *gradient descent* dengan *learning rate* yang sangat kecil (Kusumadewi, 2003).

2.2.6 Algoritma *Nguyen Widrow*

Algoritma *nguyen-widrow* merupakan algortima untuk inisialisasi bobot dan bias ke unit tersembunyi sehingga menghasilkan iterasi yang lebih cepat (Siang, 2005: 109). Algortima inisialisasi *nguyen-widrow* adalah sebagai berikut:

1) Inisialisasi semua bobot $\left(v_{ji}(lama) \right)$ dengan bilangan acak dalam interval $|-0.5,\ 0.5|$
2) Hitung $\|v_j\| = \sqrt{v_{ji}^2 + v_{j2}^2 + \ldots + v_{jn}^2}$
3) Bobot yang di pakai sebagai inisialisasi $v_{ji} = \dfrac{\beta v_{ji}(lama)}{\|v_j\|}$
4) Bias yang di pakai sebagai inisialisasi = v_0 ¿ bilangan acak antara $-\beta\ dan\ \beta$.

di mana : n = jumlah unit masukan

p = jumlah unit tersembunyi

β = faktor skala = $0.7\sqrt[n]{p}$

Analisis metode *Nguyen Widrow* didasarkan atas fungsi tangen *Sigmoid* yang memiliki interval nilai dari -1 sampai 1. Dalam fungsi tangen *sigmoid* ketika digunakan nilai $x = 1$ akan menghasilkan pendekatan nilai 0.75 dan ketika $x = -1$ akan mendekati nilai 0.25. Sedangkan Nilai β mempunyai interval 0 sampai 1 $(0 < \beta < 1)$. Nilai yang paling dekat dengan 0.75 yang berada dalam interval 0 sampai dengan 1 adalah 0.7 dan 0.8. Jika digunakan 0.8 maka nilainya akan melebihi batas interval fungsi *sigmoid* yaitu lebih dari 1. Hal inilah yang menyebabkan faktor skala β yang digunakan dalam metode *Nguyen Widrow* menggunakan nilai 0.7. Nilai 0.7 dalam faktor skala metode *Nguyen Widrow* diharapkan dapat menghasilkan bias dan bobot yang mampu menyesuaikan dengan pola pelatihan dalam *backpropagation*.

2.3 *Wavelet*

2.3.1 Pengertian *Wavelet*

Gelombang (*wave*) adalah sebuah fungsi yang bergerak naik turun secara periodik. *Wavelet* merupakan gelombang yang dibatasi atau dapat dikatakan sebagai gelombang pendek. *Wavelet* pertama kali digunakan dalam analisis dan pemrosesan digital dari sinyal gempa bumi, yang tercantum dalam literatur oleh A. Grossman dan J. Morlet. Penggunaan *wavelet* pada saat ini sudah semakin berkembang dengan penerapannya pada ilmu sains yang berhubungan dengan analisis *wavelet* dan teori transformasi *wavelet*. Dengan munculnya area sains ini *wavelet* mulai digunakan secara luas dalam filterasi dan pemrosesan data, pengenalan citra, sintesis dan pemrosesan berbagai variasi sinyal, kompresi dan pemrosesan citra, dll.

Analisis dekomposisi *wavelet* merupakan fungsi basis yang memberikan alat baru sebagai pendekatan yang dapat digunakan dalam merepresentasikan data atau fungsi-fungsi yang lain. Algoritma *wavelet* mampu memproses data pada skala atau resolusi yang berbeda. *Wavelet* merupakan fungsi basis yang dapat digunakan dalam merepresentasikan data atau suatu fungsi tertentu kedalam 2 posisi yang diskalakan dengan variabel tertentu. Fungsi *wavelet* mempunyai nilai yang berbeda dari nol dalam interval waktu yang relatif pendek. Bentuk basis dari *wavelet* dikatakan sebagai transformasi *wavelet* yaitu membagi data dari sebuah fungsi kedalam beberap komponen yang berbeda frekuensinya, serta menganalisa setiap komponen dengan skala yang berbeda.

2.3.2 Transformasi *Wavelet*

Transformasi *wavelet* merupakan sebuah fungsi konversi yang dapat digunakan untuk membagi suatu fungsi atau sinyal ke dalam komponen frekuensi yang berbeda, yang selanjutnya komponen-komponen tersebut dapat dipelajari sesuai dengan skalanya. Secara sederhana transformasi *wavelet* digunakan untuk mengubah suatu fungsi dengan domain waktu menjadi domain frekuensi. Perubahan tersebut diperlukan untuk mempermudah analisa yang akan dilakukan, dalam bidang pengolahan sinyal maka pengubahan tersebut dalam dilakukan terhadap sinyal maupun sistemnya. Tujuannya untuk mencari nilai frekuensi didapat dari waktu yang dibutuhkan untuk mencapai satu gelombang penuh maka akan didapatkan periode sinyal.

Fungsi Wavelet mempunyai nilai yang berbeda dari nol dalam interval waktu yang relatif pendek. Dalam hal ini wavelet berbeda dengan fungsi normal, ataupun fungsi gelombang seperti sinusoida, yang semuanya ditentukan dalam suatu domain waktu (-1,1). Wavelet dibedakan menjadi dua jenis, yaitu wavelet ayah (ϕ) dan wavelet ibu (ψ) yang mempunyai sifat:

$$\int_{-\infty}^{\infty} \phi(x)dx = 1 \quad \text{dan} \quad \int_{-\infty}^{\infty} \psi(x)dx = 0 \qquad (19)$$

Terdapat dua jenis dari transformasi *wavelet*: *Continue Wavelet Transform* (CWT) dan *Discrete Wavelet Transform* (DWT). CWT digunakan untuk sebuah fungsi yang berdomain bilangan real atas sumbu □, dan DWT digunakan untuk sebuah fungsi atas domain bilangan bulat (biasanya □=0,1,…,□−1, dimana □ dinotasikan sebagai banyaknya nilai dalam runtun waktu). Pada penelitian ini, jenis *wavelet* yang digunakan adalah DWT karena data runtun waktu dari nilai tukar dolar AS berdomain bilangan bulat.

2.3.3 *Discrete Wavelet Transform* (DWT)

Discrete Wavelet Transform pada level *J* dari *X* adalah transformasi ortonormal yang diberikan oleh $W = wX$, dengan *W* merupakan koefisien DWT adalah vektor berdimensi N, vektor *X* berdimensi N diartikan sebagai elemen analisis runtun waktu bernilai riil $\{X_t, t = 0,1,2,...,N-1\}$, dan w adalah matriks bernilai riil dengan ukuran N x N yang mendefinisikan DWT dan memenuhi $w^T w = I_N$. Dengan $N = 2^J$, dengan J merupakan integer positif.

Transformasi wavelet diskret dioperasikan pada fungsi dengan domain diskrit atau runtun waktu $x(t)$, biasanya menggunakan domain waktu $t = 0,1,...,N-1$. Wavelet menganalisa data runtunwaktu untuk dilatasi dan translasi data diskrit dengan menggunakan mother wavelet $\psi(t)$. Dasar proses dilatasi menggunakan interval $2_{j-1}, j = 1,2,3,...,J$. Translasi nilai t menggunakan interval 2_j.

Dari analisis multi resolusi, diketahui fungsi $f(t)$ dapat didekati dengan resolusi multi level. Diberikan $f_t(t)$ merupakan pendekatan pada level *j*, sehingga diperoleh:

$$f_j(t) = \sum_k f_{jk}\phi_{jk}(t) \tag{20}$$

Dimana

$$f_{jk} = \langle \phi_{jk}, f \rangle = \langle \phi_{jk}, f_j \rangle \tag{21}$$

Selanjutnya pendekatan f_j pada level $j-1$, dapat dituliskan sebagai berikut:

$$f_j(t) = f_{j-1}(t) + d_{j-1}(t) = \sum_k f_{jk}\phi_{jk}(t) + \sum_k d_{jk}\psi_{jk}(t) \tag{22}$$

Fungsi trasnsformasi wavelet $f(t)$ diperoleh dengan menghitung f_{jk} dan d_{jk}.

Dimana:

$$f_{j-1,k} = \sum_l h_{t-2k} f_{jl} \tag{23}$$

$$d_{j-1,k} = \sum_{l} g_{t-2k} f_{jl} \tag{24}$$

Hampiran deret wavelet orthogonal pada sinyal $f(t)$ diberikan oleh:

$$f_j(t) \approx \sum_k f_{j,k}\phi_{j,k}(t) + \sum_k d_{j,k}\psi_{j,k}(t) + \sum_k d_{j-1,k}\psi_{j-1,k}(t) + \ldots + \sum_k d_{1,k}\psi_{1,k}(t) \tag{25}$$

Dimana *j* adalah banyaknya komponen dalam multiresolusi, nilai k dari 1 sampai banyaknya koefisien dalam komponen spesifik. dengan:

$$\phi_{j,k}(t) = 2^{-\frac{j}{2}}\phi\left(\frac{t-2^j k}{2^j}\right) \tag{26}$$

$$\psi_{j,k}(t) = 2^{-\frac{j}{2}}\psi\left(\frac{t-2^j k}{2^j}\right) \tag{27}$$

DWT menghitung koefisien hampiran pada persamaan (12) untuk time series $f_0, f_1, \ldots, f_{N-1}$. DWT memetakan vektor $\mathrm{f} = (f_0, f_1, \ldots, f_{N-1})$ untuk vektor koefisien wavelet *W*.

Vektor *W* terdiri dari koefisien f_{jk} dan $d_{jk}, j = 1, 2, \ldots, J$ pada hampiran pada persamaan (12). $f_j(t)$ disebut koefisien pemulusan (*smooth coefficients*) dan $d_{j,k}$, disebut *detail* koefisien.

Koefisien Wavelet dan skala DWT didefinisikan sebagai

$$W_{1,t} = \sum_{l=0}^{L-1} h_k X_{(2t-k)\text{modulo N}} \tag{28}$$

$$V_{1,t} = \sum_{l=0}^{L-1} g_k X_{(2t-k)\text{modulo N}} \tag{29}$$

dengan $t = 0,1,\ldots,\frac{N}{2}-1$

$W_{1,t}$ dinyatakan sebagai koefisien wavelet DWT pada level 1 dan pada waktu ke t,

sedangkan $V_{1,t}$ merupakan koefisien skala DWT level 1 dan pada waktu ke t.

1) *Algoritma Level 1*

Transformasi Wavelet diskrit level 1 dari X

$$W = wX \tag{30}$$

$$\begin{bmatrix} W_1 \\ V_1 \end{bmatrix} = \begin{bmatrix} w_1 \\ v_1 \end{bmatrix} X$$

Dan karena w ortogonal yaitu $w^T w = 1$, maka

$$X = \begin{bmatrix} w_1 \\ v_1 \end{bmatrix}^T \begin{bmatrix} W_1 \\ V_1 \end{bmatrix}$$

$$X = \begin{bmatrix} w_1^T & v_1^T \end{bmatrix} \begin{bmatrix} W_1 \\ V_1 \end{bmatrix}$$

$$X = w_1^T W_1 + v_1^T V_1 \tag{31}$$

$$X = d_1 + f_1 \tag{32}$$

Dimana d_1 dan f_1 masing-masing merupakan detail dan smooth pada level 1.

2) *Algoritma Level j*

Algoritma piramida dimulai dengan $j=1$ dan selanjutnya diulang dengan $j=2,3,\ldots,J_0$ kemudain diperoleh semua vektor dari koefisien yang diperlukan untuk membentuk $\mathbf{W}$ yaitu $\mathbf{W_1},\mathbf{W_2},\ldots,\mathbf{W_{j0}}$ dan koefisien skala $\mathbf{V_{j0}}$.

Sama halnya dengan transformasi $\mathbf{V_2}$ ke $\mathbf{W_1}$ dan $\mathbf{V_1}$, transformasi $\mathbf{V_{j-1}}$ ke $\mathbf{V_{j-1}}$ ke $\mathbf{W_j}$ dan $\mathbf{V_j}$ dinyatakan sebagai:

$$\begin{bmatrix} W_j \\ V_j \end{bmatrix} = \begin{bmatrix} A_j \\ B_j \end{bmatrix} V_{j-1} = P_j V_{j-1} \tag{33}$$

Dimana A_j dan B_j matriks berukuran $N_j \times N_{j-1}$, karena matriks P_j ortonormal, maka $P_j P_j^T = P_j^T P_j = I$, kemudian karena $\mathbf{V_1} = A_1 X$ secara rekursif menghasilkan

$W_j = B_j A_{j-1} A_{j-2} \ldots A_1 X = w_j X$ dan $V_j = A_{j-1} A_{j-2} \ldots A_1 X = v_j X$

Karena P_j ortonormal, V_{j-1} dapat diperoleh dengan

$$V_{j-1} = P_j^T \begin{bmatrix} W_j \\ V_j \end{bmatrix} = \begin{bmatrix} A_j^T & B_j^T \end{bmatrix} \begin{bmatrix} W_j \\ V_j \end{bmatrix} = A_j^T W_j + B_j^T V_j \tag{34}$$

Dari persamaan diatas, V_{j-1} dapat dikonstruksi dari W_j dan V_j melalui invers algoritma piramida tingkat j, sehingga

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_j \\ v_j \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 B_2 \\ \vdots \\ A_j B_{j-1} \cdots B_1 \\ A_j A_{j-1} \cdots A_1 \end{bmatrix}$$

Dengan detail dan smooth diberikan oleh $d_j = B_1^T \ldots B_{j-1}^T A_j^T W_j$ dan $f_j = A_1^T \ldots A_{j-1}^T B_j^T V_j$.

Berdasarkan sifatnya, *Discrete Wavelet Transform* (DWT) dapat dibedakan menjadi dua, yaitu (Daubechies, 1992):

1) *Maximum Overlap Discrete Wavelet Transformation* (MODWT)
2) *Wavelet* Basis Ortonormal

2.3.4 *Maximum Overlap Discrete Wavelet Transformation* (MODWT)

Transformasi dengan menggunakan DWT tidak dapat dilakukan jika sampel yang diamati berukuran sembarang yang tidak dapat dinyatakan dalam bentuk 2^J dengan J bilangan bulat positif. Sebagai alternatif, penghitungan dapat dilakukan dengan *Maximal Overlap Discrete Transform* (MODWT). Keuntungan MODWT adalah dapat mengeliminasi reduksi data menjadi setengahnya (*down* sampling) sehingga dalam setiap level akan terdapat koefisien *wavelet* dan skala sebanyak panjang data (Percival & Walden, 2000). Pada MODWT koefisien *wavelet* pada setiap level selalu sama sehingga lebih sesuai untuk pemodelan pada *time series* dibandingkan dengan DWT. Prediksi data *time series* satu langkah ke depan dimodelkan secara linear berdasarkan koefisien *wavelet* hasil dekomposisi pada waktu-waktu sebelumnya.

2.4 Sistem *Fuzzy*

Fuzzy didefinisikan sebagai sesuatu yang kabur atau samar, tidak jelas, membingungkan tetapi istilah sistem *fuzzy* tidak dimaksudkan untuk mengacu pada sebuah sistem yang tidak jelas (kabur/samar-samar) definisi, cara kerjanya, atau deskripsinya. Sistem *fuzzy* didasarkan pada teori logika *fuzzy*. Logika *fuzzy* digunakan untuk mengekspresikan suatu besaran ke dalam suatu bahasa (*linguistic*), seperti jumlah produksi telur yang dapat dinyatakan dengan produksi tinggi, sedang dan rendah.

2.5 Himpunan *Fuzzy*

Himpunan *fuzzy* merupakan perluasan dari himpunan klasik dimana keberadaan suatu elemen tidak lagi bernilai benar atau salah, tetapi akan selalu bernilai benar jika mempunyai derajat keanggotaan yang berada dalam rentang [0,1]. Himpunan klasik (*crisp set*) adalah himpunan yang membedakan anggota dan bukan anggota dengan batasan yang jelas (Ross, 2010:26).

Pada himpunan klasik nilai keanggotaan suatu elemen x dalam himpunan A, memiliki 2 kemungkinan (Kusumadewi dan Hartati, 2010) yaitu:

1) Satu (1), yang berarti bahwa suatu elemen menjadi anggota dalam suatu himpunan.
2) Nol (0), yang berarti bahwa suatu elemen tidak menjadi suatu anggota dalam suatu himpunan.

Adanya 2 kemungkinan tersebut menjelaskan bahwa cara pengambilan keputusan pada himpunan klasik dinilai kurang bijaksana untuk menyatakan hal-hal yang bersifat kontinu. Nilai keanggotaannya menunjukkan bahwa semesta pembicaraan tidak berada pada 0 atau 1, namun juga nilai yang terletak diantaranya. Mengakibatkan nilai kebenaran suatu item tidak bernilai benar atau salah (Widodo dan Hardayanto, 2012:22). Jangkauan fungsi karakteristik tersebut memuat bilangan pada interval [0 1] sehingga didapat konsep himpunan *fuzzy* yang menjadi dasar dari logika *fuzzy*.

Antara keanggotaan *fuzzy* dengan probabilitas, keduanya memiliki nilai pada interval [0,1], namun interprestasi nilainya sangat berbeda antara kedua kasus tersebut. Keanggotaan *fuzzy* memberikan suatu ukuran terhadap pendapat atau keputusan, sedangkan probabilitas mengindikasikan proporsi terhadap keseringan suatu hasil bernilai benar dalam jangka panjang.

Misalkan S adalah himpunan semesta dan x adalah anggota S. Suatu himpunan *fuzzy* A dan S didefinisikan sebagai suatu fungsi keanggotaan $\mu A[x]$, yang memetakan setiap objek di S menjadi suatu nilai real dalam interval [0,1]. Didefinisikan himpunan semesta X adalah himpunan bilangan real yang senantiasa naik secara monoton dari kiri ke kanan. Nilai semesta pembicaraan atau domain dapat berupa bilangan positif maupun negatif. Adakalanya nilai semesta pembicaraan atau domain ini tidak dibatasi batas atasnya (Kusumadewi dan Hartati, 2010)

Definisi 2.1 (Zimmermann, 1991:11-12):

Suatu himpunan *fuzzy* $\tilde{A}$ dalam himpunan semesta X dinyatakan sebagai himpunan pasangan berurutan,

$$\tilde{A} = \left\{ \left(x, \mu_{\tilde{A}}(x) \right) \middle| x \in X \right\} \tag{35}$$

dengan $\mu\tilde{A}(x)$ adalah derajat keanggotaan x di $\tilde{A}$ yang terletak pada interval [0,1].

Menurut Wang (1997), terdapat 2 cara untuk mempresentasikan fungsi keanggotaan *fuzzy*, yaitu:

1) Himpunan *fuzzy* direpresentasikan sebagai pasangan berurutan dengan elemen pertama nama elemen dan elemen kedua nilai keanggotaan dengan notasi:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\} \tag{36}$$

2) Himpunan *fuzzy* dengan menggunakan notasi sebagai berikut:

   - Himpunan semesta diskret

$$\tilde{A} = \frac{\mu_{\tilde{A}}(x_1)}{x_1} + \frac{\mu_{\tilde{A}}(x_1)}{x_1} + \ldots + \frac{\mu_{\tilde{A}}(x_n)}{x_i} = \sum_{i=1}^{n} \frac{\mu_{\tilde{A}}(x_i)}{x_i} \tag{37}$$

   - Himpunan semesta kontinu

$$\int_x \frac{\mu_{\tilde{A}}(x)}{x} \tag{38}$$

2.5.1 Fungsi Keanggotaan *Fuzzy*

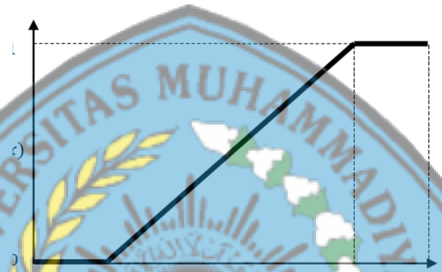
Fungsi keanggotaan (*membership function*) adalah suatu kurva yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaannya (derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Apabila *U* menyatakan himpunan universal dan $\tilde{A}$ adalah himpunan fungsi *fuzzy* dalam *U*, maka $\tilde{A}$ dapat dinyatakan sebagai pasangan terurut (Wang, 1997:22). Fungsi keanggotaan yang dapat dibangun dan digunakan untuk mempresentasikan himpunan *fuzzy* antara lain (Kusumadewi dan Hartati, 2010):

a) Representasi Linear

Pada representasi linear, pemetaan *input* ke derajat angggotanya digambarkan sebagai suatu garis lurus sehingga merupakan bentuk yang paling sederhana. Terdapat 2 keadaan pada himpunan *fuzzy* yang linear, yaitu:

1) Representasi Linear Naik

Representasi linear naik dimulai dari domain yang memiliki derajat keanggotaan nol [0] dan bergerak ke kanan menuju nilai domain yang memiliki derajat keanggotaan satu [1]. Seperti pada gambar di bawah ini:



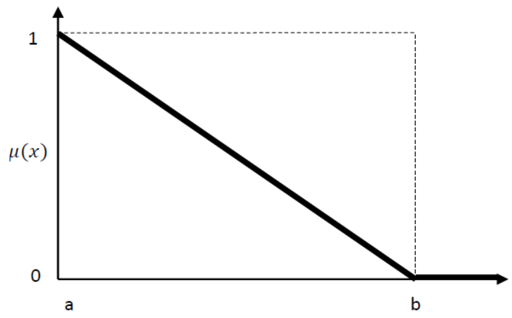
Gambar 2.8 Representasi Linear Naik (Kusumadewi dan Hartati, 2010)

Dengan fungsi keanggotaan Kurva Representasi Linear Naik:

$$\mu(x) = \begin{cases} 0; & x > b \\ (x-a)/(b-a); & a < x \leq b \\ 1; & x \leq a \end{cases} \tag{39}$$

2) Representasi Linear Turun

Representasi nilai turun merupakan kebalikan dari representasi linear naik yaitu berupa garis lurus yang dimulai dari nilai domain dengan derajat keanggotaan tertinggi pada sisi kiri kemudian bergerak menurun ke nilai domain dengan derajat keanggotaan yang lebih rendah. Seperti pada gambar di bawah ini:



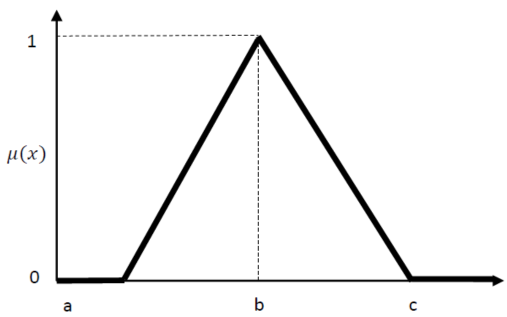
Gambar 2.9 Representasi Linear Turun (Kusumadewi dan Hartati, 2010)

Dengan fungsi keanggotaan Kurva Representasi Linear Turun:

$$\mu(x) = \begin{cases} 0; & x > b \\ (b-x)/(b-a); & a < x \leq b \\ 1; & x \leq a \end{cases} \tag{40}$$

3) Representasi Kurva Segitiga

Representasi kurva segitiga pada dasarnya terbentuk dari gabungan 2 garis linear, yaitu linear naik dan linear turun. Kurva segitiga hanya memiliki satu nilai x dengan derajat keanggotaan tertinggi, hal tersebut terjadi ketika $x = b$. Nilai yang tersebar dipersekitaran b memiliki perubahan derajat keanggotaan menurun dengan menjauhi 1. Seperti pada gambar di bawah ini:



Gambar 2.10 Representasi Segitiga (Kusumadewi dan Hartati, 2010)

Dengan fungsi keanggotaan Kurva Segitiga

$$\mu(x)=\begin{cases} 0; & x \le a \text{ dan } x > c \\ (b-x)/(b-a); & a < x \le b \\ (c-x)/(c-b); & b < x \le c \end{cases} \quad (41)$$

2.5.2 Logika *Fuzzy*

Logika *fuzzy* adalah suatu cara untuk memetakan suatu ruang *input* ke dalam suatu ruang *output* (Widodo dan Handayanto, 2012). Logika *fuzzy* adalah pengembangan dari logika Boolean yang berdasarkan pada konsep kebenaran sebagian. Dimana logika Boolean (logika klasik) menyatakan bahwa segala hal dapat diekspresikan dengan istilah binary (0 dan 1, ya atau tidak). Logika *fuzzy* menggantikan kebenaran Boolean dengan tingkat kebenaran. Logika *fuzzy* memberikan nilai kebenaran suatu item berada pada interval [0,1]. Alasan digunakannya logika *fuzzy* antara lain (Kusumadewi, 2003):

1. Konsep logika *fuzzy* mudah dimengerti dengan konsep matematis sebagai dasar dari penalaran *fuzzy* yang sangat sederhana dan mudah dimengerti.
2. Logika *fuzzy* sangat fleksibel, artinya mampu beradaptasi dengan perubahan-perubahan, dan ketidakpastian yang menyertai permasalahan.
3. Logika *fuzzy* memiliki toleransi terhadap data-data yang tidak tepat. Jika diberikan sekelompok data yang cukup homogeny, dan kemudian ada beberapa data yang “ekslusif”, maka logika *fuzzy* memiliki kemampuan untuk menangani data ekslusif.
4. Logika *fuzzy* mampu memodelkan fungsi-fungsi nonlinear yang sangat komples.
5. Logika *fuzzy* dapat mengaplikasikan pengalaman-pengalaman para ahli secara

langsung tanpa harus melalui proses pelatihan. Dalam hal ini, sering dikenal dengan nama *Fuzzy Expert System* menjadi bagian terpenting.

6. Logika *fuzzy* dapat bekerjasama dengan teknik-teknik kendali secara konvensional. Hal ini umumnya terjadi pada aplikasi di bidang teknik mesin maupun teknik elektro.
7. Logika *fuzzy* didasarkan pada bahasa alami. Logika *fuzzy* menggunakan bahasa sehari-hari sehingga mudah dimengerti.

2.5.3 Tahapan Sistem *Fuzzy*

Menurut Wang (1997), sistem *fuzzy* terdiri dari 3 tahapan, yaitu:

1. Fuzzifikasi

   Fuzzifikasi merupakan tahap pertama dari perhitungan *fuzzy*, yaitu mengubah masukan (*input*) yang berupa derajat keanggotaan. Sehingga, tahap ini mengambil nilai-nilai crisp dan menentukan derajat dimana nilai-nilai tersebut menjadi anggota dari setiap himpunan *fuzzy* yang sesuai.
2. Inferensi

   Inferensi adalah melakukan penalaran menggunakan *fuzzy input* dan aturan *fuzzy* yang telah ditentukan sehingga menghasilkan *fuzzy output*. Secara sintaks, suatu aturan *fuzzy* dituliskan sebagai berikut:

   IF anteseden THEN konsekuen
3. Defuzzifikasi

   Input dari proses defuzzifikasi adalah suatu himpunan *fuzzy* yang diperoleh dari komposisi aturan-aturan *fuzzy*, sedangkan *output* yang dihasilkan merupakan suatu bilangan pada domain himpunan *fuzzy* tersebut. Sehingga, jika diberikan suatu himpunan *fuzzy* dalam *range* tertentu, maka harus dapat diambil suatu nilai *crips* tertentu sebagai *output*.

2.5.4 Sistem Inferensi *Fuzzy*

Sistem inferensi fuzzy (*Fuzzy Inference System* atau FIS) merupakan suatu kerangka komputasi yang didasarkan pada teori himpunan *fuzzy*, aturan *fuzzy* yang berbentuk IF-THEN, dan penalaran *fuzzy.* Ada beberapa metode dalam sistem inferensi *fuzzy* yang biasa digunakan, yaitu:

1. Metode Mamdani

Metode Mamdani sering juga dikenal dengan nama Metode Max-Min. Metode ini diperkenalkan oleh Ebrahim Mamdani pada tahun 1975. Pada sistem ini untuk mendapatkan *output* diperlukan 4 tahap, antara lain:

a. Pembentukan Himpunan *Fuzzy* (Fuzzifikasi)

Pada tahap ini, variabel *input* dan variabel *output* dibagi menjadi satu atau lebih himpunan *fuzzy*.

Contoh: variabel *input* kurs dolar AS terhadap rupiah dibagi menjadi 3 himpunan *fuzzy* yaitu tinggi, sedang dan rendah.

b. Aplikasi Fungsi Implikasi

c. Komposisi Aturan

Ada 3 metode yang digunakan dalam melakukan inferensi system *fuzzy*, yaitu:

1) Metode Max (Maksimum)

Pada metode max solusi himpunan *fuzzy* diperoleh dengan cara mengambil nilai maksimum aturan yang kemudian digunakan untuk memodifikasi daerah *fuzzy* dan mengaplikasikannya ke *output* dengan menggunakan operator *OR* (*union*/gabungan). Jika semua proposisi telah dievaluasi maka *output* akan berisi suatu himpunan *fuzzy* yang menggambarkan kontribusi dari tiap-tiap proposisi. Secara umum dapat dituliskan:

$$\mu_{sf}[x_i] \leftarrow \max\left(\mu_{sf}[x_i], \mu_{kf}[x_i]\right)$$

Dengan

$\mu_{sf}[x_i]$: derajat keanggotaan solusi *fuzzy* aturan ke-*i*.

$\mu_{kf}[x_i]$: derajat keanggotaan konsekuen *fuzzy* sampai aturan ke-*i*.

2) Metode *Additive* (Sum)

Pada metode ini, solusi himpunan *fuzzy* diperoleh dengan cara melakukan *bounded-sum* terhadap semua *output* daerah *fuzzy*. Secara umum dituliskan:

$$\mu_{sf}[x_i] \leftarrow \min\left(1, \mu_{sf}[x_i] + \mu_{kf}[x_i]\right)$$

Dengan

$\mu_{sf}[x_i]$: derajat keanggotaan solusi *fuzzy* aturan ke-*i*.

$\mu_{kf}[x_i]$: derajat keanggotaan konsekuen *fuzzy* sampai aturan ke-*i*.

3) Metode Probabilistik OR (Probor)

Pada metode ini, solusi himpunan *fuzzy* diperoleh dengan cara melakukan *product* terhadap semua *Output* daerah *fuzzy*. Secara umum dituliskan sebagai berikut:

$$\mu_{sf}[x_i] \leftarrow \mu_{sf}[x_i] + \mu_{kf}[x_i] - \left(\mu_{sf}[x_i] * \mu_{kf}[x_i]\right)$$

Dengan

$\mu_{sf}[x_i]$: derajat keanggotaan solusi *fuzzy* aturan ke-*i*.

$\mu_{kf}[x_i]$: derajat keanggotaan konsekuen *fuzzy* sampai aturan ke-*i*.

d. Defuzzifikasi (Penegasan)

Terdapat beberapa metode defuzzifikasi pada komposisi aturan Mamdani:

1) Metode *Centroid*

Pada metode ini, solusi tegas diperoleh dengan cara mengambil titik pusat (z*) daerah *fuzzy*, secara umum dirumuskan:

$$Z^* = \frac{\int z\mu(z)dz}{\int \mu(z)}$$, untuk semesta kontinu

$$Z^* = \frac{\sum_{j=1}^{n} z_j \mu\left(z_j\right)}{\sum_{j=1}^{n} \mu\left(z_j\right)}$$, untuk semesta diskret

2) Metode Bisektor

Pada metode ini, solusi tegas diperoleh dengan cara mengambil nilai pada domain *fuzzy* yang memiliki nilai keanggotaan setengah dari jumlah total nilai keanggotaan pada daerah *fuzzy*. Secara umum dituliskan:

z_p sedemikian sehingga $\int_{R1}^{p} \mu(z)\, dz = \int_{p}^{Rn} \mu(z)\, dz$

3) Metode *Mean of Maximum* (MOM)

Pada metode ini, solusi tegas diperoleh dengan cara mengambil nilai rata-rata domain yang memiliki nilai keanggotaan maksimum.

4) Metode *Largest of Maximum* (LOM)

Pada metode ini, solusi tegas diperoleh dengan cara mengambil nilai rata-rata domain yang memiliki nilai keanggotaan maksimum.

2. Metode Takagi Sugeno Kang (TSK)

Metode Takagi Sugeno Kang hampir sama dengan metode Mamdani, hanya saja metode ini mempunyai konsekuen bukan merupakan himpunan *fuzzy* melainkan berupa konstanta atau persamaan linear dengan variabel-variabel yang disesuaikan dengan variabel *input*nya. Metode ini pertama kali diperkenalkan oleh Takagi Sugeno Kang pada tahun 1985. Terdapat 2 model untuk metode TSK yaitu:

a) Metode TKS orde-0

Secara umum bentuk model *fuzzy* TSK orde-0 adalah:

$if\,(x_1 \text{ is } A_1) \circ (x_2 \text{ is } A_2) \circ \ldots \circ (x_i \text{ is } A_i) \text{ then } z = k$

Dengan,

A_i = himpunan *fuzzy* ke-*i* pada variabel x_i

k = konstanta tegas sebagai konsekuen

$\circ$= operator *fuzzy*

b) Metode TKS orde-1

Secara umum bentuk model *fuzzy* TSK orde-0 adalah:

$if\,(x_1 \text{ is } A_1) \circ (x_2 \text{ is } A_2) \circ \ldots \circ (x_i \text{ is } A_i) \text{ then } z = p_1 x_1 + \ldots + p_i x_i + q$

A_i = himpunan *fuzzy* ke-*i* pada variabel x_i

p_i = konstanta tegas ke-i pada variabel x_i

q = konstanta tegas sebagai konsekuen

o= operator *fuzzy*

Apabila komposisi aturan menggunakan metode TKS, maka defuzzifikasi dilakukan dengan cara menilai rata-ratanya.

2.6 Metode *Fuzzy Wavelet*

Pada langkah pertama metode peramalan *fuzzy wavelet*, data runtun waktu dipanggil ke dalam sistem. Selanjutnya, data mentah tersebut ditransformasi dengan menggunakan transformasi *wavelet* MODWT pada level yang ditentukan. Setiap komponen hasil transformasi, baik bagian detail maupun smooth, kemudian diprediksi dengan membangun suatu sistem inferensi *fuzzy*. Hasil prediksi yang telah dilakukan selanjutnya digabung untuk memberikan hasil peramalan data runtun waktu secara keseluruhan. (Popoola, 2004)

2.7 Pemilihan Model Terbaik

Untuk mendapatkan jaringan yang optimum, hasil ouput pada tahap pelatihan dan pengujian, masing-masing dibandingkan dengan target atau data aslinya untuk mengetahui seberapa besar perbedaan atau kesalahan yang terjadi. Kesalahan tersebut dirumuskan sebagai berikut:

$$MAPE = \frac{1}{n}\left(\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100\%\right) \qquad (42)$$

Dengan n = banyak data

y_t = nilai aktual pada waktu t

$\hat{y}_t$ = nilai ramalan pada waktu t

n = jumlah data

(Makridarkis, 1998: 59)