



### **Hak cipta dan penggunaan kembali:**

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

### **Copyright and reuse:**

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

## BAB II

### LANDASAN TEORI

#### 2.1. Nilai Tukar

Nilai tukar didefinisikan sebagai harga mata uang dalam negeri dari mata uang asing (Salvatore, 2014). Kurs atau nilai tukar adalah pertukaran antara dua mata uang yang berbeda, yaitu merupakan perbandingan nilai atau harga antara kedua mata uang tersebut (Triyono, 2008). Nilai tukar mata uang asing adalah jumlah mata uang domestik yang dibutuhkan, yaitu banyaknya nilai rupiah yang dibutuhkan untuk memperoleh satu unit mata uang asing. Kurs mata uang asing juga menunjukkan harga atau nilai mata uang suatu negara yang dinyatakan dalam mata uang asing lain. Terdapat empat jenis nilai tukar atau kurs (Sukirno, 2013):

- a. *Selling rate* (kurs jual), yakni kurs yang ditentukan oleh suatu Bank untuk penjualan valuta asing tertentu pada saat tertentu.
- b. *Buying rate* (kurs beli), adalah kurs yang ditentukan oleh suatu bank untuk pembelian valuta asing tertentu pada saat tertentu.
- c. *Middle rate* (kurs tengah), adalah kurs tengah antara kurs jual dan kurs beli valuta asing terhadap mata uang nasional yang ditetapkan oleh Bank Sentral suatu saat tertentu.
- d. *Flat rate* (kurs flat), adalah kurs yang berlaku dalam transaksi jual beli bank *notes* dan *traveler cheque*, di mana dalam kurs tersebut sudah diperhitungkan promosi dan biaya lainnya.

Sistem nilai tukar dapat diklasifikasikan menurut tingkat dimana nilai tukar dikendalikan oleh pemerintah, yaitu sistem nilai tukar tetap, sistem nilai tukar mengambang bebas, sistem nilai tukar mengambang terkendali dan sistem nilai tukar terpatok (Madura, 2012). Berikut uraian dari empat jenis sistem nilai tukar :

a. Sistem Nilai Tukar Tetap

Sistem nilai tukar tetap atau *fixed exchange rate system* adalah sistem nilai tukar yang dibuat konstan oleh pemerintah atau diperbolehkan berfluktuasi hanya dalam batas-batas yang sangat sempit. Jika nilai tukar berubah terlalu besar, maka pemerintah akan mengintervensi untuk mempertahankannya dalam batas-batas yang dikehendaki.

b. Sistem Nilai Tukar Mengambang Bebas

Sistem nilai tukar mengambang bebas atau *freely floating exchange rate system* adalah sistem nilai tukar yang ditentukan sepenuhnya oleh kekuatan pasar tanpa intervensi dari pemerintah. Saat ini Bank Indonesia menggunakan sistem nilai tukar mengambang bebas.

c. Sistem Nilai Tukar Mengambang Terkendali

Sistem nilai tukar mengambang terkendali atau *managed floating exchange rate system* adalah sistem nilai tukar yang terletak diantara *fixed exchange rate system* dan *freely floating exchange rate system*. Sistem ini menyerupai sistem nilai tukar mengambang bebas karena nilai tukar dibiarkan berfluktuasi dan tidak dibatasi oleh batas-batas eksplisit. Sistem ini juga menyerupai sistem nilai tukar tetap karena pemerintah dapat melakukan

intervensi untuk menjaga agar nilai mata uang tidak berubah terlalu banyak dalam arah tertentu.

d. Sistem Nilai Tukar Terpatok

Sistem nilai tukar terpatok atau *pegged exchange rate system* adalah sistem nilai tukar dimana nilai suatu valuta dipatok ke valuta lain atau suatu unit pengukuran dan bergerak bersama-sama valuta tersebut terhadap valuta-valuta lain.

## 2.2. Peramalan

Peramalan (*forecasting*) adalah seni dan ilmu untuk memprediksi kejadian di masa depan dengan melibatkan pengambilan data historis dan memroyeksikannya ke masa mendatang dengan model pendekatan sistematis. Peramalan mungkin melibatkan pengambilan data historis dan memroyeksikannya ke masa depan dengan semacam model matematis. Metode *forecast* dilakukan dengan menggunakan model matematis yang beragam dengan data historis yang terkait dengan peramalan dan variabel sebab akibat untuk meramalkan permintaan. Analisis deret waktu (*time series*) digunakan untuk meramalkan permintaan yang pola permintaan di masa lalunya cukup konsisten dan akurat dalam periode waktu yang lama. (Heizer & Render, 2011).

## 2.3. Metode ARIMA

Pengertian metode ARIMA (*Autoregressive Integrated Moving Average*) adalah metode peramalan yang tidak menggunakan teori atau pengaruh antar variabel seperti pada model regresi. Sehingga metode ini tidak memerlukan penjelasan mengenai mana variabel bebas atau terikat. Metode ini juga tidak perlu

melihat pola data seperti pada *time series decomposition*, artinya data yang akan diprediksi tidak perlu dibagi menjadi komponen *trend*, musiman, siklis atau *irregular* (acak). Metode ini secara murni melakukan prediksi hanya berdasarkan data-data historis yang ada (Santoso, 2009).

### 2.3.1. Model dalam ARIMA

Terdapat empat macam model yang digunakan dalam ARIMA, berikut model tersebut:

a. Model *Autoregressive* (AR)

Model *Autoregressive* dengan orde AR ( $p$ ) atau model ARIMA ( $p,0,0$ ) dinyatakan sebagai berikut (Box, Jenkins, Reinsel, & Ljung, 2016):

$$\check{z}_t = \phi_1 \check{z}_{t-1} + \phi_2 \check{z}_{t-2} + \dots + \phi_p \check{z}_{t-p} + a_t$$

Rumus 2.1. *Autoregressive*

Keterangan:

$\phi_p$  = parameter *Autoregressive* ke- $p$

$a_t$  = nilai kesalahan pada saat  $t$

$\check{z}_{t-p}$  = variabel independen

Variabel independen merupakan deretan nilai dari variabel yang sejenis dalam beberapa periode  $t$  terakhir. Sedangkan  $a_t$  adalah kesalahan atau unit residual yang menggambarkan gangguan acak yang tidak dapat dijelaskan oleh model. Perhitungan *Autoregressive* dapat dilakukan dalam proses sebagai berikut:

1. Menentukan model yang sesuai dengan deret waktu
2. Menentukan nilai orde  $p$  (panjang persamaan yang terbentuk)
3. Mengestimasi nilai koefisien *autoregressive*  $\phi_1, \phi_2, \phi_3$ , dst.

b. Model *Moving Average* (MA)

Model *Moving Average* dinotasikan dalam orde MA ( $q$ ) atau ARIMA (0,0, $q$ ) yang ditulis dalam persamaan berikut (Box, Jenkins, Reinsel, & Ljung, 2016):

$$\check{z}_t = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - \dots - \theta_q a_{t-q}$$

**Rumus 2.2. *Moving Average***

Keterangan:

- $\theta_q$  = parameter *Moving Average*
- $a_t$  = nilai kesalahan atau unit residual
- $a_{t-q}$  = selisih nilai aktual dengan nilai prakiraan

Persamaan diatas menunjukkan bahwa nilai  $\check{z}_t$  tergantung nilai kesalahan sebelumnya dari nilai variabel itu sendiri. Model *Moving Average* mengukur autokorelasi antara nilai kesalahan atau residual.

c. Model *Autoregressive Moving Average* (ARMA)

Penggabungan model *autoregressive* (AR) dan *Moving Average* (MA) akan membentuk model baru, yaitu ARMA (*Autoregressive Moving Average*) dengan orde ARMA ( $p,q$ ). Bentuk umum persamaan ARMA merupakan gabungan dari persamaan AR dan MA yang dinotasikan sebagai berikut (Box, Jenkins, Reinsel, & Ljung, 2016):

$$\check{z}_t = \phi_1 \check{z}_{t-1} + \dots + \phi_p \check{z}_{t-p} + a_t - \theta_1 a_{t-1} - \dots - \theta_q a_{t-q}$$

**Rumus 2.3. *Autoregressive Moving Average***

d. Model *Autoregressive Integrated Moving Average* (ARIMA)

Jika data belum stasioner, maka model ARMA perlu ditambah proses *differencing* untuk data aslinya, dan untuk *error*-nya tidak perlu, menjadi model ARIMA ( $p,d,q$ ) dengan notasi berikut (Box, Jenkins, Reinsel, & Ljung, 2016):

$$\omega_t = \phi_1 \omega_{t-1} + \dots + \phi_p \omega_{t-p} + a_t - \theta_1 a_{t-1} - \dots - \theta_q a_{t-q}$$

#### Rumus 2.4. ARIMA

Keterangan:

$\omega_t$  = proses *differencing* (data ke- $t$  dikurang data ke-  $t-1$ )

### 2.3.2. Differencing

Pemodelan ARMA memiliki teori dasar korelasi dan stasioneritas. Maksudnya ARMA dapat digunakan ketika deret waktu telah membentuk grafik yang stasioner, atau tidak membentuk trend naik maupun turun. Namun bila data deret waktu tidak stasioner, maka perlu dilakukan *differencing* atau proses diferensiasi untuk mengubah data hingga menjadi stasioner dahulu sebelum dapat diproses melalui ARMA. Data yang telah di diferensiasi lalu dioleh dengan ARMA ini disebut dengan ARIMA dengan parameter ARIMA  $(p,d,q)$  dengan  $d$  menunjukkan jumlah proses diferensiasi yang dilakukan (Purnomo, 2015).

### 2.3.3. Fungsi Autokorelasi dan Autokorelasi Parsial

Identifikasi model untuk pemodelan data deret waktu memerlukan perhitungan dan penggambaran dari hasil fungsi autokorelasi atau *Autocorrelation Function* (ACF) dan fungsi autokorelasi parsial atau *Partial Autocorrelation Function* (PACF). Hasil perhitungan ini diperlukan untuk menentukan model ARIMA yang sesuai, apakah ARIMA  $(p,0,0)$  atau AR  $(p)$ , ARIMA  $(0,0,q)$  atau MA  $(q)$ , ARIMA  $(p,0,q)$  atau ARMA  $(p,q)$ , ARIMA  $(p,d,q)$ . Sedangkan untuk menentukan ada atau tidaknya nilai  $d$  dari suatu model, ditentukan oleh data itu sendiri. Jika bentuk datanya stasioner,  $d$  bernilai nol (0), sedangkan jika bentuk datanya tidak stasioner, nilai  $d$  bukan nol ( $d > 0$ ) (Purnomo, 2015).

#### **2.3.4. Uji Diagnostik**

Model yang telah melewati uji signifikansi parameter dalam tahapan estimasi, kemudian akan dilakukan uji diagnostik untuk meyakinkan apakah spesifikasi modelnya telah benar. Terdapat uji asumsi normalitas, uji autokorelasi, dan uji homoskedastisitas.

#### **2.4. Metode Artificial Neural Network**

Artificial Neural Network (ANN) atau Jaringan Saraf Tiruan (JST) adalah sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan saraf biologi. Jaringan Saraf Tiruan dibentuk sebagai generalisasi model matematika dari jaringan saraf biologi, dengan asumsi (Siang, 2005):

- a. Pemrosesan informasi terjadi pada banyak elemen sederhana (neuron).
- b. Sinyal dikirirkan diantara neuron-neuron melalui penghubung-penghubung.
- c. Penghubung antar neuron memiliki bobot yang akan memperkuat atau memperlemah sinyal.
- d. Untuk menentukan output, setiap neuron menggunakan fungsi aktivasi (biasanya bukan fungsi linier) yang dikenakan pada jumlahan input yang diterima. Besarnya output ini selanjutnya dibandingkan dengan suatu batas ambang.

##### **2.4.1. Lapisan Artificial Neural Network**

Secara umum jaringan saraf tiruan dapat dibagi menjadi tiga bagian, bernama lapisan (*layer*), yang dikenal sebagai (da Silva, 2017):

- a. Lapisan masukan (*input*)

Lapisan ini bertanggung jawab untuk menerima informasi (data), sinyal, fitur, atau pengukuran dari lingkungan eksternal. *Input* (sampel atau pola) biasanya dinormalisasi dalam nilai-nilai batas yang dihasilkan oleh fungsi aktivasi. Normalisasi ini menghasilkan lebih baik numerik presisi untuk operasi matematika yang dilakukan oleh jaringan.

b. Lapisan tersembunyi (*hidden*)

Lapisan ini terdiri dari neuron yang bertanggung jawab untuk mengekstrak pola terkait dengan proses atau sistem dianalisis. Lapisan ini melakukan sebagian besar proses internal dari jaringan.

c. Lapisan keluaran (*output*)

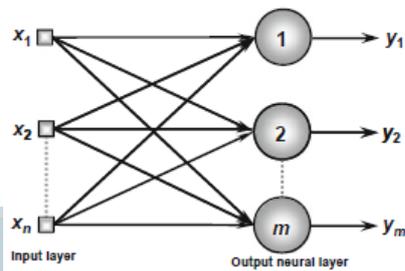
Lapisan ini juga terdiri dari neuron, dan dengan demikian bertanggung jawab untuk memproduksi dan menyajikan keluaran akhir jaringan, yang dihasilkan dari pengolahan yang dilakukan oleh neuron di lapisan sebelumnya. Neuron adalah unit pemroses informasi yang menjadi dasar dalam pengoperasian Jaringan Saraf Tiruan.

#### **2.4.2. Arsitektur Artificial Neural Network**

Arsitektur utama jaringan saraf buatan, mengingat letak neuron, serta bagaimana mereka saling berhubungan dan bagaimana lapisan tersusun, dapat dibagi menjadi *single-layer feedforward network*, *multilayer feedforward networks*, *reccurent networks*, dan *mesh networks*.

a. *Single-layer feedforward network*

Jaringan saraf tiruan ini memiliki hanya satu lapisan *input* dan lapisan saraf tunggal, yang juga merupakan lapisan *output*.

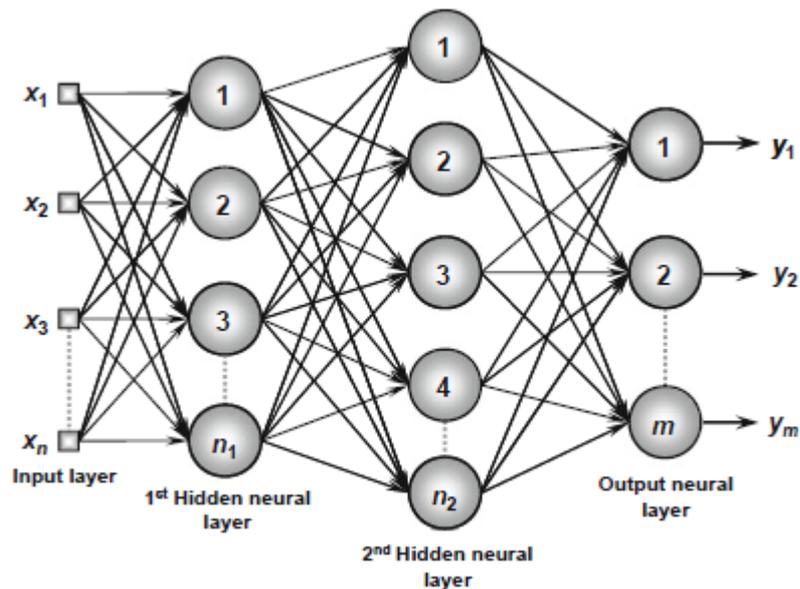


**Gambar 2.1. Single-Layer Feedforward Network**

Gambar 2.1 menggambarkan jaringan lapisan *feedforward* sederhana terdiri dari  $n$  *input* dan  $m$  *output*. Informasi yang selalu mengalir dalam satu arah yang merupakan dari lapisan *input* ke lapisan *output*. Dari Gambar 2.1, dapat dilihat bahwa dalam jaringan milik arsitektur ini, jumlah *output* jaringan akan selalu bertepatan dengan yang jumlah neuron. Jaringan ini biasanya digunakan dalam klasifikasi pola dan penyaringan masalah linier.

b. *Multiple-layer feedforward network*

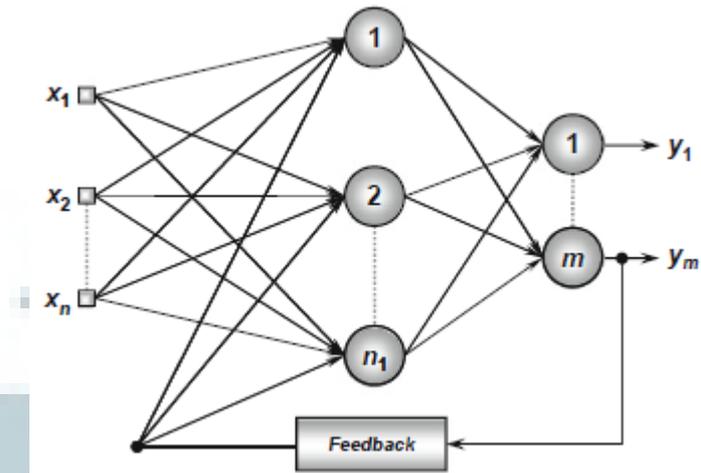
Jaringan *feedforward* dengan *multiple layers* terdiri dari satu atau lebih lapisan *hidden neural*. Gambar 2.2 Menunjukkan jaringan *feedforward* dengan *multiple layers* yang terdiri dari satu lapisan *input* dengan  $n$  sinyal sampel, dua lapisan *hidden neural* terdiri dari  $n_1$  dan  $n_2$  neuron masing-masing, dan, akhirnya, satu lapisan saraf *output* terdiri dari  $m$  neuron mewakili masing-masing nilai *output* dari masalah yang dianalisis.



Gambar 2.2. Multiple-Layer Feedforward Network

c. *Recurrent Architecture*

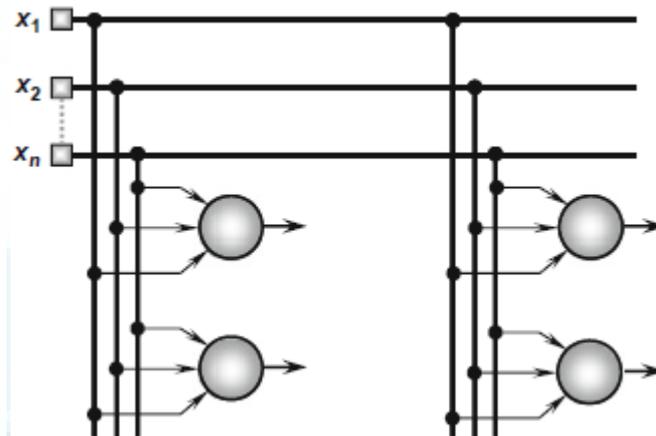
Pada jaringan ini, *output* dari neuron digunakan sebagai umpan balik untuk neuron lainnya. Gambar 2.3 menggambarkan jaringan Perceptron dengan umpan balik, dimana salah satu sinyal *output* dijadikan umpan balik ke lapisan tengah. Dengan demikian, jaringan arsitektur ini menghasilkan *output* juga mempertimbangkan nilai-nilai *output* sebelumnya. Fitur umpan balik yang memenuhi syarat jaringan ini untuk pengolahan informasi yang dinamis, mengartikan bahwa mereka dapat digunakan pada sistem waktu-varian, seperti prediksi *time series*, sistem identifikasi dan optimasi, kontrol proses, dan sebagainya.



Gambar 2.3. Recurrent Network

d. Mesh Network

Arsitektur Mesh menunjukkan neuron yang disusun dalam ruang dua dimensi. Gambar 2.4 menunjukkan mungkinnya untuk memverifikasi dalam kategori jaringan ini, beberapa sinyal *input* dibaca oleh seluruh neuron dalam jaringan.



Gambar 2.4. Mesh Network

### 2.4.3. Fungsi Aktivasi

Terdapat beberapa fungsi aktivasi yang sering digunakan dalam jaringan saraf tiruan, antara lain (Panji, 2008) :

#### a. Fungsi Sigmoid Bipolar

Fungsi sigmoid memiliki nilai maksimum 1. Untuk pola yang targetnya lebih dari 1, pola masukan dan keluaran harus terlebih dahulu ditransformasi sehingga semua polanya memiliki *range* yang sama seperti fungsi sigmoid yang dipakai. Fungsi sigmoid bipolar dirumuskan sebagai berikut:

$$f_2(x) = \frac{2}{1 + e^{-x}} - 1$$

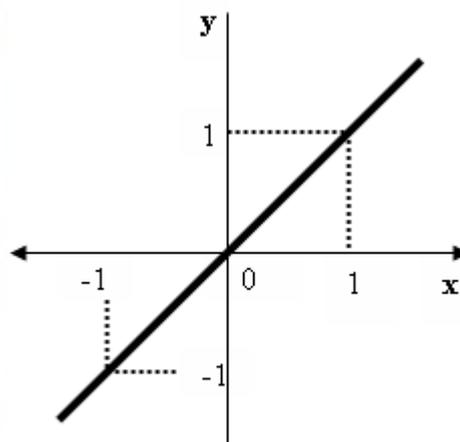
Rumus 2.5. Fungsi Sigmoid Bipolar

#### b. Fungsi Linear (Identitas)

Fungsi linear memiliki nilai output yang sama dengan nilai inputnya. Fungsi ini dirumuskan sebagai:

$$y = x$$

Rumus 2.6. Fungsi Linear



Linear (identitas)

Gambar 2.5. Fungsi Linear

c. Fungsi Sigmoid Biner

Fungsi ini digunakan untuk jaringan saraf 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 saraf yang membutuhkan nilai output yang terletak pada interval 0 sampai 1. Namun, fungsi ini bisa juga digunakan oleh jaringan saraf yang nilai outputnya 0 atau 1. Fungsi sigmoid biner dirumuskan sebagai:

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

Rumus 2.7. Fungsi Sigmoid Biner

#### 2.4.4. Proses Pelatihan dan Pembelajaran

Setelah jaringan belajar hubungan antara *input* dan *output*, jaringan dapat menghasilkan *output* yang dekat dengan *output* diharapkan (atau diinginkan) dari setiap nilai *input* yang diberikan. Oleh karena itu, proses pelatihan jaringan saraf terdiri dari menerapkan langkah-langkah ordinar yang diperlukan untuk menyesuaikan bobot *synaptic* dan ambang batas dari neuronnya, untuk menggeneralisasi solusi yang dihasilkan oleh *output* nya. Serangkaian langkah-langkah ordinar yang digunakan untuk pelatihan jaringan disebut pembelajaran algoritma (da Silva, 2017). Berikut dua jenis pembelajaran (*learning*):

a. Pembelajaran Terawasi

Pembelajaran terawasi atau *supervised learning* dalam kasus ini, penerapan pembelajaran terawasi hanya bergantung pada ketersediaan

tabel atribut atau nilainya, dan berperilaku seolah “pelatih” mengajari jaringan respon apa yang benar untuk setiap sampel yang disajikan untuk inputnya.

b. Pembelajaran Tidak Terawasi

Pembelajaran tidak terawasi atau *unsupervised learning* penerapan algoritma tidak memerlukan pengetahuan tentang masing-masing *output* yang diinginkan. Dengan demikian, jaringan perlu mengatur sendiri ketika ada sifat tertentu di antara elemen yang membentuk seluruh set sampel, mengenali *sub set* (atau *cluster*) yang menyajikan kesamaan.

#### 2.4.5. Algoritma Backpropagation

Pada algoritma *backpropagation*, vektor input dan vektor target digunakan untuk melatih ANN sampai ANN dapat mengaproksimasi sebuah deret data. *Error* atau selisih yang terjadi antara target dengan data yang sebenarnya, dipropagasikan atau ditransmisikan kembali ke dalam jaringan melalui *hidden layer* menuju ke neuron *input*. *Weights* atau bobot kemudian akan dihitung kembali dan masing-masing neuron akan mentransmisikan kembali sinyal ke *hidden layer* dan neuron *output* untuk kemudian akan dihitung kembali *error* antara target dengan data sebenarnya. Proses ini diulang terus menerus sampai *error* yang terjadi berada dalam *range* yang ditentukan di awal. Terdapat beberapa jenis *backpropagation training*, seperti *gradient descent backpropagation*, *gradient descent backpropagation with momentum*, *conjugate gradient backpropagation with powell-beale restarts*, *scaled conjugate gradient backpropagation*, *bfgs quasi-newton backpropagation*, *levenberg-marquardt*

*backpropagation, batch training with weight and bias learning rules, bayesian regulation backpropagation, gradient descent with adaptive learning rate backpropagation, gradient descent with momentum and adaptive learning rate backpropagation, one-step secant backpropagation, resilient backpropagation,* dan *sequential order incremental training with learning functions* (Demuth, 2009). Terdapat tiga fase pelatihan *backpropagation* (Siang, 2005) antara lain:

1. Fase pertama, yaitu propagasi maju. Dalam propagasi maju, setiap sinyal masukan dipropagasi (dihitung maju) ke layar tersembunyi hingga layar keluaran dengan menggunakan fungsi aktivasi yang ditentukan.
2. Fase kedua, yaitu propagasi mundur. Kesalahan (selisih antara keluaran jaringan dengan target yang diinginkan) yang terjadi dipropagasi mundur mulai dari garis yang berhubungan langsung dengan unit-unit di lapisan *output*.
3. Fase terakhir, yaitu perubahan bobot. Pada fase ini dilakukan modifikasi bobot untuk menurunkan kesalahan yang terjadi. Ketiga fase tersebut diulang-ulang terus hingga kondisi penghentian dipenuhi.

## **2.5. Kriteria Model Peramalan**

Kriteria model peramalan terdapat kriteria *Log Likelihood*, *AIC*, *Schwarz*, *Hannan-Quinn*, *R-Value*, *MSE*, dan *RMSE (Root of Mean Square Error)*. *RMSE* digunakan untuk mendapatkan informasi dari keseluruhan standar deviasi yang muncul saat menunjukkan perbedaan hubungan atau model yang dimiliki. Adapun rumus perhitungan *RMSE* adalah sebagai berikut (Wei, 2006). Berikut rumus *RMSE*:

$$RMSE = \sqrt{\left(\frac{1}{n} \times \sum_{t=1}^n e^2_t\right)}$$

Rumus 2.8. RMSE

## 2.6. Tools Peramalan

Beberapa *tools* yang dapat digunakan untuk peramalan *time series* diantaranya adalah EViews dan Matlab.

### 2.6.1. Tools EViews

EViews adalah paket ekonometrik, statistik, dan peramalan modern yang menawarkan *tools* analisis dengan antarmuka fleksibel dan mudah digunakan. EViews dapat digunakan untuk mengatur data, melakukan analisis ekonometrik dan statistik, menghasilkan peramalan atau simulasi model, serta menghasilkan grafik dan tabel untuk publikasi atau penyertaan dalam aplikasi lain. EViews menawarkan prosedur ARIMA otomatis. Antarmuka pengguna EViews yang menyederhanakan setiap langkah proses, mulai dari *input* data dan impor, hingga visualisasi data, analisis statistik, estimasi, peramalan dan pemecahan model, serta *output* presentasi publikasi (EViews).

### 2.6.2. Tools Matlab

Kegunaan umum dari Matlab diantaranya untuk Matematika dan Komputasi, Pengembangan Algoritma, Akuisisi Data, Pemodelan dan Simulasi, Pembuatan Prototipe, Analisis Data, Eksplorasi, Visualisasi, dan Pengembangan Aplikasi termasuk *Graphical User Interface* (GUI). Matlab memiliki fungsi *toolbox* yang memungkinkan pengguna untuk belajar dan menerapkan teknologi khusus seperti pemrosesan sinyal, sistem kontrol, jaringan saraf, logika *fuzzy*, simulasi dan teknologi lainnya (Away, 2006).