BAB II
STUDI PUSTAKA

Pada bab ini akan dijelaskan mengenai tinjauan pustaka dimana merupakan penjelasan mengenai penelitian terdahulu dan juga landasan teori yang merupakan teori-teori yang digunakan dalam penelitian.

2.1 Tinjauan Pustaka

Tabel 2.1Analisis Penelitian Sebelumnya

No	Peneliti	Metode	Kelebihan	Kekurangan
1	Edy Supriyanto (2004)	"Jaringan Saraf Tiruan *Backpropagation"*	Proses belajar dapat menyesuaikan bobot-bobot koneksi	Untuk menentukan parameter yang berpengaruh baik dilakukan pengujian berulang kali, karna hasil tidak pasti
2	Prisa Marga Kusumantara, I Gede Susrama (2007)	"Jaringan Syaraf Tiruan *Backpropagation* dan *Exponential Smoothing".*	*Exponential Smoothing* lebih besar menghasilkan nilai akurasi *MAPE,* karena metode tersebut tidak cocok untuk prediksi jangka pendek.	*Metode Backpropagation* menghasilkan nilai akurasi yang kecil, karena proses pelatihan yang dilakukan dengan menggunakan *hidden layer* yang terbaik.
3	Warih Maharani Jurnal SNATI (2009)	"*JST Backpropagation Momentum* Dengan *Adaptive learning Rate*".	Adannya *momentum* dan *adaptive learning rate*, memparcepat proses pembelajaran.	Jika kurang tepat memilih nilai *momentum* dan *learning rate*, proses pembelajaran akan lambat.

4	Dwi Efri Rufiyanti (20015)	"Penerapan Jaringan Saraf tiruan *backpropagation*, dan ARIMA "	Menggunakan *JST BP* lebih optimal dengan nilai akurasi terkecil dari metode ARIMA dan *metode Hibrid ARIMA-JST*	Proses pembelajaran atau pelatihan lebih lambat dari faktor banyak data.
5	Yure Firdaus Arifin,Dian Eka Ratnawati , Putra Pandu Adikara (2017)	"Metode *Bayesian* dan Metode *Backpropagation"*	Gabungan antara *Backpropagation* dan metode *Bayesian.* Karena lebih sedikit iterasi	Hasil prediksi paling optimal pada tahun pertama, untuk tahun berikutnya hasil semakin tidak akurat

2.2 Landasan Teori

Landasan teori adalah kumpulan teori yang diambil dari beberapa sumber referensi untuk merancang sistem peramalan harga saham menggunakan metode *Backpropagation.* Teori-teori tersebut akan dijelaskan di bawah ini.

1.2.1 Pengertian Saham

Saham adalah sertifikat atau tanda otentik yang mempunyai kekuatan hukum bagi pemegangnya sebagai keikutsertaan di dalam perusahaan serta mempunyai nilai nominal (mata uang) serta dapat diperjualbelikan.Saham merupakan surat berharga yang bersifat kepemilikan. Artinya si pemilik saham merupakan pemilik perusahaan. Semakin besar saham yang dimiliki, maka semakin besar pula kekuasaannya di perusahaan tersebut [1].

1.2.2 Peramalan (*Forecasting*)

Peramalan (*forecasting)* merupakan suatu usaha untuk meramalkan keadaan di masa mendatang melalui pengujian keadaan di masa lalu. Proses peramalan merupakan suatu unsur yang sangat penting dalam pengambilan keputusan, sebab efektif tidaknya suatu keputusan sering kali dipengaruhi beberapa faktor yang tidak tampak pada saat keputusan itu diambil. Peramalan

bertujuan untuk mendapatkan perkiraan atau prediksi yang bisa meminimumkan kesalahan dalam meramal yang biasanya diukur dengan *Mean Square Error*[2].

1.2.3 Teknik Peramalan

Situasi peramalan sangat beragam dalam horizon waktu peramalan, factor yang menentukan hasil sebenarnya, tipe pola data dan berbagai aspek lainnya. Untuk menghadapi penggunaan yang luas seperti itu, beberapa teknik telah dikembangkan. Teknik tersebut dibagi kedalam dua kategori utama, yaitu :

a. Metode kuantitatif
b. Metode kualitatif atau teknologi

Metode kuantitatif dapat dibagi ke dalam deret berkala (*time series*) dan *metode kausal*, sedangkan metode kualitatif atau teknologis dapat dibagi menjadi *metode eksploratif* dan *normatif*. Peramalan *kuantitatif* dapat diterapkan bila terdapat tiga kondisi berikut [2]:

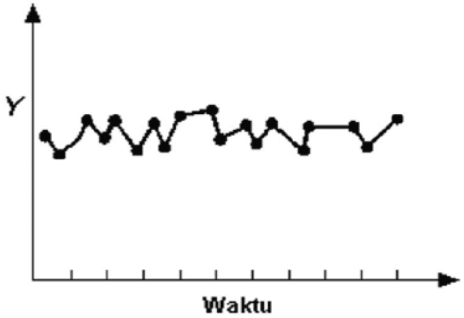
a. Tersedia informasi tentang masa lalu
b. Informasi tersebut dapat dikuantitatifkan dalam bentuk data numeric
c. Dapat diasumsikan bahwa aspek pola masa lalu akan terus berlanjut di masa mendatang.

Suatu dimensi tambahan untuk mengklasifikasi metode peramalan kuantitatif adalah dengan memperhatikan model yang mendasarinya. Terdapat dua jenis model peramalan yang utama, yaitu model deret berkala dan model regresi *(kausal*). Pada jenis pertama, pendugaan masa depan dilakukan berdasarkan nilai masa lalu dari suatu variable dan atau kesalahan masa lalu. Tujuan metode peramalan seperti itu adalah menemukan pola dalam deret data *historis* dan mengekstrapolasikan pola dalam menemukan pola dalam deret *historis* dan mengekstrapolasikan data tersebut ke masa depan.

Langkah penting dalam memilih sutu metode deret berkala yang tepat adalah dengan mempertimbangkan jenis pola data, sehingga metode yang paling tepat dengan pola tersebut dapat di uji. Menurut *Makridakis, Whellwright dan McGee* (1955:10) pola dapat dibedakan menjadi empat jenis *siklis* dan *trend* yaitu :

a. Pola horizontal (H)

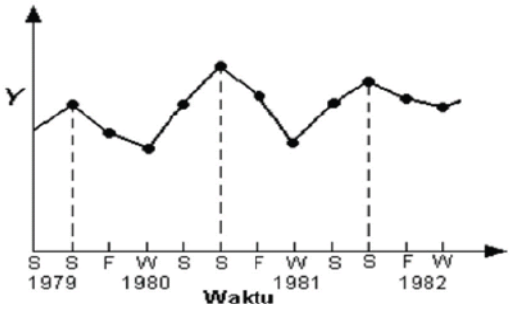
Pola horizontal terjadi bilamana nilai data berfluktuasi di sekitar nilai rata-rata konstan. Suatu produk yang penjualannya tidak meningkat atau menurun selama waktu tertentu termasuk jenis ini.



Gambar 2.1. Pola data *stationer / horizontal*

b. Pola musiman (S)

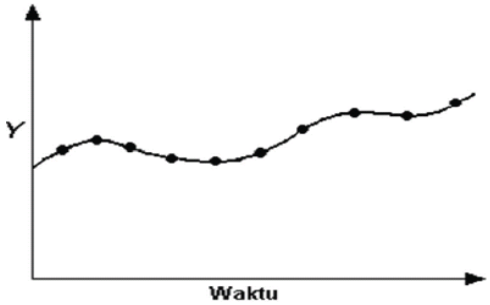
Pola musiman terjadi bilamana suatu deret terpenuhi oleh factor musiman (misalnya kuartal tahun tertentu, bulan , atau harian).



Gambar 2.2. Pola data musiman

c. Pola siklis (C)

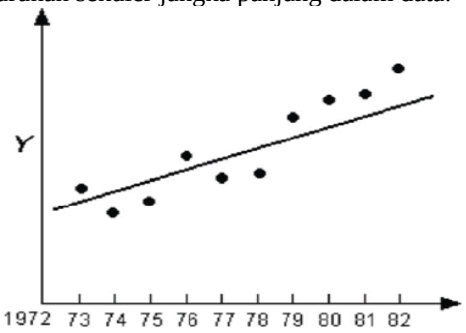
Pola siklis terjadi bilamana datanya dipengaruhi oleh fluktuasi ekonomi jangka panjang seperti yang berhubungan dengan siklus bisnis seperti mobil baja dan peralatan utama lain.



Gambar 2.3. Pola data siklis

d. Pola trend (T)

Pola trend terjadi bilamana terdapat kenaikan atau penurunan sekuler jangka panjang dalam data.



Gambar 2.4. Pola data *Trend*

1.2.4 Peramalan dan Horizon Waktu

Dalam hubungannya dengan horizon waktu peramalan, maka kita bisa mengklasifikasikan peramalan tersebut kedalam 3 kelompok, yaitu[3]:

1. Peramalan Jangka Panjang, umumnya 2 sampai 10 tahun. Peramalan ini digunakan untuk perencanaan produk dan perencanaan sumber daya.
2. Peramalan Jangka Menengah, umumnya 1 sampai 24 bulan. Peramalan ini lebih mengkhusus dibandingkan peramalan jangka panjang, biasanya digunakan untuk menentukan aliran kas, perencanaan produksi, dan penentuan anggaran.

3. Peramalan Jangka Pendek, umumnya 1 sampai 5 minggu. Peramalan ini digunakan untuk mengambil keputusan dalam hal perlu tidaknya lembur, penjadwalan kerja dan lain-lain keputusan control jangka pendek.

1.2.5 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut. Istilah buatan disini digunakan karena jaringan syaraf ini diimplementasikan dengan menggunakan program komputer yang mampu menyelesaikan sejumlah proses perhitungan selama proses pembelajaran[4].

Jaringan Syaraf Tiruan ditentukan oleh tiga hal :

1. Pola hubungan antar neuron
2. Metode untuk menentukan bobot penghubung (metode *Training/Learning*/Algoritma)
3. Fungsi aktifasi

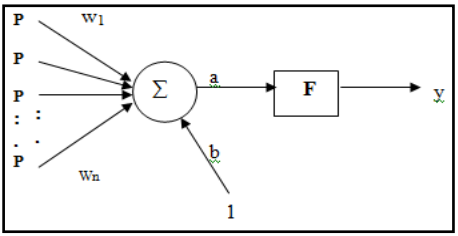
Beberapa istilah dalam jaringan syaraf tiruan yang sering ditemui antara lain:

1. Neuron atau node atau unit
   Sel syaraf yang merupakan elemen pengolahan jaringan syaraf 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.
2. 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.
3. Output atau keluaran
   Solusi atau hasil pemahaman jaringan terhadap data input. Tujuan pembangunan jaringan syaraf tiruan sendiri adalah untuk mengetahui nilai *output*.

4. Lapisan tersembunyi (*Hidden layer*)
   Lapisan yang tidak secara langsung berinteraksi dengan dunia luar. Lapisan ini memperluas kemampuan jaringan syaraf tiruan dalam menghadapi masalah-masalah yang kompleks.
5. Bobot
   Bobot dalam jaringan syaraf tiruan merupakan nilai matematis dari koneksi, yang mentransfer data dari satu lapisan ke lapisan lainnya. Bobot ini digunakan untuk mengatur jaringan sehingga jaringan syaraf tiruan bisa menghasilkan output yang diinginkan sekaligus bertujuan membuat jaringan tersebut belajar.
6. *Summation function*
   Fungsi yang digunakan untuk mencari rata-rata bobot dari semua elemen *input.*
7. Fungsi aktivasi atau fungsi transfer
   Fungsi yang meggambarkan hubungan antara tingkat aktivasi internal (*summation function*) yang mungkin berbentuk linier atau nonlinier.
8. Paradigma pembelajaran
   Cara pembelajaran atau pelatihan jaringan syaraf tiruan yaitu apakah terawasi, tidak terawasi, atau merupakan gabungan keduanya (*hybrid*).
9. Aturan pembelajaran .
   Aturan kerja secara umum dari teknik/algoritma jaringan syaraf tiruan. [4].

1.2.6 Cara Kerja Komponen Jaringan Saraf Tiruan

Ada beberapa tipe jaringan syaraf tiruan, tetapi hampir semuanya memiliki komponen yang sama. Sama halnya seperti otak manusia, jaringan syaraf juga terdiri dari beberapa *neuron,* dan ada hubungan antara neuron tersebut. *Neuron-neuron* tersebut akan mentransformasikan informasi yang diterima melalui sambungan keluarnya menuju ke *neuron-neuron* yang lain. Pada JST hubungan ini dikenal dengan nama bobot. Informasi tersebut disimpan pada suatu nilai tertentu pada bobot tersebut.



Gambar 2.5 Satu layer jaringan sebagai penyusun *multilayer.*

Keterangan:
p = Masukan (*input*)
w = Bobot pada lapisan keluaran
b = Bias
F = Fungsi aktivasi
y = Keluaran (*output*) hasil

1.2.7 Algoritma Backpropagation

Salah satu algoritma pelatihan jaringan syaraf tiruan yang dapat dimanfaatkan dalam manyelesaikan sistem pendukung keputusan adalah propagasi balik. Algoritma ini umumnya digunakan pada jaringan syaraf tiruan yang berjenis *multi-layer feed-forward*, yang tersusun dari beberapa lapisan dan sinyal dialirkan secara searah dari input menuju output[5].

Algoritma pelatihan *propagasi* balik pada dasarnya terdiri dari tiga tahapan yaitu:

a. *Input* nilai data pelatihan sehingga diperoleh nilai *output.*
b. *Propagasi* balik dari nilai *error* yang diperoleh.
c. Penyesuaian bobot koreksi untuk meminimalkan nilai *error*.

Ketiga tahapan tersebut diulangi terus-menerus sampai mendapatkan nilai error yang diinginkan. Setelah *training* selesai dilakukan, hanya tahap pertama yang diperlukan untuk memanfaatkan jaringan syaraf tiruan tersebut. informasi *error* dipropagasikan secara berurutan bermula dari *output* layer dan berakhir pada *input* layer, sehingga algoritma ini diberi nama *propagasi* balik (*backpropagation*).

Notasi yang digunakan dalam algoritma pelatihan jaringan syaraf tiruan adalah :

x = Data training *input* $x = (x_1,\ldots,x_i,\ldots,x_n)$
t = Data *training* untuk target *output* $t = (t_1,\ldots,t_k,\ldots,t_m)$

α =*Learning rate* yaitu parameter untuk mengontrol perubahan bobot selama pelatihan. Semakin besar *learning rate*,maka jaringan syaraf tiruan akan semakin cepat belajar tetapi hasilnya kurang akurat. Semakin kecil *learning rate*,maka jaringan syaraf tiruan akan semakin lambat belajar tetapi hasilnya lebih akurat[5].

X_i= Unit *input* ke-i
Z_j = Hidden unit ke-j
Y_k= Unit *output* ke-k
v_{0j}= Bias untuk *hidden* unit ke-j
v_{ij}= Bobot antara unit *input* ke-i dengan hidden unit ke-j
w_{0k}= Bias untuk unit *output* ke-k
W_{jk}= Bobot antara hidden unit ke-j dengan unit *output* ke-k
δ_k= Faktor koreksi *error* untuk bobot wjk
δ_j= Faktor koreksi *error* untuk bobot vij
m = Momentum

Langkah 0 : Inisialisasi bobot (ambil bobot awal dengan nilai random yang cukup kecil di sekitar -0.5 sampai 0.5)

Langkah 1 : Jika *stopping condition* masih belum terpenuhi, jalankan langkah 2-9

Langkah 2 : Untuk setiap data pelattihan, lakukan langkah 3-8

Feedforward:

Langkah 3 : Untuk tiap unit masukan (X_i, i=1,…,n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada diatasnya (lapisan tersembunyi).

Langkah 4 : Untuk tiap unit tersembunyi (Z_j, j=1,2,3,…,p) akan menjumlahkan sinyal-sinyal *input* yang sudah terbobot :

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij}$$

(2.1)

Kemudian dihitung nilai *output* dengan menggunakan fungsi aktivasi yang dipilih :

$$z_j = f(z_{ij})$$

(2.2

dimana fungsi aktifasi yang digunakan ialah fungsi *sigmoid biner* yang mempunyai persamaan : $\boldsymbol{f1(x)} = \frac{\mathbf{1}}{\mathbf{1+e(-x)}}$ (2.3)

Lalu mengirim sinyal *output* ini ke seluruh unit pada unit *output.*

Langkah 5 : Untuk tiap unit *output* (Y_k, k=1,2,3,…,m) akan menjumlahkan sinyal-sinyal *input* yang sudah terbobot :

$$\boldsymbol{y\_in_k} = \boldsymbol{w_{0k}} + \sum_{j=1}^{p} \boldsymbol{z_j w_{jk}} \quad (2.4)$$

Kemudian dihitung nilai *output* dengan menggunakan fungsi aktivasi :

$$\boldsymbol{y_k} = \boldsymbol{f(y\_in_k)} \quad (2.5)$$

Propagasi error (*backpropagasi of error*) :

Langkah 6: Untuk tiap unit *output* (Y_k, k=1,…,m) menerima pola target yang bersesuaian dengan pola masukan, dan kemudian dihitung informasi kesalahan :

$$\boldsymbol{\delta_k} = \boldsymbol{(t_k - y_k) f'(y\_in_k)} \quad (2.6)$$

Sebagaiman *input* data pelatihan ,*output* data pelatihan t_k juga telah diskalakan menurut fungsi aktivasi yang dipakai.

Faktor $\boldsymbol{\delta_k}$ digunakan untuk menghitung koreksi error (ΔW_{jk}) yang nantinya akan dipakai untuk memperbaharui $W_{jk,}$dimana :

$$\boldsymbol{\Delta W_{jk}} = \boldsymbol{\alpha \delta_k z_j} \quad (2.7)$$

Selain itu, juga dihitung koreksi biasa ΔW_{0k} yang nantinya akan dipakai untuk memperbaharui $\Delta W_{0k,}$ dimana :

$$\boldsymbol{\Delta W_{0k}} = \boldsymbol{\alpha \delta_k} \quad (2.8)$$

Faktor δ_k kemudian dikirimkan ke *layer* yang berada pada langkah 7.

Langkah 7 :Setiap *hidden unit* ($Z_{j,}$ j = 1,…,p) menjumlah *input* delta (yang dikirim dari *layer* langkah 6) yang sudah berbobot.

$$\boldsymbol{\delta\_in_j} = \sum_{k=1}^{m} \boldsymbol{\delta_k W_{jk}} \quad (2.9)$$

Kemudian hasilnya dikalikan dengan turunan dari fungsi aktivasi yang digunakan jaringan untuk menghasilkan factor koreksi *error* j, dimana:

$$\boldsymbol{\delta_j = \delta_{in_j} f'(z\_in_j)}$$

(2.10)

$$\boldsymbol{\Delta v_{ij} = \alpha \delta_j x_i} \quad (2.11)$$

Selain itu juga dihitung koreksi bias ΔV_{0j} yang nantinya akan dipakai untuk memperbaharui $V_{0j,}$ dimana :

$$\boldsymbol{\Delta v_{0j} = \alpha \delta_j} \quad (2.12)$$

Pembaharuan bobot (*adjustment*) dan bias

Langkah 8 : Setiap unit *output* ($Y_{k,}$ k = 1,…,m) akan memperbaharui bias dan bobotnya dari setiap *hidden unit* (j = 0,…,p)

$$\boldsymbol{w_{jk}(\text{baru}) = w_{jk}(\text{lama}) + \Delta w_{jk}} \quad (2.13)$$

Demikian pula, setiap *hidden unit* ($Z_{j,}$ j = 1,…,p) akan memperbaharui bias dan bobotnya dari setiap $unit_{input}$(I= 0,…,n)

$$\boldsymbol{v_{ij}(\text{baru}) = v_{ij}(\text{lama}) + \Delta v_{ij}} \quad (2.14)$$

Langkah 9 : Memeriksa *stop condition.*

Jika *stop condition* terpenuhi, pelatihan jaringan dapat dihentikan.

Untuk menentukan *stopping condition* terdapat tiga cara yang biasa dipakai yaitu sebagai berikut :

- Pertama, dengan membatasi iterasi yang ingin dilakukan.
- Cara kedua adalah membatasi *error.* Pada metode *Backpropagation*, dipakai metode *Mean Square Error* untuk menghitung rata-rata *error output* yang dikehendaki pada data pelatihan dengan *output* yang dihasilkan oleh jaringan.

$$MSE = 0.5\, x\, \{(t_{k1} - y_{k1})^2 + (t_{k2} - y_{k2})^2 + \ldots + (t_{km} - y_{km})^2\} \quad (2.15)$$

- Ketiga, ada kalanya sebelum mencapai kondisi seperti yang diinginkan, *error* justru semakin besar (*overtraining*). Jika salah satu dari *training set error* atau *test set error* bertambah besar, pelatihan harus dihentikan[6].

1.2.8 Variasi Backpropagation

Disamping model standart *Backpropagation*, kini sudah berkembang berbagai variasinya. Variasi tersebut bisa berupa model *Backpropagation* yang digunakan untuk keperluan khusus, atau teknik modifikasi bobot untuk mempercepat pelatihan dalam kasus tertentu. Beberapa variasi diantaranya[7]:

1.2.9 Momentum

Modifikasi dalam *Backpropagation* standart dapat dilakukan dengan melakukan perubahan bobot yang didasarkan atas arah gradient pola terakhir pola sebelumnya yang disebut momentum.

Penambahan momentum ditujukan untuk menghindari peruabhan bobot yang mencolok.Akibat adanya data yangb diberikan ke jaringan memiliki pola serupa, maka perubahan bobot dilakukan secara cepat.Namun apabila data terakhir yang dimasukkan memilki pola yang berbeda dengan pola sebelumnya, maka perubahan dilakukan secara lambat.

Dengan penambahan momentum, bobot baru pada waktu ke (t+1) didasarkan atas bobot pada waktu t dan (t-1).Disini harus dtambahkan 2 variabel baru yang mencatat besarnya momentum untuk 2 iterasi terakhir. Jika μ adalah konstanta ($0 \leq \mu \leq 1$) yang menyatakn parameter momentum maka bobot baru dihitung berdasarkan persamaan :

$$\boldsymbol{w_{jk}(t+1) = w_{kj}(t) + \alpha\delta_k z_j + \mu\left[w_{jk}(t) - w_{jk}(t-1)\right]}$$

atau$$\boldsymbol{\Delta w_{jk}(t+1) = \alpha\delta_k z_j + \mu\Delta w_{jk}(t)}$$

(2.16)

Persamaan pada lapisan *hidden*

$$\boldsymbol{v_{ij}(t+1) = v_{ij}(t) + \alpha\delta_j x_i + \mu\left[v_{ij}(t) - v_{ij}(t-1)\right]}$$

atau$$\boldsymbol{\Delta v_{ij}(t+1) = \alpha\delta_j x_i + \mu\Delta v_{ij}(t)}$$

(2.17)

Dimana μ adalah konstanta *momentum* yang biasanya berharga positif dengan range dari 0 sampai 1[7].

1.2.10 Delta-Bar-Delta

Modifikasi dengan Delta-Bar-Delta dilakuakn dengan memberikan laju pemahaman yang berbeda-beda untuk setiap bobotnya. Apabila perubahan bobot berada dalam arah yang sama dalam beberapa pola terakhir, maka laju pemahaman yang bersesuaian dengan bobot w_{jk} ditambah, dan sebaliknya.

Perubahna bobot dalam aturan delta-bar-delta adalah sebagai berikut :

$$\boldsymbol{w_{jk}(t+1) = w_{kj}(t) + \alpha_{kj}(t+1)\,\delta_k z_j} \qquad (2.18)$$

1.2.11 Model *Autoregresif Integrated Moving Average* (ARIMA)

ARIMA sering juga disebut metode runtun waktu Box-Jenkins. ARIMA sangat baik ketepatannya untuk peramalan jangka pendek, sedangkan untuk peramalan jangka panjang ketepatan peramalannya kurang baik. Biasanya akan cenderung *flat* (mendatar/konstan) untuk periode yang cukup panjang.

ARIMA cocok jika observasi dari deret waktu (*time series*) secara statistik berhubungan satu sama lain (*dependent*) [8]. Langkah-langkah analisis runtun waktu sebagai berikut :

1. Plot data Langkah pertama yang harus dilakukan adalah memplot data asli, dari plot tersebut bisa dilihat apakah data sudah stasioner. Jika data belum stasioner dalam mean maka perlu dilakukan proses *differencing.*
2. Identifikasi model Setelah data stasioner dalam *mean* dan variansi langkah selanjutnya adalah melihat plot ACF dan PACF. Dari plot ACF (*autocorrelation function*) dan PACF (*partial autocorrelation function*) tersebut bisa diindentifikasi beberapa kemungkinan model yang cocok untuk dijadikan model.
3. Estimasi model Setelah berhasil menetapkan beberapa kemungkinan model yang cocok dan mengestimasikan parameternya. Lalu dilakukan uji signifikansi pada koefisien. Bila koefisien dari model tidak signifikan maka model tersebut tidak layak digunakan untuk peramalan.
4. Uji asumsi residual (*diagnostic checking*) Dari beberapa model yang signifkan tersebut dilakukan uji asumsi residual.
5. Pemilihan model terbaik Hal-hal yang perlu diperhatikan dalam mengambil model adalah sebagai berikut :
    a. Prinsip parsimony yaitu model harus bisa sesederhana mungkin. Dalam arti mengandung sesedikit mungkin parameternya, sehingga model lebih stabil.
    b. Model sebisa mungkin memenuhi (paling tidak mendekati) asumsiasumsi yang melandasinya.

c. Dalam perbandingan model, selalu pilih model yang paling tinggi akurasinya, yaitu yang memberikan galat (*error*) terkecil.

6. Peralaman Langkah terakhir dari proses runtun waktu adalah prediksi atau peramalan dari model yang dianggap paling baik, dan bisa diramalkan nilai beberapa periode ke depan

1.2.12 Fungsi Autokorelasi

Salah satu bentuk analisis dalam teori statistika adalah analisis data deret waktu*(time series),* yaitu analisis terhadap data yang merupakan fungsi atas waktu atau tempat. Analisis data deret waktu merupakan analisis khusus dari analisis regresi, sebab dalam data deret waktu terlibat suatu besaran yang dinamakan *Autokorelasi*. Keberadaan autokorelasi bisa merupakan autokorelasi periodik, yaitu autokorelasi dengan nilai periodesitasnya lebih dari satu, dan autokorelasi seperti ini banyak terdapat pada data deret waktu yang yang memiliki komponen musiman-periodik. Perumusan *autokorelasi* sama dengan perumusan korelasi antar dua variabel. Dalam metode Statistika, jika dimiliki sampel atas data bivariat (X , Y). [6]

Dalam hal ini data harga saham berdasarkan waktunya akan diproses dengan rumus autokorelasi sehingga didapatkan *time lags* atau waktu yang bersignifikan dengan waktu yang diramalkan. Kemudian data harga saham pada waktu yang bersignifikan tersebut akan menjadi data masukan pada proses pelatihan *Jaringan Syaraf Tiruan Backpropagation.*

Rumus untuk mencari fungsi autokorelasi :

$$r_k = \frac{\left[\sum^{n}\left(Y_t - \overline{Y}_t\right)\left(Y_{t-k} - \overline{Y}_t\right)\right]}{\left[\sum_{t=1}^{n}\left(Y_t - \overline{Y}_t\right)^2\right]} \qquad \text{k = time lag} \qquad (2.19)$$

1.2.13 Normalisasi Data

Data-data yang ada dilakukan normalisasi dengan membagi nilai data tersebut dengan nilai *range* data (nilai data maksimum-nilai data minimum). Normalisasi data *input* bertujuan untuk menyesuaikan nilai *range* data dengan fungsi aktivasi dalam sistem *backpropagation*. Ini berarti nilai kudrat *input* harus berada pada *range* 0 sampai 1. Sehingga *range input* yang memenuhi syarat adalah nilai data *input* dari 0 sampai 1 atau dari –1 sampai 1. Oleh

karena itu output yang dihasilkan pun akan berada pada *range* 0 sampai 1. kemudian untuk mendapatkan nilai sebenarnya dari *output* perlu dilakukan proses denormalisasi.

Normalisasi data dengan menggunakan rumus[9] :

$$f(x) = \frac{X_i - X_{min}}{X_{max} - X_{min}} \quad (2.20)$$

Dimana:

X_i = data ke-i
X_{min} = data dengan nilai minimum
X_{max} = data dengan nilai maksimum

Pada proses *testing*, *output* yang dihasilkan oleh jaringan berkisar antara 0 sampai dengan 1 sehingga perlu dilakukan denormalisasi yang berguna untuk mengkonversikan kembali hasil *output* jaringan menjadi harga material normal. setelah itu akan dilakukan perbandingan antara data sebenarnya dengan data hasil prediksi, sehingga dapat dihitung *error* atau prosentase *errornya*.

Denormalisasi data dengan menggunakan rumus :

$$X_i = y(X_{max} - X_{min}) + X_{min}$$

(2.21)

Dimana:

X_i = harga material normal
Y = hasil output jaringan
X_{min} = data dengan nilai minimum
X_{max} = data dengan nilai maksimum

1.2.14 Ukuran Ketepatan Metode Peramalan

Ukuran akurasi hasil peramalan yang merupakan ukuran kesalahan peramalan merupakan ukuran tentang tingkat perbedaan antara hasil peramalan dengan permintaan yang sebenarnya terjadi. Ada empat ukuran yang biasa digunakan, yaitu: [5]

1. Mean Absolute Deviation

Mean absolute Deviation (MAD) mengukur akurasi persamaan dengan merata-ratakan nilai *absolut* kesalahan peramalan.

$$MAD = \sum \left| \frac{A_t - F_t}{n} \right| \quad (2.22)$$

Dimana:

A = Permintaan aktual pada periode –t

F_t = Peramalan permintaan *(forecast)* pada periode-t

N = Jumlah periode peramalan yang terlibat.

2. *Means Square Error* (MSE)

MSE adalah nilai tingkat kesalahan dari nilai yang diramalkan. Semakin kecil niali MSE maka semakin kecil pula nilai kesalahan peramalan yang dihasilkan. MSE dapat ditulis dengan rumus :

$$MSE = \sum \frac{(A_t - F_t)^2}{n} \quad (2.23)$$

3. Mean Absolute Persentage Error

Mean Absolute Persentage Error (MAPE) memberikan petunjuk seberapa kesalahan peramaln yang dibandingkan dengan nilai sebenarnya. Hasil perhitungan ditunjukkan dalam satuan persentase.

$$MAPE = \left(\frac{100}{n}\right)\sum \left| A_t - \frac{F_t}{A_t} \right| \quad (2.24)$$

1.2.15 K-Folds Cross Validation

K-*fold cross validation* merupakan teknik yang membagi data ke dalam k bagian untuk kemudian masing – masing bagian data tersebut akan dilakukan proses klasifikasi.

Gambar 2.6 Dataset menggunakan *K-Fold Cross Validation.*

Pada gambar 2.6 merupakan arsitektur dataset menggunakan *k-fold cross validation.* Dimana data akan dibagi dalam k-bagian. Dengan menggunakan k-*fold cross validation* akan dilakukan percobaan sebanyak k buah. Tiap percobaan itu akan menggunakan satu buah data *testing* dan k-1 bagian menjadi data *training*, dan kemudian data *testing* tersebut akan ditukar dengan satu buah data *traning* sehingga untuk tiap percobaan akan didapatkan data *testing* yang berbeda – beda[13].