BAB II

TINJAUAN PUSTAKA

2.1. Transportasi Kereta Api

Transportasi merupakan perpindahan manusia atau barang dari suatu tempat menuju ke tempat lainnya dengan menggunakan sebuah alat yang digerakan manusia, hewan atau mesin.(Zulfiar Sani, 2012) Transportasi di bagi menjadi 3 macam yaitu transportasi udara, transportasi darat dan transportasi laut.

Kereta api merupakan transportasi rel yang terdiri dari kombinasi satu atau lebih gerbong kereta yang dipasang untuk mengangkut manusia atau barang yang sering digunakan di daerah perkotaan maupun pedesaan, karena kereta api merupakan salah satu transportasi yang murah dan terjangkau untuk semua kalangan masyarakat. Berdasarkan jenis kelasnya, kereta apidibagi menjadi tiga yaitu kereta api eksekutif, kereta api bisnis dan kereta api ekonomi.

2.2.Deret Waktu *(Time Series)*

Time Series atau runtun waktu adalah sekumpulan data pengamatan yang disusun dalam urutan waktu (Hanke&Winchern, 2005:58 dalam kresna 2019). Metode *time series* adalah metode peramalan yang menggunakan

pola hubungan antara variabel yang akan diperkirakan dengan variabel waktu. Peramalan suatu data *time series* perlu memperhatikan tipe atau pola data. Secara umum, terdapat empat macam pola data *time series* yaitu:

1 Pola Horisontal yaitu pola data tidak berfluktuasi di sekitaran nilai rata–rata yang konstan.

2 Pola Musiman yaitu pola data yang dipengaruhi oleh faktor musiman.

3 Pola Siklus yaitu pola data yang dipengaruhi fluktuasi ekonomi jangka panjang seperti siklus bisnis.

4 Pola trend yaitu pola data yang dipengaruhi kenaikan ataupun penurunan jangka panjang pada data.

2.3.Statsioneritas

Suatu data dapat dikatakan stasioner jika pola data berada pada kesetimbangan di sekitar nilai rata–rata yang konstan dan variasi disekitar rata–rata tersebut konstan dalam jangka waktu tertentu. Jika tidak ada tren dalam data dan tidak ada musiman atau rata–rata dan variansnya tetap maka data runtun waktu tersebut stasioner. Stasioneritas data deret waktu dibedakan menjadi dua, yaitu sebagai berikut:

1 Stasioneritas dalam *mean* (rata-rata)

Stasioneritas dalam mean adalah fluktuasi data yang terletak di sekitaran nilai rata–rata yang konstan, tidak bergantung pada waktu serta variansi dari fluktuasi tersebut. Secara visual data stasioner atau belum stasioner dapat diketahui dari bentuk grafiknya.

2 Stasioner dalam Varians

Data runtun waktu dikatakan stasioner dalam variansi apabila struktur data memiliki fluktuasi data yang tetap serta tidak berubah–ubah dari waktu ke waktu. Secara visual data stasioner atau belum stasioner dapat menggunakan plot atau grafik.

Pengujian stasioneritas dari data runtun waktu dapat dibagi dengan berbagai cara, yaitu sebagai berikut:

a Pengujian stasioneritas dalam *mean* digunakan plot dari data runtun waktu yaitu plot fungsi autokorelasi (ACF) dan plot fungsi autokorelasi parsial (PACF). Data dikatakan mengandung non stasioner apabila data mengandung komponen trend dan plot ACF/PACF akan meluruh secara perlahan.

b Pengujian kestasioneran varians dapat melalui plot ACF dan PACF dari residual kuadrat

Adapun cara lain untuk melakukan pengujian terhadap stasioneritas, yaitu Uji Akar Unit atau *Unit Root Test* dengan menggunakan *Augmented Dickey-Fuller* (ADF). Apabila nilai ADF lebih kecil dari t-statistik maka dapat dikatakan data tidak stasioner dan begitu sebaliknya. (Kuncoro,2007:172 dalam Al Hikmah 2017).

Hipotesis :

$H_0: \emptyset = 1$ (Terdapat akar unit atau data tidak stasioner)

$H_1: -1 < \emptyset < 1$ (Tidak terdapat akar unit atau data stasioner)

Statistika uji :

$$t = \frac{\hat{\emptyset}}{se(\emptyset)} \quad (2.1)$$

Taraf signifikan : α =5%

Kriteria pengujian :

Jika nilai t-statistik>nilai kritis 5% maka H_0 ditolak

Berikut ini beberapa cara untuk mengatasi ketidakstasioneran data deret waktu yaitu sebagai berikut:

a. *Differencing*

Jika deret waktu tidak berfluktuasi pada sekitaran *mean* dapat dikatakan terjadi ketidakstasioneran dalam *mean*. Untuk mengatasi hal tersebut dapat dilakukan *differencing*. *Differencing* yaitu mencari selisih antara data deret waktu. *Differencing* pada selisih pertama dapat ditulis sebagai berikut :

$$W_t = Z_t - Z_{t-1} \quad (2.2)$$

Dimana :

W_t : barisan selisih (*differencing*) tingkat pertama

Z_t : data pada waktu ke-t

Z_{t-1} : data pada waktu ke t-1

Apabila pada selisih pertama data belum stasioner, maka di cari selisih tingkat dua dengan rumus sebagai berikut:

$$Y_t = W_t - W_{t-1}$$

$$Y_t = (Z_t - Z_{t-1}) - (Z_{t-1} - Z_{t-2})$$

$$Y_t = Z_t - 2Z_{t-1} + Z_{t-2}$$

(2.3)

Dimana :

Y_t : barisan selisih (differencing) tingkat kedua

Z_t : data pada waktu ke-t

Z_{t-1} : data pada waktu ke t-1

Z_{t-2} : data pada waktu ke t-2

Proses differencing berhenti apabila hasil *differencing* telah stasioner dengan *time series plot* bergerak mengikuti garis mean yaitu nilai 0.

b. Transformasi

Suatu deret waktu dikatakan tidak stasioner dalam *varians* jika deret tersebut tidak berfluktuasi dalam varians yang konstan. Untuk menstabilkan varians suatu deret waktu yang tidak stasioner, maka diperlukan transformasi terlebih dahulu. Transformasi yang sering digunakan adalah transformasi *Box-Cox* sebagai berikut :

$$T(Y_t) = \begin{cases} \dfrac{{Y_t}^{\lambda} - 1}{\lambda}, untuk\ \lambda \neq 0 \\ \lim\limits_{\lambda \to 0} \dfrac{{Y_t}^{\lambda} - 1}{\lambda} = \ln(Y_t), untuk\ \lambda = 0 \end{cases}$$

Beberapa nilai λ yang umumnya digunakan adalah sebagai berikut:

Tabel 2.1 bentuk transformasi Box-Cox berdasarkan nilai λ

Nilai Estimasi λ	Transformasi
-1	$\frac{1}{Y_t}$

-0,5	$\frac{1}{\sqrt{Y_t}}$
0	$\text{Ln}(Y_t)$
0,5	$\sqrt{Y_t}$
1	Y_t

2.4.Autokorelasi

Autokorelasi bertujuan untuk menentukan koefisien korelasi pada deret runtun waktu untuk mempelajari pola data trend atau musiman. Fungsi autokorelasi yaitu himpunan autokorelasi pada semua lag k yang diberi simbol ρ_k, k= 1,2,3,… dan $\rho_0 = 0$.

2.4.1. Fungsi Autokorelasi (ACF)

Autokorelasi adalah hubungan antar nilai *continue* yang berkaitan dengan variabel atau derer waktu yang sama. Fungsi autokorelasi mempunyai peranan penting untuk deteksi dini model dan kestasioneran data. Fungsi autokorelasi merupakan suatu fungsi yang menunjukkan derajat korelasi antara pengamatan waktu saat ini dengan pengamatan sebelumnya $(t-1, t-1, \ldots, t-k)$. ACF pada lag k dilambanglan dengan ρk yang didefinisikan sebagai berikut:

$$\rho_k = \frac{Y_k}{Y_0} = \frac{kovarians\ pada\ lag\ k}{varians}$$

(2.4)

Jika diagram cenderung lambat turun atau turun secara linier maka dapat disimpulkan bahwa data belum stasioner dalam *mean*.

2.4.2. Fungsi Parsial Autokorelasi (PACF)

Fungsi autokorelasi parsial adalah suatu fungsi yang menunjukkan besarnya korelasi parsial antara pengamatan pada waktu sekarang dengan pengamatan pada waktu sebelumnya. Pendugaan PACF merupakan koefisien autokorelasi dari persamaan Yule-Walker untuk $j = 1,2,3, \dots, k$. Pendugaan dari PACF adalah sebagai berikut :

$$P_k \phi_{kk} = \rho_k \quad (2.5)$$

Dalam penentuan pola PACF dibedakan menjadi dua yaitu menurun secara eksponensial menuju nol (*dies down*) dan signifikan pada semua lag (*cut off lag*). Untuk penentuan model dengan menggunakan pola ACF dan PACF dapat dilihat pada tabel model sebagai berikut(Wei,2006 dalam Novelia Purba 2016):

Tabel 2.2 Model pola ACF dan pola PACF

Model	Pola ACF	Pola PACF
Autoregressive (p)	*dies down*	*cut off setelah lag-p.*

Moving Average (q)	*cut off setelah lag p.*	*Dies Down.*
Autoregressive-Moving Average (p,q)	*dies down*	*dies down*

2.5. Model ARIMA Box-Jenkins

ARIMA biasanya juga disebut sebagai metode deret waktu *Box-Jenkins*. ARIMA atau *Autoregressive Integrated Moving Average* ditemukan oleh George Edward Pelham Box dan Gwilym Meirion Jenkins adalah metode prediksi untuk menyelesaikan deret berkala untuk menganalisis deret waktu. Metode ARIMA sangat cocok untuk peramalan data jangka pendek sedangkan untuk peramalan jangka panjang akurasi peramalannya kurang bagus. Klasifikasi model ARIMA terbagi menjadi tiga kelompok, diantaranya yaitu model *Autoregressive* (AR), model *Moving Average* (MA) dan model campuran yang memiliki karakteristik dari dua model pertama (ARMA). Model ARIMA merupakan model gabungan antara model *Autoregressive* (AR) dan model *Moving Average* (MA) dengan data yang telah mengalami *differencing* atau pembedaan sebanyak d kali.

2.5.1. Model *Autoregressive* (AR)

Model *Autoregressive* didasarkan pada ide bahwa saat ini pada deret Xt, dapat dinyatakan sebagai fungsi dari p nilai di masa lampau, $X_{t-1}, X_{t-2}, \dots, X_{t+p}$, dengan p adalah banyaknya langkah menuju ke masa lampau yang diperlukan untuk meramalkan nilai saat ini.

Suatu model *autoregressive* dengan orde p, yang dinotasikan AR(p) atau model ARIMA(p,0,0) mempunyai bentuk sebagai berikut:

$$X_t = \emptyset_1 X_{t-1} + \emptyset_2 X_{t-2} + \cdots + \emptyset_p X_{t-p} + \alpha$$

(2.6)

Dimana:

Xt : Variabel yang diramalkan

X_{t-1} : variabel yang menentukan

$\emptyset_p$: koefisien komponen AR derajat ke -p

α : Nilai Kesalahan

2.5.2. Model *Moving Average* (MA)

Sebagai suatu alternative dari representasi *autoregressive* dengan Xt diasumsikan sebagai kombinasi linier, model *moving aveag*e dengan orde q, ditulis dengan MA(q), mengasumsikan *white noise* merupakan suatu kombinasi linier untuk membentuk data yang diobservasikan.

Model *moving average* dengan orde q atau model MA(q) didefinisikan sebagai :

$$X_t = \alpha_t - \theta_1 \alpha_{t-1} - \theta_2 \alpha_{t-2} - \cdots - \theta_q \alpha_{t-q}$$

(2.7)

Dimana:

α_t : Nilai kesalahan saat t

α_{t-q} : Nilai kesalahan terdahulu

θ_q : Koefisien komponen MA derajat ke-q

2.5.3. Model Campuran Autoregressive Moving Average (ARMA)

Model ini merupakan gabungan antara model AR dan MA. Suatu runtun waktu {Xt} merupakan ARMA(p,q). model ARMA(p,q) dapat dituliskan sebagai berikut:

$$X_t = \emptyset_1 X_{t-1} + \emptyset_2 X_{t-2} + \cdots + \emptyset_p X_{t-p} + \alpha - \theta_1 \alpha_{t-1} - \theta_2 \alpha_{t-2} - \cdots - \theta_q \alpha_{t-q} \quad (2.8)$$

Beberapa kejadian khusus pada model ARMA, yaitu :

- Saat q= 0, model ini disebut model *autoregressive* dengan orde p, AR(p)
- Saat p = 0, model ini disebut model *moving average* dengan orde q, MA(q)

2.5.4. Model Autoregressive Integrated Moving Average (ARIMA)

Merupakan gabungan antara *Autoregressive* (AR) orde p dan *Moving Average* (MA) orde q serta proses *differencing* orde d . Model ARIMA meliputi model ARIMA non-musiman dan musiman.

Model ARIMA non-musiman adalah model *time series* yang tidak stasioner terhadap rata – rata dan memerlukan proses *differencing* sebanyak d agar stasioner. Model ARIMA non-musiman dapat dituliskan sebagai ARIMA (p, d, q) dengan persamaan sebagai berikut (Wei,2006 dalam Novelia Purba 2016):

$$\phi_p(B)(1-B)^d X_t = \theta_0 + \theta_q(B)\alpha_t$$

(2.9)

Model *Seasonal Autoregressive Integrated Moving Average* (SARIMA) adalah teknik *time series* berupa pengembangan dari ARIMA yang berisi efek musiman yang mempunyai sifat berulang setelah beberapa periode waktu tertentu, misalnya satu tahun, satu bulan, triwulan dan seterusnya. Oleh karena itu, deret berkala musiman mempunyai karakteristik yang ditunjukkan adanya korelasi yang kuat. Persamaan model SARIMA (*Seasonal Autoregressive Integrated Moving Average*) atau SARIMA musiman $(p, d, q)(P, D, Q)^s$ dengan rumus umum:

$$\emptyset_p(B^s)\,\emptyset_p(B)(1-B)^d(1-B^s)^d X_t = \theta_q(B)\theta_Q(B^s)a_t \qquad (2.10)$$

Dimana:

X_t : data deret waktu dengan rata – rata μ

$\emptyset_p(B)$: persamaan polynomial AR(p)

$\theta_q(B)$: persamaan polynomial MA(q)

$\emptyset_p(B^s)$: persamaan polynomial musiman AR(P)

$\theta_q(B^s)$: persamaan polynomial musiman MA(q)

$(1-B)^d$: operator pembeda non musiman

$(1-B^s)^d$: operator pembeda musiman dengan periode S

α_t : residual deret waktu (diasumsikan white noise)

s : jumlah periode musiman

D : ordo *differencing* musiman

Berdasarkan analisis deret waktu menggunakan model ARIMA *Box-*

Jenkins terdiri dari empat langkah dasar yaitu :

a Identifikasi model

Pembuatan model ARIMA berawal dengan membuat plot data *time series*, plot tersebut dapat dilihat pada hasil grafik ACF dan PACF pada data yang sudah stasioner.

b Estimasi Model

Pada tahap ini, setelah model awal ditentukan, akan diperoleh estimasi koefisien dari model yang diperoleh pada tahap identifikasi. Jika koefisien hasil estimasi signifikan maka model yang dipilih akan diterima. Sebaliknya jika koefisien yang diestimasi tidak signifikan maka model akan ditolak.

c Diagnostic Checking

Pada tahap ini dilakukan verifikasi kesesuaian model dan dilakukan uji *Q-Ljung-Box* untuk pemilihan model terbaik. Uji *Ljung Box* digunakan untuk menentukan apakah residual memenuhi asumsi *white noise* dan perlu dilakukan uji asumsi untuk mengetahui kenormalan residual model

d Peramalan

Model peramalan yang baik didapatkan dari hasil peramalan yang memiliki nilai ketepatan yang tinggi, sehingga model tersebut dapat digunakan untuk meramalkan data di masa yang akan datang.

2.6. Asumsi *White Noise*

Deret waktu dikatakan *white noise* jika ada sebuah barisan dari variabel bebas yang tidak berkorelasi dengan rata–rata, varians konstan dan kovarians. Pengujian *white noise* dapat dilihat melalui plot ACF dan PACF. Kriteria *error white noise* yaitu jika tidak ada lag yang melewati garis putus-putus merah atau selang kepercayaan. Uji *Q-Ljung-Box* digunakan untuk mengetahui apakah residual memenuhi asumsi *white noise.*

Hipotesis:

H_0 : parameter sama dengan nol atau tidak signifikan (residual tidak berkorelasi)

H_1 : parameter tidak sama dengan nol atau signifikan (residual berkorelasi)

Taraf signifikansi : α =5%

Kriteria pengujian:

Jika *p-value* < α =5%, maka H_0 ditolak sedangkan jika *p-value* > α =5% maka H_0 diterima

2.7.Uji Asumsi Distribusi Normal

Uji distribusi normal ini bertujuan untuk mengetahui apakah data telah memenuhi asumsi kenormalan atau tidak. Salah satu cara yang dapat digunakan untuk melakukan uji asumsi kenormalan adalah uji *Kolmogorov–Smirnov* dengan pengambilan keputusan jika nilai *p-value* < 0,05 maka data

tidak berasal dari populasi yang berdistribusi normal sedangkan jika nilai *p-value <-0,05* maka data berasal dari populasi yang berdistribusi normal.

2.8. Uji Linieritas

Uji linieritas dapat digunakan dengan menggunakan Uji Terasvirta. Uji Terasvirta merupakan pengujian linieritas dengan tipe LM (*Lagrange Multiplier*). Langkah-langkah pengujian linieritas pada Uji Terasvirta yaitu sebagai berikut:

1. Regresikan X_t pada 1 dan X_{t-k}, k=1,2,…,p sehingga diperoleh model linear

   $X_t = f_t + \widehat{e_t}$

   (2.11)

   Dimana $f_t = \emptyset_1 X_{t-1} + \emptyset_2 X_{t-2} + \cdots + \emptyset_p X_{t-p}$

   Kemudian menghitung nilai jumlah kuadrat nilai residual $SSR_0 = \sum \widehat{e_t}^2$
2. Meregresikan $\widehat{e_t}$ pada $1, X_{t-1}, X_{t-2}, \ldots, X_{t-p}$ dan m prediktor tambahan dari hasil pendekatan ekspansi Taylor. Selanjutnya menghitung jumlah kuadrat nilai residual $SSR_1 = \sum \widehat{v_t}^2$
3. Menghitung nilai statistika uji

   $$Fhit = \frac{\frac{SSR_0 - SSR_1}{m}}{\frac{SSR_1}{(n-p-1-m)}}$$

   (2.12)

   Hipotesis :

   H_0 : Model linier

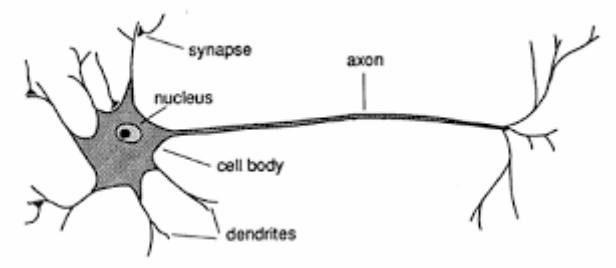
   H_1 : model non linier

Kriteria pengujian H_0 ditolak jik $F_{hitung} > F_{(\alpha,m,(n-p-1-m))}$ atau nilai *p-value* < α(0,05)

2.9.*Artificial Neural Network* (ANN)

Artificial Neural Network atau yang lebih dikenal dengan sebutan *Neural Network* (NN) yang dalam bahasa indonesia disebut Jaringan Saraf Tiruan (JST) ialah sistem pengelolaan informasi yang karakteristik menyerupai jaringan saraf biologis (Fauset, 1994:3 dalam Novelia Purba, 2016). Dalam pengoperasian NN terdapat dasar unit pengelolaan informasi yang disebut dengan *neuron*. Menurut Fausett (1994 dalam Novelia Purba, 2016), Komponen – komponen utama dari sebuah *neuron* dikelompokkan menjadi tiga bagian yaitu :

1 Dendrite, berfungsi untuk menerima informasi
2 Badan sel (Soma), berfungsi untuk mengelola informasi
3 Akson/ Axon (neurit), berfungsi mengirim implus – implus ke sel saraf lainnya.



Gambar 2.1. Jaringan Saraf Biologi

Prinsip kerja NN sama dengan kerja neuron pada otak manusia. NN bekerja dengan menggunakan tiga lapisan komponen, yaitu lapisan *input*,

lapisan tersembunyi, dan lapisan *output* . Lapisan *input* menerima *input* dari luar yang berupa deskripsi dari suatu permasalahan. Lapisan tersembunyi menghubungkan lapisan *input* dan lapisan *output*. Keluaran dari lapisan *output* yaitu hasil dari NN terhadap permasalahan yang diterima lapisan *input*. Proses pengiriman sinyal pada otak manusia melalui sinapsis sama dengan pembobotan pada setiap variabel *input*. Sedangkan lapisan tersembunyi pada NN memiliki fungsi yang sama dengan badan sel di neuron otak manusia, lapisan tersembunyi pada NN memproses sinyal sebagai *output* dimana *output* tersebut berperan sebagai akson di *neuron* otak manusia. Sama halnya dengan badan sel yang memproses informasi yang diterima *dendrit*, lapisan tersembunyi pada NN memproses informasi yang diterima dari lapisan *input* dengan menjumlahkan hasil kali bobot dengan *input* yang kemudian menggunakan fungsi aktivasi untuk memproses. Beberapa jenis algoritma pembelajaran dalam *Artificial Neural Network* yang dikembangkan untuk peramalan data deret waktu antara lain yaitu *Backpropagation, recureent network, self organizing map, radial basis function neural network* dan sebagainya. Terdapat beberapa komponen penting dalam pembentukan model ANN yaitu *neuron, layer,* fungsi aktivasi dan bobot.

Pada metode *Neural Network* mempunyai dua tahap pemrosesan informasi yaitu tahap pelatihan dan pengujian :

1 Pelatihan Jaringan (*Training*)

Tahap pelatihan dimulai dengan pemasukan pola-pola pelatihan ke dalam jaringan. Pada setiap iterasi dilakukan evaluasi terhadap *output* jaringan. Tahap ini berlangsung sampai menemukan bobot yang sesuai dengan nilai *error* yang diinginkan atau jumlah iterasi menvapai nilai maksimum yang tela ditetapkan. Selanjutnya, bobot ini menjadi dasar pengetahuan pada tahap pengujian (Warsito,2009). Pada dasarnya proses pelatihan dibagi menjadi dua macam yaitu:

a Metode pembelajaran terawasi (*supervised learning*)

Pada proses pembelajaan ini, satu *input* yang telah diberikan pada satu *neuron* di lapisan *input* akan dijalankan di sepanjang jaringan saraf sampai ke *neuron* pada lapisan *output*. Hasil *output* yang diperoleh akan dicocokkan dengan pola target, apabila terjadi perbedaan, maka akan muncul error. Jika nilai *error* cukup besar, akan dilakukan pembelajaran yang lebih banyak lagi.tujuan dari pembelajaran terawasi yaitu menghasilkan pemetaan vektor *input* ke nilai *output* sehingga diperoleh *output* yang sesuai dengan target. Pengubahan nilai bobot dilakukan untuk mencapai tujuan ini.

b Metode pembelajaran tanpa terawasi (*unsupervised learning*).

Pada proses ini, nilai bobot disusun dalam suatu interval atau range tertentu tergantung dari nilai *input* yang diberikan. Pembelajaran ini bertujuan mengelompokkan unit-unit yang hampir serupa.

2 Pengujian Jaringan

Pada proses ini dilakukan pengujian terhadap suatu pola masukan yang belum pernah dilatih (data *Testing*) dengan bobot-bobot yang telah diperoleh dari tahap pelatihan. Dengan proses ini diharpkan menghasilkan *error* minimum pada bobot-bobot hasil pelatihan.

2.10. *Radial Basis Function Neural Network*

Radial Basis Function neural network (RBFNN) merupakan salah satu metode dari *Neural Network* dengan metode pelatihan hibrida yaitu menggabungkan metode pelatihan terbimbing dan metode pelatihan tak terbimbing. RBFNN memiliki 3 lapisan layer yang meliputi *input layer*, *hidden layer* dan *output layer*.

Lapisan *input* menerima suatu vektor x yang kemudian dibawa ke lapisan tersembunyi yang akan mengolah data *input* secara nonlinear dengan fungsi aktivasi. *Output* dari lapisan tersembunyi selanjutnya diproses pada lapisan *output* secara linier (Wei *et al*,2011 dalam Fajarani Juliaristi 2014). Model RBFNN menggunakan fungsi basis sebagai fungsi aktivasi untuk setiap *neuron* pada lapisan tersembunyi. Beberapa fungsi radial basis yang banyak digunakan adalah sebagai berikut (Brodjol,Sutijo 2008 dalam Fajarani Juliaristi 2014):

1 Fungsi Multikuadratik

$$\psi(x, c_j) = \left((x - c_j)^2 + \sigma^2\right)^{\frac{1}{2}}$$

(2.13)

2 Fungsi Invers Multikuadratik

$$\psi(x, c_j) = \frac{1}{\left((x-c_j)^2+\sigma^2\right)^{\frac{1}{2}}}$$

(2.14)

3 Fungsi Gaussian

$$\psi(x, c_j) = e^{\left(\frac{(x-c_j)^2}{2\sigma^2}\right)}$$

(2.15)

Dimana σ^2 adalah varian dari c, dengan $c = (c_1, c_2, \dots, c_m)$ dan $\sigma^2 \neq 0$

2.10.1. Arsitektur *Radial Basis Function Neural Network*

Arsitektur jaringan *Radial Basis Function Neural Network* pada gambar2.2 terdiri dari p komponen vektor *input* x, m buah fungsi basis dan 1 *output*. *Output* yang dihasilkan oleh RBFNN merupakan kombinasi linier dari bobot $\{w_j\}_{j=1}^{m}$ dengan fungsi aktivasi $\phi_j(x)$ sehingga rumus model RBFNN adalah sebagai berikut:

$$f(x) = \sum_{j=1}^{m} w_j \phi_j(x)$$

(2.16)

Dengan :

m : banyak fungsi basis *neuron* tersembunyi

w_j : bobot *output* ke-j

$\phi_j(x)$: fungsi aktivasi neuron tersembunyi ke-j

x : vector *input* $[x_1, x_1, \ldots, x_p]$

Fungsi aktivasi dengan RBF (Gaussian) diperoleh persamaan sebagai berikut :

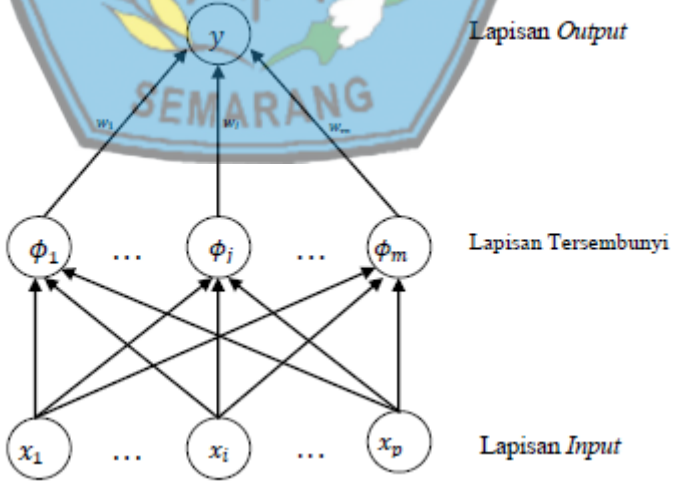
$$\phi_j(x) = exp\left(-\sum_{i=1}^{k}\left(\frac{|x_i - c_j|^2}{2\sigma_j^2}\right)\right) \quad (2.17)$$

Keterangan :

σ_j : standar deviasi pada node ke j j=1,2,3,…,m

c_j : vektor center pada node ke j j=1,2,3,…,m

x : vektor *input* ke-I i=1,2,3,…,p



Gambar 2.2. Arsitektur jaringan RBFNN

Berdasarkan arsitektur diatas, berikut merupakan algoritma pelatihan *Radial Basis function Neural Network* (Wulandari, Ayu , 2017):

1. Menentukan banyak input jaringan
2. Menentukan banyak lapisan tersembunyi
3. Menentukan hasil aktivasi jaringan RBFNN dengan menggunakan fungsi Gaussian pada persamaan (2.17)
4. Menghitung bobot pelatihan mengunakan metode *least square*
5. Menghitung nilai keseluruhan output jaringan RBFNN berdasarkan persamaan (2.16)

2.10.2. Algoritma Pelatihan *Radial Basis Function Neural Network*

Radial Basis Function Neural Network memiliki algoritma pelatihan yang terdiri dari cara terbimbing dan tak terbimbing sekaligus. Pelatihan *Radial Basis Function Neural Network* terdiri dari dua tahap yaitu :

1 Tahap *Clustering* Data

Pada tahap pertama, data dikelompokkan berdasarkan kedekatan tertentu, misalnya kedekatan jarak antar dua titik. Penentuan *cluster* akan menghasilkan *center* atau pusat dari kelompok data. Jumlah *cluster* yang dihasilkan akan menentukan *hidden layer* yang dipakai.Ada dua cara yang dapat digunakan untuk menentukan *center*. Cara pertama yaitu menentukan *center* secara acak dari kelompok data. Sedangkan cara kedua yaitu dengan menggunakan algoritma *clustering* yang dianggap lebih baik dari cara pertama. Algoritma *clustering* yang sering dugunakan yaitu algoritma *K-means*. Dengan menggunakan algoritma tersebut, NN dapat mencari sendiri

center yang terbaik dari data. Dengan melihat tahap pertama dari pelatihan tersebut, disimpulkan bahwa tahap ini bersifat *unsupervised*.

2 Tahap Pembaharuan Bobot

Tahap pelatihan berikutnya berfungsi untuk mendapatkan bobot dari *neuron*. Pada tahap ini diperlukan perhitungan untuk memperbaharui bobot dan dibutuhkan data *training* beserta targetnya. Dapat disimpulkan pada tahap kedua ini bersifat *supervised*.

2.10.3. Algoritma *K-Means Clustering*

Salah satu ciri model RBFNN ialah terletak pada fungsi aktivasi yang perhitungannya membutuhkan nilai pusat dan varians *neuron* tersembunyi. Metode *K-Means* ini mengelompokkan data *input* menjadi beberapa kelompok atau *cluster* untuk menghitung nilai pusat dan varians. Pusat cluster adalah rata–rata (*mean*) *cluster* tersebut. Algoritma *K-Means Clustering* adalah sebagai berikut :

1 Tentukan nilai k kluster dan nilai k nilai pusat

2 Tempatkan setiap data/objek ke *cluster* terdekat. Kedekatan dua objek berdasarkan jarak kedua objek tersebut. Jarak dihitung dengan menggunakan jarak *Euclide.* Persamaan jarak *Euclide* antara dua titik sebarang P dan Q dengan P $(x_1, x_2, \dots, x_p)$ dan Q $(y_1, y_2, \dots, y_p)$ adalah sebagai berikut:

$$d(P,q) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2}$$

(2.18)

Hitung ulang nilai pusat *cluster* yang menerima data baru dan *cluster* yang kehilangan data.

3 Ulangi langkah ke-2 sampai nilai pusat lama sama dengan nilai pusat baru (stabil)

2.11. Metode Kuadrat Terkecil (*Least Square)*

Desain dari RBFNN akan membentuk pemetaan secara nonlinier dari variabel *input* ke lapisan *hidden* dan pemetaan linier dari lapisan *hidden* ke lapisan *output*. Oleh karena itu, model RBFNN memerlukan optimasi pada lapisan *output* yang dilakukan dengan menggunakan metode kuadrat kecil (*least square*). Metode kuadrat terkecil pada penelitian ini akan digunakan untuk menentukan nilai bobot dengan nilai *error* yang minimum. Pada metode ini dikenal dengan istilah *training set* yang memuat elemen dari pasangan nilai variabel *input* dan variabel *output*. Model linier yang digunakan yaitu $y = \sum_{j=1}^{m} w_j \phi_j (x)$ dan *training set* $\{(x_i, \hat{y}_i)\}_{i=1}^{n}$ maka prinsip dari kuadrat terkecil yaitu meminimalkan jumlah kuadrat kesalahan (*Sum Square Error*) dengan rumus sebagai berikut :

$$SSE = \sum_{i=1}^{n} (\hat{y}_i - y)^2$$

(2.19)

Keterangan :

$\hat{y}_i$: nilai prediksi variabel *output* ke-i

y : nilai variabel output ke-i

n : banyak pengamatan

Kemudian ditentukan nilai optimal dari bobot ke-j. langkah pertama yaitu diturunkannya fungsi SSE dengan persamaan sebagai berikut:

$$\frac{\partial s}{\partial w_j} = 2\sum_{i=1}^{n}(\hat{y}_i - y)\frac{\partial y}{\partial w_j}$$

(2.20)

Dengan persamaan (2.16) maka diperoleh persamaan :

$$\frac{\partial y}{\partial w_j} = \phi_j(x)$$

(2.21)

Dari persamaan (2.19) disubstitusikan ke persamaan (2.18), sehingga diperoleh persamaan sebagai berikut :

$$2\sum_{i=1}^{n}(\hat{y}_i - y)\,\phi_j(x) = 0$$

(2.22)

$$\sum_{i=1}^{n}(\hat{y}_i)\,\phi_j(x) = \sum_{i=1}^{n}(y)\,\phi_j(x)$$

(2.23)

Karena j=1,2,3,…m maka diperoleh m persamaan seperti persamaan (2.23) untuk menentukan m bobot. Untuk memperoleh penyelesaian tunggal persamaan (2.23) ditulis dengan notasi vector, sehingga didapatkan persamaan :

$$\phi_j{}^T y = \phi_j{}^T \hat{y}$$

(2.24)

$$\phi_j = \begin{bmatrix} \phi_j(x_1) \\ \phi_j(x_2) \\ \vdots \\ \phi_j(x_n) \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \hat{y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix}$$

Karena terdapat m persamaan untuk setiap nilai j, maka persamaan (2.25) ditulis sebagai berikut :

$$\begin{bmatrix} {\phi_1}^T y \\ {\phi_2}^T y \\ \vdots \\ {\phi_m}^T y \end{bmatrix} = \begin{bmatrix} {\phi_1}^T \hat{y} \\ {\phi_2}^T \hat{y} \\ \vdots \\ {\phi_m}^T \hat{y}\,) \end{bmatrix}$$

Dengan menerapkan hukum perkalian vektor maka persamaan diatas ditulis menjadi :

$$\Phi^T y = \Phi^T \hat{y}$$

(2.25)

$$\Phi = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \cdots & \phi_m(x_1) \\ \phi_1(x_2) & \phi_1(x_2) & \cdots & \phi_m(x_2) \\ \vdots & \vdots & \cdots & \vdots \\ \phi_1(x_n) & \phi_1(x_n) & \cdots & \phi_m(x_n) \end{bmatrix}$$

Matriks Φ disebut dengan matriks desai. Menurut Howelett & Jain (2001:4) komponen ke-i dari y ketika bobot pada nilai optimum didapatkan persamaan sebagai berikut:

$$y = \sum_{j=1}^{m} w_j \phi_j(x) = \bar{\phi}_i^T \hat{w}$$

(2.26)

$$\overline{\phi_i} = \begin{bmatrix} \phi_1(x_i) \\ \phi_2(x_i) \\ \vdots \\ \phi_m(x_i) \end{bmatrix}$$

ϕ_j adalah salah satu dari kolom Φ sedangkan $\bar{\phi}_i^T$ salah satu baris dari Φ. Oleh karena itu, dari persamaan diatas diperoleh :

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \bar{\phi}_1^T \hat{w} \\ \bar{\phi}_2^T \hat{w} \\ \vdots \\ \bar{\phi}_n^T \hat{w} \end{bmatrix} = \Phi \hat{w}$$

(2.27)

Persamaan (2.27) disubsitusikan ke persamaan (2.26) sehingga diperoleh persamaan :

$$\Phi^T \hat{y} = \Phi^T y$$

$$\Phi^T \hat{y} = \Phi^T \Phi \hat{w}$$

(2.28)

Pada beberapa kasus ditemukan nilai invers dari $\Phi^T\Phi$ tidak dapat ditemukan karena $\Phi^T\Phi$ adalah matriks singular. Untuk menyelesaikan masalah matriks singular dapat menggunakan *weight-decay* atau sama dengan *ridge regression. Ridgre Regression* dibedakan menjadi dua bentuk yaitu *global ridge* dan *local ridge*.

2.12. Metode *Global Ridge-Regression*

Metode *global ridge regression* mengestimasi bobot dengan cara menambahkan nilai parameter regulasi yang bernilai positif pada SSE sehingga diperoleh fungsi sebagai berikut (Orr,1996:21 dalam Fajarani Juliaristi,2014):

$$C = \sum_{i=1}^{n} (y - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m} {w_j}^2$$

(2.29)

Keterangan :

$\hat{y}_i$: nilai prediksi variabel *output* ke-i

y : nilai variabel *output*

λ : parameter regulasi

w_j : bobot ke-j

n : banyak pengamatan

Untuk memperoleh bobot yang optimum digunakan differensial dari persamaan (2.31) dengan variabel bebas yang ada, penyelesaiannya adalah sebagai berikut :

$$\frac{\partial C}{\partial w_j} = 2\sum_{i=1}^{n} y - \hat{y}_i + 2\lambda \frac{\partial y}{\partial w_j} w_j$$

$$\frac{\partial C}{\partial w_j} = \sum_{i=1}^{n} y \frac{\partial y}{\partial w_j} - \sum_{i=1}^{n} \hat{y}_i \frac{\partial y}{\partial w_j} + \lambda w_j$$

$$\sum_{i=1}^{n} y \frac{\partial y}{\partial w_j} - \sum_{i=1}^{n} \hat{y}_i \frac{\partial y}{\partial w_j} + \lambda w_j = 0$$

$$\sum_{i=1}^{n} y \frac{\partial y}{\partial w_j} = \sum_{i=1}^{n} \hat{y}_i \frac{\partial y}{\partial w_j} + \lambda w_j$$

Berdasarkan persamaan $\frac{\partial y}{\partial w_j} = \phi_j(x)$, maka persamaan diatas menjadi :

$$\sum_{i=1}^{n} y\, \phi_j(x_i) + \lambda w_j = \sum_{i=1}^{n} \hat{y}\, \phi_j(x_i)$$

Dinyatakan dalam notasi vektor menjadi sebagai berikut :

$$\phi_j^T y + \lambda \widehat{w_j} = \phi_j^T \hat{y}$$

(2.30)

$$\begin{bmatrix} \phi_1^T y \\ \phi_2^T y \\ \vdots \\ \phi_m^T y \end{bmatrix} + \begin{bmatrix} \lambda\widehat{w_1} \\ \lambda\widehat{w_2} \\ \vdots \\ \lambda\widehat{w_m} \end{bmatrix} = \begin{bmatrix} \phi_1^T \hat{y} \\ \phi_2^T \hat{y} \\ \vdots \\ \phi_m^T \hat{y} \end{bmatrix}$$

$$\Phi^T y + \lambda\widehat{w_j} = \Phi^T \hat{y}$$

(2.31)

Keterangan :

λ : parameter regulasi

$\hat{w}$: vektor prediksi bobot

$\hat{y}$: vektor prediksi nilai output

Berdasarkan definisi diatas, diperoleh persamaan sebagai berikut (Orr,1996:21):

$$\Phi^T \hat{y} = \Phi^T y + \lambda\widehat{w_j}$$

$$\Phi^T \hat{y} = \Phi^T \Phi \hat{w} + \lambda\widehat{w_j}$$

$$\Phi^T \hat{y} = (\Phi^T \Phi + \lambda I_m)\hat{w}$$

(2.32)

I_m merupakan matriks identitas berukuran $m \times m$, sehingga diperoleh persamaan normal prediksi bobot adalah sebagai berikut :

$$\hat{w} = (\Phi^T \Phi + \lambda I_m)^{-1} \Phi^T \hat{y}$$

(2.33)

2.13. Model *Hybrid* ARIMA-RBFNN

Hybrid adalah kombinasi dari satu atau lebih metode atau model dalam satu sistem. Dalam hal ini adalah kombinasi antara ARIMA dan RBFNN. Penggunaan model *hybrid* diharapkan akan mendapatkan model yang lebih akurat. Di karenakan, pada dasarnya dalam dunia nyata sulit ditemukan kejadian deret waktu yang murni linier ataupun murni non linear. Secara umum, rumus kombinasi deret waktu yaitu sebagai berikut (Zhang,2003):

$$Y_t = N_t + L_t$$

(2.34)

Dimana N_t merupakan komponen non linier sedangkan L_t adalah komponen linier. Terdapat dua komponen yang digunakan untuk meramalkan data yaitu metode ARIMA digunakan untuk menyelesaikan model linier, dimana hasil residual dari model linier masih mengandung hubungan non linier. Dapat ditunjukkan dalam rumus berikut ini :

$$e_t = x_t + \widehat{N_t}$$

(2.35)

Dimana e_t merupakan residual dari model linier, $\widehat{N_t}$ merupakan nilai hasil peramalan ARIMA pada waktu t. Persamaan residual untuk model RBFNN dapat dituliskan sebagai berikut:

$$e_t = \mu + \emptyset_1 e_{t-1} + \emptyset_2 e_{t-2} + \cdots + \emptyset_p e_{t-p} + \varepsilon_t$$

(2.36)

Dimana ε_t merupakan error. Sehingga kombinasi untuk peramalan menjadi sebagai berikut:

$$Y_t = \widehat{N_t} + \widehat{L_t}$$

(2.37)

2.14. Ketepatan Model Terbaik

Proses analisis runtun waktu atau *time series*, ada beberapa model yang dapat menunjukkan suatu keadaan data. Kriteria yang dapat digunakan untuk pemilihan Model ARIMA terbaik setelah dilakukan proses identifikasi model yaitu nilai *Akaike's Information Criterion* (AIC).

Akaike's Information Criterion (AIC) diperkenalkan pertama kali oleh Akaike untuk mengidentifikasi model pada suatu kumpulan data. Model terbaik adalah model yang memiliki nilai AIC terkecil. Rumus untuk menentukan nilai AIC dinyatakan dalam persamaan berikut ini:

$$AIC = \log\left(\frac{\sum_{t=1}^{n}(X_t - \widehat{X_t})^2}{n}\right) + \frac{2m}{n}$$

(2.38)

Dengan :

n : Banyak data

X_t *:* Nilai aktual periode ke-t

$\widehat{X_t}$ *: Forecast* (ramalan) untuk periode t

2.15. Ketepatan Model Peramalan

Dalam membuat pemodelan peramalan dilakukan validasi untuk mengetahui kinerja metode peramalan yang digunakan. Pengujian tersebut digunakan untuk mengetahui *error* yang ada dalam model peramlaan yang

dibuat, salah satunya dengan menggunakan *Mean Absolute Percentage Error* (MAPE). *Mean Absolute Percentage Error* (MAPE) digunakan untuk mengukur kesalahan nilai dugaan dalam bentuk persentase rata-rata *absolute* kesalahan. Rumus menentukan nilai MAPE dapat dinyatakan dalam persamaan berikut ini:

$$MAPE = \frac{\sum_{t=1}^{n} \frac{|X_t - \widehat{X_t}|}{X_t}}{n} \times 100\%$$

(2.39)

Dengan :

X_t : Nilai aktual periode ke-t

$\widehat{X_t}$: *Forecast* (ramalan) untuk periode t

n : Banyak data