BAB II

KAJIAN TEORI

Bab II berisi teori-teori yang digunakan dalam penelitian ini. Teori yang digunakan yakni deskripsi kanker otak (*brain cancer*) dan diagnosisnya, citra digital, pengolahan citra, ekstraksi citra, *neural network* (NN), dan *radial basis function neural network* (RBFNN).

A. Kanker Otak

Kanker otak adalah sebuah penyakit dimana terjadi pertumbuhan sel otak yang tidak terkendali. Kanker otak dapat menyerang siapapun tanpa memandang usia maupun jenis kelamin. Kanker otak terdiri dari kanker otak primer dan kanker otak sekunder. Berdasarkan Kementrian Kesehatan Republik Indonesia (2015), terdapat dua jenis kanker otak, yakni:

1. Kanker otak primer

   Kanker otak primer terdiri dari kanker otak ganas dan kanker otak jinak. Kanker otak ganas merupakan sekumpulan sel yang tumbuh secara tidak normal atau tidak terkontrol dan dapat menyebar kejaringan sel lain. Kanker otak ganas terdiri dari tumor sel glial (glioma) yang meliputi glioma derajat rendah (astrositoma grade I/II, oligodendroglioma), glioma derajat tinggi (astrositoma anaplastik grade III, glioblastoma grade IV, dan anaplastik oligodendroglioma. Sedangkan kanker otak jinak merupakan sekumpulan sel yang tumbuh secara tidak normal dan tidak menyebar ke jaringan sel lain. Kanker otak primer yang tergolong jinak meliputi meningioma, tumor hipofisis dan schwannoma.

2. Kanker otak sekunder

Kanker otak sekunder atau yang disebut metastasis merupakan kanker otak yang disebabkan oleh kanker lainnya, seperti kanker paru (50%), payudara (15-25%), melanoma (5-20%), kolorektal dan ginjal. Lesi metastasis dapat tumbuh di parenkim otak (sekitar 75%) maupun di laptomeningeal.

Berdasarkan berbagai jenis kanker otak, terdapat gejala-gejala yang timbul pada penderita. Gejala-gejala yang timbul tergantung dari lokasi dan tingkat pertumbuhan tumor. Secara umum, gejala-gejala yang sering ditemukan adalah (Suwondo, 2011:1):

1. Gejala serebral umum

   Gejala serebral umum dapat berupa perubahan mental yang ringan (psikomotor asthenia) yang dapat dirasakan oleh keluarga dekat penderita, seperti mudah tersinggung, emosi, labil, pelupa, perlambatan aktivitas mental dan sosial, kehilangan inisiatif dan spontanitas, dan dapat mengalami depresi.

2. Nyeri kepala

   Diperkirakan 30% gejala awal kanker otak adalah nyeri kepala. Sifat dari nyeri kepala bervariasi, mulai dari ringan dan episodik sampai keras dan berdenyut. Pada umumnya, nyeri kepala bertambah berat pada malam hari dan pada pagi hari saat bangun tidur serta pada keadaan dimana terjadi peninggian tekanan tinggi intrakranial.

3. Muntah

4. Kejang

Berdasarkan gejala-gejala yang timbul, dapat dilakukan pemeriksaan dini atau diagnosis dini mengenai kanker otak. Berdasarkan Kementrian Kesehatan Republik Indonesia (2015), diagnosis kanker otak terdiri dari beberapa langkah yakni:

1. Anamnesis dan pemeriksaan fisik

   Secara umum, pemeriksaan anamnesis merupakan pemeriksaan dengan menghubungkan gejala-gejala yang timbul dari pasien, seperti sakit kepala, muntah, kejang, dan lain-lain.

2. Pemeriksaan laboraturium

   Pemeriksaan laboraturium dilakukan untuk mengetahui keadaan umum pasien, seperti darah lengkap, homostasis, LDH, fungsi hati, ginjal, gula darah, serolosi hepatitis B dan C, dan elektrolit lengkap. Pemeriksaan ini juga digunakan untuk mengetahui kesiapannya untuk terapi yang akan dijalankan (bedah, radiasi, maupun kemoterapi).

3. Pemeriksaan radiologi

   Pemeriksaan radiologi merupakan pemeriksaan yang dilakukan dengan mengambil gambar organ tubuh dalam menggunakan sebuah alat. Pemeriksaan radiologi standar terdiri dari CT scan dan MRI. CT scan berguna untuk melihat adanya tumor pada langkah awal penegakkan diagnosis. Sedangkan MRI dapat melihat gambaran jaringan lunak dengan lebih jelas. Hasil dari pemeriksaan radiologi berupa gambar atau citra digital.

B. Citra Digital

Menurut Gonzales & Woods (2002:1), citra digital merupakan fungsi pada bidang dua dimensi $f(x, y)$, dimana (x, y) adalah koordinat spasial (bidang), dan amplitudo *f* pada sembarang pasangan koordinat (x, y) disebut intensitas dari citra pada titik tersebut. Menurut Rinaldi Munir (2004:18-19), citra digital berbentuk persegi panjang yang dimensi ukuranya dinyatakan sebagai lebar kali panjang atau disimbolkan dengan $P \times Q$. Citra digital yang berukuran $P \times Q$ biasanya direpresentasikan dengan matrik berukuran $P \times Q$, yang ditunjukkan sebagai berikut:

$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0, Q-1) \\ f(1,0) & f(1,1) & \dots & f(1, Q-1) \\ \vdots & \vdots & \dots & \vdots \\ f(P-1,0) & f(P-1,1) & \dots & f(P-1, Q-1) \end{bmatrix} \qquad (2.1)$$

Elemen pada matriks merupakan elemen citra digital yang disebut piksel. Menurut Darma Putra (2010:40-44), citra digital terbagi menjadi tiga jenis yakni:

1. Citra biner (*binary image*)

   Citra biner merupakan cita yang paling sederhana. Citra ini hanya memiliki dua kemungkinan nilai piksel, yakni hitam dan putih. Warna hitam ditunjukkan dengan nilai 0 dan warna putih ditunjukkan dengan nilai 1. Citra biner juga disebut citra B&W (*black and white*) atau citra monokrom.

2. Citra Keabuan (*grayscale*)

   Citra keabuan merupakan citra yang mampu menghasilkan gradasi warna abu-abu dari warna hitam hingga putih. Citra keabuan memiliki kedalaman warna 8 bit (256 kombinasi warna keabuan). Nilai ini yang

digunakan untuk menunjukkan nilai intensitas citra. Nilai 0 untuk warna hitam, nilai 256 untuk nilai putih, dan nilai antara 0-256 untuk warna keabuan.

3. Citra Warna (*true Color*)

Citra warna merupakan perpaguan dari tiga kombinasi utama pembentuk warna, yang dikenal sebagai warna rgb. Rgb terdiri dari 3 warna dasar yakni merah (*red*), hijau (*green*), dan biru (*blue*) yang berukuran sama.

C. Ekstraksi Citra

Proses ekstraksi citra merupakan salah satu karakteristik penting yang digunakan dalam mengidentifikasi objek atau pola citra. Salah satu metode ekstraksi citra adalah *Gray Level Co-occurrence Matrix* (GLCM). GLCM merupakan salah satu metode ekstraksi citra yang digunakan dalam klasifikasi citra karena mampu memberikan informasi detail tentang tekstur citra (Gadkari, 2004). Hasil dari ekstraksi citra menggunakan metode GLCM berupa 14 fitur ekstraksi, diantaranya:

1. Energy

Nilai *energy* menggambarkan keteraturan penyebaran derajat suatu citra keabuan. Rumus *energy* adalah (Mohanaiah, *et al*., 2013:2):

$$E = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P_{ij}^2 \qquad (2.2)$$

i, j = koordinat spasial fungsi P_{ij}

P_{ij} = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

2. *Contrast*

Contrast digunakan untuk menghitung *range* perbedaan derajat keabuan dalam sebuah citra. Rumus *contrast* adalah (Haralick, Shanmugam & Dinstein, 1973: 619):

$$C = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i-j)^2 P_i \quad (2.3)$$

i, j = koordinat spasial fungsi P_i

P_i = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

3. *Correlation*

Correlation adalah fitur yang digunakan untuk menghitung ketergantungan linier sebuah citra. Rumus *correlation* adalah (Chiang & Weng, 2013:77):

$$C = \frac{\sum_{i-1}^{N_g} \sum_{j-1}^{N_g} (i.j.P_i) - \mu_x.\mu_y}{o_x.o_y} \quad (2.4)$$

i, j = koordinat spasial fungsi P_i

P_i = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

$\mu_x.\mu_y$ = rata-rata dari peluang marginal $P_x(i)$ dan $P_y(j)$

$o_x.o_y$ = standar deviasi dari peluang marginal $P_x(i)$ dan $P_y(j)$

4. *Sum of square (Variance)*

Variance merupakan ukuran heterogenitas atau variasi elemen-elemen matrik. Rumus *variance* adalah (Haralick, Shanmugam & Dinstein, 1973: 619):

$$V = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} (i-\mu)^2 P(i,j) \quad (2.5)$$

i, j = koordinat spasial fungsi P_i

P_{ij} = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

5. *Inverse Difference Moment* (IDM)

IDM merupakan ukuran dari homogenitas lokal. Rumus IDM adalah (Mohanaiah *et al*, 2013: 2):

$$IDM = \frac{\sum_{i=0}^{N_g-1}\sum_{j=0}^{N_g-1} P_{ij}}{1+(i-j)^2} \quad (2.6)$$

i, j = koordinat spasial fungsi P_{ij}

P_{ij} = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

6. Sum Average

Sum average merupakan fitur yang menunjukkan seberapa banyak nilai rata-rata *pixel* yang ada didalam citra. Rumus *sum average* adalah (Haralick, Shanmugam & Dinstein, 1973: 619):

$$SUM\ AVERAGE = \sum_{k=2}^{2N_g} kP_{x+y}(k) \quad (2.7)$$

$$P_{x+y}(k) = \sum_{i=1}^{N_g}\sum_{j=1}^{N_g} P(i,j)\,;\ i+j=k; k=2,3,\ldots,2N_g$$

N_g = level keabu-abu

7. Sum Variance

Sum variance menyatakan heterogenitas spasial (perbedaan) gambar (Abouelatta, 2013: 217). Rumus *sum variance* adalah (Haralick, Shanmugam & Dinstein, 1973: 619):

$$SUM\ VARIANCE = \sum_{i=2}^{2N_g}(i - SUM\ AVERAGE)^2 P_{x+y}(k) \quad (2.8)$$

$$P_{x+y}(k) = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P(i,j)\,;\ \ i+j=k; k=2,3,\ldots,2N_g$$

N_g = level keabu-abu

8. *Sum Entropy* (SE)

   Sum Entropy adalah fitur yang menunjukkan seberapa banyak level keabu-abuan yang acak. Rumus *Sum Entropy* (SE) adalah sebagai berikut (Haralick, Shanmugam & Dinstein, 1973: 619):

$$Si \quad E \qquad = -\sum_{k=2}^{2N_g} P_{x+y}(k) \log\{P_{x+y}(k)\} \qquad (2.9)$$

$$P_{x+y}(k) = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P(i,j)\,;\ \ i+j=k; k=2,3,\ldots,2N_g$$

N_g = level keabu-abu

9. *Entropy*

   Nilai *entropy* menunjukkan ketidakteraturan distribusi derajat keabuan sebuah citra. Rumus *entropy* adalah (Mohanaiah, *et al*., 2013: 2):

$$E \qquad = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} -P_i \,.\log P_i \qquad (2.10)$$

i, j = koordinat spasial fungsi P_i

P_i = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

10. *Difference Variance*

    Difference variance (DV) menyatakan ukuran variabilitas lokal. Rumus *difference variance* adalah (Haralick, Shanmugam & Dinstein, 1973: 619):

$$D \quad = v \qquad d \quad P_{x+y}(k) \qquad (2.11)$$

$$P_{x+y}(k) = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P(i,j)\,;\ \ i+j=k; k=2,3,\ldots,2N_g$$

11. *Difference Entropy*

Difference entropy (DE) merupakan ukuran variabilitas perbedaan mikro (lokal). Rumus *difference entropy* adalah (Haralick, Shanmugam & Dinstein, 1973:619):

$$D \quad = -\sum_{j=0}^{N_g-1} P_{x-y}(k) \log\{P_{x+y}(k)\} \qquad (2.12)$$

$$P_{x+y}(k) = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P(i,j)\,;\ \ i+j=k; k=2,3,\ldots,2N_g$$

N_g = level keabu-abu

12. *Maximum Probability*

Maximum probability menghitung tingkat keabu-abuan yang yang mempunyai peluang maksimum pada GLCM. Berikut rumus *maximum probability* (Soh & Tsatsoulis, 1999: 781):

$$M \qquad P \qquad = \max_{i,j}\{P(i,j)\} \qquad (2.13)$$

i, j = koordinat spasial fungsi P_i

P_i = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

13. *Homogeneity*

Fitur *homogeneity* menghitung keseragaman variasi derajat keabuan sebuah citra. Rumus *homogeneity* adalah (Soh & Tsatsoulis, 1999: 781):

$$H \qquad = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{1}{1+(i-j)^2} P_i \qquad (2.14)$$

i, j = koordinat spasial fungsi P_i

P_{ij} = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

14. Dissimiliraity

Dissimiliraity mirip dengan *contrast*. Nilai *dissimilarity* akan tinggi ketika daerah lokalnya memiliki nilai *contrast* yang tinggi. Rumus *dissimiliraity* adalah (Abouelatta, 2013: 216):

$$D = \sum_{i=0}^{N_g-1} \sum_{j=0}^{N_g-1} P(i,j)|i-j| \qquad (2.15)$$

i, j = koordinat spasial fungsi P_{ij}

P_{ij} = peluang nilai level keabuan pada baris ke-i dan kolom ke-j

N_g = level keabu-abu

Berikut ini merupakan contoh hasil ekstraksi citra dari 14 fitur citra berdasarkan Persamaan (2.2) sampai Persamaan (2.15). Gambar 2.1 merupakan contoh citra yang akan diekstraksi.

Gambar 2.1 Gambar Citra MRI yang diekstraksi

Hasil ekstraksi pada Gambar 2.1 dapat dilihat pada tabel berikut:

Tabel 2.1 Hasil Ekstraksi Citra MRI (Gambar 2.1)

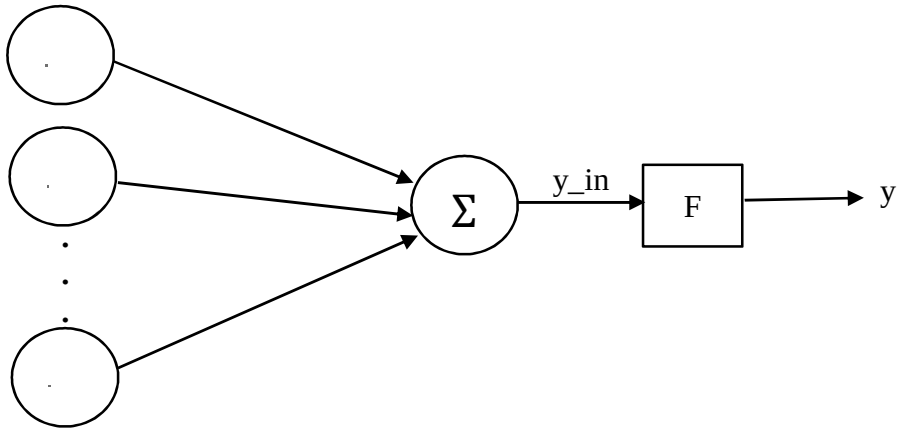
Contrast	*Correla-tion*	*Dissimi-larity*	*Energy*	*Entropy*	*Homo-genity*	*Max.Prob-babiliti*
0,2844	0,9759	0,1579	0,3253	1,7866	0,9369	0,5216
Sum of Square	*SumAva rage*	*SumVari ance*	*Sum Entropy*	*Diff.Vari ance*	*Diff.Entr opy*	*IDM*
39,755	11,664	124,577	1,606	0,2844	0,4477	0,9959

D. *Neural Network (NN)*

Neural Network (NN) atau yang disebut jaringan syaraf tiruan merupakan sistem pemrosesan informasi yang memiliki karakteristik mirip dengan jaringan saraf biologis (Fausett, 1994: 3). Menurut Siang (2005: 2), *Neural Network* dibentuk sebagai generalisasi model matematika dari jaringan syaraf biologi. *Neural network* pertama kali diperkenalkan pada tahun 1943 oleh seorang ahli syaraf Warren McCulloch dan seorang ahli logika Walter Pitss. Hingga saat ini, *neural network* sudah banyak diaplikasikan dalam berbagai permasalahan, seperti pengenalan pola (misal huruf, angka, suara atau tanda tangan), *signal processing* untuk menekan *noise* dalam saluran telepon, peramalan, klasifikasi, optimisasi, dll (Siang, 2005: 5).

Suatu *neural network* ditandai dengan (1) arsitektur yakni pola dari hubungan antar *neuron*, (2) algoritma pembelajaran yakni metode untuk menentukan bobot pada hubungan *neuron*, dan (3) fungsi aktivasi (Fausett, 1994:3). *Neural network* terdiri dari beberapa *neuron* dan saling berhubungan antar *neuron* yang lain. *Neuron-neuron* akan mentransformasikan informasi yang diterima melalui sambungan keluarnya menuju ke *neuron-neuron* yang lain,

hubungan ini sering disebut bobot. Gambar 2.2 menunjukkan struktur *neuron* pada jaringan syaraf.



Gambar 2.2 *Neural network* sederhana

Gambar 2.2 menunjukkan arsitektur jaringan dengan *p neuron input* $(x_1, x_2, \ldots, x_p)$ dan $(w_1, w_2, \ldots, w_p)$ merupakan bobot antara jaringan *input* dan *output.* F merupakan vektor *output* dan *y* merupakan nilai *output.*

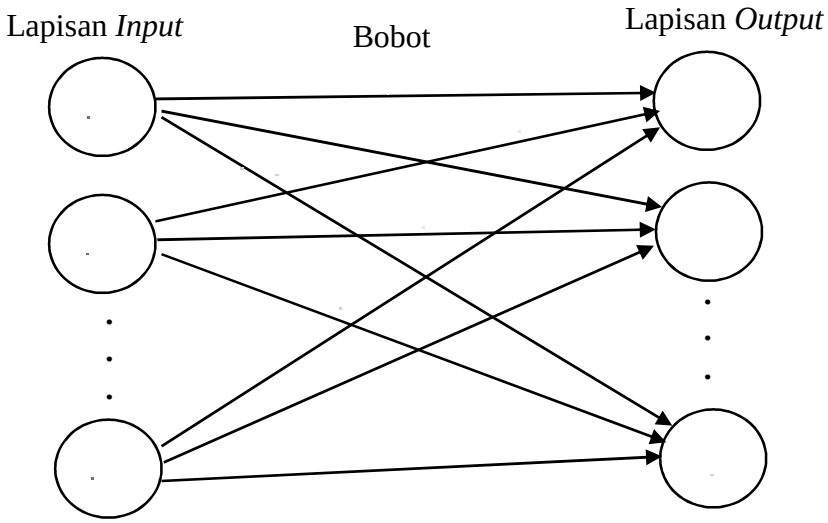
Pada *neural network* terdiri dari beberapa lapisan yang disebut dengan lapisan *neuron* (*neuron layer*). Secara umum, *neural network* terdiri dari 3 lapisan yakni lapisan masukan (*input layer)*, lapisan tersembunyi (*hidden layer*), dan lapisan keluaran (*output layer*). *Neuron-neuron* pada satu lapisan akan dihubungkan dengan lapisan-lapisan sebelum dan sesudahnya (kecuali lapisan *input* dan *output*). Menurut Fausett (1994: 3) karakteristik dari *neural network* ditentukan oleh beberapa hal, yakni:

1. Arsitektur

   Terdapat tiga jenis arsitektur dalam *neural network*, diantaranya adalah (Fausett, 1994: 12-15):

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

Pada jaringan layar tunggal, sekumpulan *input neuron* dihubungkan langsung dengan sekumpulan outputnya. Dalam beberapa model (misal *perceptron*), hanya ada sebuah *neuron output*.



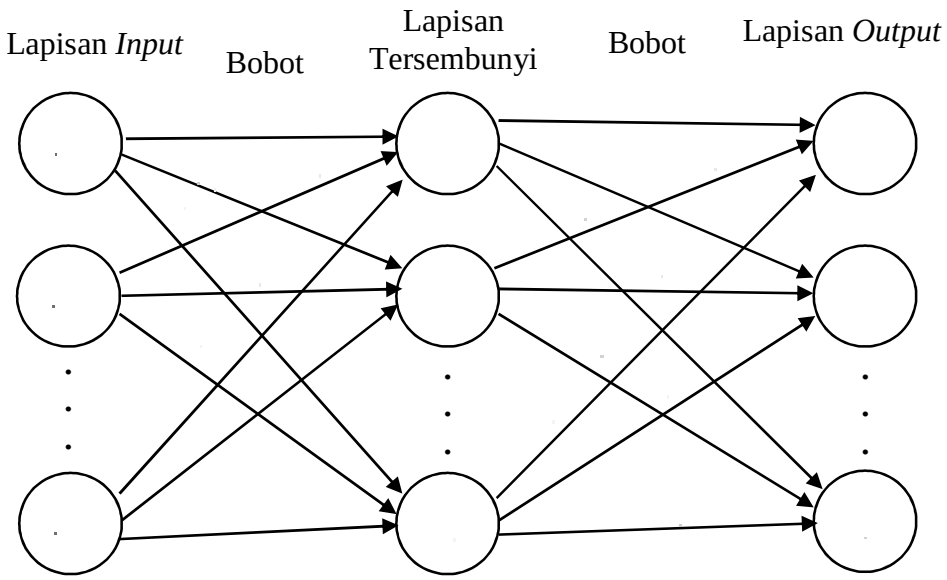
Gambar 2.3 Jaringan Layar Tunggal

Gambar 2.3 menunjukkan arsitektur jaringan dengan *n neuron input* ($x_1, x_2, \ldots, x_n$) dan *m neuron output* ($y_1, y_2, \ldots, y_m$). Pada jaringan ini, semua *neuron input* dihubungkan dengan semua *neuron output*, meskipun dengan bobot yang berbeda, tidak satu pun *neuron input* yang saling berhubungan. Demikian pula dengan *neuron output*.

b. Jaringan Layar Jamak (*Multilayer Network*)

Jaringan layar jamak merupakan jaringan dengan satu atau lebih layar simpul (*hidden neuron*) antara *neuron input* dan *neuron output*.

Terdapat layar bobot antara dua tingkat *neuron* yang berdekatan (*input, hidden, output*).

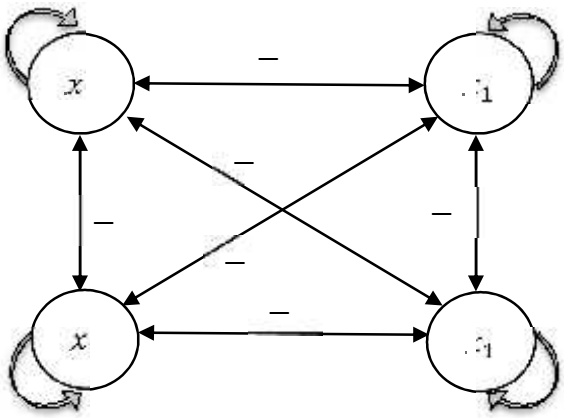


Gambar 2.4 Jaringan Layar Jamak

Gambar 2.4 adalah jaringan dengan *neuron input* ($x_1, x_2, \ldots, x_n$), sebuah layar tersembunyi yang terdiri dari *k neuron* ($\varphi_1, \varphi_2, \ldots, \varphi_k$) dan *neuron output* ($y_1, y_2, \ldots, y_m$). Jaringan ini dapat menyelesaikan masalah yang lebih kompleks dibandingkan dengan layar tunggal, meskipun terkadang proses pelatihan lebih kompleks dan lama.

c. Jaringan Layar Kompetitif (*Competitive Layer Network*)

Arsitektur ini memiliki bentuk yang berbeda, dimana antar *neuron* dapat saling dihubungkan. Gambar 2.5 merupakan salah satu contoh arsitektur jaringan layar kompetitif.



Gambar 2.5 Jaringan Layar Kompetitif

2. Fungsi Aktivasi

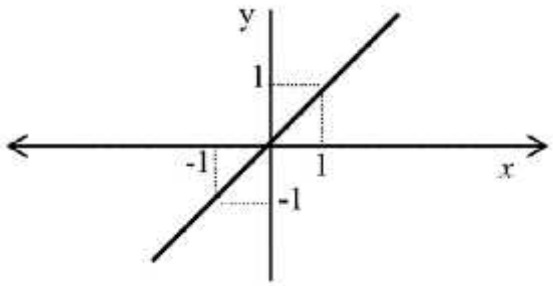
Dalam *neural network*, fungsi aktivasi digunakan untuk menentukan nilai *output* suatu *neuron*. Menurut Fausett (1994:17-19), fungsi aktivasi terdiri dari beberapa fungsi yakni:

a. Fungsi Identitas (*Identity Function*)

Pada fungsi ini, nilai *output* yang dihasilkan sama dengan nilai *input*nya. Fungsi identitas sering digunakan apabila menginginkan *output* berupa sembarang nilai riil. Berikut persamaan fungsi identitas:

$$y = f(x) = x, x \in \mathbb{R} \qquad (2.16)$$

Grafik fungsi identitas dapat digambar dengan bantuan Matlab R2013a. Sintaks untuk fungsi identitas adalah *purelin*. Grafik fungsi identitas dapat dilihat pada Gambar 2.6



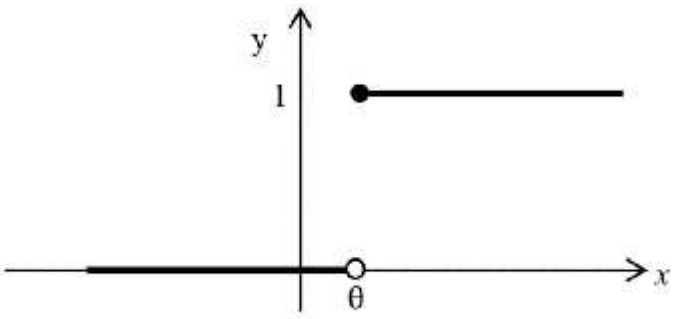
Gambar 2.6 Fungsi Indentitas

b. Fungsi Undak Biner

Fungsi undak biner digunakan untuk mengkonversikan *input* suatu variabel yang bernilai kontinu ke suatu *output* biner (0 atau 1). Fungsi undak biner (dengan *threshold* θ) dirumuskan sebagai berikut:

$$y = f(x) = \begin{cases} 1, x \geq \theta \\ 0, x < \theta \end{cases} \quad (2.17)$$

Grafik fungsi undak biner dapat digambar dengan bantuan Matlab R2013a. Sintaks untuk fungsi undak biner adalah *hardlim*. Grafik fungsi undak biner dapat dilihat pada Gambar 2.7



Gambar 2.7 Fungsi Undak Biner

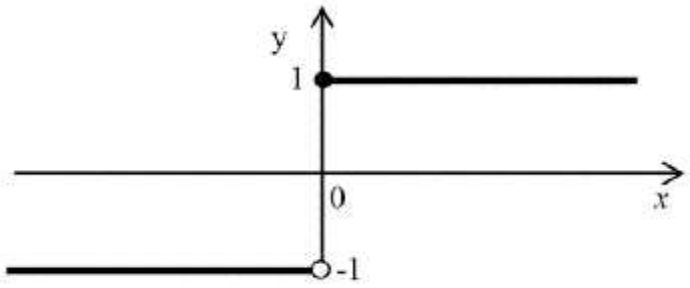
c. Fungsi *Bipolar*

Fungsi *bipolar* hampir sama dengan fungsi undak biner, perbedaannya terletak pada nilai *output* yang dihasilkan. Nilai *output*

fungsi bipolar berupa 1 dan -1. Fungsi *bipolar* dirumuskan sebagai berikut:

$$y = f(x) = \begin{cases} 1, x \geq 0 \\ -1, x < 0 \end{cases} \quad (2.18)$$

Grafik fungsi bipolar dapat digambar dengan bantuan Matlab R2013a. Sintaks untuk fungsi bipolar adalah *hardlims*. Grafik fungsi *bipolar* dapat dilihat pada Gambar 2.8



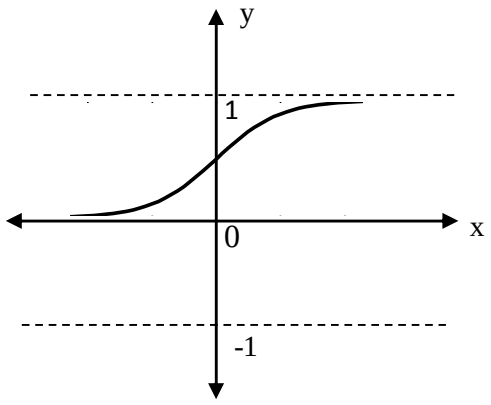
Gambar 2.8 Fungsi Bipolar

d. Fungsi *Sigmoid Biner*

Fungsi *sigmoid biner* sering digunakan karena nilai fungsinya yang terletak antara 0 dan 1. Fungsi *sigmoid biner* dirumuskan sebagai berikut:

$$y = f(x) = \frac{1}{1+e^{-x}}, x \in \mathbb{R} \quad (2.19)$$

Grafik fungsi sigmoid biner dapat digambar dengan bantuan Matlab R2013a. Sintaks untuk fungsi sigmoid biner adalah *logsig*. Grafik fungsi *sigmoid biner* dapat dilihat pada Gambar 2.9



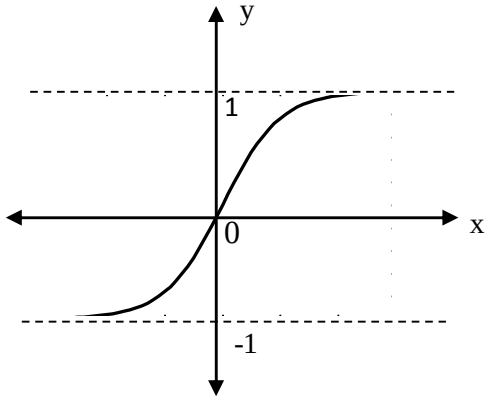
Gambar 2.9 Fungsi Sigmoid Biner

e. Fungsi *Sigmoid Bipolar*

Fungsi *sigmoid bipolar* hampir sama dengan fungsi *sigmoid biner*, perbedaannya terletak pada rentang nilai *output*nya. Rentang nilai *output* fungsi *sigmoid bipolar* adalah -1 sampai 1. Fungsi *sigmoid bipolar* dirumuskan sebagai berikut:

$$y = f(x) = \frac{1-e^{-x}}{1+e^{-x}}, x \in \mathbb{R} \qquad (2.20)$$

Grafik fungsi sigmoid bipolar dapat digambar dengan bantuan Matlab R2013a. Sintaks untuk fungsi sigmoid bipolar adalah *tansig*. Grafik fungsi *sigmoid bipolar* dapat dilihat pada Gambar 2.10



Gambar 2.9 Fungsi Sigmoid Bipolar

3. Algoritma Pembelajaran

Algoritma pembelajaran dalam *neural network* didefinisikan sebagai metode untuk menentukan nilai-nilai bobot penghubung yang tepat dalam mengirimkan suatu informasi antar *neuron* (Siang, 2005: 3). Menurut Siang (2005:28-29), metode pembelajaran *neural network* berdasarkan strategi pembelajaran dibagi menjadi 2 metode yakni:

a. Pembelajaran Tak Terawasi (*Unsupervised Learning*)

Dalam pembelajaran tak terawasi, tidak terdapat target *output* yang akan mengarahkan proses pembelajaran. Dalam pembelajarannya, perubahan bobot jaringan dilakukan berdasarkan parameter tertentu dan jaringan dimodifikasi menurut ukuran parameter tersebut. Tujuan pembelajaran ini adalah untuk mengelompokkan unit-unit yang hampir sama ke dalam suatu area tertentu sesuai nilai *input* yang diberikan.

b. Pembelajaran Terawasi (*Supervised Learning*)

Dalam pembelajaran terawasi terdapat sejumlah data (*input* – target output) yang dipakai untuk melatih jaringan hingga diperoleh bobot yang diinginkan. Pasangan data tersebut berfungsi untuk melatih jaringan hingga diperoleh bentuk yang terbaik. Pasangan data yang diberikan akan memberikan informasi yang jelas tentang bagaimana sistem harus mengubah dirinya untuk meningkatkan kinerjanya.

Pada setiap pembelajaran, suatu *input* diberikan ke jaringan. Jaringan akan memproses dan mengeluarkan *output*. Selisih antara *output* jaringan dengan target (*output* yang diinginkan) merupakan kesalahan

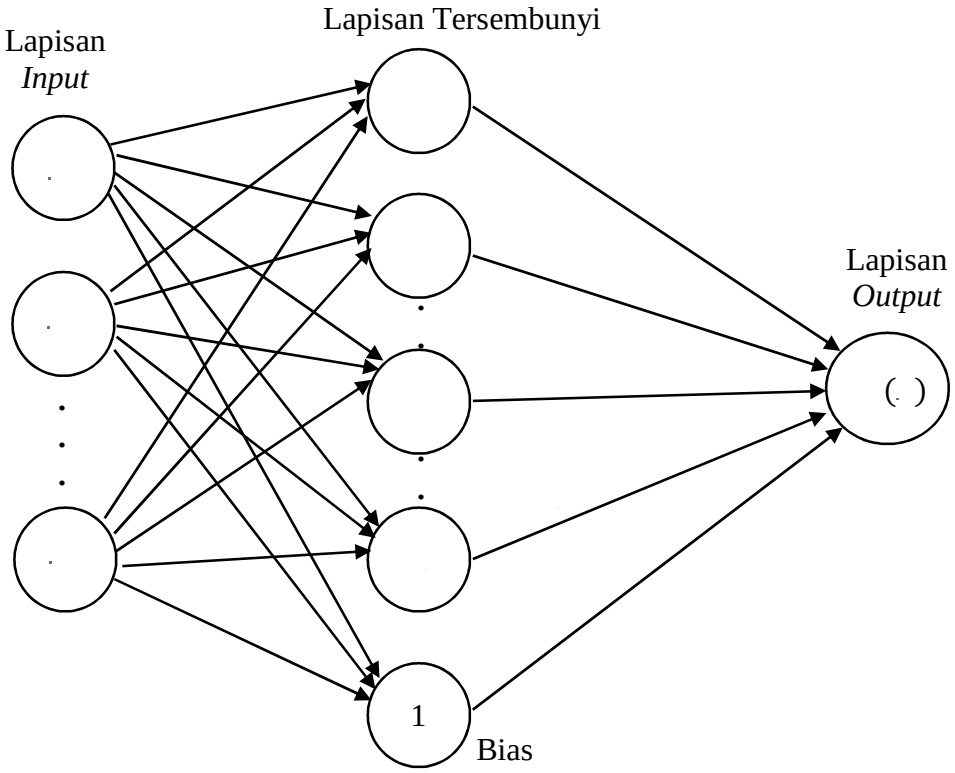
(*error)* yang terjadi. Jaringan akan memodifikasi bobot sesuai kesalahan tersebut.

E. *Radial Basis Function Neural Network (RBFNN)*

Radial Basis Function Neural Network (RBFNN) merupakan salah satu model *neural network*. Menurut Qasem, *et al* (2013: 165), RBFNN adalah jaringan *feed-forward* dengan tiga lapisan yakni, sebuah lapisan *input* dengan *i neuron*, sebuah lapisan tersembunyi dengan *j neuron*, dan lapisan *output* dengan satu atau beberapa *neuron*. Kinerja RBFNN tergantung pada pilihan yang tepat tiga parameter penting (pusat *cluster*, jarak variabel *input* ke pusat *cluster*, dan bobot antar lapisan). Nilai parameter ini umumnya diketahui dan dapat ditemukan selama proses pembelajaran jaringan (Pislaru & Shebani, 2014: 1678).

1. Arsitektur *Radial Basil Function Neural Network*

   Dalam RBFNN, variabel *input* masing-masing ditetapkan pada *neuron* dalam lapisan *input* dan masuk secara langsung ke lapisan tersembunyi tanpa bobot (Balasubramanie et al, 2009: 136). Hal ini yang membedakan RBFNN dengan model *neural network* yang lain. Pada lapisan tersembunyi RBFNN dilakukan transformasi *nonlinear* terhadap data dari lapisan *input* menggunakan fungsi radial basis sebelum diproses secara linier pada lapisan *output* (Wei, *et al*, 2011: 65). Arsitektur *Radial Basis Function Neural Network* ditunjukkan pada Gambar 2.11 berikut:



Gambar 2.11 Arsitektur RBFNN Sederhana

Pada Gambar 2.11, $(x_i, i = 1, 2, \ldots, n)$ merupakan *neuron* pada lapisan *input*, $(\varphi_j,\ j = 1, 2, \ldots k)$ merupakan *neuron* pada lapisan tersembunyi, dan $f(x)$ merupakan *neuron* pada lapisan *output*. Bobot antara lapisan tersembunyi dan lapisan output disimbolkan dengan w_j. Dalam arsitektur RBFNN juga ditambahkan sebuah *neuron* bias pada lapisan tersembunyi. Bias tersebut berfungsi untuk membantu *neural network* dalam mengolah informasi dengan lebih baik.

2. Model *Radial Basis Function neural network* (RBFNN)

Pada RBFNN, lapisan tersembunyi menghitung jarak variabel *input* dengan pusat *cluster*, kemudian dengan fungsi aktivasi $\varphi_j(x)$ menuju lapisan

output (Pislaru & Shebani, 2014:1678). Menurut Andrew (2002: 63), ada beberapa fungsi aktivasi dalam RBFNN yakni:

a) Fungsi Gaussian

$$\varphi(x) = \exp\left(-\frac{(x-c)^2}{r^2}\right) \tag{2.21}$$

b) Fungsi Multikuadratik

$$\varphi(x) = \sqrt{(x-c)^2 + r^2} \tag{2.22}$$

c) Fungsi Invers Multikuadratik

$$\varphi(x) = \frac{r}{\sqrt{(x-c)^2+r^2}} \tag{2.23}$$

d) Fungsi Cauchy

$$\varphi(x) = \frac{\left((x-c)^2+r^2\right)^{-1}}{r} \tag{2.24}$$

dengan,

r = jarak maksimal variabel *input* ke pusat *cluster*

x = nilai *input* variabel

c = nilai pusat pada *neuron* tersembunyi

$\varphi(x)$ = fungsi aktivasi *neuron* tersembunyi

Output y yang dihasilkan dari model RBFNN merupakan kombinasi linier dari bobot w_j dengan fungsi aktivasi $\varphi_j(x)$ dan bobot bias w_0. Rumus *output* $f(x)$ dirumuskan sebagai berikut (Ali & Dale, 2003: 809):

$$f(x) = \sum_{j=1}^{K} w_j \varphi_j(\boldsymbol{x}) + w_0 \tag{2.25}$$

dengan,

$$f(x) = \sum_{j=1}^{K} w_j \exp\left(-\sum_{i=1}^{n} \frac{(x_i - c_j)^2}{r_j^2}\right) + w_0$$

dimana,

k = banyak *neuron* tersembunyi

w_j = bobot dari *neuron* lapisan tersembunyi ke-*j* menuju *neuron output*

w_0 = bobot bias menuju *neuron output*

$\varphi_j(x)$ = fungsi aktivasi *neuron* tersembunyi ke- *j*

$\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$ merupakan vektor *input*

$j = 1, 2, \ldots, k$

3. Algoritma Pembelajaran *Radial Basis Function Neural Network*

Proses pembelajaran dalam RBFNN sedikit berbeda pada proses *learning* dengan model *neural network* lainnya. RBFNN menggabungkan antara pembelajaran terawasi (*supervised learning*) dan pembelajaran tak terawasi (*unsupervised learning*). Metode pembelajaran tidak terawasi (*unsupervised learning*) digunakan pada proses dari lapisan *input* menuju lapisan tersembunyi dan metode pembelajaran terawasi (*supervised learning*) digunakan pada proses yang terjadi dari lapisan tersembunyi menuju lapisan *output* (Chen, *et al*, 2005: 320).

Algoritma pembelajaran RBFNN terbagi menjadi tiga bagian yakni (Andrew, 2002:70):

a) Menentukan pusat dan jarak dari setiap fungsi basis.

Pusat dan jarak dari setiap fungsi basis dapat dicari menggunakan metode pengelompokan (*clustering*). Jarak yang umumnya digunakan yakni jarak *Euclide*. Jarak *Euclide* dapat digunakan dalam mengetahui

tingkat kemiripan. Semakin kecil nilai jarak *Euclide* maka semakin tinggi tingkat kemiripan, berlaku untuk sebaliknya.

b) Jumlah *neuron* pada lapisan tersembunyi sama dengan jumlah fungsi basis.

c) Bobot lapisan *output* jaringan optimum.

F. *K-means Clustering*

Terdapat beberapa metode yang dapat digunakan dalam proses pengelompokan (*clustering*), salah satu metodenya adalah *K-means clustering*. *K-means* merupakan algoritma untuk mengelompokan atau mengklasifikasi objek/data berdasarkan unsur/fitur ke sejumlah *k* kelompok/*cluster*, dengan *k* adalah bilangan bulat positif. Berikut merupakan algoritma *K-means clustering* (Johnson & Wichern, 2007: 696):

1) Partisi data ke dalam *k cluster*
2) Tempatkan setiap data/obyek ke *cluster* terdekat. Kedekatan dua obyek ditentukan berdasarkan jarak kedua obyek tersebut. Persamaan jarak *Euclide* antara dua titik sebarang P dan Q dengan koordinat P($x_1, x_2, \ldots, x_n$) dan Q($y_1, y_2, \ldots, y_n$) adalah sebagai berikut:

$$d(P,Q) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2} \qquad (2.26)$$

Hitung ulang nilai pusat untuk *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).

Contoh penggunaan metode *K-Means clustering*:

Misalkan akan diukur 14 variabel $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_1$, x_1 , x_1 , x_1 , dan x_1 untuk masing-masing empat item A, B, C, dan D. Data yang diberikan dalam Tabel 2.2:

Tabel 2.2 Data Pengamatan

Item	Pengamatan						
	x_1	x_2	x_3	x_4	x_5	x_6	x_7
A	-0,29	0,36	0,29	-0,21	0,75	-0,58	0,46
B	-0,68	0,63	-0,73	1,03	-0,68	0,708	1,21
C	1,31	-1,01	1,11	-0,31	0,101	-0,95	-1,102
D	-0,69	0,48	-1,05	1,37	-1,27	1,18	0,902
Item	Pengamatan						
	x_8	x_9	x_1	x_1	x_1	x_1	x_1
A	0,53	0,45	0,26	0,84	-0,29	0,43	0,22
B	1,03	0,89	1,06	-0,62	-0,68	-0,709	0,706
C	-1,112	-0,98	-0,97	-0,095	1,316	1,094	-1,309
D	0,93	0,89	1,138	-1,27	-0,69	-1,15	0,74

Data pada Tabel 2.2 dikelompokkan menjadi 2 *cluster* ($k = 2$). Untuk mengimplementasi metode *K-Means* dengan dua *cluster*, pertama dipartisi item menjadi 2 *cluster* (AB) dan (CD), kemudian hitung koordinat pusat *cluster* (rata-rata *cluster*), seperti pada Tabel 2.3:

Tabel 2.3 Koordinal Pusat *Cluster* partisi pertama

Cluster	Koordinat Pusat			
	x_1	x_2	x_3	x_4
AB	$\frac{-0,29 + (-0,68)}{2} = -0,485$	$\frac{0,36 + 0,63}{2} = 0,51$	$\frac{-0,29 + (-0,73)}{2} = -0,22$	$\frac{-0,21 + 1,03}{2} = 0,41$

	Koordinat Pusat			
	x_5	x_6	x_7	x_8
	$\frac{0,75+(-0,68)}{2} = 0,035$	$\frac{-0,58+0,706}{2} = 0,063$	$\frac{0,46+1,21}{2} = 0,835$	$\frac{0,53+1,03}{2} = 0,78$
	Koordinat Pusat			
	x_9	x_1	x_1	x_1
	$\frac{0,45+0,89}{2} = 0,67$	$\frac{0,26+1,06}{2} = 0,66$	$\frac{0,84+(-0,62)}{2} = 0,11$	$\frac{-0,29+(-0,68)}{2} = -0,485$
	Koordinat Pusat			
	x_1		x_1	
	$\frac{0,43+(-0,709)}{2} = -0,1395$		$\frac{0,22+0,706}{2} = 0,463$	
Cluster	Koordinat Pusat			
	x_1	x_2	x_3	x_4
CD	$\frac{-1,31+(-0,69)}{2} = 0,31$	$\frac{-1,01+0,48}{2} = -0,265$	$\frac{1,11+(-1,05)}{2} = 0,03$	$\frac{-0,31+1,37}{2} = 0,53$
	Koordinat Pusat			
	x_5	x_6	x_7	x_8
	$\frac{0,101+(-1,27)}{2} = -0,5845$	$\frac{-0,95+1,18}{2} = 0,115$	$\frac{-1,102+0,902}{2} = -0,1$	$\frac{-1,112+0,93}{2} = -0,091$
	Koordinat Pusat			
	x_9	x_1	x_1	x_1
	$\frac{-0,98+0,89}{2} = -0,045$	$\frac{-0,97+1,138}{2} = 0,084$	$\frac{-0,09+(-1,27)}{2} = -0,6825$	$\frac{1,316+(-0,69)}{2} = 0,313$
	Koordinat Pusat			
	x_1		x_1	
	$\frac{1,094+(-1,15)}{2} = -0,528$		$\frac{-1,309+0,74}{2} = -0,2845$	

Selanjutnya menghitung jarak *Euclide* untuk masing-masing item dari pusat *cluster* dan menempatkan kembali masing-masing item ke *cluster* terdekat. Jika sebuah item berpindah dari konfigurasi awal, pusat *cluster* harus dihitung kembali. Untuk koordinat ke-*i*, $i = 1, 2, \ldots, p$, pusat *cluster* dapat dihitung kembali dengan cara:

$\bar{x}_{i,n\ w} = \frac{n\bar{x}_i + x_j}{n+1}$ jika item ke-*j* ditambahkan ke dalam *cluster*

$\bar{x}_{i,n} = \frac{n\bar{x}_i + x_j}{n-1}$ jika item ke-*j* dihilangkan dari *cluster*

Dengan *n* adalah jumlah item pada *cluster* sebelumnya. Misal item D dengan koordinat $(-0{,}69; 0{,}48; -1{,}05; 1{,}37; -1{,}27; 1{,}18; 0{,}902; 0{,}93; 0{,}89; 1{,}138; -1{,}27; -0{,}69; -1{,}15; 0{,}74)$ dipindahkan ke dalam *cluster* (AB). *Cluster* baru ABD dan C.

Untuk menghitung jarak *Euclide* didapatkan:

$$d(A, (A\ \ )) = \sqrt{\begin{matrix}(-0{,}29 + 0{,}485)^2 + (0{,}36 - 0{,}51)^2 + (0{,}29 + 0{,}22)^2 \\ +(-0{,}21 - 0{,}41)^2 + (0{,}75 - 0{,}035)^2 + (-0{,}58 - 0{,}063)^2 \\ +(0{,}46 - 0{,}835)^2 + (0{,}53 - 0{,}78)^2 + (0{,}45 - 0{,}67)^2 \\ +(0{,}26 - 0{,}66)^2 + (0{,}84 - 0{,}11)^2 + (-0{,}29 + 0{,}485)^2 \\ +(0{,}43 + 0{,}1395)^2 + (0{,}22 - 0{,}463)^2\end{matrix}}$$

$$= 1{,}7284$$

$$d(A, (C\ \ )) = \sqrt{\begin{matrix}(-0{,}29 - 0{,}31)^2 + (0{,}36 + 0{,}265)^2 + (0{,}29 - 0{,}03)^2 \\ +(-0{,}21 - 0{,}53)^2 + (0{,}75 + 0{,}5845)^2 + (-0{,}58 - 0{,}115)^2 \\ +(0{,}46 + 0{,}1)^2 + (0{,}53 + 0{,}091)^2 + (0{,}45 + 0{,}045)^2 \\ +(0{,}26 - 0{,}084)^2 + (0{,}84 + 0{,}6825)^2 + (-0{,}29 - 0{,}313)^2 \\ +(0{,}43 + 0{,}528)^2 + (0{,}22 + 0{,}2845)^2\end{matrix}}$$

$$= 2{,}915$$

Karena jarak A dengan (AB) lebih dekat, maka A tetap pada *cluster* (AB).

$$d(B,(A\ \ )) = \sqrt{\begin{matrix}(-0{,}689+0{,}485)^2+(0{,}63-0{,}51)^2+(-0{,}73+0{,}22)^2\\ +(1{,}03-0{,}41)^2+(-0{,}68-0{,}035)^2+(0{,}706-0{,}063)^2\\ +(1{,}21-0{,}835)^2+(1{,}03-0{,}78)^2+(0{,}89-0{,}67)^2\\ +(1{,}06-0{,}66)^2+(-0{,}62-0{,}11)^2+(-0{,}68+0{,}485)^2\\ +(-0{,}709+0{,}1395)^2+(0{,}706-0{,}463)^2\end{matrix}}$$

$$= 1{,}7284$$

$$d(B,(C\ \ )) = \sqrt{\begin{matrix}(-0{,}68-0{,}31)^2+(0{,}63+0{,}265)^2+(-0{,}73-0{,}03)^2\\ +(1{,}03-0{,}53)^2+(-0{,}68+0{,}5845)^2+(0{,}706-0{,}115)^2\\ +(1{,}21+0{,}1)^2+(1{,}03+0{,}091)^2+(0{,}89+0{,}045)^2\\ +(1{,}06-0{,}084)^2+(-0{,}62+0{,}6825)^2+(-0{,}68-0{,}313)^2\\ +(-0{,}709+0{,}528)^2+(0{,}706+0{,}2845)^2\end{matrix}}$$

$$= 3{,}1257$$

Karena jarak B dengan (AB) lebih dekat, maka B tetap pada *cluster* (AB).

$$d(C,(A\ \ )) = \sqrt{\begin{matrix}(1{,}31+0{,}485)^2+(-1{,}01-0{,}51)^2+(1{,}11+0{,}22)^2\\ +(-0{,}31-0{,}41)^2+(0{,}101-0{,}035)^2+(-0{,}95-0{,}063)^2\\ +(-1{,}102-0{,}835)^2+(-1{,}112-0{,}78)^2+(-0{,}98-0{,}67)^2\\ +(-0{,}97-0{,}66)^2+(-0{,}095-0{,}11)^2+(1{,}316+0{,}485)^2\\ +(1{,}094+0{,}1395)^2+(-1{,}309-0{,}463)^2\end{matrix}}$$

$$= 5{,}2954$$

$$d(C,(C\ \ )) = \sqrt{\begin{matrix}(1{,}31-0{,}31)^2+(-1{,}01+0{,}265)^2+(1{,}11-0{,}03)^2\\ +(-0{,}31-0{,}53)^2+(0{,}101+0{,}5845)^2+(-0{,}95-0{,}115)^2\\ +(-1{,}102+0{,}1)^2+(-1{,}112+0{,}091)^2+(-0{,}98+0{,}045)^2\\ +(-0{,}97-0{,}084)^2+(-0{,}095+0{,}682)^2+(1{,}316-0{,}313)^2\\ +(1{,}094+0{,}528)^2+(-1{,}309+0{,}2845)^2\end{matrix}}$$

$$= 3{,}4424$$

Karena jarak C dengan (CD) lebih dekat, maka C tetap pada *cluster* (CD).

$$d(D,(A\ \ )) = \sqrt{\begin{matrix}(-0{,}69+0{,}485)^2+(0{,}48-0{,}51)^2+(-1{,}05+0{,}22)^2\\ +(1{,}37-0{,}41)^2+(-1{,}27-0{,}035)^2+(1{,}18-0{,}063)^2\\ +(0{,}902-0{,}835)^2+(0{,}93-0{,}78)^2+(0{,}89-0{,}67)^2\\ +(1{,}138-0{,}66)^2+(-1{,}27-0{,}11)^2+(-0{,}69+0{,}485)^2\\ +(-1{,}15+0{,}1395)^2+(0{,}74-0{,}463)^2\end{matrix}}$$

$$= 2{,}8199$$

$$d(D,(C\ \ )) = \sqrt{\begin{matrix}(-0{,}69-0{,}31)^2+(0{,}48+0{,}265)^2+(-1{,}05-0{,}03)^2\\ +(1{,}37-0{,}53)^2+(-1{,}27+0{,}5845)^2+(1{,}18-0{,}115)^2\\ +(0{,}902+0{,}1)^2+(0{,}93+0{,}091)^2+(0{,}89+0{,}045)^2\\ +(1{,}138-0{,}084)^2+(-1{,}27+0{,}6825)^2+(-0{,}69-0{,}313)^2\\ +(-1{,}15+0{,}528)^2+(0{,}74+0{,}2845)^2\end{matrix}}$$

$$= 3{,}4424$$

Karena jarak D dengan (AB) lebih dekat, maka D dipindah ke *cluster* (AB).

Berdasarkan pengelompokan kembali dengan jarak minimum seperti diatas, didapatkan *cluster* baru yang terbentuk yakni (ABD) dan (C) dengan nilai pusat baru:

Tabel 2.4 Koordinat Pusat *Cluster* Partisi Kedua

Cluster	Koordinat Pusat			
	x_1	x_2	x_3	x_4
ABD	$\frac{2(-0{,}4)+(-0{,}6)}{2+1} = -0{,}553$	$\frac{2(0{,}51)+0{,}48}{2+1} = 0{,}5$	$\frac{2(-0{,}2)+(-1{,}1)}{2+1} = -0{,}496$	$\frac{2(0{,}41)+1{,}37}{2+1} = 0{,}73$
	Koordinat Pusat			
	x_5	x_6	x_7	x_8
	$\frac{2(0{,}03)+(-1{,}27)}{2+1} = 0{,}035$	$\frac{2(0{,}06)+1{,}18}{2+1} = 0{,}435$	$\frac{2(0{,}83)+0{,}902}{2+1} = 0{,}857$	$\frac{2(0{,}78)+0{,}93}{2+1} = 0{,}83$
	Koordinat Pusat			
	x_9	x_1	x_1	x_1
	$\frac{2(0{,}67)+0{,}89}{2+1} = 0{,}743$	$\frac{2(0{,}66)+1{,}13}{2+1} = 0{,}819$	$\frac{2(0{,}11)+(-1{,}2)}{2+1} = -0{,}35$	$\frac{2(-0{,}4)+(-1{,}2)}{2+1} = -0{,}553$
	Koordinat Pusat			
	x_1		x_1	
	$\frac{2(-0{,}13)+(-1{,}15)}{2+1} = -0{,}476$		$\frac{2(0{,}46)+0{,}74}{2+1} = 0{,}555$	

Cluster	Koordinat Pusat			
	x_1	x_2	x_3	x_4
C	$\frac{2(-0,3)-(-0,6)}{2-1} = 1,31$	$\frac{2(-0,2)-0,48}{2-1} = -1,01$	$\frac{2(0,03)-(-1,1)}{2-1} = 1,11$	$\frac{2(0,53)-1,37}{2-1} = -0,31$
	Koordinat Pusat			
	x_5	x_6	x_7	x_8
	$\frac{2(-0,58)+1,27}{2-1} = 0,101$	$\frac{2(0,11)-1,18}{2-1} = -0,95$	$\frac{2(-0,1)-0,902}{2-1} = -1,102$	$\frac{2(-0,09)-0,93}{2-1} = -1,112$
	Koordinat Pusat			
	x_9	x_1	x_1	x_1
	$\frac{2(-0,04)-0,89}{2-1} = -0,98$	$\frac{2(0,08)-1,13}{2-1} = -0,97$	$\frac{2(-0,6)-(-1,2)}{2-1} = -0,095$	$\frac{2(0,31)-(-1,2)}{2-1} = 1,316$
	Koordinat Pusat			
	x_1		x_1	
	$\frac{2(-0,52)-(-1,15)}{2-1} = 0,094$		$\frac{2(-0,28)-0,74}{2-1} = -1,309$	

Pusat *cluster* baru yang terbentuk adalah ABD $(-0,55; 0,5; -0,49; 0,73; -0,4; 0,43; 0,85; 0,83; 0,74; 0,81; -0,35; -0,55; -0,47; 0,55)$ dan C $(1,31; -1,01; 1,11; -0,31; 0,101; -0,95; -1,102; -1,112; -0,98; -0,97; -0,095; 1,316; 0,094; -1,309)$. Selanjutnya perhitungan jarak *Euclide* dan pengelompokan dilakukan kembali hingga didapat nilai pusat yang sama dengan sebelumnya (stabil). Pada contoh ini, dilakukan perhitungan jarak *Euclide* dan pengelompokan kembali dan didapatkan nilai pusat yang sama pada *cluster* ABD dan C. Langkah selanjutnya mencari jarak maksimum setiap item terhadap masing-masing *cluster*.

$$d(A,(A\quad)) = \sqrt{\begin{matrix}(-0{,}29+0{,}55)^2+(0{,}36-0{,}5)^2+(0{,}29+0{,}49)^2\\ +(-0{,}21-0{,}73)^2+(0{,}75+0{,}4)^2+(-0{,}58-0{,}43)^2\\ +(0{,}46-0{,}85)^2+(0{,}53-0{,}83)^2+(0{,}45-0{,}74)^2\\ +(0{,}26-0{,}81)^2+(0{,}84+0{,}35)^2+(-0{,}29+0{,}55)^2\\ +(0{,}43+0{,}47)^2+(0{,}22-0{,}55)^2\end{matrix}}$$

$$= 2{,}6463$$

$$d(B,(A\quad)) = \sqrt{\begin{matrix}(-0{,}68+0{,}55)^2+(0{,}63-0{,}5)^2+(-0{,}73+0{,}49)^2\\ +(1{,}03-0{,}73)^2+(-0{,}68+0{,}4)^2+(0{,}706-0{,}43)^2\\ +(1{,}21-0{,}85)^2+(1{,}03-0{,}83)^2+(0{,}89-0{,}74)^2\\ +(1{,}06-0{,}81)^2+(-0{,}62+0{,}35)^2+(-0{,}68+0{,}55)^2\\ +(-0{,}709+0{,}47)^2+(0{,}706-0{,}55)^2\end{matrix}}$$

$$= 0{,}8594$$

$$d(C,(C)) = \sqrt{\begin{matrix}(1{,}31-1{,}31)^2+(-1{,}01+1{,}01)^2+(1{,}11-1{,}11)^2\\ +(-0{,}31+0{,}31)^2+(0{,}101-0{,}101)^2+(-0{,}95+0{,}95)^2\\ +(-1{,}102+1{,}102)^2+(-1{,}112+1{,}112)^2+(-0{,}98+0{,}98)^2\\ +(-0{,}97+0{,}97)^2+(-0{,}095+0{,}095)^2+(1{,}316-1{,}316)^2\\ +(1{,}094-1{,}094)^2+(-1{,}309+1{,}309)^2\end{matrix}}$$

$$= 0$$

$$d(D,(A\quad)) = \sqrt{\begin{matrix}(-0{,}69+0{,}55)^2+(0{,}48-0{,}5)^2+(-1{,}05+0{,}49)^2\\ +(1{,}37-0{,}73)^2+(-1{,}27+0{,}4)^2+(1{,}18-0{,}43)^2\\ +(0{,}902-0{,}85)^2+(0{,}93-0{,}83)^2+(0{,}89-0{,}74)^2\\ +(1{,}13-0{,}81)^2+(-1{,}27+0{,}35)^2+(-0{,}69+0{,}55)^2\\ +(-1{,}15+0{,}47)^2+(0{,}74-0{,}55)^2\end{matrix}}$$

$$= 5{,}7208$$

Berdasarkan perhitungan di atas, didapatkan jarak maksimum masing-masing *cluster* yakni 0 untuk C dan 2,6463 untuk (ABD) dengan koordinat pusat $(1{,}31; -1{,}01; 1{,}11; -0{,}31; 0{,}101; -0{,}95; -1{,}102; -1{,}112; -0{,}98; -0{,}97; -0{,}095; 1{,}316; 0{,}094; -1{,}309)$ dan $(-0{,}55; 0{,}5;\ -0{,}49; 0{,}73; -0{,}4; 0{,}43; 0{,}85; 0{,}83; 0{,}74; 0{,}81; -0{,}35; -0{,}55; -0{,}47; 0{,}55)$.

Terdapat beberapa keunggulan dari algoritma *K-Means clustering* yakni (Zhang C & Fang Z, 2013: 193):

1) Algoritma *K-Means* merupakan algoritma klasik untuk menyelesaikan masalah pengelompokkan. Algoritma ini relatif sederhana dan cepat.
2) Untuk data yang besar, algoritma ini relatif fleksibel dan efisien.
3) Memberikan hasil yang relatif baik.

Beberapa kekurangan *K-Means clustering* antara lain (Zhang C & Fang Z, 2013: 193):

1) Sensitif terhadap nilai awal, sehingga apabila nilai awal berbeda, mungkin akan terbentuk *cluster* yang berbeda.
2) Algoritma *K-Means clustering* memiliki ketergantungan yang lebih tinggi dari pusat *cluster* awal. Jika pusat *cluster* awal benar-benar jauh dari pusat *cluster* data itu sendiri, jumlah iterasi cenderung tak terbatas dan menghasilkan pengelompokan yang tidak tepat.
3) Algoritma *K-Means clustering* memiliki sensitifitas yang kuat terhadap *noise* objek data. Jika terdapat sejumlah data *noise* pada kumpulan data, ini akan mempengaruhi hasil pengelompokan akhir yang menyebabkan error pada hasil.

G. *Ridge Regression*

Ridge regression dari perfektif bias dan varian mempengaruhi persamaan untuk vektor bobot yang optimal, matriks variansi, dan matriks proyeksi.

1. Bias dan variansi

Ketika *input* x merupaka model terlatih memprediksi *output* sebagai $f(x)$. Jika terdapat kumpulan data *training* dan diketahui *output* $y(x)$ yang benar, maka dapat dihitung *mean squared error* (MSE), yakni

$$M = \langle \big(y(x) - (f(x))\big)^2 \rangle \quad (2.27)$$

Nilai MSE menunjukkan seberapa baik prediksi rata-rata yang dapat dipecah menjadi dua komponen, yakni

$$M = (y(x) - \langle f(x) \rangle)^2 + \langle (f(x) - \langle f(x) \rangle)^2 \rangle \quad (2.28)$$

Bagian pertama merupakan bias dan bagian kedua merupakan variansi.

Jika $y(x) = \langle f(x) \rangle$ untuk semua x maka model ini tidak memiliki bias (bias sama dengan nol). Namun model tersebut mungkin memiliki *mean squared error* yang bernilai besar jika memiliki nilai variansi yang besar. Ini akan terjadi jika $f(x)$ sangat sensitif dengan kekhasan (seperti *noise* dan pilihan titik sampel) dari setiap data *training* tertentu, sensitivitas ini menyebabkan masalah regresi menjadi *ill-posed*. Variansi dapat dikurangi secara signifikan dengan memasukkan sejumlah kecil bias sehingga terjadi pengurangan *mean squared error*.

Masuknya bias setara dengan pembatasan jangkauan pada fungsi dimana model dapat dijelaskan. Hal ini dicapai dengan menghapus derajat kebebasan. Misalnya akan menurunkan urutan polinomial atau mengurangi jumlah bobot dalam *Neural Network*, *ridge regression* menghapus derajat kebebasan tidak secara eksplisit tetapi mengurangi jumlah efektif dari parameter. Hasilnya berupa hilangnya fleksibilitas yang membuat model kurang sensitif.

$$C = \sum_{s=1}^{m}(y_s - f(x_s))^2 + \lambda \sum_{j=1}^{k} w_j^2 \qquad (2.29)$$

Ini merupakan *ridge regression* (*weight decay*) dengan y_m merupakan nilai variabel *output* ke-*s* dengan *m* adalah banyaknya pengamatan dan w_j adalah bobot dari *neuron* lapisan tersembunyi ke-*j*. Parameter regularisasi $\lambda > 0$ mengatur keseimbangan antara penyesuaian data dan pencegahan *penalty*. λ bernilai kecil menunjukkan bahwa data tersebut tepat tanpa menyebabkan *penalty* bernilai besar. Sedangkan λ bernilai besar menunjukkan ketepatan data tidak bisa didapatkan jika membutuhkan bobot besar. Bias merupakan solusi yang melibatkan bobot bernilai kecil dan hasilnya untuk proses fungsi *output* karena bobot yang besar diperlukan untuk menghasilkan fungsi *output* yang sangat bervariasi. (Orr, 1996: 23-24).

Berdasarkan pernyataan-pernyataan sebelumnya, diketahui bahwa masalah yang mungkin muncul ketika bekerja dengan *noise* pada data *training*, *input* dalam jumlah besar, dan kumpulan *training* dalam jumlah kecil. Hal ini disebut *over-fitting*. Dalam mengatasi masalah tersebut, sebuah *roughness penalty*, yaitu ukuran kemulusan kurva dalam memetakan data, dapat ditambahkan pada *Sum Square Error* (SSE). Ini yang disebut *global ridge regression*. Metode *global ridge regression* mengestimasi bobot dengan menambahkan parameter regulasi tunggal yang bernilai positif pada *Sum Square Error* (SSE) untuk mendapatkan vektor bobot yang lebih kuat terhadap *noise* pada kumpulan data *training* (Leondes, 2005: 128).

Estimasi bobot terbaik didapatkan dari hasil akhir dengan SSE terkecil. SSE terkecil atau jumlah kuadrat kesalahan minimal didapatkan dengan metode kuadrat terkecil (*least square*). Penerapannya pada analisis regresi, metode kuadrat terkecil bertujuan untuk memudahkan menyelesaikan masalah optimasi. Model linear yang digunakan adalah $f(x) = \sum_{j=1}^{k} w_j \varphi_j(\boldsymbol{x}) + w_0$, *input* data $\{(x_i)\}_{i=1}^{n}$, dan target klasifikasi variabel *output* $\{(y_s)\}_{s=1}^{m}$,

$$S = \sum_{s=1}^{m} (y_s - f(\boldsymbol{x}_s))^2 \quad (2.30)$$

dengan,

y_s = target klasifikasi variabel *output* data ke-*s*

$f(\boldsymbol{x}_s)$ = nilai variabel *output* data ke-*s*

m = banyak pengamatan

Untuk menentukan nilai optimum bobot w_j, menurunkan fungsi SSE menjadi:

$$\frac{\partial}{\partial w_j} = 2 \sum_{s=1}^{m} (f(\boldsymbol{x}_s) - y_s) \frac{\partial (\boldsymbol{x}_s)}{\partial w_j} \quad (2.31)$$

Berdasrkan persamaan (2.25) didapatkan

$$\frac{\partial (\boldsymbol{x}_s)}{\partial w_j} = \varphi_j(\boldsymbol{x}) \quad (2.32)$$

Selanjutnya persamaan (2.32) disubtitusikan kepersamaan (2.31) dengan hasil sama dengan nol

$$2 \sum_{s=1}^{m} (f(\boldsymbol{x}_s) - y_s)\, \varphi_j(\boldsymbol{x}) = 0 \quad (2.33)$$

$$2 \sum_{s=1}^{m} (f(\boldsymbol{x}_s))\, \varphi_j(\boldsymbol{x}) = 2 \sum_{s=1}^{m} (y_s)\, \varphi_j(\boldsymbol{x}) \quad (2.34)$$

$$\sum_{s=1}^{m} (f(\boldsymbol{x}_s))\, \varphi_j(\boldsymbol{x}) = \sum_{s=1}^{m} (y_s)\, \varphi_j(\boldsymbol{x}) \quad (2.35)$$

Karena $j = 1, 2, 3, \ldots, k$ maka diperoleh *k* persamaan seperti persamaan (2.35) untuk menentukan *k* bobot. Penyelesaian tunggal untuk persamaan (2.35) ditulis dengan notasi vektor menjadi:

$$\boldsymbol{\varphi}_j^T \boldsymbol{f} = \boldsymbol{\varphi}_j^T \boldsymbol{y} \tag{2.36}$$

dengan,

$$\boldsymbol{\varphi_j} = \begin{bmatrix} \varphi_j(\boldsymbol{x}_1) \\ \varphi_j(\boldsymbol{x}_2) \\ \vdots \\ \varphi_j(\boldsymbol{x}_m) \end{bmatrix}, \boldsymbol{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Karena terdapat *k* persamaan untuk setiap nilai *j* maka persamaan (2.36) dapat ditulis sebagai:

$$\begin{bmatrix} \boldsymbol{\varphi}_1^T \boldsymbol{f} \\ \boldsymbol{\varphi}_2^T \boldsymbol{f} \\ \vdots \\ \boldsymbol{\varphi}_k^T \boldsymbol{f} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\varphi}_1^T \boldsymbol{y} \\ \boldsymbol{\varphi}_2^T \boldsymbol{y} \\ \vdots \\ \boldsymbol{\varphi}_k^T \boldsymbol{y} \end{bmatrix}$$

$$\boldsymbol{\varphi}^T \boldsymbol{f} = \boldsymbol{\varphi}^T \boldsymbol{y} \tag{2.37}$$

dengan,

$$\boldsymbol{\varphi} = [\boldsymbol{\varphi}_1 \quad \boldsymbol{\varphi}_2 \quad \cdots \quad \boldsymbol{\varphi}_k]$$

$$\boldsymbol{\varphi} = \begin{bmatrix} \varphi_1(\boldsymbol{x}_1) & \varphi_2(\boldsymbol{x}_1) & \ldots & \varphi_k(\boldsymbol{x}_1) \\ \varphi_1(\boldsymbol{x}_2) & \varphi_2(\boldsymbol{x}_2) & \ldots & \varphi_k(\boldsymbol{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(\boldsymbol{x}_m) & \varphi_2(\boldsymbol{x}_m) & \ldots & \varphi_k(\boldsymbol{x}_m) \end{bmatrix}$$

Matriks $\boldsymbol{\varphi}$ merupakan matriks desain.

Komponen ke-*s* dari $\boldsymbol{f}$ saat bobot pada nilai optimum adalah (Orr, 1996:43):

$$f_s = f(\boldsymbol{x}_s) = \sum_{j=1}^{k} \hat{w}_j \varphi_j(\boldsymbol{x}) = \overline{\boldsymbol{\varphi}}_s^T \hat{\boldsymbol{w}} \tag{2.38}$$

dimana,

$$\bar{\varphi}_s = \begin{bmatrix} \varphi_1(\boldsymbol{x}_s) \\ \varphi_2(\boldsymbol{x}_s) \\ \vdots \\ \varphi_k(\boldsymbol{x}_s) \end{bmatrix}$$

Sehingga diperolah persamaan berikut:

$$\boldsymbol{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{\varphi}}_1^T \hat{\boldsymbol{w}} \\ \bar{\boldsymbol{\varphi}}_2^T \hat{\boldsymbol{w}} \\ \vdots \\ \bar{\boldsymbol{\varphi}}_m^T \hat{\boldsymbol{w}} \end{bmatrix} = \boldsymbol{\varphi} \hat{\boldsymbol{w}} \qquad (2.39)$$

Persamaan (2.39) disubtitusikan kepersamaan (2.37) menjadi:

$$\boldsymbol{\varphi}^T \boldsymbol{\varphi} \hat{\boldsymbol{w}} = \boldsymbol{\varphi}^T \boldsymbol{y} \qquad (2.40)$$

$$\hat{\boldsymbol{w}} = (\boldsymbol{\varphi}^T \boldsymbol{\varphi})^{-1} \boldsymbol{\varphi}^T \boldsymbol{y} \qquad (2.41)$$

$$\hat{\boldsymbol{w}} = \boldsymbol{\Delta}^{-1} \boldsymbol{\varphi}^T \boldsymbol{y} \qquad (2.42)$$

$\hat{w}$ merupakan nilai bobot dan Δ merupakan matrik perkalian antara φ^T dan φ. Selanjutnya ditambah parameter regulasi yang berilai positif pada SSE yakni (Orr, 1996:24):

$$C = \sum_{s=1}^{m} (y_s - f(\boldsymbol{x}_s))^2 + \lambda \sum_{j=1}^{k} w_j^2 \qquad (2.43)$$

dengan,

y_s = target klasifikasi variabel *output* data ke-*s*

$f(\boldsymbol{x}_s)$ = nilai variabel *output* data ke-*s*

m = banyaknya data pengamatan

λ = parameter reguasi

w_j = bobot dari *neuron* lapisan tersembunyi ke-*j* menuju lapisan *output*

Bobot optimum diperoleh dengan mendeferensialkan persamaan (2.43) dengan variabel bebas yang ada, kemudian ditentukan penyelesaianya untuk diferensial sama dengan nol.

$$\frac{\partial}{\partial w_j} = 2\sum_{s=1}^{m}(f(\boldsymbol{x}_s) - y_s)\frac{\partial\ (x_s)}{\partial w_j} + 2\lambda w_j \quad (2.44)$$

$$0 = 2\sum_{s=1}^{m} f(\boldsymbol{x}_s)\frac{\partial\ (x_s)}{\partial w_j} - 2\sum_{s=1}^{m} y_s\frac{\partial\ (x_s)}{\partial w_j} + 2\lambda w_j \quad (2.45)$$

$$0 = \sum_{s=1}^{m} f(\boldsymbol{x}_s)\frac{\partial\ (x_s)}{\partial w_j} - \sum_{s=1}^{m} y_s\frac{\partial\ (x_s)}{\partial w_j} + \lambda w_j \quad (2.46)$$

$$\sum_{s=1}^{m} f(\boldsymbol{x}_s)\frac{\partial\ (x_s)}{\partial w_j} + \lambda w_j = \sum_{s=1}^{m} y_s\frac{\partial\ (x_s)}{\partial w_j} \quad (2.47)$$

Berdasarkan persamaan (2. 32), persamaan (2.47) menjadi:

$$\sum_{s=1}^{m} f(\boldsymbol{x}_s)\varphi_j(\boldsymbol{x}_s) + \lambda w_j = \sum_{s=1}^{m} y_s\varphi_j(\boldsymbol{x}_s) \quad (2.48)$$

Dapat dinotasikan sebagai:

$$\boldsymbol{\varphi}_j^T\boldsymbol{f} + \lambda\hat{w}_j = \boldsymbol{\varphi}_j^T\boldsymbol{y} \quad (2.49)$$

Karena terdapat *k* persamaan dari setiap nilai *j*, maka persamaan (2.49) dapat ditulis sebagai:

$$\begin{bmatrix}\boldsymbol{\varphi}_1^T\boldsymbol{f}\\ \boldsymbol{\varphi}_2^T\boldsymbol{f}\\ \square\\ \boldsymbol{\varphi}_k^T\boldsymbol{f}\end{bmatrix} + \begin{bmatrix}\lambda\hat{w}_1\\ \lambda\hat{w}_2\\ \square\\ \lambda\hat{w}_K\end{bmatrix} = \begin{bmatrix}\boldsymbol{\varphi}_1^T\boldsymbol{y}\\ \boldsymbol{\varphi}_2^T\boldsymbol{y}\\ \square\\ \boldsymbol{\varphi}_k^T\boldsymbol{y}\end{bmatrix}$$

$$\boldsymbol{\varphi}^{\boldsymbol{T}}\boldsymbol{f} + \lambda\hat{w} = \boldsymbol{\varphi}^{\boldsymbol{T}}\boldsymbol{y} \quad (2.50)$$

dengan,

λ = parameter regulasi

$\hat{w}$ = vektor bobot klasifikasi

y = vektor target klasifikasi

φ = matrik desain dengan $\{\varphi_j\}_{j=1}^{K}$ sebagai kolom

f = pekalian matriks desain dan vektor bobot

Berdasarkan definisi-definisi yang telah disebutkan, diperoleh persamaan berikut (Orr, 1996:21):

$$\boldsymbol{\varphi}^T\boldsymbol{y} = \boldsymbol{\varphi}^T\boldsymbol{f} + \lambda\hat{\boldsymbol{w}} \tag{2.51}$$

$$= \boldsymbol{\varphi}^T\boldsymbol{\varphi}\hat{\boldsymbol{w}} + \lambda\hat{\boldsymbol{w}}\boldsymbol{I}_K$$

$$= (\boldsymbol{\varphi}^T\boldsymbol{\varphi} + \lambda\boldsymbol{I}_K)\hat{\boldsymbol{w}}$$

Dimana $\boldsymbol{I}_K$ adalah matriks identitas berukuran $k \times k$. Diperoleh persamaan normal untuk bobot pengklasifikasian berikut:

$$\hat{\boldsymbol{w}} = (\boldsymbol{\varphi}^T\boldsymbol{\varphi} + \lambda\boldsymbol{I}_K)^{-1}\boldsymbol{\varphi}^T\boldsymbol{y} \tag{2.52}$$

Kriteria pemilihan model mencakup estimasi prediksi *error*, yaitu estimasi seberapa baik model pada data *training* akan bekerja pada *input* selanjutnya yang tidak diketahui. Model yang terbaik adalah model dengan estimasi prediksi error yang kecil. Salah satu kriteria tersebut yaitu *Generalised Cross-Validation* (GCV) untuk menghitung prediksi *error*. Rumus GCV adalah sebagai berikut. (Orr, 1996: 20).

$$\hat{\sigma}_G^2 = \frac{m\boldsymbol{y}^T\boldsymbol{P}^2\boldsymbol{y}}{(t \quad (\boldsymbol{P}))^2} \tag{2.53}$$

m = Banyaknya data

$\boldsymbol{P} = \boldsymbol{I}_m - \boldsymbol{\varphi}\ ^{-1}\boldsymbol{\varphi}^T$

$\boldsymbol{A} = \boldsymbol{\varphi}^T\boldsymbol{\varphi} + \lambda\boldsymbol{I}_K$

P = Matriks proyeksi

y = Vektor target klasifikasi

2. Pengoptimalan Parameter Regulasi

Pemilihan model digunakan untuk memilih nilai untuk parameter regularisasi λ. Nilai yang dipilih adalah salah satu yang terkait dengan

estimasi *error* terendah. Karena semua kriteria pemilihan model bergantung secara nonlinier pada λ dibutuhkan metode optimasi nonlinier. Sehingga digunakan salah satu teknik standar untuk ini, seperti ketika turunan dari estimasi *error* GCV disamadengankan nol, persamaan yang dihasilkan dapat dimanipulasi sehingga hanya λ yang muncul di sisi kiri.

$$\hat{\lambda} = \frac{\mathbf{y}^T \mathbf{P}^2 \mathbf{y}\, t_i \quad (\mathbf{A}^{-1} - \lambda \mathbf{A}^{-2})}{\hat{\mathbf{w}}^T \mathbf{A}^{-1} \hat{\mathbf{w}} t_i \quad (\mathbf{P})} \tag{2.54}$$

Persamaan tersebut bukan solusi, namun merupakan rumus estimasi ulang karena sisi kanan bergantung pada $\hat{\lambda}$ (secara eksplisit maupun implisit melalui A^{-1} dan P). Untuk menggunakannya, nilai awal dari $\hat{\lambda}$ dipilih dan digunakan untuk menghitung nilai untuk sisi kanan, ini menyebabkan perkiraan baru dan proses dapat diulang sampai konvergen.

H. Ketepatan Hasil Klasifikasi

Setelah proses pembelajaran selesai dilakukan, tahapan selanjutnya adalah pengujian ketepatan hasil klasifikasi (diagnosa). Sensitivitas, spesifisitas dan akurasi secara luas digunakan untuk menggambarkan hasil klasifikasi. Kemungkinan yang dapat terjadi pada hasil klasifikasi ditunjukkan Tabel 2.5 berikut (Zhu, *et al*, 2010: 1-9).

Tabel 2.5 Hasil Klasifikasi Uji Diagnosa

Hasil Diagnosa	Kondisi penyakit sebagaimana ditetapkan oleh Standar Kebenaran		
	Positif	Negatif	Jumlah Baris

Positif	TP	FP	TP+FP
Negatif	FN	TN	FN+TN
Jumlah Kolom	TP+FN	FP+TN	N=TP+FN+FP+TN

Ada beberapa istilah yang umum digunakan bersama dengan deskripsi sensitivitas, spesifisitas dan akurasi yakni TP = *True Positive*, FP = *False Positive,* TN = *True Negative*, dan FN = *False Negative*. Jika penyakit terbukti ada dalam tubuh pasien, tes diagnostik yang diberikan juga menunjukkan adanya penyakit, hasil tes diagnostik dianggap *True Positive* (TP). Demikian pula, jika penyakit terbukti tidak ada pada tubuh pasien, tes diagnostik menunjukkan penyakit tidak ada juga, sehingga hasil tes *True Negative* (TN). Kedua *True Negative* and *True Positive* menunjukkan hasil yang konsisten antara tes diagnostik dan kondisi terbukti (juga disebut standar kebenaran). Namun, tidak ada tes medis yang sempurna.

Jika tes diagnostik menunjukkan adanya penyakit pada pasien yang sebenarnya tidak memiliki penyakit tersebut, hasil tes *False Positive* (FP). Demikian pula, jika hasil tes diagnosis menunjukkan bahwa penyakit ini tidak ada pada pasien yang sebenarnya terjangkit penyakit, hasil tes *False Negative* (FN). Kedua *False Positive* dan *False Negative* menunjukkan bahwa hasil tes berlawanan dengan kondisi yang sebenarnya. (Zhu, *et al*, 2010: 1-9).

1. Sensitivitas

   Sensitivitas mengacu pada kemampuan tes untuk mengidentifikasi pasien dengan penyakit secara tepat. Rumus Sensitivitas adalah sebagai berikut.

$$S = \frac{T}{T + F} \quad (2.55)$$

Misalnya, jika sensitivitas = 99%, artinya ketika dilakukan tes diagnostik pada pasien dengan penyakit tertentu, pasien ini berpeluang 99% teridentifikasi positif terjangkit penyakit tersebut.

2. Spesifisitas

Spesifisitas mengacu pada kemampuan tes untuk mengidentifikasi pasien tanpa penyakit secara tepat. Rumus Spesifisitas adalah sebagai berikut.

$$S = \frac{T}{T + F} \quad (2.56)$$

Nilai Spesifisitas merupakan peluang tes diagnosa penyakit tertentu tanpa memberikan hasil False Positive. Misalnya, jika spesifisitas suatu tes 99%, ini artinya ketika dilakukan tes diagnosa pada pasien tanpa penyakit tertentu, pasien ini berpeluang 99% teridentifikasi negatif terjangkit penyakit tersebut.

3. Akurasi

Akurasi adalah proporsi dari hasil yang benar (True), baik True Positive maupun True Negative, dalam suatu populasi. Akurasi mengukur atau mengidentifikasi dengan benar kondisi pasien. Rumus untuk menghitung akurasi adalah sebagai berikut.

$$A = \frac{T + T}{T + T + F + F} \quad (2.57)$$

Besar nilai akurasi merepresentasikan tingginya keakuratan hasil diagnosa pada pasien yang melakukan uji diagnosa, baik pasien yang terjangkit penyakit maupun tidak.

I. Penelitian-penelitian Terdahulu

Dewasa ini telah banyak penelitian mengenai kanker otak yang dikaitkan dengan berbagai bidang ilmu. Salah satu bidang ilmu yang sering digunakan dalam penelitian kanker otak adalah bidang ilmu matematika. Salah satu fokus kajian dari penelitian adalah deteksi dan diagnosa kanker otak dengan citra *magnetic resinance images* (MRI) yang diterapkan pada sistem *fuzzy* dan *neural network*. Kathalkar & Chopade (2013) melakukan penelitian yang bertujuan untuk mengklasifikasi kanker otak menggunakan *Artificial Neural Network* (ANN) berdasarkan data MRI otak. Data yang digunakan terdiri dari 38 data. *Input* data berupa hasil estraksi citra MRI menggunakan GLCM (*Gray Level Co-occuence Matrix*) yang terdiri dari ASM, *contrast*, *Energy*, IDM, dan *dissimilarity*. Hasil ekstraksi tersebut merupakan *neuron* pada lapisan *input*. Penelitian ini juga dibuat ke dalam bentuk *Graphical User Interface* (GUI) sebagai salah satu teknik deteksi kanker otak.

Al-Naami, *et al*. (2014) melakukan penelitian yang bertujuan untuk mengklasifikasi kanker otak berdasarkan hasil segmentasi citra MRI. Pada penelitian tersebut, mereka mengkombinasikan beberapa metode dari *artificial intelligent system* yakni *Adaptive neuro-fuzzy inference system* (ANFIS), *Elamn Neural Network* (Elman NN), *Nonlinear AutoRegressive with exogenous neural networks* (NARXNN), dan *feedforward* NN. Hasil klasifikasi yang digunakan adalah *malignant* dan *benign* pada 107 pasien kanker otak. Dari ke-empat AI yang digunakan, disimpulkan bahwa dengan mengunakan NARX NN menghasilkan keakurasian yang baik yakni dengan nilai akurasi sebanyak 99,1%. Nayak & Verma (2014) meneliti tentang klasifikasi kanker otak menggunaka

backpropagation neural network (BPNN) dan *principle component analysis* (PCA). Pada penelitian tersebut, variabel *input* yang digunakan berupa hasil ekstraksi dari segmentasi dan pengolahan citra. Metode yang digunakan untuk mengklasifikasi kanker otak ada 3 metode yakni yang pertama, menggunakan BPNN dengan membuat klasifikasi pada dua bagian yaitu bagian *training* dan bagian *testing;* kedua, mengkombinasikan PCA dan BPNN dimana PCA berguna untuk mengurangi dimensi pada matriks hasil ekstraksi dan BPNN berguna untuk klasifikasi kanker otak; ketiga, PCA dengan mengurangi dimensi dari himpunan data dan membuat klasifikasi.

Pada tahun 2014, Ramaraju & Baji melakukan penelitian untuk mendiagnosa kanker otak berdasarkan data *magnetic resonance images* (MRI) otak menggunakan segmentasi citra dan metode *probabilistic neural network* (PNN). Pada tahun 2015, Pergad & Shingare melakukan penelitian yang serupa yakni untuk mengkasifikasi kanker otak berdasarkan segmentasi citra MRI dan menggunakan metode *Probabilistic neural network* (PNN). Penelitian tersebut juga dibuat dalam bentuk *Graphical User Interface* (GUI).

Maghana & Rekha pada tahun 2015 melakukan penelitian untuk mengklasifikasi kanker otak menggunakan *Artificial Neural Network* (ANN) berdasarkan data segmentasi citra MRI menggunakan *Fuzzy C-Means* (FCM) dan metode *Bounding Box*. Pada penelitian tersebut, variabel *input* berupa hasil ekstraksi dari segmentasi citra MRI, sedangkan variabel *output* berupa hasil klasifikasi yakni normal dan abnormal. Pada tahun 2015, Suhay & Saini melakukan penelitian yang bertujuan untuk mendiagnosa dan menglasifikasi

kanker otak menggunakan *Support Vektor Machine* (SVM) dan Multi-SVM. Pada penelitian tersebut, diawali dengan pengolahan citra, segmentasi citra menggunakan *Fuzzy C-Means clustering* (FCM), ekstraksi citra menggunakan GLCM, dan deteksi kanker serta klasifikasi kanker otak. Multi-SVM digunakan untuk mengkasifikasi jenis-jenis kanker. Secara keseluruhan metode SVM menghasilkan niali akurasi sebesar 91% dari 100 data.

Pada tahun 2016, Rathod & Kapse melakukan penelitian yang bertujuan untuk mendiagnosa kanker otak menggunakan *Artificial Neural Network* (ANN). Pada penelitian tersebut, proses segmentasi citra menggunakan *K-means clustering* dan proses ekstraksi citra menggunakan GLCM. Fitur-fitur hasil ekstrasi citra yakni *contrast, correlation, homogenity, entropy*, *energy, variance*, *maximum probability*, dan *dissimilarity*. Fitur-fitur tersebut digunakan sebagai variabel *input* sedangkan variabel *output* berupa hasil klasifikasi yakni normal dan abnormal. Penelitian ini juga dibuat dalam bentuk GUI pada MATLAB. Roy, *et al*. (2016), melakukan penelitian untuk mengklasifikasi kanker otak berdasarkan data MRI menggunakan *Adaptive Neuro-Fuzzy Inference System* (ANFIS). Penelitian tersebut menggunakan dua klasifikasi yakni *Artificial Neural Network* dengan *Backpropagation Learning Model* dan *Artificial Neural Network* dengan *K-Nearest Neighbors*. Variabel *input* berurupa fitur-fitur hasil ekstraksi cita yakni *energy, correlation, contrast, absolute value*, dan *entropy*. Sedangkan variabel *output* berupa kasifikasi kanker otak yakni glioma, maningioma, metastatic adenocarcinoma, metastatic bronchogenic carcinoma, dan sarcoma. Nilai akurasi tertinggi dihasilkan dengan menggunakan ANFIS yakni 98, 25%.

Beberapa penelitian yang terkait dengan diagnosa kanker otak menggunakan *radial basis function neural netwok* (RBFNN) diantaranya, Devadas & Ganesan (2012) yang meneliti tentang diagnosa kanker otak yang mengunakan data CT otak normal dan otak abnormal. Penelitian tersebut, variabel *input* yang digunakan merupakan hasil dari ekstraksi menggunakan metode analisis statistik tekstur. Shah, *et al*. (2014) meneliti diagnosa kanker otak menggunakan 25 data MRI. Penelitian tersebut, variabel *input* yang digunakan berupa hasil ekstraksi citra menggunakan *principle component analysis* (PCA) dengan menentukan semua *eigen values* dan *eigen vectors*. Pada tahun 2016, Padmapriya, *et al*. melakukan penelitian tentang diagnosa kanker otak berdasarkan sinyal digital *Electroencephalograph* (EEG) yang diekstraksi menggunakan *Principal Component Analysis*. Hasil ekstraksi berupa *mean, variance, co-variance, eigen values,* dan *eigen vectors*. Pada penelitian tersebut, metode yang digunakan untuk menentukan bobot antara lapisan tersembunyi dengan lapisan *output* adalah *standard gradient descent technique* seperti *LMS algorithm*. Vaiabel *output* berupa klasifikasi kanker otak yakni normal dan abnormal.