BAB II

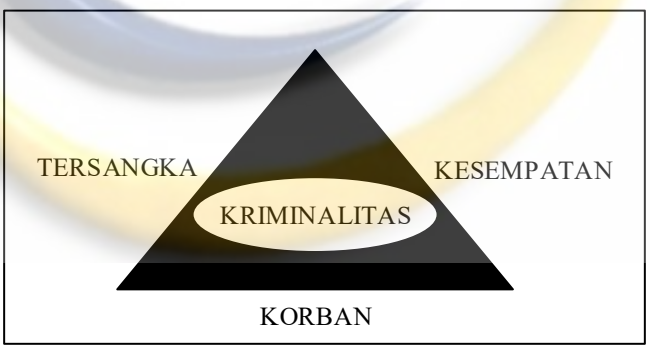
LANDASAN TEORI

2.1 Data Kejadian Kriminalitas

Kriminalitas merupakan segala macam bentuk tindakan dan perbuatan yang merugikan secara ekonomi dan psikologi yang melanggar hukum yang berlaku dalam negara Indonesia serta norma-norma sosial dan agama. Dapat diartikan bahwa, tindak kriminalitas adalah segala sesuatu perbuatan yang melanggar hukum dan melanggar norma-norma sosial, sehingga masyarakat menentangnya[3]. Kriminalitas terdiri dari tiga faktor yang berhubungan, yaitu[9] :

1) Korban yang menjadi sasaran,
2) Tersangka,
3) Kesempatan untuk melakukan.

Seperti pada Gambar 2.1, semua faktor tersebut terjadi secara bersamaan. Misalnya, jika korban dan pelaku ada tapi kesempatan tidak ada, maka kejahatan tidak akan terjadi.



Gambar 2.1 Faktor kejadian kriminalitas

Dalam analisis kejadian kriminalitas, data kriminalitas merupakan jenis data yang paling penting. Data kriminalitas bersifat dinamis dan bisa dilaporkan setiap hari, minggu, bulan, bahkan bertahun-tahun setelah kejadian terjadi. Salah satu data kriminalitas yang penting adalah data yang berkaitan dengan kejadian kriminalitas. Ada dua kategori kejadian kriminalitas, yaitu kriminalitas indeks dan kriminalitas non-indeks[9] Seperti yang ditampilkan pada Tabel 2.1. Menurut definisi kategori

kriminalitas pada Tabel 2.1, kriminalitas indeks lebih memiliki nilai yang berharga apabila dibandingkan dengan kriminalitas non-indeks.

Tabel 2.1 Kategori kejadian kriminalitas[9]

Kategori Kriminalitas	Definisi
Kriminalitas indeks	Kriminalitas yang biasanya dilaporkan dan memiliki pengaruh yang cukup untuk dianggap sepenting indikasi terhadap tingkat kriminalitas.
Kriminalitas non-indeks	Kriminalitas yang tidak dianggap sebagai pengukuran tingkat kejahatan.

Daftar jenis kriminalitas yang diidentifikasi sebagai kejadian kriminalitas indeks berbeda pada setiap negara. Di Indonesia, Satuan Polisi Republik Indonesia (Polri) membagi kriminalitas indeks ke dalam 9 kelompok seperti yang ditampilkan pada Tabel 2.2 berikut

Tabel 2.2 Kelompok kejadian kriminalitas

Kelompok kriminalitas	Jenis kriminalitas
Kejahatan terhadap nyawa	- Pembunuhan
Kejahatan terhadap fisik (badan)	- Penganiayaan berat - Penganiayaan ringan - Kekerasan dalam rumah tangga
Kejahatan terhadap kesusilaan	- Perkosaan - Pencabulan
Kejahatan terhadap kemerdekaan orang	- Penculikan - Mempekerjakan anak dibawah umur
Kejahatan terhadap hak milik/barang dengan penggunaan kekerasan	- Pencurian dengan kekerasan - Pencurian dengan kekerasan menggunakan senjata api - Pencurian dengan kekerasan menggunakan senjata tajam

Kelompok kriminalitas	Jenis kriminalitas
Kejahatan terhadap hak milik/barang	- Pencurian - Pencurian dengan pemberatan - Pencurian kendaraan bermotor - Pengrusakan dengan sengaja - Penadahan
Kejahatan terkait narkoba	- Narkotika dan psikotropika
Kejahatan terkait penipuan, penggelapan dan korupsi	- Penipuan/perbuatan curang - Penggelapan - Korupsi
Kejahatan terhadap ketertiban umum	- Terhadap ketertiban umum

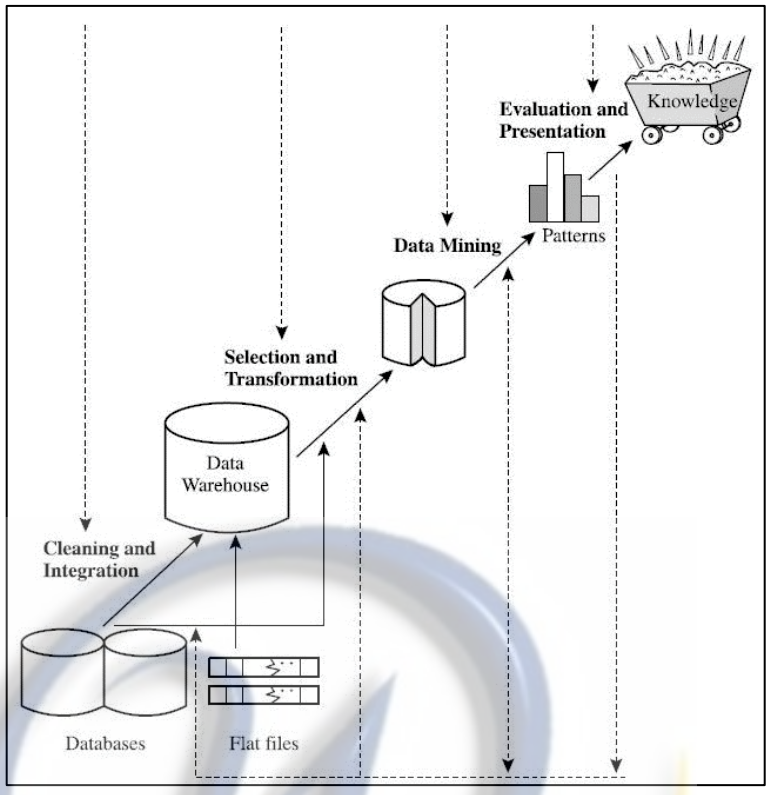
2.2 Data *Mining*

Data *mining* adalah suatu istilah yang digunakan untuk menguraikan penemuan pengetahuan didalam *database*. Data *mining* merupakan suatu proses kegiatan yang meliputi pengumpulan dan pemakaian data historis untuk menemukan keteraturan pola maupun hubungan dalam suatu set data[10]. Data *mining* adalah tentang memecahkan suatu masalah dengan menganalisis data yang sudah ada. Data *mining* juga didefinisikan sebagai proses menemukan pola dalam data, dimana pola yang didapat harus memiliki beberapa keuntungan[14].

Data *mining* sering disebut *knowledge discovery in database* (KDD), yaitu kegiatan yang meliputi kumpulan pemakaian data historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar. Keluaran dari data *mining* ini bisa dipakai untuk memperbaiki pengambilan keputusan di masa depan. Sehingga istilah *pattern recognition* sekarang jarang digunakan karena ia termasuk bagian dari data *mining*[10].

2.2.1 *Knowledge Discovery in Database* (KDD)

Proses KDD secara garis besar terdiri dari urutan yang berulang dari langkah-langkah berikut[2]:



Gambar 2.2 Proses *KDD*[2]

a) Data *Cleaning*

Pembersihan data merupakan proses menghilangkan *noise* dan data yang tidak konsisten atau data tidak relevan. Pada umumnya data yang diperoleh, baik dari *database* suatu perusahaan maupun hasil eksperimen, memiliki isian-isian yang tidak sempurna seperti data yang hilang, data yang tidak valid atau juga hanya sekedar salah ketik. Selain itu, ada juga atribut-atribut data yang tidak relevan dengan hipotesa data *mining* yang dimiliki. Data-data yang tidak relevan itu juga lebih baik tidak digunakan. Pembersihan data juga akan mempengaruhi kinerja dari teknik data *mining*, karena data yang ditangani akan berkurang jumlah dan kompleksitasnya.

b) Data *Integration*

Integrasi data merupakan penggabungan data dari berbagai *database* ke dalam satu *database* baru. Tidak jarang data yang diperlukan untuk data *mining* tidak hanya berasal dari satu *database* tetapi juga berasal dari beberapa *database* atau *file* teks. Integrasi data dilakukan pada atribut-aribut yang mengidentifikasikan entitas-entitas yang unik seperti atribut nama, jenis produk, nomor pelanggan dan lainnya. Integrasi data perlu dilakukan secara cermat karena kesalahan pada

integrasi data bisa menghasilkan hasil yang menyimpang dan bahkan menyesatkan pengambilan aksi nantinya. Sebagai contoh bila integrasi data berdasarkan jenis produk ternyata menggabungkan produk dari kategori yang berbeda maka akan didapatkan korelasi antar produk yang sebenarnya tidak ada.

c) Data *Selection*

Data yang ada pada *database* sering kali tidak semuanya dipakai, oleh karena itu hanya data yang sesuai untuk proses analisis yang akan diambil dari *database*. Sebagai contoh, sebuah kasus yang meneliti faktor kecenderungan orang membeli dalam kasus *market basket analysis,* tidak perlu mengambil nama pelanggan, cukup dengan *id* pelanggan saja.

d) Data *Transformation*

Data diubah atau digabung ke dalam format yang sesuai untuk proses data *mining*. Beberapa metode data *mining* membutuhkan format data yang khusus sebelum diaplikasikan. Sebagai contoh beberapa metode standar seperti analisis asosiasi dan *clustering* hanya bisa menerima *input* data kategorikal. Karenanya data berupa angka numerik yang berlanjut perlu dibagi menjadi beberapa interval. Proses ini sering disebut transformasi data.

e) Data *Mining*

Merupakan suatu proses utama saat metode diterapkan untuk menemukan pengetahuan berharga dan tersembunyi dari data.

f) *Pattern Evaluation*

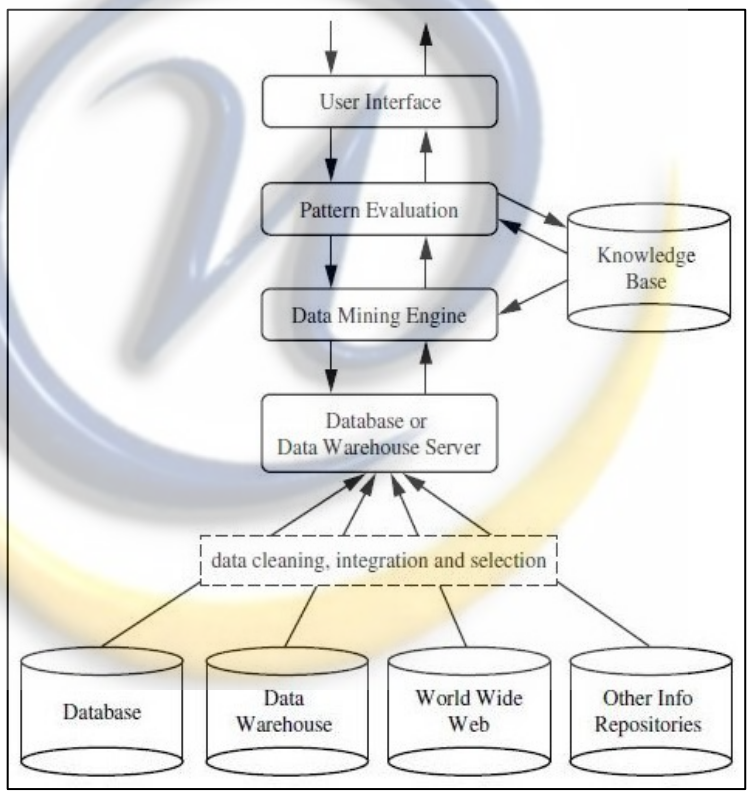
Untuk mengidentifikasi pola-pola menarik ke dalam *knowledge based* yang ditemukan. Dalam tahap ini hasil dari teknik data *mining* berupa pola-pola yang khas maupun model prediksi dievaluasi untuk menilai apakah hipotesa yang ada memang tercapai. Bila ternyata hasil yang diperoleh tidak sesuai hipotesa ada beberapa alternatif yang dapat diambil seperti menjadikannya umpan balik untuk memperbaiki proses data *mining*, mencoba metode data *mining* lain yang lebih sesuai, atau menerima hasil ini sebagai suatu hasil yang diluar dugaan yang mungkin bermanfaat.

g) *Knowledge Presentation*

Merupakan visualisasi dan penyajian pengetahuan mengenai metode yang digunakan untuk memperoleh pengetahuan yang diperoleh pengguna. Tahap

terakhir dari proses data *mining* adalah bagaimana memformulasikan keputusan atau aksi dari hasil analisis yang didapat. Ada kalanya hal ini harus melibatkan orang-orang yang tidak memahami data *mining*. Karenanya presentasi hasil data *mining* dalam bentuk pengetahuan yang bisa dipahami semua orang adalah satu tahapan yang diperlukan dalam proses data *mining*. Dalam presentasi ini, visualisasi juga bisa membantu mengkomunikasikan hasil *data mining*[2].

Langkah 1 sampai 4 merupakan berbagai bentuk *precrocessing*, dimana data disusun untuk proses *mining*. Langkah data *mining* dapat berinteraksi dengan pengguna atau basis pengetahuan. Berdasarkan pandangan ini, berikut komponen utama dari arsitektur sistem data mining pada Gambar 2.3.



Gambar 2.3 Arsitektur sistem data *mining*[2]

Database, Data *Warehouse, World Wide Web, dan Other Info Repositories* merupakan satu atau kumpulan dari *database*, *spreadsheet* atau jenis lain dari informasi *repository*. *Cleaning*, integrasi dan pemilihan data dapat dilakukan pada data tersebut.

Database atau *Data Warehouse Server*: bertanggung jawab untuk mengambil data yang relevan berdasarkan permintaan data *mining* pengguna.

Knowledge Base: merupakan domain pengetahuan yang digunakan untuk memandu pencarian atau mengevaluasi pola yang dihasilkan. Pengetahuan tersebut dapat mecakup hirarki konsep yang digunakan untuk mengatur atribut atau nilai atribut ke dalam berbagai tingkat abstraksi.

Data Mining Engine: penting untuk sistem *data mining* seperti untuk tugas-tugas karakteristik, asosiasi dan analisis korelasi, klasifikasi, prediksi, analisis *cluster*, analisis *outlier*, dan analisis *evolusi*.

Pattern Evaluation: melakukan pemusatan pencarian terhadap pola yang menarik dengan berinteraksi dengan modul *data mining*, atau dapat diintegrasikan dengan modul *mining*, tergantung pada pelaksanaan metode *data mining* yang digunakan.

User Interface: diperlukan sebagai perantara pengguna dengan sistem untuk berkomunikasi yang memungkinkan pengguna untuk berinteraksi dengan sistem, dengan menentukan pemberian tugas, memberi informasi untuk membantu memfokuskan pencarian. Selain itu, memungkinkan pengguna untuk menelusuri *database* dan skema *data warehouse* atau struktur data, mengevaluasi pola, dan memvisualisasikan pola dalam bentuk yang berbeda.

2.2.2 Teknik data mining

Terdapat beberapa teknik *data mining* berdasarkan tugas yang bisa dilakukan, diantaranya adalah[6]:

a) Deskripsi

Peneliti dan analis mencoba mencari cara untuk menggambarkan pola dan kecenderungan yang terdapat dalam data. Sebagai contoh, petugas di tempat pengumpulan suara mungkin tidak dapat menemukan keterangan atau fakta bahwa terdapat pemilih yang tidak cukup profesional akan sedikit berkontribusi dalam pemilihan presiden. Deskripsi dari pola dalam data dapat memberikan kemungkinan penjelasan untuk suatu pola.

b) Estimasi

Estimasi hampir sama dengan klasifikasi, kecuali variabel target estimasi lebih ke arah numerik dari pada ke arah kategori. Model dibangun menggunakan *record* lengkap yang menyediakan nilai dari variabel target sebagai nilai prediksi. Selanjutnya, pada peninjauan berikutnya estimasi nilai dari variabel target dibuat berdasarkan nilai variabel prediksi. Sebagai contoh, akan dilakukan estimasi tekanan darah sistolik pada pasien rumah sakit berdasarkan umur pasien, jenis kelamin, berat badan. Hubungan antara tekanan darah sistolik dan nilai variabel prediksi dalam proses pembelajaran akan menghasilkan model estimasi. Model estimasi yang dihasilkan dapat digunakan untuk kasus baru lainnya.

c) Prediksi

Prediksi memiliki kemiripan dengan estimasi dan klasifikasi. Hanya saja, prediksi memberikan hasil yang menunjukkan sesuatu yang belum terjadi (mungkin terjadi di masa depan).

d) Klasifikasi

Dalam klasifikasi variabel, tujuan bersifat kategorik. Misalnya, kita akan mengklasifikasikan pendapatan dalam tiga kelas, yaitu pendapatan tinggi, pendapatan sedang, dan pendapatan rendah.

e) *Clustering*

Clustering adalah salah satu teknik multivarian yang bertujuan mengelompokkan sejumlah objek ke dalam beberapa kelompok berdasarkan informasi/ karakteristik data. Pengelompokan sejumlah objek dengan tujuan dalam suatu kelompok atau *cluster* memiliki tingkat kesaman yang tinggi, sedangkan anggota antar kelompok memiliki tingkat keragaman yang tinggi. Analisis *cluster* biasa digunakan dalam segmentasi pasar konsumen, memahami perilaku pembeli, mengidentifikasi peluang produk baru, meringkas atau mereduksi data, dan lain-lain. Terdapat beberapa tipe *clustering*, diantaranya hirarki dan non-hirarki untuk mengelompokkan setiap objek tepat dalam suatu kelompok. Pengelompokan objek tersebut tidak mampu menunjukkan variabel dominan dalam *cluster*, tidak demikian dengan *Neural Network-based Clustering* dalam hal ini *Self Organizing*

Map. Metode tersebut mengelompokkan suatu objek dengan menganggap setiap titik sebagai *neuron*[1].

f) Asosiasi

Mengidentifikasi hubungan antara berbagai peristiwa yang terjadi pada satu waktu.

2.3 Sistem *Fuzzy*

Sistem *fuzzy* pertama kali diperkenalkan oleh Prof. L. A. Zadeh dari Barkelay pada tahun 1965. Sistem *fuzzy* merupakan penduga numerik yang terstruktur dan dinamis. Sistem ini mempunyai kemampuan untuk mengembangkan sistem intelijen dalam lingkungan yang tak pasti. Sistem ini menduga suatu fungsi dengan logika *fuzzy*. Dalam logika *fuzzy* terdapat beberapa proses yaitu penentuan himpunan *fuzzy*, penerapan aturan *IF-THEN* dan proses inferensi *fuzzy*[4]. Sistem *Fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang *input* ke dalam ruang *output*.

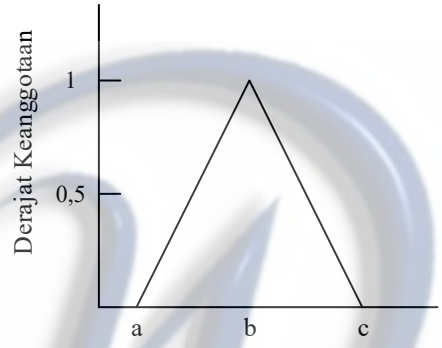
Ada beberapa metode untuk merepresentasikan hasil logika *fuzzy* yaitu metode Tsukamoto, Sugeno dan Mamdani. Pada penelitian ini, metode Mamdani digunakan untuk melakukan fuzzifikasi data dengan tujuan mendapatkan hasil pembelajaran yang maksimal. Pada metode Mamdani, aplikasi fungsi implikasi menggunakan *MIN*, sedang komposisi aturan menggunakan metode *MAX*. Metode Mamdani dikenal juga dengan metode *MAX-MIN*. Sistem *fuzzy* secara umum terdapat 5 langkah dalam melakukan penalaran, yaitu:

1. Memasukkan *input fuzzy*.
2. Mengaplikasikan operator *fuzzy*.
3. Mengaplikasikan metode implikasi.
4. Komposisi semua output.
5. Defuzifikasi.

Pada penelitian ini, dataset yang akan dilakukan pembelajaran oleh *Self Organizing Map* terlebih dahulu dilakukan proses fuzzifikasi. Fuzzifikasi merupakan proses transformasi data menggunakan sistem *fuzzy* untuk menjadikan

setiap variabel bernilai antara 0 sampai 1 dengan tujuan menghilangkan variabel dengan nilai yang dominan[8].

Untuk memetakan *input* data ke dalam nilai antara 0 sampai 1, digunakan fungsi keanggotaan (*membership function*). Fungsi keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik *input* data ke dalam nilai keanggotaannya yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan adalah dengan melalui pendekatan fungsi. Pada penelitian ini, penulis menggunakan representasi kurva segitiga (*triangular*).



Gambar 2.4 Fungsi keanggotaan *triangular*

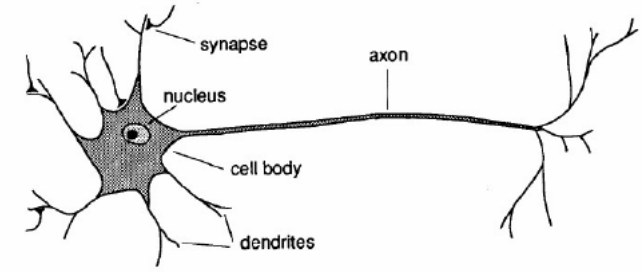
Fungsi keanggotaan *triangular* dideskripsikan oleh 3 parameter a,b,c dan diberi pernyataan sebagai berikut,

$$f(x;a,b,c) = max\left\{min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right\}$$

Didefinisikan a merupakan batas bawah, c adalah batas atas, dan nilai b merupakan nilai tengah, dimana $a < b < c$ seperti pada Gambar 2.9.

2.4 Jaringan Saraf Tiruan

Jaringan saraf tiruan adalah representasi buatan untuk mensimulasikan proses pembelajaran seperti pada kinerja otak manusia. Istilah buatan didasari oleh penerapannya menggunakan program komputer dalam menyelesaikan sejumlah proses perhitungan selama proses pembelajaran[5]. Bentuk sederhana dari *neuron* dapat dilihat pada gambar 2.4.



Gambar 2.5 struktur sederhana neuron

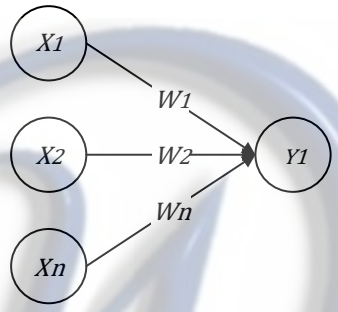
Pada saraf biologis, setiap inti sel saraf (*neuron*) yaitu *nucleus* bertugas melakukan pemrosesan informasi. Berikut adalah komponen utama dari jaringan saraf biologi:

a. *Dendrites*, bertugas menerima informasi,
b. *Cell body*, tempat menerima informasi dari *dendrites* untuk diteruskan pada *axon*
c. *Synapse*, pengirim informasi dari *axon* pada sel saraf lain,
d. *Axon*, mengirim impuls pada sel saraf lain.

Dari beberapa jaringan saraf tiruan, hampir semua memiliki kesamaan pada setiap komponen. Seperti halnya pada otak manusia, jaringan saraf tiruan juga terdiri dari beberapa *neuron* yang saling berhubungan. Pada jaringan saraf tiruan, hubungan ini disebut bobot. *Neuron* jaringan saraf tiruan bekerja dengan cara yang sama dengan jaringan saraf biologi. Informasi (*input*) dikirim pada *neuron* dengan bobot tertentu. Informasi tersebut diproses oleh suatu fungsi perambatan yang akan dijumlahkan dengna nilai semua bobot. Hasil penjumlahan tersebut dibandingkan dengan fungsi aktivasi setiap *neuron*. *Neuron* akan diaktifkan apabila *input* tersebut melewati suatu nilai ambang tertentu, begitu pula sebaliknya. *Neuron* yang diaktifkan akan mengirim *output* melalui bobot-bobot output kepada seluruh *neuron* yang terhubung dengannya, demikian seterusnya. Secara berurutan perbandingan jaringan saraf biologi dengna jaringan saraf tiruan dan ilustrasi jaringan saraf tiruan dapat dilihat pada Tabel 2.3.

Tabel 2.3 Perbandingan jaringan saraf biologi dan jaringan saraf tiruan

Jaringan saraf biologi	Jaringan saraf tiruan
Dendrites	*Input*
Cell body	*Node* atau *input*
Synapse	Bobot
axon	*Output*



Gambar 2.6 Sel saraf tiruan

Dimana: *neuron* input: $X_1, X_2,...,X_n$; bobot: $W_1,W_2,...,W_n$; *neuron output*: Y_1.

Proses perubahan bobot dibagi menjadi 2, yaitu pembelajaran terawasi (*supervised learning*) dan pembelajaran tidak terawasi (*unsupervised learning*). Pembelajaran terawasi merupakan suatu jaringan dengan *output* yang memiliki pola yang sama dengan pola *input*, sedangkan pembelajaran tak terawasi ialah suatu jaringan dengan *output* tidak ditentukan hasilnya seperti apakah yang diharapkan selama proses pembelajaran. Selama proses pembelajaran tak terawasi, nilai bobot disusun dalam suatu *range* tertentu tergantung pada nilai *input* yang diberikan. Tujuan pembelajaran tak terawasi ialah mengelompokkan unit-unit (*node*) hampir sama dalam suatu area tertentu. Pembelajaran ini biasa digunakan dalam *clustering*[11].

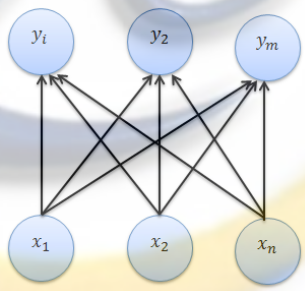
2.5 *Self Organizing Map*

Self Organizing Map adalah metode jaringan saraf tiruan yang diperkenalkan pertama kali oleh Teuvo Kohonen pada tahun 1981, sehingga sering

disebut dengan jaringan Kohonen. Dinamakan "*Self Organizing*" karena tidak memerlukan pengawasan atau tak terawasi (*unsupervised learning*) dan disebut "*Map*" karena *Self Organizing Map* berusaha memetakan bobotnya agar sesuai dengan *input* data yang diberikan.

Self Organizing Map merupakan algoritma yang melakukan pemetaan dari data yang ada di ruang vektor berdimensi tinggi ke ruang vektor dua dimensi yang terletak pada lokasi yang berdekatan. *Self Organizing Map* terdiri dari dua lapisan (*layer*), yaitu lapisan *input* dan lapisan *output*. Setiap *neuron* dalam lapisan *input* terhubung dengan setiap *neuron* pada lapisan *output*. Setiap *neuron* pada lapisan *output* merepresentasikan kelas (*cluster*) dari *input* yang telah diberikan.

Neuron pada jaringan ini menyusun dirinya sendiri berdasarkan nilai *input* tertentu dalam suatu kelompok, biasa disebut *cluster*. Selama proses penyusunan diri, *cluster* dengan vektor bobot paling cocok dengan pola bobot (jarak paling dekat) akan terpilih sebagai pemenang. *Neuron* pemenang beserta *neuron* tetangga akan memperbaiki bobotnya masing-masing[5]. Ilustrasi *Self Organizing Map* dapat dilihat pada Gambar 2.7.

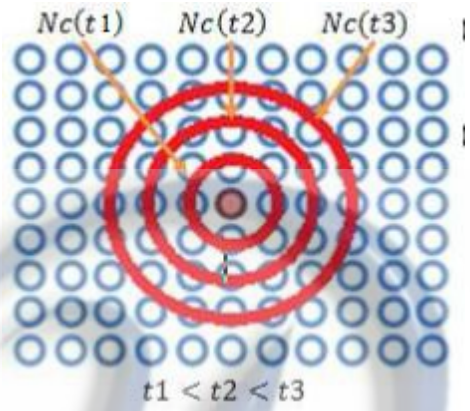


Gambar 2.7 Arsitektur *Self Organazing Map*

Dengan x_i adalah vektor input dengan *n* dimensi dan y_i adalah vektor output dengan *m* dimensi.

Pada algoritma pembelajaran *Self Organizing Map*, akumulasi sinyal yang didapat tidak perlu diaktivasi, karena fungsi aktivasi tidak memberikan pengaruh pada pemilihan *winner neuron* yang akan memperbarui bobotnya dan bobot tetangganya. Dengan demikian, pada proses pelatihan error tidak dihitung pada setiap iterasi pelatihan. Kriteria berhentinya proses pelatihan dalam *Self Organizing Map* akan menggunakan jumlah iterasi tertentu[7].

Pada Gambar 3.2, jika *neuron* yang di tengah adalah *winner neuron* untuk suatu *input* vektor, maka *neighbour neuron* untuk *winner neuron* ini adalah mereka yang terletak di dalam lingkaran area, yang didefinisikan dengan *Nc*(*t*1), *Nc*(*t*2), …dst. *Nc*(*t*1) adalah batas area pada iterasi ke-1, *Nc*(*t*2) adalah batas area pada iterasi ke-2, dan seterusnya. *Neuron* yang secara topografi terletak jauh dari *winner neuron* tidak di*update*.



Gambar 2.8 Ilustrasi *Self Organizing Map*

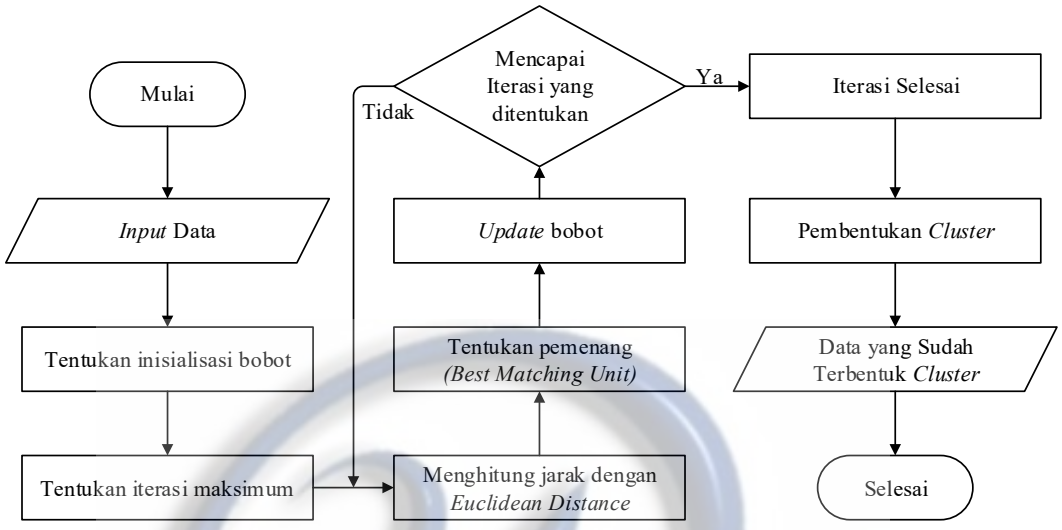
Self Organizing Map memperlihatkan tiga karakteristik:

a. Kompetitif, setiap vektor bobot berlomba menjadi simpul pemenang.
b. Kooperasi, setiap simpul pemenang bekerjasama dengan simpul tetangga, dan
c. Adaptasi, perubahan simpul pemenang dengan simpul tetangga.

Pada penelitian ini, diterapkan logika *fuzzy* terhadap pembentukan dataset yang bisa dikatakan merupakan modifikasi dari *Self Organizing Map*. Logika *fuzzy* memetakan satu set vektor *input* berdimensi tinggi ke dalam grid 2 dimensi. Dalam modifikasi *Self Organizing Map* menggunakan *logika fuzzy*, aturan digunakan untuk mengganti *neuron*. Setiap aturan *fuzzy* mencakup area tertentu dari ruang *input*, yang didefinisikan oleh rangkaian *fuzzy* itu sendiri. *Output* dari aturan *fuzzy* digabungkan secara bersamaan oleh bobot rata-rata, dimana *fire strength* dari aturan *fuzzy* menjadi bobot. Oleh karena itu, struktur *Self Organizing Map* akan sangat mirip dengan *fuzzy logic controller*[9].

2.5.1 Algoritma *Self Organizing Map*

Proses *Self Organizing Map* dilakukan dengan mengikuti algoritma sebagai berikut[9]:



Gambar 2.9 *Flowchart* algortima *Self Organizing Map*

Berdasarkan *flowchart* diatas, prosedur untuk pembelajaran *Self Organizing Map* dapat dideskripsikan sebagai berikut :

Langkah 1	Seluruh data frekuensi kejadian kriminalitas dijadikan data input. Data input tersebut, sebelumnya dilakukan fuzzifikasi menggunakan aturan fuzzy. Data input yang digunakan adalah data yang berbentuk matrik $i \times j$, dimana i adalah jumlah polsek dan j adalah jumlah variabel. Selanjutnya, dilakukan proses *clustering* menggunakan metode *Self Organizing Map*. Inisialisasi bobot *fuzzy* dalam nilai antara 0 sampai 1 secara acak dan tentukan jarak kedekatan radius R dan *learning rate* η
Langkah 2	Pada perhitungan menggunakan metode Self Organizing Map, diawali dengan inisialisasi bobot secara *random* (acak).
Langkah 3	Menentapkan nilai parameter *epoch* (iterasi) maksimum.

Langkah 4	Untuk setiap data dilakukan perhitungan terhadap bobot menggunakan rumus Euclidean Distance. Kemudian dipilih nilai terkecil. $D(j) = \sum_{i=0}^{i=n} (I_i - W_i)^2$ Keterangan : I = vektor input W = bobot vektor n = jumlah bobot
Langkah 5	Data yang memiliki nilai terkecil dari langkah 4 digunakan untuk proses update bobot. Dalam menentukan bobot terbaru pada waktu t, maka diasumsikan objek saat ini $x(i)$ dan *centroid* yang terbentuk w_j. Kemudian untuk menentukan *centroid* yang baru untuk waktu berikutnya *t*+1. $w_{ij}(baru) = w_{ij}(lama) + \alpha[x_i - w_{ij}(lama)]$ α adalah *learning rate*. Pada langkah selanjutnya nilai *learning rate* yang digunakan adalah $\alpha(baru) = a \times b$ dimana nilai b berada diantara 0 dengan 1. Pada akhir iterasi, nilai α akan menuju nilai *laerning rate* minimum.
Langkah 6	Melakukan pengecekan syarat berhenti, iterasi akan berhenti apabila *threshold* terpenuhi, untuk mencapai nilai *threshold* terpenuhi. Adapun nilai *threshold* dikatakan terpenuhi apabila nilai parameter telah terpenuhi.
Langkah 7	Selanjutnya dilakukan proses pengelompokkan atau *clusterisasi*, pada penelitian ini menggunakan metode hirarki.
Langkah 8	Hasil akhir dari proses ini yaitu data ter-*cluster*.

2.5.2 Membangun Jaringan Pada Metode *Self Organizing Map*

Dalam proses membangun suatu jaringan, hal pertama yang harus dilakukan adalah dengan menentukan *input* jaringan. Untuk memudahkan visualisasi, diberikan koordinat (*a*,*b*), penentuan nilai *a* dan *b* akan mempengaruhi hasil *output*. Hal ini disebabkan karena jumlah *cluster* pada akhir proses pembelajaran algoritma *Self Organizing Map* akan sama banyak dengan hasil perkalian *a* dan *b*. Sebagai contoh jika ditentukan nilai *a* adalah 2 dan nilai *b* adalah 3 maka jaringan akan menghasilkan *output neuron* sebanyak 6 *neuron*. Penentuan nilai *a* dan *b* juga akan mempengaruhi topologi *neuron output*. Sebagai contoh, pada model dengan hasil 6 *neuron*, maka terdapat 2 kemungkinan bentuk jaringan yang dapat dibangun, yaitu jaringan dengan nilai *a* adalah 1 dan *b* adalah 6, dan jaringan dengan nilai *a* adalah 2 dan *b* adalah 3. Kedua jaringan ini pada hasil *output* akan mengelompokkan objek ke dalam 6 *neuron*, meskipun demikian topologi yang dihasilkan oleh kedua model jaringan ini berbeda. karena topologi yang dihasilkan oleh kedua model jaringan ini berbeda maka hasil yang diperoleh dapat berbeda, meskipun tidak ada jaminan bahwa jaringan dengan penghubung tunggal antar *neuron* lebih baik atau lebih buruk bila dibandingkan dengan jaringan dengan penghubung jamak antar *neuron*.

2.6 *Fuzzy Logic Toolbox*

Fuzzy logic toolbox merupakan fungsi yang disediakan oleh *Matlab* untuk menganalisa, merancang dan mensimulasikan sistem berdasarkan logika *fuzzy*. *Fuzzy logic toolbox* memberikan fasilitas untuk merancang *Fuzzy Inference System* (FIS). Dengan menggunakan editor dan viewer pada *fuzzy logic toolbox*, dapat dibuat *rule*, menentukan fungsi keanggotaan dan menganalisis *rule* yang telah dibuat. Berikut adalah fitur yang disediakan :

- *FIS Editor* – Berfungsi untuk menampilkan informasi umum mengenai FIS.
- *Membership function editor* – Berfungsi untuk menampilkan dan merubah fungsi keanggotaan yang terkait dengan variabel input dan output FIS.
- *Rule Editor* – Berfungsi untuk melihat dan merubah aturan *fuzzy* yang ditetapkan .

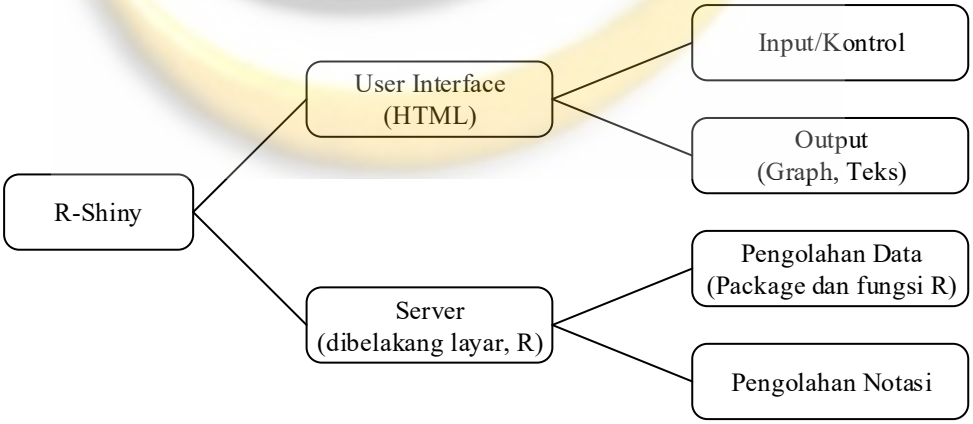
- *Rule Viewer* – Berfungsi untuk melihat *rule* FIS yang terperinci untuk membantu memeriksa *rule* tertentu atau mempelajari pengaruh perubahan variabel *input*.

2.7 *R-Shiny*

R-Shiny merupakan *framework* bahasa pemrograman R berbasis *web*, komponen program Shiny dibedakan menjadi dua kelompok, yaitu[13]:

1. *User interface*. Bagian ini berfungsi untuk:
   a. Panel kontrol, untuk mengendalikan *input* berupa data, variabel dan model. Tampilan kontrol dapat berupa *slider*, *radio button*, *check box*, dan lain-lain.
   b. Nilai *input* data dengan berbagai jenis variabel yang diperlukan.
   c. Penyajian *output* terkait hasil analisis atau pengujian.
2. *Server*. Bagian ini merupakan inti dari program yang bertugas melakukan simulasi, berbagai analisis data sesuai pilihan penggunaan dan selanjutnya mengirim hasilnya ke bagian *output*. *Server* didukung oleh berbagai prosedur dan analisis data yang pada umumnya telah tersedia pada berbagai paket R. Bagian ini disimpan daam file *server.r*.

Struktur umum komponen pemrograman dengan R-Shiny dapat dilihat pada Gambar 2.9.



Gambar 2.10 Struktur komponen *R-Shiny*