BAB II
DASAR TEORI

Pada bab ini akan dijelaskan mengenai definisi *data mining* beserta teknik-teknik dalam data mining yang dipakai di dalam thesis ini.

2.1 Data mining

Seiring dengan berjalannya waktu, pertambahan data pada *database* menjadikan volume data yang disimpan serta kompleksitas data dalam *database* tersebut menjadi sangat besar. Organisasi bisnis membutuhkan lebih dari hanya sekedar menyimpan data saja. Mereka berharap keberadaan data bervolume besar tersebut bisa memberikan keuntungan dari sisi bisnis. Pendekatan tradisional seperti *query* sederhana tidak banyak membantu dalam mengindentifikasi *trend* serta hubungan antar data yang ada di dalam *database*.

Volume data yang besar ini merupakan sebuah tambang emas informasi yang sangat berharga jika bisa digali. Seiring dengan berkembangnya teknologi, maka proses penggalian informasi dari lautan data yang sangat luas dan besar tersebut menjadi sesuatu hal yang sangat mungkin diwujudkan. Salah satu teknologi yang memungkinkan hal tersebut diwujudkan adalah *data mining*.

Menurut Kurt Thearling, [1] *data mining* merupakan sebuah ekstraksi otomatis atas informasi prediktif yang berasal dari *database* yang besar. Sedangkan menurut D. Hand, H. Mannila, P. Smyth[2], *data mining* adalah sebuah ilmu untuk mengekstraksi informasi yang yang berguna dari sebuah *database* berukuran besar.

Selain itu *Data mining* yang dikenal juga sebagai penemuan pengetahuan dalam *database* (*Knowledge Discovery in Databases)* [15], didefinisikan sebagai ekstraksi secara implisit informasi yang berpotensi bermanfaat, dan sebelumnya tidak diketahui dari data [16].

Banyak kalangan saat ini menggunakan komputer serta perangkat lunak untuk melakukan *data mining.* Walaupun perangkat lunak *data mining* bukanlah satu-satunya *tool* yang bisa digunakan untuk melakukan analisis data, namun *tool* ini menghadirkan sebuah kesempatan bagi penggunanya untuk menganalisa data dari berbagai dimensi sudut pandang, mengkategorikan data tersebut, dan menyimpulkan relasi antar data yang teridentifikasi, hingga menghasilkan sebuah pengetahuan.

Data mining sendiri merupakan sebuah terminologi yang baru, akan tetapi teknologi yang ada didalamnya bukanlah teknologi yang benar-benar baru. Beberapa, bahkan sebagian besar alat analisis yang terdapat dalam *data mining* merupakan turunan dari ilmu statistik. Namun dengan semakin berkembangnya teknologi komputasi dari komputer, dan kapasitas media yang semakin besar, menjadikan teknologi-teknologi tersebut bisa diterapkan sebagai sebuah perangkat lunak dan sistem yang memiliki tingkat akurasi dan daya analisa yang cukup tinggi, dan memiliki biaya lebih murah.

2.2 Teknik-teknik Data mining

Data mining telah berkembang dengan sangat luas. Algoritma-algoritma baru serta teknik-teknik baru muncul untuk membantu melakukan analisa data. Teknik-teknik yang ada pada *data mining* terdiri atas beberapa algoritma yang memiliki keuntungan dan kelemahan tersendiri. Pada sub-bab ini akan dibahas

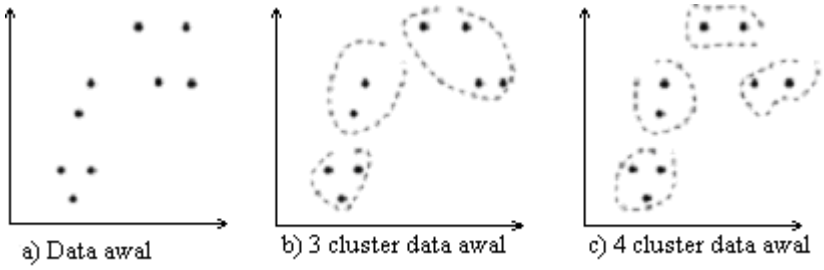
teknik-teknik yang akan dipakai sebagai alat bantu untuk menganalisa problem dari tesis yang akan dibuat nanti.

2.2.1 Clustering[5]

Menurut Kurt Thearling [1]. *Clustering* adalah sebuah teknik *data mining* yang membagi sebuah *dataset* ke dalam kelompok-kelompok secara eksklusif yang memiliki ketergantungan satu dengan yang lain (*mutually exclusive*), sehingga anggota dari tiap kelompok memiliki kedekatan yang sedekat mungkin dengan anggota sesama kelompok. Pada saat yang sama teknik ini juga menciptakan jarak sejauh mungkin antara kelompok-kelompok yang terbentuk.

Dunham[3], menyatakan bahwa *clustering* adalah pengelompokan data yang dilakukan melalui pencarian kemiripan antara data yang satu dengan data yang lain berdasarkan karakteristik yang ditemukan pada data yang ada. Elemen-elemen dari kelompok yang satu berbeda atau tidak serupa dengan elemen dari kelompok yang lain. Kelompok-kelompok ini kemudian dikenal sebagai *cluster*(kluster).

Mehmed Kantardzic[5] menyatakan, dalam penggunaan *clustering* sering ditemukan masalah-masalah yang sangat sulit untuk dipecahkan, hal ini dikarenakan data input bisa menggambarkan kluster dengan bentuk dan ukuran yang berbeda-beda, selain itu pemetaan yang dihasilkan berada dalam sebentuk ruang data berdimensi-n (*n-dimentional data space*). Untuk mengatasi hal ini sering digunakan pendekatan dengan mengubah resolusi dari hasil pemetaan. Gambar 1 bisa menjelaskan permasalahan ini dengan lebih jelas.



Gambar 1. Masalah yang dihadapi dalam penggunaan clustering[5]

Saat ini telah tersedia banyak ragam algoritma *clustering* yang bisa digunakan untuk memecahkan permasalahan *data mining* yang ada. Pemilihan algoritma yang akan digunakan didasarkan pada pemahaman masalah dan tipe data yang digunakan. Secara garis besar, algoritma-algoritma *data mining* yang ada digolongkan kedalam dua pendekatan populer, yaitu pendekatan *Hierarchical clustering* dan pendekatan *Iterative square-error partitional clustering*.

2.2.1.1 Mengukur kemiripan

Dasar dari teknik *clustering* adalah kemampuan untuk membedakan kemiripan. Untuk dapat menilai dua buah *pattern* adalah mirip maka dibutuhkan sebuah metode pengukuran (*measure*) tertentu. Metode pengukuran ini harus dipilih secara hati-hati sebab qulitas hasil *clustering* sangat tergantung pada metode pengukuran yang dipilih. Ketidakmiripan (*dissimilarity*) antara dua contoh (sampel) bisa dihitung berdasarkan berdasarkan pengukuran jarak. Pengukuran jarak bisa berbentuk sebuah *metric* atau sebuah *quasi-metric* yang berada diatas ruang contoh, *metrik* tersebut juga bisa digunakan untuk mengkuantifikasi ketidakmiripan antar contoh.

Kata ”mirip” dalam *clustering* memiliki arti: jika nilai dari s(x,x’) adalah besar ketika x dan x’ adalah dua contoh yang sama. Jika nilai s(x,x’) adalah kecil

maka kedua contoh x dan x' adalah tidak mirip. Selain itu ukuran kemiripan s yang dipakai adalah simetris dengan persamaan matematis berikut:

$$s(x, x') = s(x', x), \qquad \forall \mathrm{x}, \mathrm{x}' \in \mathrm{X}$$

Dimana X adalah sebuah ruang sample yang berisi vektor-vektor data tunggal *x*.

Salah satu rumus pengukuran jarak metrik (*metric distance*) adalah jarak Eucledian (*Eucledian distance*) dalam sebuah ruang berdimensi-*m*, yaitu:

$$d_2(x_i, x_j) = \left(\sum_{k=1}^{m} (x_{ik} - x_{jk})^2 \right)^{1/2}$$

Model *eucledian distance* berdimensi m tersebut tidak hanya bisa dipakai untuk menghitung jarak antar dua sampel, namun juga bisa dipakai untuk menghitung nilai kemiripan antara dua sampel. Rumus yang dipakai untuk menghitung kemiripan tersebut dikenal sebagai *cosinus-correlation,* yaitu:

$$S_{\cos}(x_i, x_j) = \left[\sum_{k=1}^{m} (x_{ik} . x_{jk}) \right] / [\sum_{\mathrm{k}=1}^{m} \mathrm{x}_{\mathrm{ik}}^2 . \sum_{k=1}^{m} x_{jk}^2 \Bigg]^{1/2}$$

Sehingga bisa dengan mudah dilihat,

$$s_{\cos}(x_i, x_j) = 1 \Leftrightarrow \forall i, j \text{ dan } \lambda > 0 \text{ dimana } \mathrm{x}_{\mathrm{i}} = \lambda . x_j$$
$$s_{\cos}(x_i, x_j) = -1 \Leftrightarrow \forall i, j \text{ dan } \lambda < 0 \text{ dimana } \mathrm{x}_{\mathrm{i}} = \lambda . x_j$$

2.2.1.2 Hierarchical clustering

Dalam analisa kluster hirarkis tidak didefinisikan secara jelas jumlah kluster-kluster yang menjadi bagian dari input. Input terhadap sistem dinamakan (X,s),

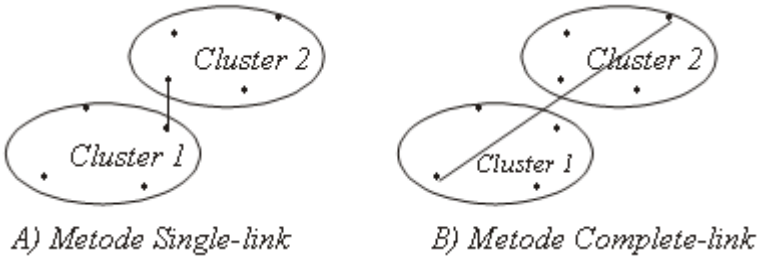
dimana X adala sekumpulan contoh (sampel) dan *s* adalah ukuran kemiripan. Output dari sistem ini adalah kluster-kluster yang tersusun secara hirarkis. Sebagian besar algoritma yang masuk dalam kategori pendekatan ini tidak berdasarkan pada konsep optimasi. Tujuan dari sistem *clustering hirarkis* ini adalah untuk mencari beberapa pendekatan, solusi sub-optimal dengan mengunakan iterasi untuk perbaikan pada partisi sampai sebuah konvergensi didapat. Algoritma-algoritma yang ada dalam pendekatan ini dibagi menjadi dua kategori, yaitu algoritma *divisible* dan algoritma *agglomerative*.

Sebuah algoritma *divisible* memulai proses dengan membagi sekaligus seluruh kumpulan data kedalam sekumpulan partisi subset, kemudian membagi subset-subset tersebut ke dalam sub-subset yang lebih kecil demikian seterusnya. Lalu algoritma ini mengurutkan partisi-partisi yang dihasilkan dengan urutan dari yang terkasar sampai yang terhalus.

Algoritma *agglomerative* memulai proses dengan menganggap seluruh data sebagai sekumpulan kluster-kluster awal. Lalu kluster-kluster tersebut digabungkan dalam sebuah kelompok yang lebih besar, proses ini terus berulang sampai didapatkan kelompok yang diinginkan. Sebagian besar algoritma *clustering* untuk aplikasi dunia nyata menggunakan pendekatan algoritma *agglomerative*.

Kebanyakan algoritma *clustering agglomerative* hirarkis adalah varian dari algoritma *single-link* atau algoritma *complete-link*. Perbedaan kedua algoritma ini terletak pada cara mereka mengkarakterisasikan kemiripan antara sepasang kluster. Pada metode *single-link,* jarak antara dua kluster adalah jarak terkecil yang mungkin timbul antara seluruh pasangan sampel yang diambil dari kedua kluster (satu elemen dari kluster pertama, elemen yang lain diambil dari kluster kedua).

Sedangkan pada metode *complete-link,* jarak yang diambil adalah jarak maksimum. Gambar dibawah ini menjelaskan perbedaan antara kedua metode tersebut.



Gambar 2. Perbedaan metode single-link dengan metode Complete-link[5]

2.2.1.3 Partitional Clustering

Teknik *clustering* partisional menghasilkan kluster-kluster dengan cara mengoptimalkan sebuah fungsi untuk mengkategorisasi data baik secara lokal (dalam sebuah subset dari sampel) maupun global (dilihat dari sudut pandang keseluruhan sampel). Sebuah kriteria global seperti penilaian *Eucledian square-error*, menggambarkan tiap kluster dengan sebuah prototype atau centroid, dan menentukan penempatan sampel-sampel yang ada berdasarkan kemiripan antara sampel tersebut dengan prototipe yang ada. Sebuah *criterion* lokal, seperti MND(*Mutual Neighbor Distance*) membentuk kluster-kluster melalui penggunaan struktur local atau konteks yang terdapat di dalam data. Cara *criterion* membentuk kluster didasarkan pada identifikasi area-area dengan kepadatan yang tinggi di dalam ruang data.

Strategi yang paling sering digunakan untuk algoritma *clustering* partisional adalah dengan memakai *square-error cluster*. Cara strategi ini untuk membentuk partisi adalah meminimalisasi total *square-error* dari sejumlah kluster yang tetap. Misalkan, sejumlah sampel dalam sebuah ruang berdimensi n (*n-dimensional*)

adalah N, akan dipartisi kedalam K kluster $\{C_1, C_2, C_3,\ldots,C_k\}$. maka tiap C_K akan memiliki n_K sampel dan tiap sampel berada tepat pada satu kluster sehingga,

$$\sum_{k=1}^{K} n_k = N$$

dimana k = 1,…,K. Maka *centroid* dari kluster C_k adalah vector tengah (*mean vector*) dari kluster tersebut, atau secara matematis

$$M_k = \left(1/n_k\right)\sum_{i=1}^{n_k} X_{ik}$$

Dimana X_{ik} adalah sampel ke-i milik kluster C_k. *Square-error* untuk kluster C_k adalah jumlah kuadrat *Eucledian distances* antara tiap sampel di dalam C_k dan *centroid* miliknya. Error ini disebut juga variasi antar kluster (*within-cluster variation*), yang bisa dirumuskan secara matematis sebagai berikut:

$$e_k^2 = \sum_{i=1}^{n_k} \left(x_{ik} - M_k\right)^2$$

Kuadrat error dari keseluruhan ruang yang berisi K kluster adalah jumlah dari variasi antar kluster, yaitu

$$E_k^2 = \sum_{k=1}^{K} e_k^2$$

Algoritma K-means adalah implementasi dari algoritma *clustering* partisional yang paling sederhana dan paling banyak digunakan. K-means juga menggunakan kuadrat *error criterion*. Algoritma ini mulai mempartisi ruang data secara acak sambil penunjukan (*assignment*) sampel yang ada ke dalam kluster-

kluster berdasarkan kemiripan antara kluster dan sampel, sampai sebuah *criterion* yang convergen ditemukan. Syarat sebuah *criterion* telah ditemukan adalah ketika tidak ada lagi pemindahan sampel (*reassignment*) dari satu kluster ke kluster yang lain yang akan menyebabkan berkurangnya total error yang dikuadratkan (*error square*). Algoritma ini populer digunakan karena kemudahan implementasinya, dan memiliki kecepatan yang cukup baik.

Langkah-langkah dasar yang diambil oleh algoritma K-means adalah:

1. Pilih partisi awal dengan K kluster yang berisi beberapa sampel yang dipilih secara random, kemudian hitung centroid dari kluster
2. Buat sebuah partisi baru dengan melakukan *assigning* dari tiap sampel ke pusat kluster yang terdekat
3. Hitung pusat kluster baru sebagai *centroid* dari kluster
4. Ulangi langkah 2 dan 3 sampai nilai optimum fungsi *criterion* ditemukan atau sampai keanggotaan kluster sudah stabil (tidak ada lagi perpindahan antar kluster dari sampel).

2.2.1.4 Sequence clustering

Sequence clustering adalah teknik *data mining* yang menggabungkan teknik clustering dengan teknik *sequence analysis*. Teknik berupaya untuk mengelompokkan *sequence* yang ada berdasarkan kemiripan yang dimiliki diantara *sequence* tersebut tersebut. Data S*equence* bisa berbentuk rangkaian kondisi dalam bentuk diskrit seperti rangkaian rantai DNA yang tersusun dari empat buah kondisi diskrit A(*Adenosine*), G(*Guanin*), T(*Thymine*), C(*Cytosine*).

Salah satu algoritma *sequence clustering* yang ada adalah algoritma *Microsoft Sequence Clustering*. Algoritma ini menggabungkan *Markov chain model* dengan teknik *EM clustering*.

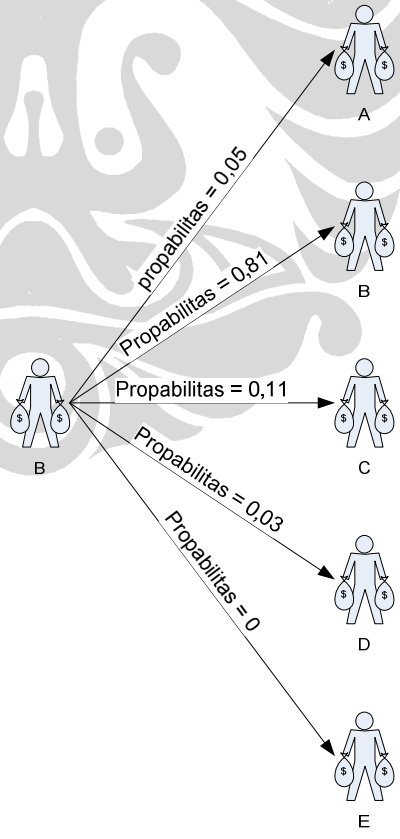
2.2.1.5 Markov Chain Model (Model Rantai Markov)

Rantai Markov adalah sebuah model matematis yang menggambarkan pola perubahan perilaku berdasarkan waktu. Model ini bisa memiliki banyak bentuk dari bentuk yang sederhana dan mudah dimengerti sampai ke bentuk yang sangat kompleks.

Rantai Markov berisi sekumpulan keadaan (*state*) yang beruntai dalam interval-interval satuan waktu yang terpisah (*discrete*) seperti jam, hari, minggu, bulan atau tahun. Transisi dari satu state ke state yang lain terdistribusi dalam sebuah matrik transisi.

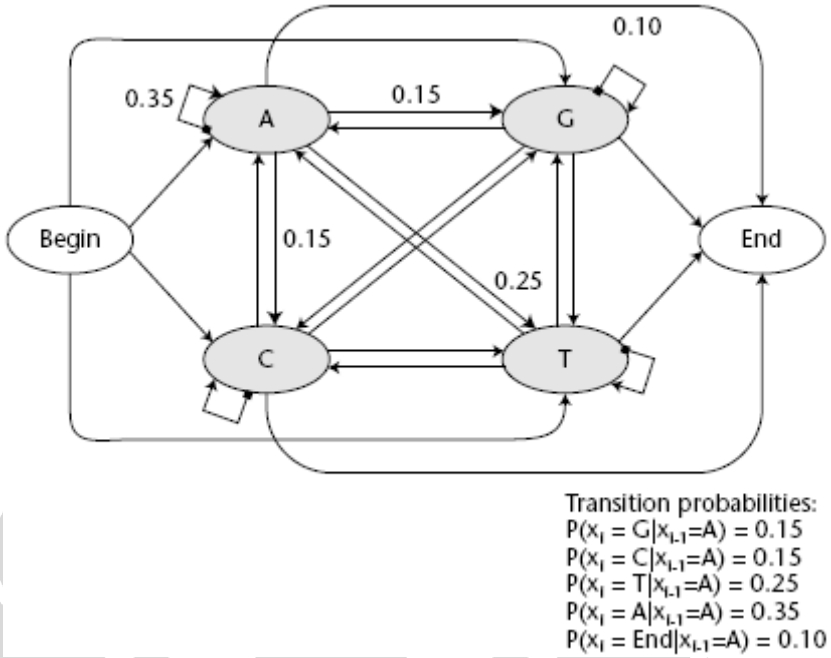
Sebagai contoh, jika kita dapat menentukan semua keadaan yang mungkin sebagai jumlah produk perbankan yang dimiliki oleh seorang nasabah bank, maka kita dapat mendefinisikan sejumlah keadaan seperti A, B, C, D, E. Dimana keadaan tersebut merepresentasikan jumlah produk yang dimiliki nasabah pada waktu t ("A" merepresentasikan keadaan dimana nasabah tersebut adalah nasabah yang baru bergabung atau berpindah ke bank yang lain, "B", "C", "D" secara berturut-turut merepresentasikan jumlah produk yang dimiliki sebanyak 1, 2, atau 3. Sedangkan "E" merepresentasikan seorang nasabah yang memiliki 4 atau lebih produk). Kumpulan semua keadaan tersebut (A, B, C, D, E) dinamakan ruang keadaan (*state space*)[11]

"Jika bulan ini seorang nasabah berada pada keadaan "B" (memiliki sebuah produk), akan seperti apakah keadaan nasabah tersebut bulan depan?" Rantai Markov akan menjawab pertanyaan tersebut melalui dengan menggambarkan distribusi propabilitas keadaan pada waktu *t* ke keadaan pada waktu *t + 1*. Gambar 3 dibawah menggambarkan distribusi tersebut, dimana untuk contoh diatas, terdapat kemungkinan sebesar 0,11 nasabah akan menambah jumlah produknya menjadi 2, dan ada juga kemungkinan sebesar 0,81 nasabah tersebut tidak akan menambah atau mengurangi jumlah produknya atau dengan kata lain nasabah tersebut tetap berada pada keadaan yang sama. Dari gambar tersebut juga bisa terlihat bahwa terdapat 5 % (0,05) kecenderungan nasabah tersebut akan berpindah ke kompetitor yang lain.



Gambar 3. Distribusi Propabilitas perpindahan antara keadaan satu ke keadaan lain

Contoh lain yang menggambarkan rantai markov seperti terlihat pada Gambar 4 dibawah yaitu model rantai markov dari rangkaian DNA.



Gambar 4. Rantai Markov [10]

Rantai Markov merangkum seluruh propabilitas perpindahan ke dalam sebuah matrik transisi. Baris pada matrik transisi seperti terlihat pada gambar 5 dibawah merepresentasikan keadaan saat ini (*t*), sedangkan kolom dari matrik transisi menggambarkan keadaan pada saat yang akan datang (*t+1*). Sebagai contoh nilai 0,13 pada gambar 2-5 (baris ke-3, kolom ke-2), merupakan nilai propabilitas dari keadaan "C" ke keadaan "B". Jika mengambil contoh nasabah bank diatas, nilai tersebut adalah propabilitas seorang nasabah yang bulan ini memiliki jumlah produk perbankan 2 buah akan mengurangi jumlah produk perbankannya menjadi 1 pada bulan depan.

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{bmatrix} 0{,}95 & 0{,}04 & 0{,}01 & 0 & 0 \\ 0{,}05 & 0.81 & 0{,}11 & 0{,}03 & 0 \\ 0{,}01 & 0{,}13 & 0{,}80 & 0{,}15 & 0{,}01 \\ 0{,}01 & 0{,}02 & 0{,}08 & 0{,}83 & 0{,}06 \\ 0 & 0 & 0{,}03 & 0{,}1 & 0{,}87 \end{bmatrix} \end{array}$$

Gambar 5. Matrix transisi keadaan

Secara matematis, jika diketahui rantai markov dengan panjang rangkaian L adalah $x\{x_1, x_2, x_3, ..., x_L\}$, maka dapat dihitung propabilitas sebuah untaian sebagai berikut:

$$\begin{aligned} P(x) &= P(x_L \,.\, x_{L-1}, \ldots, x_1) \\ &= P(x_L \mid x_{L-1}, \ldots, x_1)\, P(x_{L-1} \mid x_{L-2}, \ldots, x_1) \ldots P(x_1) \end{aligned}$$

2.2.2 Classification

Classification (klasifikasi) adalah teknik *data mining* yang memetakan data kedalam grup atau *class* yang berbeda berdasarkan kelas atau grup yang telah ditentukan sebelumnya (*predefined*). Teknik ini sering juga disebut sebagai *supervise learning*, sebab klasifikasi data ditentukan sebelum pengkajian terhadap data dilakukan.

Definisi matematis dari klasifikasi menurut Dunham[3] adalah sebagai berikut. Jika diberikan sebuah *database* D = {t_1, t_2, t_3,...,t_n} dari *tuple* (*items, record*) dan sebuah *set* Kelas C = {C_1, C_2, C_3,...,t_m}, maka tugas dari klasifikasi adalah untuk menentukan sebuah pemetaan $f : D \rightarrow C$, dimana tiap t_i ditempatkan (*assign*) pada satu kelas. Sebuah kelas C_j tepat berisi kumpulan *tuple* yang dipetakan kepadanya, sehingga $C_j = \{ t_i \mid f(t_i) = C_j, 1 \leq i \leq n \text{ dan } t_i \in D\}$.

Berdasarkan definisi diatas maka klasifikasi dapat dipandang sebagai pemetaan dari *database* ke dalam kumpulan kelas-kelas, dengan catatan bahwa kelas-kelas tersebut telah didefinisikan sebelumnya (*predefined*), tidak tumpang tindih satu dengan yang lain, dan terpecah-pecah pada keseluruhan *database*. Tiap *tuple* dalam *database* ditempatkan tepat kedalam satu kelas, dan kelas-kelas yang terbentuk dari klasifikasi tersebut adalah eqivalen satu dengan yang lain. Implementasi dari klasifikasi biasanya dilakukan dalam dua tahap, yaitu:

1. Tahap pembuat model spesifik dengan mengevaluasi data latih (*training*). Tahap ini memiliki input berupa data latih yang sudah memiliki klasifikasi terdefinisi untuk tiap *tuple*-nya. Output dari tahap ini adalah sebuah definisi dari model yang dikembangkan. Model yang tercipta akan mengklasifikasi data seakurat mungkin.
2. Tahap penerapan model yang dikembangkan pada tahap pertama dengan melakukan klasifikasi *tuple* dari *database* target.

2.3 Evaluasi model

Salah satu tahapan akhir yang penting dari sebuah proses pembelajaran adalah evaluasi dari model yang dihasilkan. Evaluasi dari model yang dihasilkan bertujuan untuk mengukur peforma model tersebut. Untuk melakukan sebuah tes, seorang peneliti harus memiliki sekumpulan dataset yang terpisah dan tidak berhubungan dengan dataset yang dipakai untuk membentuk model tersebut.

Salah satu cara untuk melakukan evaluasi sebuah model adalah dengan menggunakan *confusion matrix* [14]. C*onfusion matrix* yang sering juga disebut *classification matrix* merupakan sebuah matrik yang memberikan gambaran penuh

mengenai tingkat kesalahan (*error rate*) serta kualitas prediksi sebuah model. Tabel 1 dibawah merupakan contoh sebuah *confusion matrik* dengan memakai dua buah kelas ("true", "false").

		Actual	
		True	False
Predicted	True	d	b
	False	c	a

Tabel 1. Confusion/classification matrix

Kohavi dan Provost [14], menyatakan beberapa definisi terkait dengan evaluasi yang bisa dipakai melalui *classification matrix*, diantaranya adalah:

- *Recall* atau *True Positip* (TP), adalah proporsi dari sample bernilai "true" yang diprediksi secara benar. TP dihitung dengan menggunakan persamaan : d/(c+d)
- *False Positive* (FP), yaitu proporsi antara sampel bernilai "false" yang salah diprediksi sebagai sample bernilai "true". Persamaan yang digunakan adalah: b / (a+b).
- *True Negative* (TN), didefinisikan sebagai perbandingan antara sampel bernilai "false" yang diprediksi secara benar. Persamaan yang digunakan adalah: a / (a+b).
- *False Negative* (FN), didefinisikan sebagai proporsi sampel bernilai "true" yang salah diprediksi sebagai sampel bernilai "true". Persamaan yang digunakan adalah : c / (c+d)
- Akurasi (AC), didefinisikan sebagai proporsi jumlah sampel yang diprediksi secara tepat, terhadap jumlah seluruh sampel. Persamaan yang digunakan adalah: (d+a)/(a+b+c+d)

- Presisi (PR), didefinisikann sebagai proporsi jumlah sampel bernilai "true" yang berhasil diprediksi secara tepat. Persamaan yang digunakan adalah: d / (b+d).
- [16] menambahkan sebuah definisi baru, yaitu *error rate* (ER) dimana ER = 1 – AC.

2.4 Churn Management

Seperti telah dikemukakan di bab sebelumnya, *churn management* merujuk pada strategi untuk mencegah berpindahnya pelanggan ke pihak pesaing. Strategi ini didasarkan pada sekumpulan data pendukung yang menghubungkan banyak faktor atau variabel. Istilah *churn* adalah sebuah istilah yang biasa terdengar di industri telekomunikasi. Kata "*churn*" itu sendiri dapat diartikan sebagai bergesernya pelanggan (c*ustomer attrition*) dari satu *provider* ke *provider* yang lain.

Penggunaan *churn management* itu sendiri sudah cukup meluas, terutama di industri telekomunikasi. Ketatnya persaingan, dan deregulasi yang terjadi di industri ini membuat perusahaan di industri ini menggeser fokus mereka dari yang awalnya fokus pada percepatan pengembalian modal melalui peningkatan *revenue* dan pertumbuhan *market share*, menjadi ke arah *margin enhancement* melalui *revenue enhancement* dan *cost enhancement.*

Tantangan terbesar mereka adalah meningkatnya perpindahan pelanggan (*customer churn*), akibat persaingan harga dan kualitas layanan yang semakin ketat di industri telekomunikasi. Ketatnya persaingan harga dan kualitas layanan tersebut merupakan akibat dari kondisi pasar yang semakin jenuh dan lingkungan

kompetisi yang semakin intensif. Kondisi pasar tersebut memberikan pelanggan pilihan yang lebih banyak, sekaligus meningkatkan posisi tawar mereka[4]. Tantangan ini menghasilkan semakin tingginya *cost* untuk mendapatkan pelanggan baru dan menurunkan rata-rata *billing* bulanan.

Tantangan yang hampir serupa juga terjadi di industri perbankan. Homogenitas produk yang ada di pasaran, persaingan tingkat suku bunga berakibat pada peningkatan posisi tawar dari nasabah bank. Hal ini menyebabkan nasabah dengan begitu mudahnya menutup rekening tabungan mereka dengan memindahkan dana yang ada di rekening tersebut ke rekening lain yang berada di bank pesaing. Padahal dana nasabah tersebut adalah faktor yang sangat penting dalam industri perbankan.

Churn Management pada industri telekomikasi telah secara luas dipakai, yaitu dengan menerapkan *churn prediction* yang memakai *data mining* sebagai alat bantu mereka untuk mengidentifikasi pelanggan yang memiliki kecenderungan untuk berpindah ke penyedia (*provider*) jasa telekomunikasi yang lain.

Terdapat dua jenis pendekatan terhadap pemodelan *churn*. Pendekatan pertama dinamakan model *churn binary outcome* memperlakukan *churn* sebagai hasil pasangan (*binary outcomel*) prediksi antara pelanggan yang akan tetap setia dan pelanggan yang akan meninggalkan perusahaan. Pendekatan kedua mencoba untuk melakukan estimasi terhadap daur hidup (*lifetime/tenure*) dari pelanggan yang tersisa pendekatan ini disebut *survival analysis*. Berry, Michael J. A [8].