BAB II

TINJAUAN PUSTAKA

II.1. Data Mining

Proses *Data Mining* yaitu proses mencari pola atau informasi menarik dalam data terpilih dengan menggunakan teknik, metode atau algoritma tertentu sesuai dengan tujuan dari proses KDD secara keseluruhan. (Ali Ikhwan, dkk, 2013). Data mining dapat diterapkan pada berbagai bidang yang mempunyai sejumlah data, tetapi karena wilayah penelitian dengan sejarah yang belum lama, dan belum melewati masa 'remaja', maka data mining masih diperdebatkan posisi bidang pengetahuan yang memilikinya. Maka, Daryl Pregibon menyatakan bahwa "data mining adalah campuran dari statistik, kecerdasan buatan, dan riset basis data" yang masih berkembang (Eko Prasetyo, 2011).

II.1.1. Tahapan Data Mining

Ada empat tahap yang dilalui dalam data mining, antara lain:

1. Tahap pertama : *Pricise statement of the problem*

Sebelum mengakses perangkat lunak data mining, seorang analis harus memiliki kejelasan perihal 'pertanyaan apa yang akan ingin dijawabnya'. Jika tidak ada *formula* yang tepat untuk problematika yang ada maka anda hanya akan membuang-buang dan uang dalam membuat solusinya.

2. Tahap kedua : *Initial exploration*

Tahap ini dimulai dengan mempersiapkan data yang juga juga termasuk kedalam data mining "*cleaning*" (misalnya : mengedentifikasi dan menyikirkan

data yang dikodekkan salah), transformasi data, memilih *subset record*, *data set*, langkah awal seleksi. Mendeskripsikan dan memvisualisasikan data adalah kunci dari tahap ini.

3. Tahap tiga : *Model building and validation*

Tahap ini melibatkan pertimbangan terhadap ragam permodelan dan memilih yang terbaik bagi performasi prediktif.

4. Tahap ke-empat : *Deployment*

Memilih aplikasi yang tepat berikut permodelan untuk membuat (*generate*) prediksi. Selanjutnya kita kan melihat rincian perihal tahapan-tahapan data mining. (Ali Ikhwan, dkk, 2013).

II.1.2. Pengelompokan Data Mining

Data Miningdibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, yaitu (Ali Ikhwan, dkk, 2013):

1. Deskripsi

Terkadang peneliti dan analis secara sederhana ingin mencoba mencari cara untuk menggambarkan pola dan kecenderungan yang terdapat dalam data. Sebagai contoh, petugas pengumpulan suara mungkin tidak dapat menentukan keterangan atau fakta bahwa siapa yang tidak cukup profesional akan sedikit didukung dalam pemilihan presiden.

2. Estimasi

Estimasi hampir sama dengan klasifikasi, kecuali variabel target estimasi lebih kearah numerik daripada kearah kategori. Sebagai contoh akan dilakukan estimasi tekanan darah sistolik pada pasien rumah sakit berdasarkan umur pasien, jenis

kelamin, indeks berat badan, dan level sodium darah. 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.

3. Prediksi

Prediksi hampir sama dengan klasifikasi dan estimasi, kecuali bahwa dalam prediksi nilai dari hasil akan ada di masa mendatang.

Contoh prediksi bisnis dan penelitian adalah:

a. Prediksi harga beras dalam tiga bulan yang akan datang.
b. Prediksi persentasi kenaikan kecelakaan lalu lintas tahun depan jika batas bawah kecepatan dinaikkan.

Beberapa metode dan teknik yang digunakan dalam klasifikasi dan estimasi dapat pula digunakan (untuk keadaan yang tepat) untuk prediksi.

4. Klasifikasi

Dalam klasifikasi, terdapat target variabel kategori. Sebagai contoh, penggolongan pendapatan dapat dipisahkan dalam tiga kategori, yaitu pendapatan tinggi, pendapatan sedang dan pendapatan rendah.

Contoh lain klasifikasi dalam bisnis dan penelitian adalah:

a. Menentukan apakah suatu transaksi kartu kredit merupakan transaksi yang curang atau tidak.
b. Memperkirakan apakah suatu pengajuan hipotek oleh nasabah merupakan suatu kredit yang baik atau buruk.

c. Mendiagnosis penyakit seorang pasien untuk mendapatkan termasuk kategori penyakit apa.

5. Pengklusteran

Pengklusteran merupakan pengelompokan *record*, pengamatan atau memperhatikan dan membentuk kelas objek-objek yang memiliki kemiripan.Contoh pengklusteran dalam bisnis dan penelitian adalah:

a. Mendapatkan kelompok-kelompok konsumen untuk target pemasaran dari satu suatu produk bagi perusahaan yang tidak memiliki dana pemasaran yang besar.
b. Untuk tujuan audit akuntansi, yaitu melakukan pemisahan terhadap perilaku *financial* dalam baik dan mencurigakan.
c. Melakukan pengklusteran terhadap ekspresi dari *gen*, untuk mendapatkan kemiripan perilaku dari *gen* dalam jumlah besar.

6. Asosiasi

Tugas asosiasi dalam *Data Mining* adalah menemukan *attribut* yang muncul dalam satu waktu. Dalam dunia bisnis lebih umum disebut analisis keranjang belanja. Contoh asosiasi dalam bisnis dan penelitian adalah:

a. Meneliti jumlah pelanggan dari perusahaan telekomunikasi seluler yang diharapkan untuk memberikan respon positif terhadap penawaran *upgrade* layanan yang diberikan.
b. Menentukan barang dalam supermarket yang dibeli secara bersamaan dan yang tidak pernah dibeli secara bersamaan. (Ali Ikhwan, dkk, 2013).

II.2. Metode *FP-Growth*

FP-Growth adalah salah satu alternatif algoritma yang dapat digunakan untuk menentukan himpunan data yang paling sering muncul *(frequent itemset)* dalam sekumpulan data. Struktur data yang digunakan untuk mencari *frequent itemset* dengan algoritma *fp-growth* adalah perluasan dari ssebuah pohon *prefix*, yang biasa disebut *FP-Tree*. (Ririanti, 2014).

Contoh Kasus :

Metode *FP-Growth* dapat dibagi menjadi 3 tahapan utama yaitu sebagai:

1. Tahap pembangkitan *conditional pattern base*,
2. Tahap pembangkitan *conditional FP-Tree*, dan
3. Tahap pencarian *frequent itemset*.

Ketiga tahap tersebut merupakan langkah yang akan dilakukan untuk mendapat *frequent itemset*.

Input : FP-Tree Tree

Output : Rt Sekumpulan lengkap pola *frequent*

Methode : FP-Growth (Tree, null)

Procedure : FP-Growth (Tree, α)

{

01*: if Tree* mengandung *single path* P;

02*: then* untuk tiap kombinasi (dinotasikan β) dari *node-node* dalam *path do*

03: bangkitkan pola β α dengan *support* dari *node-node* dalam *path do* β;

04: *else u*ntuk tiap a1 dalam *header* dari *tree do*

}

05: bangkitkan pola

06: bangun β = a1 α dengan *support* = a1 *support*

07*: if Tree* β =

Tabel II.2. Untuk Memberi Tanda Pada Setiap

Item
SMK Negeri (A1)
SMK Swasta(A2)
SMA Negeri (A3)
SMA Swasta (A4)
Madrasah Aliyah (A5)
Medan (B1)
Aceh (B2)
Lubuk Pakam (B3)
Deli Serdang (B4)
Perbaungan (B5)
Tebing Tinggi (B6)
Riau (B7)
IPA (C1)
IPS (C2)
TKJ (C3)
MULTIMEDIA (C4)
Sistem Informasi (D1)
Sistem Komputer (D2)
Teknik Komputer (D3)
Manajemen Informatika (D4)

Itemset

(Sumber : Ali Ikhwan, dkk ; 2013)

Tabel II.3. Frekuensi dari Setiap Transaksi *Item*

TID	ITEM
1	D1,B1
2	C1,A3,B1
3	A3,C2
4	D1,A4,B1,C2
5	D1,C1,A4
6	C1,A3
7	D1,C1,A3
8	D1

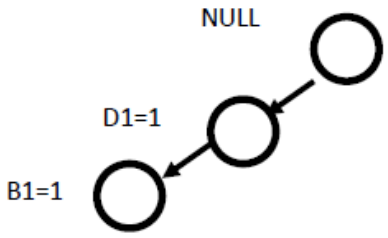
Tabel II.3. Frekuensi dari Setiap Transaksi Item (Lanjutan)

9	D1,C1,A3
10	D1,C1,A4
11	D1,C1,A4
12	A3,B1.C2
13	D1,C1
14	C1
15	D1,C1,A3
16	C1,A3
17	D1,C1,A3
18	-
19	D1,A4,C2
20	D1,A4,B1,C2
21	A3,B1,C2
22	A3,B1,C2
23	D1,B1
24	D1
25	D1,C1,A4,B1
26	D1,C1,A4,B1
27	D1,C1,A4
28	D1,C1,A3
29	C2
30	D1,A4,C2

(Sumber : Ali Ikhwan, dkk ; 2013)

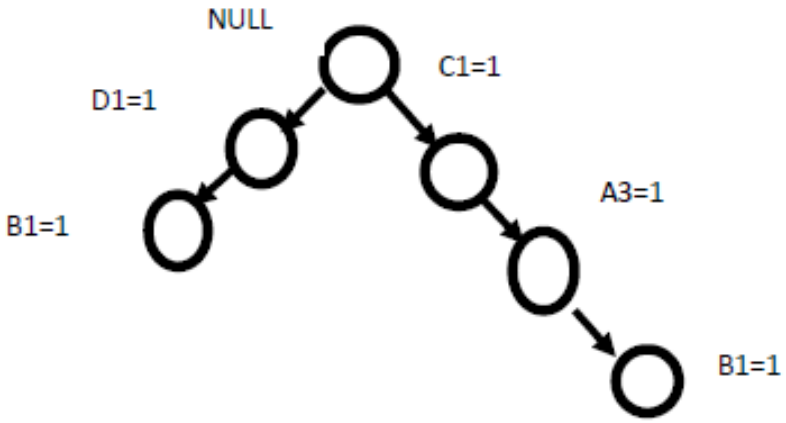
Maka langkah selanjutnya adalah membentuk pohon *FP-Tree* dengan melihat tabel II.3 sebagai berikut:

Gambar di bawah ini memberikan ilustrasi mengenai pembentukan *FP-tree* setelah pembacaan TID pada tabel II.3.

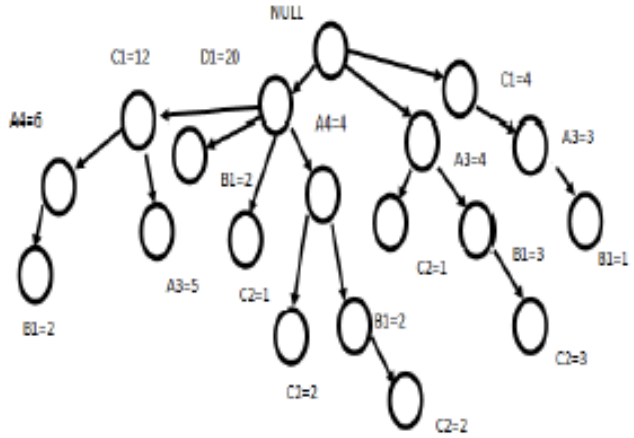


Gambar II.1. Hasil Pembentukan *FP-tree* Setelah Pembacaan TID 1

Gambar II.2 adalah penjelasan tentang pembentukan *FP-Tree* setelah pembacaan didapat setelah melakukan TID 1, Yaitu berisi : NULL- D1 (Sistem Informasi) =1 –(B1) Medan= 1.



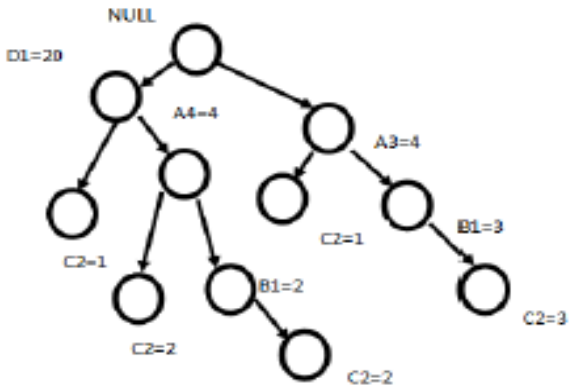
Gambar II.2. Hasil Pembentukan *FP-tree* Setelah Pembacaan TID 2 didapat setelah melakukan TID 2 ,Yaitu: NULL-C1 (IPA) =1- A3 (SMAN) =1- B1(Medan)=1.



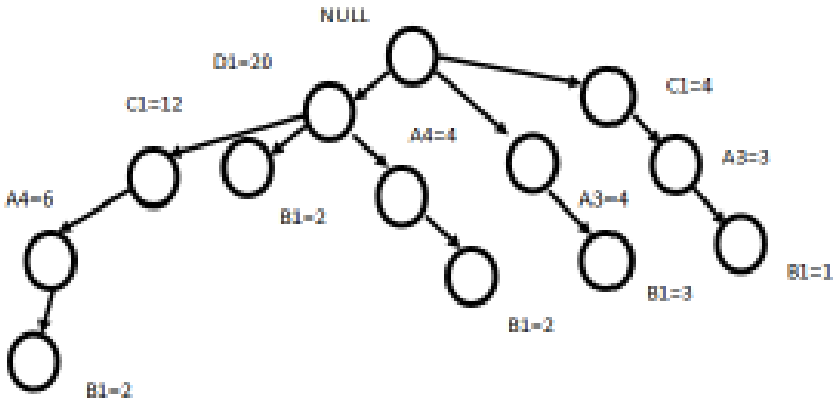
Gambar II.3 : Hasil Pembentukan *FP-tree* Setelah Pembacaan TID 30

Gambar didapat setelah melakukan TID 30 yang dijumlahkan, yaitu berisi Null-Sistem Informasi (D1) = 20 -IPA (C1) =16 -SMA Negeri (A3) =12 -SMA Swasta (A4) =10- Medan (B1) = 10-IPS (C2) = 9.

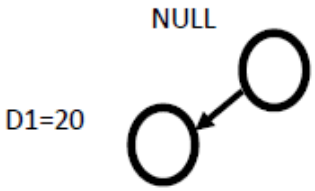
Untuk menemukan *Frequent itemset* dari tabel 4.2, maka perlu ditentukan terlebih dahulu lintasan yang berakhir dengan *support count* terkecil, yaitu C2 yang diikuti dengan A4,A3, C1,D1 dan diakhiri D1. Proses pembentukan masing-masing *node* dapat dilihat pada gambar II.4.



Gambar II.4. Lintasan yang Mengandung Simpul C2



Gambar II.5 Lintasan yang Mengandung Simpul B1



Gambar II.6 Lintasan yang Mengandung Simpul D1

Setelah mencari *frequent itemset* untuk beberapa akhiran *suffix* maka didapat hasil yang dirangkum dalam tabel berikut:

Tabel II.3 Daftar *Frequent Itemset* Diurutkan Berdasarkan Hubungan Akhiran

Suffix	*Frequent Itemset*
C2	{C2},{C2,A4}{C2,A3}{C2,D1}{C2,B1, A3},{C2,B1,A4}
B1	{B1},{B1,D1},{B1,A3},{B1,A3,C1},(B 1,A4,D1},{B1,A4,C1,D1}
A4	{A4},{A4,C1,D1}
A3	{A3},{A3,C1},{A3,C1,D1}
C1	{C1},{ C1,D1}
D1	{D1}

(Sumber : Ali Ikhwan, dkk ; 2013)

Dari *frequent itemset* yang didapat dari pembentikan *Fp-Tree* dan *FP-Growth* maka dapat dihitung nilai *Support* dan *Confidence* sebagai berikut:

Support = (Sistem Informasi,IPA,SMA Swasta,Medan) = *Count* (Sistem Informasi,IPA,SMA Swasta,Medan)/Jumlah Transaksi = 1/30.

Support = (Sistem Informasi,IPA,SMA Negeri) = *Count* (Sistem Informasi,IPA,SMA Negeri)/Jumlah Transaksi = 1/30.

Support = (Sistem Informasi, Medan) *Count* (Sistem Informasi, Medan)/Jumlah Transaksi =1/30.

Support = (Sistem Informasi,SMA Swasta, IPS) *Count*(Sistem Informasi,SMA Swasta, IPS)/Jumlah Transaksi = 1/30.

Support = (Sistem Informasi,SMA Swasta,Medan, IPS) *Count* (Sistem Informasi,SMA Swasta, Medan,IPS)/Jumlah Transaksi =1/30.

Support = (SMA Negeri ,IPS) *Count* (SMA Negeri ,IPS) / Jumlah Transaksi = 1/30.

Support = (SMA Negeri ,Medan,IPS) *Count* (SMA Negeri ,Medan,IPS) / Jumlah Transaksi = 1/30.

Support = (IPA,SMA Negeri ,Medan) *Count* (IPA,SMA Negeri ,Medan) / Jumlah Transaksi = 1/30.

Sedangkan untuk *confidence* atau nilai kepercayaannya adalah sebagai berikut :

Confidence = (SMA Negeri,IPS) = Count (SMA Negeri,IPS)/*Count* Medan= 1/9.

Confidence = (Sistem Informasi,SMA Swasta,Medan,IPS) = *Count* (Sistem Informasi,SMA Swasta,Medan,IPS /*Count* Medan = 1/9.

Confidence = (SMA Negeri,Medan,IPS) *Count* (SMA Negeri,Medan,IPS /*Count* Medan =1/9.

Confidence = (Sistem Informasi,SMA Swasta,IPS) *Count*(Sistem Informasi,SMA Swasta,IPS)/*Count* Medan= 1/19.

Confidence = (Sistem Informasi,SMA Swasta,Medan,IPS) *Count* (Sistem Informasi,SMA Swasta,Medan,IPS)/*Count* Medan =1/9.

Confidence = (SMA Negeri, Medan,IPS) *Count* (SMA Negeri, Medan,IPS) / *Count* Medan =1/9.

Confidence = (Sistem Informasi,IPS) *Count* (Sistem Informasi,IPS) / *Count* Medan =1/9.

Confidence = (Sistem Informasi,SMA Swasta,IPS) *Count* (Sistem Informasi,SMA Swasta,IPS) / *Count* Medan =1/9.

Setelah didapat nilai *support* dan *confidence* dari keseluruhan kombinasi pada data dengan perhitungan *FP-Tree* dan *FP-Growth* maka didapat niai *support* dan *confidence* yang paling tinggi dan akurat yaitu kombinasi. (D1,C1,A3) {

Sistem Informasi, IPA, SMA Negeri } yang mempunyai nilai *support* : 5/30 = 0,16 dan nilai *confidence* : 5/20 = 0.25. (Ali Ikhwan, 2012).

II.6. *Microsoft Visual Basic 2010*

Visual Basic 2010 merupakan salah satu bagian dari produk pemrograman terbaru yang dikeluarkan oleh *Microsoft*, yaitu *Microsoft Visual Studio 2010. Visual Studio* merupakan produk pemrograman andalan dari *mocrosoft corporation*, dimana di dalamnya berisi beberapa jenis *IDE* pemrograman seperti *Visual Basic*, *Visual C++*, *Visual Web Developer*, *Visual C#,* dan *Visual F#*.

Semua *IDE* pemrograman tersebut sudah mendukung penuh implementasi *.Net Framework* terbaru, yaitu *.Net Framework 4.0* yang merupakan pengembangan dari *.Net Framework 3.5*. Adapun database standar yang disertakan adalah *Microfot SQL Server 2008 express*.

Visual Basic 2010 merupakan versi perbaikan dan pengembangan dari versi pendahulunya yaitu *visual basic 2008*. Beberapa pengembangan yang terdapat di dalamnya antara lain dukungan terhadap *library* terbaru dari *Microsoft*, yaitu *.Net Framework 4.0,* dukungan terhadap pengembangan aplikasi menggunakan *Microsoft SilverLight*, dukungan terhaddap aplikasi berbasis *cloudcomputing*, serta perluasan dukungan terhadap *database-database*, baik *standalone* maupun *database server.* (Wahana Komputer, 2011).

II.7. Basis Data

Basis data dapat didefenisikan sebagai koleksi dari data-data yang terorganisasi sedemikian rupa sehingga data mudah disimpan dan dimanipulasi

(diperbarui, dicari, diolah dengan perhitungan-perhitungan tertentu, serta dihapus). Secara teoritis, basis data tidak harus berurusan dengan komputer (misalnya, catatan belanja hari ini yang dibuat oleh seorang ibu rumah tangga juga merupakan basis data dalam bentuk yang sangat sederhana). (Adi Nugroho, 2011).

II.8. Normalisasi

Normalisasi dapat dipahami sebagai tahapan-tahapan yang masing-masing berhubungan dengan bentuk normal. Bentuk normal adalah keadaan relasi yang dihasilkan dengan menerapkan aturan sederhana berkaitan dengan konsep kebergantungan fungsional pada relasi yang bersangkutan. Kita akan menggambarkannya secara garis besar sebagai berikut :

1. Bentuk Normal Pertama (1NF/ *First Normal Form*)

   Bentuk normal pertama adalah suatu bentuk relasi dimana atribut bernilai banyak (*multivalues attribute*) telah dihilangkan sehingga kita akan menjumpai nilai tunggal (mungkin saja nilai *null*) pada perpotongan setiap baris dan kolom.

2. Bentuk Normal Kedua (2ND/ *Second Normal Form*)

   Semua kebergantungan fungsional yang bersifat sebagian (*particial functional dependency*) telah dihilangkan.

3. Bentuk Normal Ketiga (3RD/ *Thrid Normal Form*)

   Semua kebergantungan transitif (*transitive dependency*) telah dihilangkan.

4. Bentuk Normal *Boyce-Codd* (BCNF/ *Boyce-Codd Normal Form*)

   Semua anomaly yang tersisa dari hasil penyempurnaan kebergantungan fungsional sebelumnya telah dihilangkan.s

5. Bentuk Normal Keempat (4NF/ *Fourth Normal Form*)

   Semua kebergantungan bernilai banyak telah dihilangkan.

6. Bentuk Normal Kelima (5NF/ *Fifth Normal Form*)

   Semua anomaly yang tertinggi telah dihilangkan.

II.9. *SQL Server 2008*

SQL Server 2008 adalah sebuah*RDBMS (Relational Database ManagementSystem)* yang sangat powerful dan telahterbukti kekuatannya dalam mengolah data.Dalam versi terbarunya ini, *SQL Server* 2008memiliki banyak fitur yang bisa dihandalkanuntuk meningkatkan performa *database*.*SQL Server* 2008 memiliki suatu *GUI(Graphic User Interface)* yang kita gunakanuntuk melakukan aktivitas sehari hariberkaitan dengan database,seperti menulis*T-SQL*, melakukan *backup* dan *restoredatabase*, melakukan *security database*terhadap aplikasi, dan sebagainya. Pada *GUI*tersebut kita bisa melakukan settinganterhadap *SQL Serve*r untuk berkerja lebihoptimal.Settingan juga bisa dilakukanmenggunakan *script* untuk memudahkandeveloper mengubah *Setting Options* pada*SQL Server* 2008. (Ruslan, 2013).

II.10. *Unified Modeling Language* (*UML*)

Menurut Windu Gata (2013) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (*UML*). *UML* adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

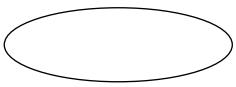
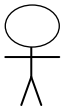
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. *UML* saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem. (Gellysa Urva dan Helmi Fauzi Siregar, 2015).

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan *UML* adalah sebagai berikut:

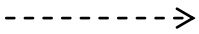
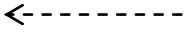
1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.4 dibawah ini:

Tabel II.4. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, dan dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem.

Tabel II.4. Simbol *Use Case*(Lanjutan)

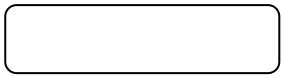
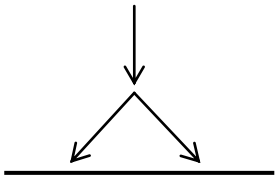
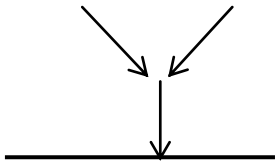
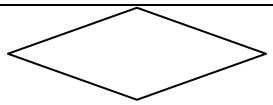
	Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki *control* terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber:Gellysa Urva dan Helmi Fauzi Siregar; 2015)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2 dibawah ini:

Tabel II.3. Simbol *Activity Diagram*

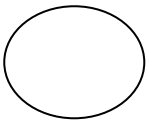
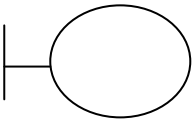
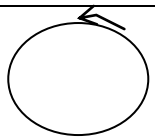
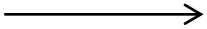
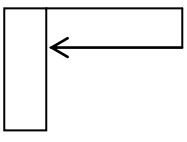
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.4 dibawah ini :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass,* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive,* menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.

Tabel II.6. Simbol *Sequence Diagram*(Lanjutan)

	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.*Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.5 dibawah ini:

Tabel II.5. *Multiplicity Class Diagram*

Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti

Tabel II.7. *Multiplicity Class Diagram*(Lanjutan)

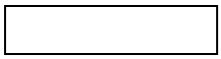
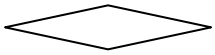
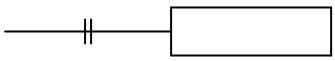
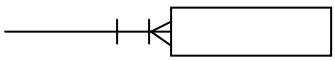
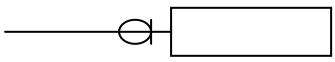
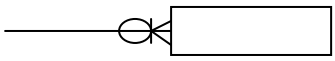
	minimal 2 maksimum 4

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015)

II.9. *Entity Relationship Diagram*

Entity Relationship Diagram (ERD) adalah bagian yang menunjukkan hubungan antara *entity* yang ada dalam sistem. Simbol-simbol yang digunakan dapat dilihat dari tabel II.8. (Yuhendra, M.T, Dr. Eng dan Riza Eko Yulianto, 2015).

Tabel II.8. Simbol Yang DigunakanPada *Entity Relationship Diagram (ERD)*

SIMBOL	KETERANGAN
	*Entity*
	Atribut Dan *Entity*
	Atribut Dan *Entity* Dengan *Key* (Kunci)
	Relasi Atau Aktifitas Antar *Entity*
	Hubungan Satu Dan Pasti
	Hubungan Banyak Dan Pasti
	Hubungan Satu Tapi Tidak Pasti
	Hubungan Banyak Tapi Tidak Pasti

(Sumber : Yuhendra, M.T, Dr. Eng dan Riza Eko Yulianto; 2015)