BAB II

TINJAUAN PUSTAKA

II.1. *Data Mining*

Data Mining adalah proses menganalisa dari data perspektif yang berbeda dan menyimpulkannya menjadi informasi-informasi penting yang dapat dipakai untuk meningkatkan keuntungan memperkecil biaya pengeluaran, atau bahkan keduanya. Secara teknis, *data mining* dapat disebut sebagai proses untuk menentukan korelasi atau pola dari ratusan atau ribuan *field* dari sebuah relasional *database* yang besar (Angga Ginanjar, dkk, 2012 : 54).

Istilah *data mining* dan *knowledge discovery in databases* (KDD) sering kali digunakan secara bergantian untuk menjelaskan proses penggalian informasi yang tersembunyi dalam suatu basis data yang besar. Sebenarnya kedua istilah tersebut memiliki konsep yang berbeda, tetapi berkaitan satu sama lain. Dan salah satu tahapan dalam keseluruhan proses KDD adalah *data mining*. Proses KDD secara garis besar dapat dijelaskan sebagai berikut :

1. *Data Selection*

   Pemilihan (seleksi) data dari sekumpulan data operasional perlu dilakukan sebelum tahap penggalian informasi dalam KDD dimulai. Data hasil seleksi yang akan digunakan untuk proses *data mining*, disimpan dalam satu berkas, terpisah dari basis data operasional.

1. *Pre-processing / Cleaning*

   Sebelum proses *data mining* dapat dilaksanakan, perlu dilakukan proses *cleaning* pada data yang menjadi fokus KDD. Proses *cleannng* mencakup antara lain membuang duplikasi data, memeriksa data yang inkonsisten, dan memperbaiki kesalahan pada data, seperti kesalahan cetak (*tipografi*). Juga dilakukan proses *encrichment*, yaitu proses "memperkaya" data yang sudah ada dengan data atau informasi lain yang relevan dan diperlukan untuk KDD, seperti data atau informasi eksternal.

2. *Tansformation*

   Coding adalah proses tranformasi pada data yang telah dipilih, sehingga data tersebut sesuai untuk proses *data mining*. Proses *coding* dalam KDD merupakan proses kreatif dan sangat tergatung pada jenis atau pola informasi yang akan dicari dalam basis data.

3. *Data Mining*

   Data mining adalah proses mencari pola atau informasi menarik dalam data terpilih dengan menggunakan teknik atau metode tertentu. Teknik, metode, atau algoritma dalam *data mining* sangat bervariasi. Pemilihan metode atau algoritma yang tepat sangatbergantung pada tujuan dan proses KDD secara keseluruhan.

4. *Interpretation/ Evaluation*

   Pola nformasi yang dihasilkan dari proses *data mining* perlu ditampilkan dalam bentuk yang mudah dimengerti oleh pihak yang berkepentingan. Tahap ini merupakan bagian dari proses KDD yang disebut *interpretation*. Tahap ini

mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesis yang ada sebelumnya.

II.1.1. Pengelompokan *Data Mining*

Data mining dibagi menjadi beberapa kelompok berdasarkan tugas yang dapat dilakukan, yaitu :

1. Deskripsi

   Terkadang peneliti dan analisi secara sederhana ingin mencoba mencari cara untuk meggambarkan pola dan kecenderungan yang terdapat dalam data. Sebagai contoh, petugas pengumpulan suara mungkin tidak dapat menemukan keterangan atau fakta bahwa siapa yang tidak cukup professional akan sedikit didukung dalam pemilihan presiden. Deskripsi dari pola dan kecenderungan sering memberikan kemungkinan penjelasan untuk suatu pola atau kecenderungan.

2. Estimasi

   Estimasi hampir sama dengan klasifikasi, kecuali variabel target estimasi lebih kearah numerik daripada 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, 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 dalam bisnis dan penelitian adalah :

   a. Prediksi harga beras dalam tiga bulan yang akan dating.
   b. Prediksi presentase kenaikan kecelakaan lalu lintas tahun depan jika batas bawah kecepatan dinaikan.

   Beberapa metode 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.

5. Pengklusteran

   Pengklusteran merupakan pengelompokan *record*, pengamatan, atau memperhatikan dan membentuk kelas objek-objek yang memiliki kemiripan. Kluster adalah kumpulan *record* yang memiliki kemiripan satu dengan yang lainnya dan memiliki ketidakmiripan sengan *record-record* dalam kluster lain.

   Pengklusteran berbeda dengan klasifikasi yaitu tidak adanya variabel target dalam pengklusteran. Pengklusteran tidak mencoba untuk melakukan klasifikasi,

mengestimasi, atau memprediksi nilai dari variabel target. Akan tetapi, algoritma pengklusteran mencoba untuk melakukan pembagian terhadap keseluruhan data menjadi kelompok-kelompok yang memiliki kemiripan (homogen), yang mana kemiripan *record* dalam satu kelompok akan bernilai maksimal, sedangkan kemiripan dengan *record* dalam kelompok lain akan bernilai minimal.

Contoh pengklusteran dalam bisnis dan penelitian adalah :

a. Mendapatkan kelompok-kelompok konsumen untuk target pemasaran dari suatu produk bagi perusahaan yang tidak memiliki dana pemasaran yang besar.
b. Untuk tujuan audit akuntansi, yaitu melakukan pemisahan terhadap perilaku finansial 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 atribut 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 respons positif terhadap penawaran *upgrade* layanan yang diberikan.
b. Menemukan barang dalam supermarket yang dibeli secara bersamaan dan barang yang tidak pernah dibeli secara bersamaan.

II.2. Algoritma C4.5

Algoritma C4.5 adalah salah satu metode untuk membuat *decision tree* berdasarkan training data yang telah disediakan. Algoritma C4.5 merupakan pengembangan dari ID3.

Beberapa pengembangan yang dilakukan C4.5 adalah bisa mengatasi *missing value*, *continue data* dan *pruning*. Algoritma C4.5 mempunyai input berupa *training samples* dan *samples*, *training samples* berupa data contoh yang akan digunakan untuk membangun sebuah *tree* yang telah diuji kebenarannya, sedangkan untuk *samples* merupakan *field-field* data yang nantinya akan kita gunakan sebagai parameter dalam melakukan klasifikasi data (Rismayanti, 2016 : 117).

Algoritma C4.5 dan pohon keputusan merupakan dua model yang tak terpisahkan, karena untuk membangun sebuah pohon keputusan, dibutuhkan algoritma C4.5. Di akhir tahun 1970 hingga di awal tahun 1980-an, J.Ross Quinlann seorang peneliti di bidang mesin pembelajaran mengembangkan sebuah model pohon keputusan yang dinamakan ID3 (*Iterative Dichotomiser*), walaupun sebenarnya proyek ini telah dibuat sebelumnya oleh E.B.Hunt, J.Marlin, dan P.T.Stone. Kemudian Quinlann membuat algoritma dari pengembangan ID3 yang dinamakan C4.5 yang berbasis *supervised learning*. (Siska Haryati, dkk, 2015 : 132)

Serangkaian perbaikan yang dilakukan pada ID3 mencapai puncaknya dengan menghasilkan sebuah sistem praktis dan berpengaruh untuk decision treeyaitu C4.5. Adapun perbaikannya adalah sebagai berikut:

1. Algoritma C4.5 menghitung *gain ratio* untuk masing-masing atribut, dan atribut yang memiliki nilai yang tertinggi akan dipilih sebagai simpul atau *node*. Penggunaan *gain ratio* ini memperbaiki kelemahan dari ID3 yang menggunakan *information gain*.
2. Pemangkasan (*prunning*) dapat dilakukan pada saat pembangunan pohon (*tree*) ataupun pada saat proses pembanguan *tree* selesai.
3. Mampu menangani *continues attribute*.
4. Mampu menangani *missing data*.
5. Mampu membangkitkan *rule* dari sebuah *tree*.

Ada beberapa tahap dalam membuat sebuah pohon keputusan dengan algoritma C4.5 yaitu:

1. Menyiapkan *data training*. *Data training* biasanya dari histori yang pernah terjadi sebelumnya dan sudah dikelompokkan ke dalam kelas-kelas tertentu.
2. Menghitung nilai *Entropy* yaitu:

$$Entropy\ (S) = \sum_{i=1}^{n} - pi * log2\ pi$$

Keterangan :

S = Himpunan kasus

n = Jumlah partisi S

P_i = Proporsi dari S_i terhadap S

3. Kemudian hitung nilai *Gain* dengan *information gain*:

$$Gain\ (S,A) = Entropy(S) - \sum_{i=1}^{n} \frac{|si|}{|s|} * Entropy$$

Keterangan :

S = Himpunan kasus

A = Atribut

N = Jumlah partisi atribut A

|Si| = Jumlah kasus pada partisi ke-i

|S| = Jumlah kasus dalam S

4. Kemudian hitung nilai *Split Info* :

$$SplitInfo(S,A) = -\sum_{i=1}^{n} \frac{S_i}{S} log_2 \frac{S_i}{S}$$

dimana:

S = himpunan kasus

A = atribut

Si = jumlah sampel untuk atribut i

5. Menghitung nilai Gain Ratio untuk masing-masing atribut. Atribut yang memiliki Gain Ratio tertinggi dipilih menjadi akar (*root*) dan atribut yang memiliki nilai Gain Ratio lebih rendah dari akar (*roo*t) dipilih menjadi cabang (*branches*). Cara Menghitung *Gain Ratio* yaitu :

$$GainRatio(S,A) = \frac{Gain(S,A)}{SplitInfo(S,A)}$$

dimana:

S = himpunan kasus

A = atribut

Gain (S,A) = info *gain* pada atribut A

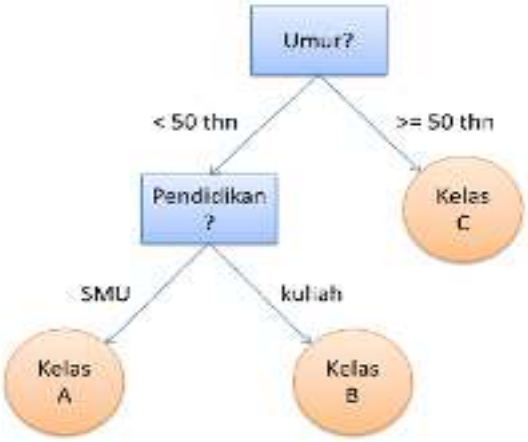
Split Info (S,A) = *split info* pada atribut A

6. Ulangi langkan ke-2 hingga semua tupel terpartisi
7. Proses partisi pohon keputusan akan berhenti saat:
    a. Semua tupel dalam node N mendapat kelas yang sama.
    b. Tidak ada atribut di dalam tupel yang dipartisi lagi.
    c. Tidak ada tupel di dalam cabang yang kosong.

II.2.1. Pohon Keputusan (*Decission Tree*)

Pohon keputusan adalah salah satu metode klasifikasi yang paling populer karena mudah di interperensi manusia. Pohon keputusan adalah model prediksi menggunakan stuktur pohon atau stuktur berhirarki. Konsep dari pohon keputusan adalah mengubah data menjadi pohon keputusan dan aturan-aturan keputusan. Data dalam pohon keputusan biasanya dinyatakan dalam bentuk tabel dengan atribut dan *record*. Atribut menyatakan suatu parameter yang dibuat sebagai kriteria dalam pembentukan *tree*. (Siska Haryati, dkk, 2015 : 132)

Pohon keputusan (*Decission tree*) biasanya digunakan untuk mendapatkan informasi untuk tujuan pengambilan sebuah keputusan, seperti terlihat pada gambar di bawah ini :



Gambar II.1. Pohon Keputusan

Untuk mengklasifikasikan sampel yang tidak diketahui, nilai atribut dari sampel tersebut dites oleh decision tree. Suatu jalur ditelusuri dari *root* ke *node leaf* yang memiliki prediksi kelas untuk sampel tersebut. *Decision tree* dengan mudah dapat diubah dari aturan (*rule*) prediksi ke klasifikasi. Saat *decision tree* dibuat, kebanyakan dari cabang mungkin memperhatikan *noise* atau *outlier* pada *training data*, *tree pruning* berusaha untuk mengidentifikasi dan membuang cabang-cabang tersebut dengan tujuan memperbaiki akurasi klasifikasi pada data yang tidak kelihatan.

II.3. Beasiswa

Beasiswa adalah pemberian bantuan keuangan yang diberikan kepada perorangan, mahasiswa atau pelajar yang digunakan demi keberlangsungan pendidikan yang ditempuh. Beasiswa dapat diberikan oleh lembaga pemerintah, perusahaan ataupun yayasan. Pemberian beasiswa dapat dikategorikan pada

pemberian cuma-cuma ataupun pemberian dengan ikatan kerja (biasa disebut ikatan dinas) setelah selesainya pendidikan.

Beasiswa dapat diberikan oleh pemerintah, perusahaan swasta, lembaga pendidikan yang mengadakan kerjasama untuk melakukan tukar menukar tenaga pendidik yang diberi kesempatan untuk meningkatkan kapasitas sumber daya manusianya melalui pendidikan. Namun bisa juga beasiswa ini dapat diwujudkan dalam bentuk lain, misalnya buku-buku pelajaran atau fasilitas sekolah yang dapat menunjang proses belajar mengajar atau memperlancar pendidikannya tanpa adanya gangguan dalam hal keuangan hingga selesai proses pendidikannya.

Tujuan pemberian beasiswa menurut pedoman umum pemberian beasiswa adalah meningkatkan pemerataan dan kesempatan belajar bagi siswa yang mengalami kesulitan ekonomi, mendorong dan mempertahankan semangat belajar para siswa agar mereka dapat menyelesaikan pendidikan tepat waktu, mendorong meningkatkan prestasi belajar sehingga memacu kualitas pendidikan.

II.4. *Database*

Database adalah sekumpulan data yang saling berhubungan secara logis beserta deskripsinya, yang digunakan secara bersama-sama dan dirancang untuk memenuhi kebutuhan informasi disuatu tempat.

Sistem Database adalah suatu sistem yang terdiri dari kumpulan file atau tabel yang saling berhubungan yang memungkinkan beberapa pemakai mengakses dan memanpulasi file-file tersebut. (Melisa Elistri, dkk, 2014 : 106).

II.5. Normalisasi

Normalisasi merupakan parameter digunakan untuk menghindari duplikasi terhadap tabel dalam basis data dan juga merupakan proses mendekomposisikan sebuah tabel yang masih memiliki beberapa anomali aau ketidak wajaran sehingga menghasilkan tabel yang lebih sederhana dan struktur yang bagus, yaitu sebuah tabel yang tidak memiliki *data redundancy* dan memungkinkan *user* untuk melakukan *insert, delete,* dan *update* pada baris (*record*) tanpa menyebabkan inkonsistensi data. Terdapat lima bentuk normal yang biasa digunakan yaitu :

1. *First Normal Form*(1 NF)

Sudah tidak ada *repeating group* yaitu pengulangan yang terjadi pada beberapa atribut atau kolom dalam sebuah tabel, dan juga setiap atribut harus bernilai tunggal. Atribut *multivalued*, *composite*, *derive* tidak tunggal. Setiap nilai dari atribut hanya mempunyai nilai tunggal.

2. *SecondNormal Form* (2 NF)

Untuk menjadikan tabel normal tingkat ke 2 maka sudah 1NF dan setiap atribut yang bukan *primary key* sepenuhnya secara *fungsional* tergantung pada semua atribut pembentuk *primary key*.

3. *Third Normal Form* (3NF)

Tabel sudah 2NF dan tidak memiliki *transitive dependencies*, *Transitive dependencies* adalah ketika ada atribut yang secara tidak langsung tergantung pada *primary key* dan atribut tersebut juga tergantung pada atribut lainyang bukan *primary key*.

4. *Fourth Normal Form* (4NF)

Relasi berada pada bentuk normal keempat apabila memenuhi syarat BNCF dan tidak mempunyai *multivalue dependency*.

5. *Fifth Normal Form* (5NF)

Tabel bentuk normal kelima sering disebut PJNF (*Projection Join Normal Form*), penyebutan PJNF karena untuk suatu relasi akan berbentuk normal kelima jika tabel tersebut dapat dipecah atau diproyeksikan menjadi beberapa tabel dan dari proyeksi-proyeksi itu dapat disusun kembali (*join*) menjadi tabel yang sama dengan keadaan semula.

II.6. *Visual Basic* 2010

Visual Basic 2010 merupakan salah satu bagian dari produk pemograman yang dikeluarkan oleh *Microsoft*, yaitu *Micrsoft Visual Studio* 2010. Sebagai produk pengembangan atau *Integrated Deveopment Environment* (IDE) andalan yang dikeluarkan oleh *microsoft*, *Visual Studio* 2010 berisi beberapa IDE pemograman seperti *Visual Basic*, *Visual C++*, *Visual Web Developer, Visual C#*, dan *Visual F#.* Semua IDE tersebut sudah mendukung penuh implementasi *.Net Framwork* terbaru, yaitu *.Net Framework* 3.5. Adapun *database* standar yang disertakan adalah *Microsoft SQL Server 2008 Express.* (Mustakim, dkk, 2013 : 27)

II.7. SQL Server 2008

SQL Server adalah sebuah *database* relasional yang dirancang untuk mendukung aplikasi dengan arsitektur *client/server* dimana *database* terdapat pada komputer puat yang disebut *server,* dan informasi digunakan bersama-sama oleh beberapa user yang menjalankan aplikasi didalam komputer lokalnya yang disebut dengan *client* arsitektur semacam ini memberikan integritas data yang tinggi karena semua *user* bekerja dengan informasi yang sama. Melalui aturan-aturan bisnis, kendali diterapkan kepada semua user mengenai informasi yang ditambahkan ke dalam *database*. (Rika Yunitarini, 2013 : 47)

II.8. UML

Unified Modeing Language (UML) adalah sebuah bahasa pemodelan yang telah menjadi standard lam industri *software* untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak. Bahasa Pemodelan UML lebih cocok untuk pembuatan perangkat lunak dalam bahasa pemrograman berorientasi objek (C, Java, VB.NET), namun demikian tetap dapat digunakan pada bahasa pemrograman procedural. (Otto Fajarianto, 2016 : 55)

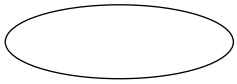
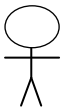
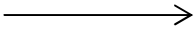
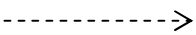
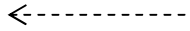
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut:

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.1 dibawah ini:

Tabel II.1. Simbol *Use Case*

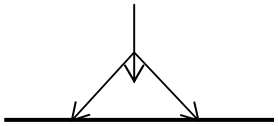
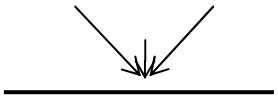
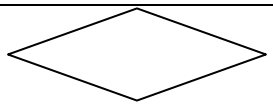
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. 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, Hal : 94)

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.2. 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.

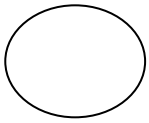
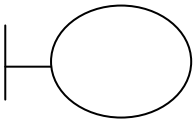
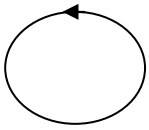
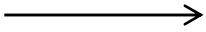
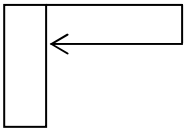
(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015, Hal : 94)

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.3 dibawah ini :

Tabel II.3. 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.
	*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, Hal : 95)

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.4 dibawah ini :

Tabel II.4. *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 minimal 2 maksimum 4

(Sumber : Gellysa Urva dan Helmi Fauzi Siregar; 2015, Hal : 95)