BAB II

TINJAUAN PUSTAKA

II.1. *Data Mining*

Data Mining adalah proses untuk mendapatkan informasi yang berguna dari gudang basis data yang besar. *Data Mining* juga dapat diartikan sebagai pengekstrakan informasi baru yang diambil dari bongkahan data besar yang membantu dalam pengambilan keputusan. Istilah *Data Mining* sering juga disebut *Knowledge Discovery*. Salah satu teknik yang dibuat dalam *Data Mining* adalah bagaimana menelusuri data yang ada untuk membangun sebuah model, kemudian menggunakan model tersebut agar dapat mengenali pola data yang lain yang yang tidak berada dalam basis data yang tersimpan. Kebutuhan untuk prediksi juga dapat memanfaatkan teknik ini. Dalam *Data Mining*, pengelompokan data juga bisa dilakukan. Tujuannya adalah agar kita dapat mengetahui pola universal data-data yang ada. Anomali data transaksi juga perlu dideteksi untuk dapat mengetahui tindak lanjut berikutnya yang dapat diambil. Semua hal tersebut bertujuan mendukung kegiatan operasional perusahaan sehingga tujuan akhir perusahaan diharapkan dapat tercapai.

Para ahli berusaha menetukan posisi bidang *Data Mining* diantara bidang-bidang yang lain. Hal ini dikarenakan ada kesamaan antara sebagian bahasan dalam *Data Mining* dengan bahasa dibidang lain. Memang tidak seratus persen sama, tetapi ada sejumlah kesamaan karakteristik dalam beberapa hal. Kesamaan bidang *Data Mining* dengan bidang statistik adalah penyampelan, estimasi, dan

pengujian hipotesis. Kesamaan dengan kecerdasan buatan *(Artificial Intelligence),* pengenalan pola, dan pembelajaran mesin adalah algoritma pencarian, teknik pemodelan dan teori pembelajaran. Bidang lain yang mempengaruhi *Data Mining* adalah teknologi basis data, yang mendukung penyediaan penyimpanan yang efisien, pengindeksan dan pemrosesan *query.* Teknik komputasi paralel sering digunakan untuk memberikan kinerja yang tinggi untuk ukuran set data yang besar, sedangkan komputasi terdistribusi dapat digunakan untuk menangani masalah ketika data tidak dapat disimpan disatu tempat.

Pekerjaan yang berkaitan dengan *Data Mining* dapat dibagi menjadi empat kelompok, yaitu Model Prediksi *(Prediction Modelling),* Analisis Kelompok *(Cluster Analysis),* Analisis Asosiasi *(Association Analysis)* dan Deteksi Anomali *(Anomali Detection).*

1. Model Prediksi *(Prediction Modelling)*

Model prediksi berkaitan dengan pembuatan sebuah model yang dapat melakukan pemetaan dari setiap himpunan variabel ke setiap targetnya, kemudian menggunakan model tersebut untuk memberikan nilai target pada himpunan baru yang didapat.

2. Analisis Kelompok *(Cluster Analysis)*

Analisis kelompok melakukan pengelompokan data-data kedalam sejumlah kelompok *(Cluster)* berdasarkan kesamaan karakteristik masing-masing data pada kelompok-kelompok yang ada. Data-data yang masuk dalam batas kesamaan dengan kelompoknya akan bergabung dalam kelompok tersebut, dan

akan terpisah dalam kelompok yang berbeda jika keluar dari batas kesamaan dengan kelompok tersebut.

3. Analisis Asosiasi *(Association Analysis)*

Analisi asosiasi digunakan untuk menemukan pola yang menggambar kan kekuatan hubungan fitur dalam data. Pola yang ditemukan biasanya merepresentasikan bentuk aturan implikasi atau subset fitur. Tujuannya adalah untuk menemukan pola yang menarik dengan cara yang efisien. Penerapan yang paling dekat dengan kehidupan sehari-hari adalah analisis data keranjang belanja.

4. Deteksi Anomali *(Anomali Detection).*

Pekerjaan deteksi anomali berkaitan dengan pengamatan sebuah data dari sejumlah data yang secara signifikan mempunyai karaktristik yang berbeda dari sisa data yang lain. Data-data yang karaktristiknya berbeda dari data yang lain disebut *Outlier.* Algoritma deteksi anomali yang baik harus harus mempunyai laju deteksi yang tinggi dan laju *Error* yang rendah. Deteksi anolami dapat diterapkan pada sistem jaringan untuk mengetahui pola data yang memasuki jaringan sehingga penyusupan bisa ditemukan jika pola kerja data yang datang berbeda. Perilaku kondisi cuaca yang mengalami anomali juga dapat dideteksi dengan algoritma ini.

II.2. *Market Basket Analysis*

Market basket analysis adalah salah satu cara yang digunakan untuk menganalisis data penjualan dari suatu perusahaan. Proses ini menganalisis perilaku pembelian konsumen dengan melihat asosiasi antar item-item yang

berbeda pada tiap transaksi pembelian (Han dan Kamber, 2006). Dengan melihat hubungan kegunaan item-item, beberapa diantaranya mudah diperkirakan *item* apa saja yang dibeli secata bersamaan, misalnya kopi dan gula, sehingga antisipasi ketersediaan kopi akan sama dengan gula. Namun mungkin saja ada pola pembelian *item* yang tidak diperkirakan sebelumnya, misalnya gula dan sabun yang tidak mempunyai hubungan sebagai barang substitusi atau komplementer, sehingga akan terjadi kesalahan dalam mengantisipasi ketersediaan gula atau sabun tersebut. Inilah salah satu manfaat yang dapat diperoleh dari melakukan *market basket analysis*. Dengan melakukan analisa ini, seorang manajer tidak perlu mengalami kesulitan untuk menemukan pola pembelian *item* apa saja yang mungkin dibeli secara bersamaan, karena data dari transaksi penjualan akan memberitahukannya sendiri. Dalam *Data Mining*, analisa ini dilakukan dengan *association rule*.

II.3. Penjualan

Penjualan merupakan tujuan utama dilakukannya kegiatan perusahaan. Perusahaan dalam menghasilkan barang/jasa, mempunyai tujuan akhir yaitu menjual barang/jasa tersebut kepada masyarakat. Oleh karena itu, penjualan memegang peranan penting bagi perusahaan agar produk yang dihasilkan oleh perusahaan dapat terjual dan memberikan penghasilan bagi perusahaan Penjualan yang dilakukan perusahaan bertujuan untuk menjual barang/jasa yang diperlukan sebagai sumber pendapatan untuk menutup semua ongkos guna memperoleh laba.

Penjualan adalah pemindahan hak milik barang atau pemberian jasa yang dilakukan penjualan kepada pembeli dengan harga yang disepakati bersama dengan jumlah yang dibebankan kepada pelangggan dalam penjualan barang/jasa dalam suatu periode akuntansi (Freddy Rangkuti ; 2010 : 206).

II.4. Algoritma Apriori

Algoritma apriori termasuk jenis aturan asosiasi pada *Data Mining*. Aturan yang menyatakan asosiasi antara beberapa atribut sering disebut *affinity analysis* atau *market basket analysis.* Analisis asosiasi atau *association rule mining* adalah teknik *Data Mining* untuk menemukan aturan suatu kombinasi *item*. Salah satu tahap analisis asosiasi yang menarik perhatian banyak peneliti untuk menghasilkan algoritma yang efisien adalah analisis pola frequensi tinggi(*frequent pattern mining*).

Penting tidaknya suatu asosiasi dapat diketahui dengan dua tolak ukur , yaitu : *support* dan *confidence*. Support (nilai penunjang) adalah persentase kombinasi *item* tersebut dalam database, sedangkan *confidence* (nilai kepastian) adalah kuatnya hubungan antar-*item* dalam aturan asosiasi.

1. Analisis Pola Frekuensi Tinggi dengan Algoritma Apriori

   Tahap ini mencari kombinasi *item* yang memenuhi syarat minimum dari nilai *support* dalam basis data. Nilai *support* sebuah *item* diperoleh dengan menggunakan rumus berikut :

$$Support\ A = \frac{Jumlah\ transaksi\ mengandung\ A}{Total\ transaksi}$$

   Sementara, nilai *support* dari 2 *item* diperoleh dengan menggunakan rumus :

$$Support\ (A,B) = P\ (A \cap B)$$

$$Support = \frac{\sum transaksi\ mengandung\ A\ dan\ B}{\sum transaksi}$$

Frequent itemset menunjukkan *itemset* yang memiliki frekuensi kemunculan lebih dari nilai minimum yang ditentukan (ꝏ). Misalkan ꝏ = 2, maka semua *itemsets* yang frekuensi kemunculannya lebih dari atau sama dengan 2 kali disebut *frequent*. Himpunan dari *frequent* k-*itemset* dilambangkan dengan Fk.

2. Pembentukan Aturan Asosiasi

   Setelah semua pola frekuensi tinggi ditemukan, barulah dicari aturan asosiasi yang memenuhi syarat minimum untuk *confidence* dengan menghitung *confidence* aturan asosiatif A → B. Nilai *confidence* dari aturan A → B diperoleh dengan rumus berikut :

$$Support = \frac{\sum transaksi\ mengandung\ A\ dan\ B}{\sum transaksi\ \ mengandung\ A}$$

   Untuk menentukan aturan asosiasi yang akan dipilih maka harus diurutkan berdasarkan *Support* × *Confidence*. Aturan diambil sebanyak *n* aturan yang memiliki hasil terbesar (Dewi Kartika Pane ; 2013 : 26).

II.5. Java

Bahasa pemrograman java adalah bahasa yang digunakan untuk menghasilkan aplikasi-aplikasi java. Pada umumnya, bahasa pemrograman hanya mendefinisikan sintaks dan perilaku bahasa. Pada saat program java dikompilasi, java akan dikonversi ke bentuk *bytecode*, yang merupakan bahasa mesin yang portable. Selanjutnya, *bytecode* tersebut dijalankan di Java *Virtual Machine* (atau

disebut Java VM atau JVM). Meskipun JVM dapat diimplementasikan langsung di perangkat kerjas, namun biasanya diimplementasikan dalam bentuk program perangkat lunak yang mengemulasi mesin (komputer) dan digunakan untuk menginterprestasi *bytecode*.

Platform dapat didefinisikan sebagai perangkat lunak pendukung aktivitas-aktivitas tertentu. *Platform Java* sendiri pada prinsipnya berbeda dengan bahasa Java atau JVM. *Platform Java* adalah himpunan kelas-kelas Java yang sudah didefinisikan sebelumnya dan eksis sejak instalasi Java. *Platform Java* juga mengacu pada lingkungan *runtime* atau API (*Application Programming Interface)* Java (Didik Dwi Prasetyo ; 2010 : 1).

II.6. *Database*

Database merupakan kumpulan data yang saling berhubungan, hubungan antar data dapat ditunjukkan dengan adanya *field* kunci dari setiap tabel yang beda. Dalam satu *file* atau tabel terdapat *record-record* yang sejenis, sama besar, sama bentuk, yang merupakan satu kumpulan entitas yang seragam. Satu *record* terdiri dari *field* yang saling berhubungan menunjukkan bahwa *field* tersebut satu pengertian yang lengkap dan disimpan dalam satu *record*. Basis data mempunyai beberapa kriteria penting yaitu :

1. Bersifat data oriented dan bukan program oriented.
2. Dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis data.
3. Dapat dikembangkan dengan mudah, baik *volume* maupun struktur.

4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah.
5. Dapat digunakan dengan cara-cara yang berbeda.

Prinsip utama *database* adalah pengaturan data dengan tujuan utama fleksibel dan kecepatan pada saat pengambilan data kembali. Adapun ciri-ciri basis data di antaranya adalah sebagai berikut :

1. Efesiensi meliputi kecepatan, ukuran dan ketepatan.
2. Data dalam jumlah besar.
3. Berbagi pakai (dipakai bersama-sama atau *sharebility*).

Mengurangi bahkan menghilangkan terjadinya duplikasi dan data yang tidak konsisten (Windu Gata ; 2013 : 19).

II.7. MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau yang dikenal dengan (*database management system*), database ini *multithread, multiuser.* MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU *General Public License (GPL).* Kekuatan MySQL tidak ditopang oleh sebuah komunitas, seperti Apache, yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh pemilik masing-masing, tetapi MySQL didukung penuh oleh pemilik sebuah perusahaan profesional dan komersial, yakni MySQL AB dari Swedia.

MySQL adalah *Relational Database Management System* (DBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Public Licensi*). Di mana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh

dijadikan produk turunan yang bersifat *closed source* atau komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu SQL (*Structured Query Language*).

Sebagai database server, MySQL dapat dikatakan lebih unggul dibandingkan database server lainnya, terutama dalam kecepatan. Berikut ini beberapa keistimewaan MySQL, antara lain :

1. Portability

   MySQL dapat berjalan stabil pada berbagai sistem operasi seperti windows, linuc, freeBSD, Mac Os X Server, Solaris, Amiga dan masih banyak lagi.

2. Multiuser

   MySQL dapat digunakan oleh beberapa user dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

3. Security

   MySQL memiliki beberapa lapisan sekuritas seperti level subnetmask, nama host, dan izin akses user dengan sistem perizinan yang mendetail serta password terenkripsi.

4. Scalability dan Limits

   MySQL mampu menangani database dalam skala besar, dengan jumlah records lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya (M Agus ; 2010 : 181).

II.8. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau ERD adalah alat pemodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antar entitas. Proses memungkinkan analis menghasilkan struktur basis data yang baik sehingga data dapat disimpan dan diambil secara efisien (Janner Simarmata ; 2010 : 67).

II.9. Kamus Data

Kamus data adalah suatu ensiklopedik dari informasi yang berkaitan dengan data perusahaan, atau dapat juga kita katakan bahwa kamus data adalah katalog atau *directory* yang berbasis komputer (*computer base catalog or directory*) yang berbasis data perubahan (metadata). Yang berkenaan dengan tahapan penjelasan data ini adalah sistem kamus data (*data description language/DDL*). Sistem kamus data berbentuk perangkat lunak yang fungsinya adalah penciptaan dan pemeliharaan serta menyediakan kamus data agar dapat digunakan. Kamus data dapat berbentuk kertas ataupun arsip (*file*) komputer (Ian Sommerville ; 2010 : 344).

II.10. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan

menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang tekah ditemukan.

II.10.1.Bentuk-bentuk Normalisasi

1. **Bentuk normal tahap pertama (1" Normal Form)**

   Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

2. **Bentuk normal tahap kedua (2^{nd} normal form)**

   Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

3. **Bentuk normal tahap ketiga (3^{rd} normal form)**

   Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

4. Boyce Code Normal Form (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

5. Bentuk Normal Tahap Keempat dan Kelima

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD).

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi *lossless* menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata, 2010 : 76).

II.11. UML (*Unified Modeling Language*)

Menurut Windu Gata (2013 : 4) 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.

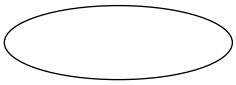
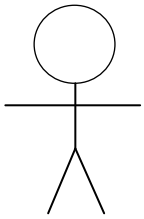
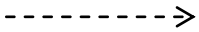
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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, yaitu :

Tabel II.1. Simbol *Use Case*

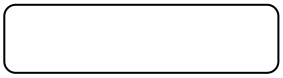
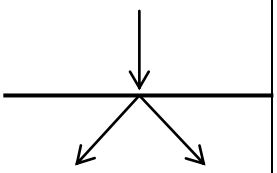
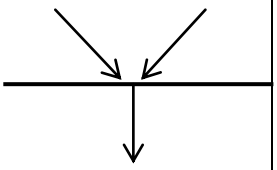
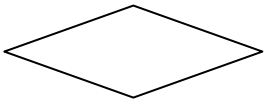
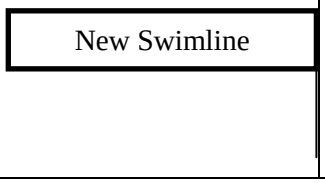
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi 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 : Windu Gata ; 2013 : 4)

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*, yaitu :

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 : Windu Gata ; 2013 : 6)

3. *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.

Tabel II.3. *Multiplicity Class Diagram*

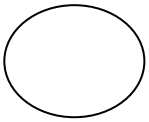
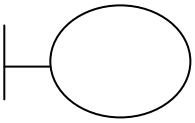
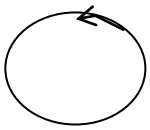
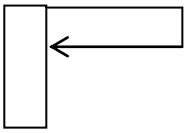
Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata ; 2013 : 9)

4. 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*, yaitu :

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class,* 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 : Windu Gata ; 2013 : 7)