**BAB 2**

**TINJAUAN PUSTAKA**

**2.1 Media Sosial Twitter**

Twitter merupakan suaatu media sosial yang dioperasikan oleh Twitter Inc., yang dimana menawarkan jaringan sosial mikroblog sehingga memungkinkan penggunannya mengirim dan membaca *tweet* [1]. Panjang *tweet* yang dapat diinput oleh pengguna Twitter saat ini 280 karakter. Berikut ini merupakan fitur yang ditawarkan oleh media sosial Twitter:

1. *Home*

Halaman *home* merupakan tempat dimana pengguna dapat melihat tweet dari akun lain yang diikuti. Isi dari *tweet* bisa berupa informasi, berita, gambar sesuai dengan keinginan dari pembuat *tweet* tersebut.

1. *Profile*

Halaman *profile* berisi *tweet* yang telah dibuat oleh suatu akun dan juga data diri yang telah dimasukkan oleh pemilik akun.

1. *Following*

Halaman *following* ini berisi kumpulan akun yang diikuti oleh suatu akun. Ketika suatu akun di*follow* maka *tweet* dari akun tersebut akan masuk ke halaman *home*.

1. *Follower*

Halaman *follower* ini berisi kumpulan akun yang mengikuti suatu akun. Ketika suatu akun memiliki *follower* maka jika akun itu menambahkan *tweet* maka *tweet* tersebut akan masuk ke halaman *home* akun yang ada pada halaman *follower*.

1. *Mention*

*Mention* merupakan fitur yang berfungsi untuk memanggil akun dari pengguna lain saat membuat suatu *tweet.*

1. *Favorite*

*Favorite* merupakan fitur yang dapat digunakan untuk menandai suatu *tweet* yang mungkin saja ingin dibuka dikemudian hari. Bisa juga digunakan untuk menandai suatu *tweet* yang disukai.

1. *Direct Message*

*Direct Message* merupakan fitur yang memungkinkan pengguna Twitter untuk bertukar pesan tanpa diketahui oleh pungguna lain.

1. *Hashtag*

*Hashtag* “#” yang ditulis di depan topik tertentu agar pengguna Twitter lainnya dapat mencari *tweet* yang memiliki topik yang sama.

1. *List*

Pengguna Twitter dapat mengelompokkan *follower* ke dalam satu grup sehingga memudahkan untuk dapat melihat informasi dari sesama anggota grup.

1. *Trending Topic*

Pengguna Twitter dapat melihat topik apa yang sedang ramai dibicarakan secara *realtime*.

**2.2 Twitter API**

*Application Programming Interface* (API) adalah sekumpulan prosedur dan protokol yang menggambarkan perilaku komponen perangkat lunak, seperti pustaka atau layanan jarak jauh, pengoperasian, *input*, dan *output* yang diizinkan. Para *developer* menggunakan Twitter API untuk membuat aplikasi, *website*, dan informasi lain yang berinteraksi dengan sosial media Twitter. Twitter API terdiri dari tiga bagian yaitu [14]:

1. *Search* API

Dirancang untuk memudahkan *user* dalam mengelola *query search* di konten Twitter. *User* dapat menggunakannya untuk mencari *keyword* berdasarkan kata khusus atau mencari *tweet* lebih spesifik berdasarkan *username* Twitter.

1. REST API

Twitter REST memberikan *core* data dan *core* Twitter *objects*. Memperbolehkan *developer* untuk mengakses inti dari Twitter seperti *timeline*, status *update*, dan informasi *user*. REST API digunakan dalam membangun sebuah aplikasi Twitter yang kompleks yang memerlukan inti dari Twitter.

1. *Streaming* API

*Streaming* API digunakan *developer* untuk kebutuhan yang lebih intensif seperti melakukan penelitian dan analisis data. *Streaming* API dapat menghasilkan aplikasi yang dapat mengetahui statistik status *update*, *follower*, dan lain sebagainya.

**2.3 *Machine Learning***

*Machine learning* merupakan isitilah yang diciptakan sekitar tahun 1960 yang terdiri dari kata mesin, yang berhubungan dengan komputer dan teknologi, dan belajar mengenai suatu kegiatan atau pola peristiwa [12]. Mesin mampu untuk belajar dengan mengikuti aturan yang dirancang oleh manusia. *Machine learning* merupakan bagian dari kecerdasan buatan (*Artificial Intelligence*). Tugas utama dari machine learning mengeksplorasi dan membangun algoritma yang daapt belajar dari data historis dan membuat prediksi pada data baru. Berdasarkan cara mempelajari data, *machine learning* terbagi menjadi tiga kategori, yaitu:

1. *Unsupervised Learning*

Data yang digunakan untuk pembelajaran atau *training* tanpa penjelasan apapun, memberi kebebasan untuk menemukan pola dari data, mengetahui informasi tersembunyi, atau untuk menentukan cara menguraikan data.

1. *Supervised Learning*

Data yang digunakan dalam pembelajaran atau *training* dilengkapi dengan deskripsi atau label. Tujuan akhirnya adalah untuk mencari aturan umum yang menentukan *input* menjadi *output*. Aturan yang dipelajari menghasilkan model yang dapat melakukan prediksi pada data baru.

1. *Reinforcement Learning*

*Reinforcement Learning* terjadi ketika kita menyajikan algoritma dengan contoh yang kekurangan label, tetapi kita dapat menyertakan contoh dengan *feedback*positif atau negatif bergantung pada solusi yang ditawarkan oleh algoritma tersebut.

**2.4 Analisis Sentimen**

Informasi yang berbentuk teks secara umum dapat dibagi menjadi informasi fakta dan opini [10]. Fakta adalah ekspresi obyektif terhadap suatu benda, kejadian dan kepunyaan benda tersebut. Opini bisaanya berupa ekspresi subyektif yang menggambarkan sentimen, penilaian, atau perasaan seseorang terhadap suatu benda, kejadian atau kepunyaan dari benda tersebut. [16] menjelaskan bahwa analisis sentimen adalah bagian dari pekerjaan yang meninjau segala sesuatu hal yang berhubungan dengan pendapat komputasi, sentimen dan subjektivitas teks. Ditambahkan oleh [5] bahwa analisis sentimen adalah alat untuk memproses koleksi hasil pencarian yang bertujuan dengan mencari atribut suatu produk dan proses memperolah hasil pendapatnya. Penelitian mengenai analisis sentimen telah berkembang sejak tahun 2003 dan merupakan bagian dari *text mining* yang merupakan penelitian komputasi berdasarkan sentimen, emoticon, pendapat, komentar dan setiap ekspresi yang diungkapkan oleh teks. Analisis sentimen difokuskan untuk melakukan klasifikasi berdasarkan polaritas. Berdasarkan klasifikasi, analisis sentimen dibagi menjadi dua kelompok utama. Yaitu dokumen klasifikasi ke pendapat atau fakta, atau dikenal sebagai klasifikasi subjektivitas dan dokumen klasifikasi ke dalam positif atau negatif, atau dikenal sebagai analisis sentimen [18].

Analisis sentimen dapat dibedakan berdasarkan sumber datanya, level yang sering digunakan dalam penelitian analisis sentimen adalah analisis sentimen pada level dokumen dan analisis sentimen pada level kalimat [6]. Berdasarkan level sumber datanya analisis sentimen terbagi menjadi *Coarse*-*Grained Sentiment Analysis* dan *Fined*-*Grained Sentiment Analysis.*

**2.4.1 *Coarse-Grained Sentiment Analysis***

*Coarse-Grained* merupakan proses analisis yang dilakukan pada level dokumen. Analisis sentiment jenis ini mencoba mengklasifikasikan orientasi sebuah dokumen secara keseluruhan. Orientasi ini ada tiga jenis yaitu positif, netral, dan negatif. Akan tetapi, ada juga yang menjadikan nilai orientasi ini bersifat kontinu atau tidak diskrit [6].

**2.4.2 *Fined-Grained Sentiment Analysis***

Obyek yang ingin diklasifikasi pada *fined-grained* bukan berada pada level dokumen melainkan sebuah kalimat pada suatu dokumen. Orientasi ini ada dua jenis yaitu positif dan negatif. Informasi berupa paper, artikel, jurnal dan lain sebagainya berisikan informasi tekstual yang bersifat objektif, yaitu informasi yang tidak menggambarkan perasaan atau pendapat penulis terhadap hal yang ditulisnya. Infomasi-informasi objektif tersebut cenderung bersifat netral, sedangkan informasi seperti opini, komentar atau pendapat masing-masing memiliki nilai atau makna yang diungkapkan oleh penulisnya, makna tersebut dapat bernilai positif atau negatif. Nilai positif dan negatif yang terkandung dalam kalimat subjektif ini disebut dengan polaritas. Kalimat opini, komentar atau pendapat yang menggambarkan penilaian atau perasaan seseorang terhadap suatu hal merupakan sentimen. Polaritas sentimen ini yang menjadi objek penelitian yang akan dilaksanakan [6].

**2.5 *Text Mining***

*Text mining* adalah penambangan yang dilakukan oleh komputer untuk mendapatkan sesuatu yang baru, sesuatu yang tidak diketahui sebelumnya atau menemukan informasi yang ditampilkan secara tersirat, yang berasal dari informasi yang diekstrak secara otomatis dari sumber data dalam bentuk teks yang berbeda-beda [17].

Pada dasarnya proses kerja dari *text mining* mengadopsi dari penelitian *data mining* namun yang menjadi perbedaan adalah pola yang digunakan oleh *text mining* diambil dari sekumpulan bahasa alami yang tidak terstruktur sedangkan *data mining* dari *database* yang terstruktur [7].

**2.6 *Text Preprocessing***

Pada saat melakukan proses *text mining*, data yang dihasilkan berisi data-data dengan struktur yang sembarang. Oleh karena itu, diperlukan proses pengubahan agar data yang didapat menjadi data yang terstruktur. Proses ini sering disebut dengan *text preprocessing* [17]. Setelah data menjadi data yang terstruktur maka data dapat dijadikan sebagai sumber data yang dapat diolah lebih lanjut. Beberapa proses yang dilakukan dalam *text preprocessing* adalah sebagai berikut:

1. *Case Folding*

*Case Folding* adalah mengkonversi keseluruhan teks dalam dokumen menjadi suatu bentuk standar dan biasanya bentuk standar tersebut adalah huruf kecil atau *lowercase* [17]*.*

1. *Cleaning*

Proses ini dilakukan untuk membersihkan kata-kata dari hal yang tidak diperlukan dan dapat menimbulkan *noise*. Hal yang dihilangkan pada proses ini adalah angka, *emoticon*, *hashtag*, *username* (@*username*), url suatu *website.*

1. *Stopword Removal*

Tahap *stopword removal* adalah tahap mengambil kata-kata yang penting dari hasil *tokenizing*. Algoritma yang biasa digunakan pada proses ini adalah algoritma *stoplist* (membuang kata yang tidak penting) atau *wordlist* (menyimpan kata yang penting). *Stoplist atau stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan *bag*-*of*-*words* [17].

**2.7 *Negation Handling***

Kata-kata negasi (tidak, bukan, belum, jangan) tidak hanya digunakan untuk mengungkapkan penolakan atau hukuman, tetapi juga dalam hubungannya dengan pertanyaan ya atau tidak. Dari survei literatur, banyak penelitian telah dilakukan sehubungan dengan masalah klasifikasi sentimen pada pendekatan level kalimat atau frasa, termasuk menangani negasi dalam subtugas analisis sentimen. Ini menjadi penting karena, dalam metode tradisional unigram seperti itu, dua kalimat dapat berakhir dengan representasi yang sama, misalnya, kalimat “responnya cepat sekali” dengan “responnya gak cepat sama sekali”. Dua kalimat ini dapat direpresentasikan dengan vektor fitur yang sama, dan dapat menyebabkan kesalahan klasifikasi [2].

Dalam Naïve bayes diasumsikan setiap kata dalam dokumen digunakan sebagai fitur terpisah, misalnya kata “baik” dalam frase “tidak baik” akan memberikan kontribusi sentimen positif daripada sentimen negatif karena kata “tidak baik” merupakan fitur yang tepisah sehingga memberikan kontribusi peluang sentimen masing-masing daripada memberikan satu kontribusi peluang yaitu negatif [3]. Jika ada kata “tidak” akan digabungkan dengan kata selanjutnya menjadi “tidak\_” + kata. Kata tersebut tidak akan digabungkan jika kata setelah “tidak” adalah tanda baca atau kata “tidak” lagi [3].

**2.8 Klasifikasi**

Klasifikasi merupakan proses untuk menentukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek. Oleh karena itu kelas yang ada lebih dari satu dan suatu dokumen akan diklasifikasikan kedalam suatu kelas.

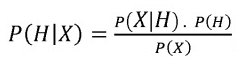
**2.9 *Lexicon Based***

Leksikon sentimen merupakan kumpulan kata yang mengekspresikan sentimen positif atau negatif [11]. *Lexicon based* dapat mengidentifikasi sentimen dari setiap kata beropini yang terdapat pada kalimat dan dapat menangani masalah multiopini di dalam suatu kalimat. *Lexicon based* menggunakan *dictionary* atau kamus *lexicon* untuk melakukan penilaian terhadap kata. Makna kata dapat berubah bergantung pada konteks dari kalimat, oleh karena itu, pendekatan *lexicon based* terkadang tidak bisa menangkap makna sebenarnya dari kata yang diprosesnya. Akan tetapi, *lexicon based* memiliki performa klasifikasi yang baik pada kasus lintas domain, dan *knowledge* dapat ditambahkan kapan saja kedalam *dictionary*[9]. Klasifikasi berbasis leksikon mengacu pada aturan klasifikasi di mana dokumen diberi label berdasarkan hitungan kata-kata dari leksikon yang terkait dengan setiap label [21].

**2.10 Naive Bayes**

Naive Bayes merupakan pengklasifikasian dengan metode probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya [4]. Keuntungan penggunaan metode Naive Bayes adalah bahwa metode ini hanya membutuhkan jumlah data pelatihan yang sedikit untuk menentukan estimasi parameter yang diperlukan dalam proses klasifikasi. Metode Naive Bayes sering bekerja lebih baik dalam kebanyakan situasi dunia nyata yang kompleks dari pada yang diharapkan.

Berikut merupakan persamaan dari teorema Bayes:

(2.1)

Keterangan:

X : Data dengan class yang belum diketahui

H : Hipotesis data merupakan suatu class spesifik

P(H|X) : Probabilitas hipotesis H berdasarkan kondisi X

P(H) : Probabilitas hipotesis H

P(X|H) : Probabilitas X berdasarkan kondisi pada hipotesis H

P(X) : Probabilitas X

**2.10.1 Multinomial Naive Bayes**

Multinomial Naive Bayesmerupakan metode yang digunakan untuk melakukan klasifikasi berdasarkan pada probabilitas yang dikemukakan oleh ilmuan Inggris Thomas Bayes. Klasifikasi dilakukan dengan memprediksi peluang dimasa depan berdasarkan pengalaman pada periode sebelumnya sehingga dikenal dengan teorema *Bayes*. Metode ini mengasumsikan bahwa fitur yang ada tidak memiliki keterkaitan satu sama lain. Secara teoritis Multinomial Naive Bayes bekerja lebih baik dibandingkan dengan metode klasifikasi lainnya. Metode Multinomial Naive Bayes terdiri dari dua tahap, yaitu tahap pembelajaran dan tahap klasifikasi.

Proses pembelajaran dimulai dengan memasukkan data latih yang digunakan untuk pembelajaran. Kemudian mulai untuk pembentukan daftar kata-kata. Daftar kata-kata merupakan kumpulan kata unik yang berasal dari data latih.

Kemudian menghitung prior atau peluang kemunculan suatu kategori pada semua dokumen latih dapat dilakukan dengan menggunakan persamaan berikut [8]:



(2.2)

Keterangan:

Nc : Banyaknya kategori c pada dokumen latih

Ndoc : Banyaknya keseluruhan dokumen latih yang digunakan

Dilanjutkan dengan menghitung peluang sebuah kata i masuk ke dalam kategori atau kelas tertentu dapat dilakukan dengan menggunakan persamaan berikut [8]:



(2.3)

Count(wi,c) merupakan jumlah kata tertentu yang muncul dalam suatu kategori atau kelas. Penambahan nilai satu berfungsi agar tidak menghasilkan probabilitas bernilai 0, modifikasi ini disebut dengan *smoothing*. Terdapat laplace *smoothing* jika nilai *smoothing* adalah satu dan terdapat *add*-k *smoothing* jika nilai *smoothing* tidak bernilai satu [8]. ∑wɛVcount(w,c) merupakan jumlah seluruh kata pada kelas. Sedangkan |V| merupakan jumlah seluruh kata unik diseluruh kelas [8]. Setelah tahap pembelajaran selesai dilakukan maka tahap selanjutnya adalah tahap klasifikasi data baru berdasarkan hasil pembelajaran. Berikut ini merupakan rumus yang digunakan dalam melakukan klasifikasi data baru [8]:



(2.4)

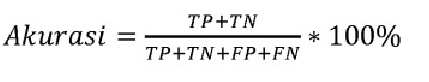
Setelah diperoleh hasil perhitungan untuk setiap kategori, maka hasil klasifikasi data baru ditentukan berdasarkan nilai tertinggi (argmax) [8].

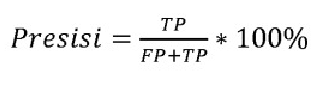
**2.11 *Confusion Matrix***

*Confusion matrix* merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi. Pada dasarnya, *confusion matrix* mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya.

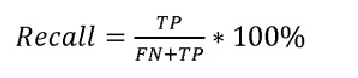
Pada pengukuran kinerja *confusion matrix* terdapat empat istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). Nilai *True Negative* (TN) merupakan jumlah data negatif yang terdeteksi secara benar, sedangkan *False Positive* (FP) merupakan jumlah data negatif namun terdeteksi sebagai data positif. Sementara itu, *True Positive* (TP) merupakan jumlah data positif yang terdeteksi benar, sedangkan *False Negative* (FN) merupakan jumlah data positif namun terdeteksi sebagai negatif [7].

Berdasarkan nilai *True Negative* (TN), *True Positive* (TP), *False Negative* (FN) dan *False Positive* (FP) dapat diperoleh nilai akurasi, presisi dan *recall*. Nilai akurasi menggambarkan seberapa akurat sistem dapat mengklasifikasikan data secara benar. Nilai presisi menggambarkan jumlah data kategori positif yang diklasifikasikan secara benar dibagi dengan total data yang diklasifikasikan positif. Sementara itu, *recall* menunjukkan berapa persen data kategori positif yang terklasifikasi dengan benar oleh sistem .



(2.5)

(2.6)

(2.7)

**2.12 Bahasa Pemrograman Python**

Bahasa pemrograman Python di-*release* pertama kali pada tahun 1991 oleh Guido van Rossum di *Scitchting Mathematisch Centrum* Belanda. Guido menggunakan nama Python karena ia merupakan penggemar grup komedi Inggris bernama Monty Python. Bahasa pemrograman Python bersifat *open source* dengan sebagian besar versinya menggunakan lisensi *GFL-compatible* [23].

Bahasa pemrograman Python terbilang baru dibandingkan dengan bahasa-bahasa pemrograman *mainstream* seperti Bahasa C, Visual Basic, Java atau PHP. Tetapi Python berkembang dengan pesat karena memiliki berbagai kelebihan seperti aspek *readibility*, multifungsi, interoperabilitas dan juga dukungan komunitas yang memadai. Berikut ini merupakan kelebihan bahasa pemrograman Python dibandingkan dengan bahasa pemrograman lainnya [23].

1. Keterbacaan atau *Readibility*

Pengembang bahasa pemrograman Python menekankan aspek *readibility* pada sintaks serta kode-kode dasar programnya. Itu berarti bahwa Python mudah dibaca dan dipahami. Hal ini tentu akan memudahkan pengembangan aplikasi, mulai dari tahap *coding*, *testing*, perbaikan jika ada kesalahan, *bug* atau *error* lainnya, serta mempermudah pemeliharaan dalam operasionalisasinya [23].

1. Efisien

Memiliki aspek keterbacaan yang baik dan *programmer friendly*, memberikan efisiensi bagi programmer dalam membuat program. Python juga memiliki *library* yang lengkap sehingga untuk mendapatkan program yang sama, Python *code* akan lebih simpel dibandingkan dengan kode yang ditulis dengan bahasa pemrograman lainnya. Hal ini tentu akan meningkatkan produktivitas dan juga menghemat waktu pembuatan aplikasi bagi para *programmer* [23].

1. Multifungsi

Python merupakan bahasa pemrograman multifungsi. Artinya, dengan menggunakan Python, kita bisa membuat *website*, membuat aplikasi jaringan, pengembangan aplikasi bidang robotika, sampai pada pengembangan aplikasi kecerdasan buatan. Hal itu dikarenakan bahasa pemrograman ini memiliki banyak dukungan *library*. Python memiliki banyak modul yang siap dipakai untuk pembuatan aplikasi [23].

1. Interoperabilitas

Python memiliki interoperabilitas yang tinggi, karena mampu berinteraksi dengan bahasa-bahasa pemrograman lain. Kode-kode program menggunakan Python dipanggil oleh bahasa lain, demikian juga sebaliknya kode bahasa pemrograman lainnya bisa dipanggil dari bahasa pemrograman Python ini. Selain itu, program yang dikembangkan menggunakan bahasa Python dapat digunakan pada hampir semua sistem operasi baik Windows, Linux, Mac OS, Unix, dan juga sistem operasi pada perangkat berbasis *mobile* seperti Android atau IOS [23].

1. Dukungan Komunitas

Sebagaimana program-program *open source*, Python memiliki dukungan komunitas yang sangat kuat. Dukungan inilah yang sampai saat ini membuat Python terus berkembang [23].

**2.12.1 Scikit-learn**

Bersumber dari [19], modul scikit-learn merupakan *machine learning library* yang ditulis dalam Bahasa pemrograman Python. Modul ini menyediakan banyak sekali algoritma dan dataset yang dapat digunakan dalam pembuatan *machine learning*. Scikit-learn menyediakan beberapa kelas untuk proses kategorisasi dokumen, seperti CountVectorizer dan TfidfTransformer. Kelas CountVectorizer berfungsi untuk mengubah dokumen menjadi representasi vektornya dengan pembobotan *term* *frequency*. Untuk membangun *classifier*, *library* ini menyediakan kelas seperti MultinomialNB (implementasi *classifier* Multinomial Naïve Bayes) dan DecisionTreeClassifier (implementasi *classifier* Decision Tree).

**2.13 *Flowchart***

*Flowchart* adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses dengan proses yang lain nya dalam suatu program [24]. *Flowchart* dibagi menjadi dua model, yaitu:

1. Sistem *Flowchart*, yaitu urutan proses dalam sistem dengan alat media *input*, *output* serta media penyimpanan yang digunakan dalam proses pengolahan data.
2. Program *Flowchart*, yaitu simbol-simbol tertentu yang digunakan untuk menggambarkan urutan proses secara detail dan hubungan antara suatu proses dengan proses lainnya dalam suatu program.

Simbol dalam diagram *flowchart* dibagi menjadi tiga, yaitu:

1. *Input* atau *output*, simbol yang digunakan untuk menggambarkan peralatan yang digunakan sebagai media *input* atau *output*. Simbol *input* atau *output* dapat dilihat pada table 2.1.

**Tabel 2.1 Simbol Input Output Flowchart**

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | Simbol *display*, menunjukkan *output* dalam layar monitor. |
|  | Simbol dokumen, mencetak *output* dalam bentuk dokumen. |
|  | Simbol *disk storage*, menunjukkan input berasal dari *disk* atau *output* disimpan ke *disk*. |
|  | Simbol *magnetic tape*, menunjukkan input berasal dari pita magnetik atau output disimpan pada pita magnetik. |
|  | Simbol *punched card*, menunjukkan input berasal dari kartu atau output ditulis ke kartu. |
|  | Simbol *input* atau *output*, menunjukkan proses *input* atau *output*. |

1. *Processing*, simbol yang digunakan untuk menggambarkan jenis operasi pengolahan dalam suatu prosedur. Simbol *processing* terdapat pada tabel 2.2.

**Tabel 2.2 Simbol Processing Flowchart**

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | Simbol proses, menyatakan suatu proses yang dilakukan. |
|  | Simbol manual, menyatakan suatu proses yang dilakukan secara manual. |
|  | Simbol decision, menunjukkan kondisi tertentu yang memiliki dua kemungkinan jawaban. |
|  | Simbol terminal, menunjukkan awal atau akhir suatu program. |
|  | Simbol *offline storage*, menunjukkan data pada simbol ini akan disimpan pada media yang *offline*. |

1. *Flow Direction*, simbol yang digunakan sebagai penghubung antar simbol. Simbol yang menggambarkan *flow direction* dapat dilihat pada tabel 2.5.

**Tabel 2.3 Simbol Flow Direction Flowchart**

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | Simbol arus, menunjukkan alur dari diagram. |
|  | Simbol *connector*, berfungsi untuk sambungan dari suatu proses dalam halaman yang sama. |
|  | Simbol *offline connector*, berfungsi untuk sambungan dari suatu proses dalam halaman yang berbeda. |