**BAB II**

**LANDASAN TEORI**

**2.1 Tinjauan Studi**

Penelitian tentang klasifikasi prestasi siswa telah banyak dilakukan oleh penulis terdahulu antara lain :

1. Penelitian yang dilakukan oleh (Amjad Abu Saa, 2016 : 212) dengan judul “ Educational Data Mining & Students’ Performance Prediction” Penelitian ini bertujuan untuk mengidentifikasi hubungan antara faktor personal sosial siswa dan kinerja pendidikan mereka di semester sebelumnya dengan menggunakan *data mining.* dalam penelitian ini, empat algoritma *decision tree* digunakan pada data siswa yang terkumpul, yaitu pohon keputusan *C4.5*, pohon keputusan *ID3*, keputusan *CART Tree*, dan *CHAID*. Algoritma C4.5 mampu memprediksi kelas 95 objek dari 270, yang memberikan nilai akurasi 35,19 %. algoritma *ID3* mampu memprediksi kelas dari 90 objek dari 270, yang memberikan nilai akurasi 33,33%. Algoritma *CART* mampu memprediksi kelas 108 objek dari 270, yang memberikan nilai akurasi 40%. Algoritma *CHAID* mampu memprediksi kelas dari 92 objek dari 270, yang memberikan nilai akurasi sebesar 34,07%.
2. Penelitian yang dilakukan oleh (Sujana, 2010 : 25-28) dengan judul “ Aplikasi Data Mining Data Mahasiswa Dengan Metode Klasifikasi *Decision Tree*”. Penelitian ini bertujuan untuk melihat apakah IPK seorang mahasiswa dapat diperkirakan berdasarkan nilai beberapa mata kuliah yang dianggap paling signifikan dalam menentukan IPK seorang mahasiswa*.* Dalam penelitian ini dua algoritma *decision tree* yang digunakan algoritma *ID3* dan *C4.5,* algoritma *C4.5* adalah ekstensi dari algoritma *ID3.*Secara rekursif membuat sebuah *decision tree* berdasarkan training data yang telah disiapkan. Algoritma ini mempunyai inputan berupa *training samples* dan *samples.* Hasil hasil uji dapat dilihat prosentase dari beberapa matakuliah terhadap nilai IPK seorang mahasiswa. Prosentase *error rate* yang dihasilkan pada hasil *testing* rata-rata adalah dibawah 50% bahkan ada yang 26% itu menandakan bahwa *rule* yang dihasilkan sudah cukup baik. Hasil tersebut diperoleh dari data *treining* pada matakuliah Algoritma I, Algoritma II, Basis Data, Statistika dan Struktur Data. Dari data treining tersebut teryata menghasilkan *rule* yang digunakan untuk data testing.
3. Penelitian yang dilakukan oleh (Fandy Ferdian Harryanto dkk, 2017) dengan judul “Penerapan Algoritma C4.5 untuk memprediks penerimaan calon pegawai baru di PT. WISE ”. Penelitian ini bertujuan untuk mengetahui kualitas orang-orang yang memiliki potensi baik sebagai pegawai pada suatu perusahaan. Oleh karena itu diperlukan cara atau metode untuk mengidentifikasi calon pegawai dengan menggunakan algoritma C4.5 berdasarkan data-data yang sudah ada. Berdasarkan metode pengukuran akurasi *ten-folel cross validation.* Dari hasil pengukuran tingkat keberhasilan prediksi calon pegawai baru sebesar 71%.

**2.2 Tinjauan Pustaka**

**2.2.1 Prestasi Siswa**

Menurut Kamus Besar Bahasa Indonesia (2005 : 895) prestasi belajar adalah (1) penguasaan pengetahuan atau keterampilan yang dikembangkan oleh mata pelajaran, lazimnya ditunjukkan dengan nilai tes atau angka nilai yang diberikan guru (2) prestasi belajar adalah kemampuan yang sungguh-sungguh ada atau dapat di amati yang dapat di ukur langsung dengan tes tertentu.Prestasi berasal dari bahasa belanda, yang berarti hasil bisnis. Prestasi dapat dicapai dengan mengandalkan kemampuan intelektual, emosonal, dan spritual serta ketahanan dalam menghadapi semua aspek situasi kehidupan. Prestasi belajar adalah hasil dari seseorang dalam kegiatan pembelajaran (Siti Pratini, 2005). Prestasi adalah sebagai rumus yang diberikan guru mata pelajaran mengenai kemajuan atau prestasi belajar selama periode tertentu.(Sumadi Suryabrata,1998).

Berdasarkan beberapa pendapat dijelaskan bahwa faktor-faktor yang mempengaruhi prestasi belajar adalah sebagai berikut:

1. Faktor internal, yaitu faktor yang berasal dari dalam diri siswa yang meliputi faktor fisiologis dan faktor psikologis. Faktor fisiologis ini menyangkut kondisi jasmani/kondisi fisik siswa selama belajar. Sedangkan faktor psikologis meliputi aspek:
2. Minat belajar siswa. Minat belajar yang besar cenderung menghasilkan prestasi yang tinggi, sebaliknya minat belajar yang kurang akan menghasilkan prestasi belajar yang rendah.
3. Kecerdasan/intelegensi. Seseorang yang memiliki intelegensi yang baik umumnya mudah belajar dan hasilnya pun cenderung baik.
4. Motivasi belajar
5. Bakat siswa
6. Kemampuan kognitif siswa
7. Sikap siswa terhadap mata pelajaran.
8. Faktor eksternal, yaitu faktor yang berasal dari luar diri siswa yang meliputi lingkungan fisik dan sosial serta instrumen yang berupa kurikulum, program, metode mengajar, guru, sarana dan fasilitas.

Namun berdasarkan penelitian yang dilakukan oleh Didik Setyadi pada tahun 2012 di jelaskan bahwa data pada rapor siswa merupakan data yang diakui oleh sekolah untuk informasi potensi akademik siswa yang dijadikan sebagai data yang akan di proses untuk menentukan klasifikasi siswa yang berpotensi dan belum berpotensi akademik sehingga dalam penelitian ini variabel yang dijadikan untuk prediksi yaitu :

1. Variabel kategori adalah kompetensi pengetahuan, kompetensi keterampilan.
2. Variabel target adalah prestasi yang paling menonjol dari siswa tersebut. Seperti nilai tertinggi pada masing-masing siswa kelas XII.

Variabel tersebut dijadikan sebagai data yang akan di proses untuk menentukan klasifikasi siswa yang berpotensi dan belum berpotensi sehingga dalam penelitian ini penulis melakukan klasifikasi prestasi siswa yang berprestasi dengan menggunakan metode data mining.Ada beberapa macam prestasi yang dapat di capai oleh setiap orang, diantaranya prestasi belajar adalah hasil yang di peroleh untuk usaha untuk belajar, prestasi siswa misalnya disekolah menjadi juara umum setiap tahun. Prestasi kerja adalah hasil yang diperoleh dari usaha kerja yang telah dilakukan, misalnya promosi kerja kerja mereka selama bertahun-tahun. Prestasi seni adalah hasil yang diperoleh dari bisnis seni, misalnya pencapaian penyanyi atau bentuk lain dari seniman upeti. Prestasi olahraga adalah hasil yang diperoleh untuk usaha dan kerja keras di bidang olahraga, contonya seorang atlet mendapat medali emas ditempat pertama diraih saat mengahdiri pekan olahraga nasional (PON). Prestasi lingkungan hidup adalah sebuah prestasi yang oleh upaya untuk menyelamatkan lingkungan, misalnya individu atau kelompok mendapatkan penghargaan untuk upaya konservasi lingkungan seperti penanaman pohon atau penghijauan. Pada tabel berikut adalah atribut yang akan digunakan dalam klasifikasi prestasi belajar siswa.

Tabel 2.1 Atribut Siswa

|  |  |  |
| --- | --- | --- |
| No | Atribut | Nilai Atribut |
|  | Jenis Kelamin | Laki-laki  Perempuan |
|  | Agama | Muslim  Non-Muslim |
|  | Pekerjaan Ayah | PNS/TNI/Polri  Petani/Nelayan/Buruh  Pegawai Swasta  Wiraswasta |
|  | Pekerjaan Ibu | Pegawai Swasta/PNS  Wiraswasta  Tidak Bekerja |
|  | Nilai UN Matematika | < 40  40 - 60  > 60 |
|  | Nilai UN Bahasa Indonesia | < 40  40 - 60  > 60 |
|  | Nilai UN IPA | < 40  40 - 60  > 60 |
|  | Penghasilan Ortu | < Rp. 1,000,000  Rp 1,000,000 – Rp 2,000,000  > Rp 2,000,000 |
|  | Klasifikasi | Berprestasi, Tidak |

(sumber: Smkn3gorontalo, 2018)

**2.2.2 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 sangat bergantung pada tujuan dan proses KDD secara keseluruhan. Data mining mampu menganalisa data yang beşar menjadi informası berupa pola yang mempunyai arti bagi pendukung keputusan.

Hasil dari aplikasi data mining tersebut dievaluasi untuk menemukan informasi/pengetahuan baru yang menarik dan bernilai bagi perusahaan, dan kemudian divisualisasikan agar mempermudah bagi user memilih informasi-informasi yang mempunyai arti bagi pendukung keputusan. Salah satu proses dalam data mining adalah klasifikasi, pada klasifikasi diberikan sejumlah record yang dinamakan *trainingşet* yang terdiri dari beberapa atribut, salah satu atribut menunjukkan kelas untuk *record*. Tujuan dari klasifikasi adalah untuk menemukan model dari *training set* yang membedakan *record* kedalam kategori atau kelas yang sesuai, model tersebut kemudian digunakan untuk mengklasifikasikan *record* yang kelasnya belum diketahui sebelumnya. Salah satu metode yang digunakan di dalam klasifıkasi adalah pengklasifikasian dengan menggunakan *decision tree* (pohon keputusan). Data Mining sebuah proses untuk menemukan pola atau pengetahuan yang bermanfaat secara otomatıs dari sekumpulan data yang berjumlah banyak. Data mining sering dianggap sebagai bagian dari *Knowledge Discovery in Database (KDD)* yaitu sebüah mencari pengetahuan yang bermanfaat dan secara garis beşar 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 unluk proses data mining, disimpan dalam suatu 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 cleaning mencakup antara lain membuang duplikasi data, memeriksa data yang inkonsisten, dan memperbaiki kesalahan pada data, seperti kesalahan cetak (tipografi). Selain itu dilakukan proses *enrichment*, yaitu proses "memperkaya” data yang sudah ada dengan data atau informasi lain yang relevan dan diperlukan untuk KDD, seperti data atau informasi eksternal.

1. ***Transformation***

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

## *Interpretation/EvaIuation*

Pola informasi 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 dengan *interpretation.* Tahap ini mencakup pemeriksaan apakah pola atau informasi yang ditemukan bertentangan dengan fakta atau hipotesa yang ada sebelumnya.

**2.2.2.1 Kelompok Data *Mining***

Berdasarkan tugasnya, data *mining* dikelompokkan menjadi (Larose, 2005):

1. Deskripsi

Mencari cara untuk menggambarkan pola dan *trend* yang terdapat dalam data. Sebagai contoh, seorang pengumpul suara mengungkap bukti bahwa mereka yang diberhentikan dari jabatannya saat ini, akan kurang mendukung dalam pemilihan Presiden. Untuk deskripsi ini bisa dilakukan dengan *exploratory data analysis*, yaitu metode grafik untuk menelusuri data dalam mencari pola dan *tren*. (Larose, 2005)

2. Estimasi

Estimasi mirip seperti klasifikasi tapi *variable* sasaran adalah *numeric*. Model dibuat menggunakan *record* yang lengkap, juga ada *variable* targetnya. Kemudian untuk data baru, estimasi nilai *variable* target dibuatberdasarkan nilai prediktor. Contoh, untuk estimasi tekanan darah pada pasien, *variable* prediktornya umur, jenis kelamin, berat badan, dan tingkat sodium darah. Hubungan antara tekanan darah, dan *variable* prediktor pada data *training* akan menghasilkan model kemudian diaplikasikan pada data baru. Untuk melakukan estimasi bisa digunakan *neural network* atau metode*statistic*seperti*point estimation* dan *conﬁdenceinter valestimations*, *simple linear regression* dan  *correlation*, dan *multiple regression* (Larose, 2005).

3. Prediksi

Prediksi mirip seperti klasifikasi dan estimasi, tapi hasilnya untuk memprediksi masa depan. Contoh, memprediksi harga barang tiga bulan mendatang, memprediksi presentasi kenaikan angka kematian karena kecelakaan tahun mendatang jika kecepatan berkendara dinaikkan. Metode dan teknik untuk klasifikasi dan estimasi, jika cocok, bisa juga digunakan untuk prediksi, termasuk metode statistik. Algoritma untuk prediksi antara lain *regression tree* dan model *tree* (Han, 2006).

4. Klasifikasi

Dalam klasifikasi, sasarannya adalah *variable* kategori, misalkan atribut penghasilan, yang bisa dikategorikan menjadi tiga kelas atau kategori yaitu, tinggi, sedang, dan rendah. Model data *mining* membaca sejumlah besar *record* tiap *record* berisi informasi pada *variable* target. Contoh, dari sebuah data set misalkan mau mengklasifikasikan penghasilan seseorang yang datanya tidak terdapat pada dataset, berdasarkan karakteristik yang berhubungan dengan orang itu seperti, umur, jenis kelamin, dan pekerjaan. Tugas klasifikasi ini cocok untuk metode dan teknik data *mining*. Algoritma akan mengolah dengan cara membaca data set yang berisi *variable predictor* dan *variable* target yang telah diklasifikasikan, yaitu penghasilan. Di sini algoritma (*software*) “mempelajari” kombinasi *variable* mana yang berhubungan dengan penghasilan yang mana. Data ini disebut *training* set. Kemudian algoritma akan melihat ke data baru yang belum termasuk klasifikasi manapun. Berdasarkan klasifikasi pada data set, kemudian algoritma akan memasukkan data baru tersebut ke dalam klasifikasi yang mana. Misalkan seorang professor wanita berusia 63 tahun bisa jadi diklasifikasikan ke dalam kelas penghasilan tinggi. Algoritma klasifikasi yang banyak digunakan secara luas untuk klasifikasi antara lain, *decision tree, bayesian classifier,* dan *neural network* (Gorunescu, 2011)

5.  *Clustering*

*Clustering* mengacu pada pengelompokkan *record-record*, observasi, atau kasus-kasus ke dalam kelas-kelas dari objek yang mirip. Pada *clustering* tidak ada *variable* sasaran. Sebuah *cluster* adalah koleksi *record* yang mirip satu sama lain, dan tidak mirip dengan *record* pada *cluster*. Tidak seperti klasifikasi, pada *clustering* tidak ada *variable* target. *Clustering* tidak menglasifikasi atau mengestimasi atau memprediksi tetapi mencari untuk mensegmentasi seluruh data set ke *subgroup* yang *relative* sejenis atau *cluster*, dimana kemiripan *record* di dalam *cluster* dimaksimalkan dan kemiripan dengan *record* di luar *cluster* diminimalkan. Contoh *clustering*, untuk *accounting* dengan tujuan audit untuk mensegmentasi *financial behaviour* ke dalam kategori ramah dan curiga, sebagai alat reduksi dimensi ketika data set memiliki ratusan atribut, untuk *clustering* ekspresi gen, dimana kuantitas gen bisa terlihat mempunyai *behavior* yang mirip. Algoritma untuk *clustering* antara lain, *hierarchical agglomerative clustering, Bayesian clustering, self – organizing feature maps, growing hierarchical self - organizing maps* (Wu, 2009).

6. Asosiasi

Tugas asosiasi untuk data *mining* adalah kegiatan untuk mencari atribut yang “*go together*.” Dalam dunia bisnis, asosiasi dikenal sebagai *affinity analysis* atau *market basket analysis*, tugas asosiasi adalah membuka *rules* untuk pengukuran hubungan antara dua atribut atau lebih. Contoh asosiasi, prediksi degradasi dalam jaringan komunikasi, menemukan barang apa di supermarket yang dibeli bersama dengan barang lain yang tidak pernah dibeli bersama, menemukan proporsi kasus dimana obat baru akan memperlihatkan efek samping yang berbahaya. Untuk menemukan*association rules*, bisa dilakukan dengan Algoritma A Priori dan algoritmaGRI (*Generalized Rule Induction*) (Larose, 2005) .

**2.2.2.2 Algoritma Klasifikasi Data *Mining***

Klasifikasi (Han, 2006) adalah proses penemuan model (atau fungsi) yang menggambarkan dan membedakan kelas data atau konsep yang bertujuan agar bisa digunakan untuk memprediksi kelas dari objek yang label kelasnya tidak diketahui.

Klasifikasi data terdiri dari 2 langkah proses. Pertama adalah *learning* (*fase training*), dimana algoritma klasifikasi dibuat untuk menganalisa data *training* lalu direpresentasikan dalam bentuk *rule* klasifikasi. Proses kedua adalah klasifikasi, dimana data tes digunakan untuk memperkirakan akurasi dari *rule* klasifikasi. (Han, 2006).

Proses klasifikasi didasarkan pada empat komponen: (Gorunescu, 2011)

a. Kelas

Variabel independen yang berupa kategorikal yang merepresentasikan labelyang terdapat pada objek. Contohnya: resiko penyakit jantung, resiko kredit, *customer loyalty*, jenis gempa.

b. *Predictor*

Variabel independen yang direpresentasikan oleh karakteristik (atribut) data. Contohnya: merokok, minum alkohol, tekanan darah, tabungan, aset, gaji.

c. *Training dataset*

Satu set data yang berisi nilai dari kedua komponen di atas yang digunakan untuk menentukan kelas yang cocok berdasarkan *predictor*.

d. *Testing dataset*

Berisi data baru yang akan diklasifikasikan oleh model yang telah dibuat dan akurasi klasifikasi dievaluasi.

Berikut ini adalah algoritma klasifikasi yang banyak digunakan secara luas: (Gorunescu, 2011).

1. *Decision/ classiﬁcation trees*

*Decision tree* digunakan untuk memprediksi keanggotaan suatu objek ke dalam kategori (kelas) yang berbeda, berdasarkan variabel prediktor. Algoritma *decision tree* yang dikenal luas antara lain *Hunt, CART (C&RT)*, ID3, C4.5&C5.0, *SLIQ, SPRINT, QUEST, DTREG, THAID, CHAID* (Gorunescu, 2011).

2. *Bayesian classiﬁers/ Naïve Bayes classiﬁers*

Klasifikasi Bayes (Kusrini, 2009) adalah pengklasifikasian statistik yang dapat digunakan untuk memprediksi probabilitas keanggotaan suatu kelas. Klasifikasi *Bayes* juga dikenal dengan *Naïve Bayes, idiot’s Bayes, simpleBayes*, dan *independence Bayes* (Wu, 2009). Klasifikasi *Bayes* didasarkan pada *teorema Bayes*, diambil dari nama seorang ahli matematika yang juga menteri *Prebysterian* Inggris, Thomas Bayes (1702-1761).

3. *Neural networks*

*Neural network* adalah (Han, 2006) satu set unit *input/output* yang terhubung dimana tiap relasinya memiliki bobot. *Neural Network* dimaksudkan untuk mensimulasikan perilaku sistem biologi susunan syaraf manusia (Alpaydin,2010).

4. Analisa Statistik

Analisa statistik selama ini juga melakukan hal yang sama seperti data *mining*. Beberapa metode estimasi dan prediksi tradisional merupakan turunan dari analisa statistik, antara lain *linearregression* dan *multiple regression* (Larose, 2005).

5. Algoritma Genetika

Algoritma genetika merepresentasikan teknik identifikasi solusi perkiraan untuk masalah optimasi dan pencarian. Algoritma genetika merupakan tipe algoritma *evolusioner* yang paling populer, berbasis pada populasi solusi potensial mengguinakan mekanisme spesifik yang diinspirasi dari evolusi biologi (genetika alamiah), seperti kromosom, reproduksi, rekombinasi, seleksi, *survival of fittest* (Gorunescu, 2011).

6. *Rough sets*

*Rough Set* dibuat oleh Zdzislaw Pawlak pada awal tahun 1980, dalam rangka mengungkap secara matematis konsep kesamaran, tujuan utamanya adalah menjadi proses otomatisasi transformasi data ke dalam pengetahuan. *Rough set* merupakan pendekatan matematis untuk pengetahuan yang tidak sempurna, hal ini penting dalam logika *fuzzy*. Konsep *rough set* secara umum bisa didefinisikan dengan 2 topologi, yaitu *interior* dan  *closure*. Ide dasar *rough set* terdapat pada kenyataan bahwa, berdasarkan satu set objek, satu set atribut dan nilai keputusan, seseorang dapat membuat *rule* untuk menemukan perkiraan *upper* dan *lower*, dan daerah batasan (*boundary region*) dari set objek. Setelah *rule* dibuat, maka objek baru bisa dengan mudah diklasifikasikan ke dalam salah satu daerah (*region*) (Gorunescu, 2011).

7. *K-Nearest Neighbor*

*K- Nearest Neighbor* (K-NN) adalah klasifikasi yang menyimpan semua data *training* dan melakukan klasifikasi dengan cara membandingkan antara atribut data baru yang paling cocok dengan atribut *record* yang terdapat pada data *training*. K-NN adalah kasus khusus dalam *instance-based learning*. Ini termasuk *case-based reasoning*, yang menangani data simbol. K-NN juga merupakan contoh teknik *lazy learning*, yaitu teknik yang menunggu sampai pertanyaan (*query*) datang agar sama dengan data *training* (Wu, 2009).

8. Metode *Rule Based*

Klasifikasi *rule based* artinya proses penggunaan data *training* dengan objek berlabeldimana*rule*akan diekstrak untuk menghasilkan klasifikasi *Gorunescu*, 2011).

9. *Memory based reasoning*

10. *Support vector machines* (SVM)

*Supportvectormachines*(SVM)dibuatolehVapnikuntukmengimplementasikansecarakonstruktifprinsip dari teori*statisticallearning*. Dalam kerangka *statistical learning, learning* artinya mengestimasi sebuah fungsi dari data *training*. Untuk melakukan ini, sebuah *machine learning* harus memilih satu fungsi untuk meminimalkan sejumlah resikoyangfungsi estimasinya berbeda dengan fungsi kenyataan (yang belum diketahui). Resiko tergantung pada kompleksitas fungsi-fungsi yang dipilih dari *training* set. Kemudian *learning machine* harus menemukan fungsi terbaik, ditentukan oleh kompleksitasnya. SVM cukup populer untuk penggunaan klasifikasi karena kelebihannya antara lain dari segi cara kerja, SVM baik untuk klasifikasi, tidak tergantung pada jumlah fitur dan bisa mengatasi masalah dimensi. Dari segi komputasi, SVM dapat melakukan proses *training* dengan cepat dan ini berguna dalam teknik *learning* ketika menghadapi masalah ketidaktegasan (Maimon, 2005).

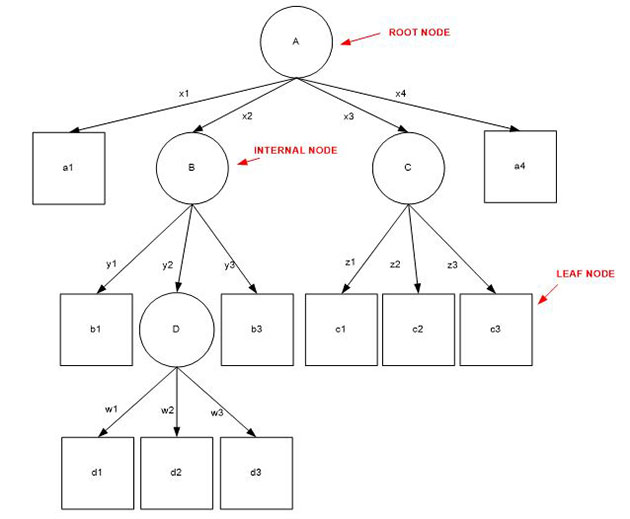
* + 1. ***Decision tree* (Pohon Keputusan)**

Manusia selalu dihadapkan dengan berbagai macam masalah dari berbagai bidang kehidupan. Masalah ini juga memiliki variasi tingkat kesulitannya. Untuk menghadapi masalah ini manusia mulai mengembangkan sebuah sistem untuk membantu mereka menyelesaikan masalah-masalah ini, salah satu sistem tersebut adalah pohon keputusan.

Pohon keputusan adalah metode klasifikasi dan prediksi yang sudah terbukti *powerfull* dan sangat terkenal. Metode ini berfungsi untuk mengubah fakta menjadi pohon keputusan yang merepresentasikan aturan yang dapat mudah dimengerti dengan bahasa alami. Proses dari pohon keputusan ini dimulai dari *node* akar hingga *node* daun yang dilakukan secara rekursif dimana setiap percabangan menyatakan kondisi dan setiap ujung pohon akan menyatakan keputusan Arsitektur pohon keputusan dibuat sedemikian rupa agar menyerupai pohon asli, dimana terdapat beberapa bagian yaitu

1. *Root nod*e atau node akar merupakan node yang terletak paling atas dari suatu pohon.
2. *Internal Node* ini merupakan node percabangan, dimana pada node ini hanya terdapat satu input dan mempunyai minimal dua output.
3. *Leaf Node* ini merupakan node akhir, hanya memiliki satu input, dan tidak memiliki output. Pada pohon keputusan setiap leaf node menandai label kelas.

  Pada pohon keputusan di setiap percabangan menyatakan kondisi yang harus dipenuhi dan tiap ujung pohon menyatakan nilai kelas data. Gambar berikut merupakan bentuk arsitektur pohon keputusan.



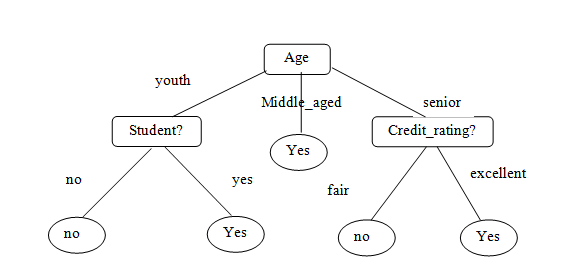
Gambar 2.1 Arsitektur pohon keputusan

1. Kelebihan Pohon Keputusan:
   * Daerah pengambilan keputusan yang kompleks dapat diubah menjadi sederhana.
   * Dapat menghilangkan perhitungan yang tidak penting karena proses pengujian hanya berdasarkan kriteria yang diperlukan saja.
   * Proses pemilihan fitur dari *internal node* yang berbeda lebih fleksibel. Fitur yang telah dipilih ini akan menjadi pembeda antara kriteria yang satu dengan kriteria lainnya.
   * Metode ini dapat menghindari munculnya permasalahan dengan cara menggunakan kriteria dengan jumlah yang sedikit pada *node internal* tanpa mengurangi kualitas keputusan yang dihasilkan.
2. Kekurangan Pohon Keputusan :
   * Dapat terjadi *overlap* apabila hasil keputusan dan kriteria yang digunakan jumlahnya sangat banyak. Hal ini juga dapat berakibat bertambahnya waktu yang digunakan untuk pengambilan keputusan dan jumlah memori yang dibutuhkan semakin tinggi.
   * Akumulasi jumlah *error* dari setiap tingkat pohon keputusan besar.
   * Mendesain pohon keputusan yang optimal sulit.
   * Kualitas keputusan yang didapatkan sangat tergantung dengan bagaimana pohon tersebut didesain.

**2.2.3.1 Algoritma C4.5**

**2.2.3.2 Konsep Algoritma C4.5**

Pada akhir tahun 1970 sampai awal tahun 1980 J. Ross Quinlan, seorang peneliti di bidang *machine learning*, membuat sebuah algoritma *decision tree* yang dikenal dengan ID3 (*Iterative Dichotomiser*). Quinlan kemudian membuat algoritma C4.5 (sering disebut dengan pohon keputusan) yang merupakan pengembangan dari algoritma ID3 (Han, 2006). Algoritma ini memiliki kelebihan, yaitu mudah dimengerti, fleksibel, dan menarik karena dapat divisualisasikan dalam bentuk gambar (pohon keputusan) (Gorunescu, 2011). Algoritma C4.5 merupakan struktur pohon dimana terdapat simpul yang mendeskripsikan atribut-atribut, setiap cabang menggambarkan hasil dari atribut yang diuji, dan setiap daun menggambarkan kelas. Algoritma C4.5secara rekursif mengunjungi setiap simpul keputusan, memilih pembagian yang optimal, sampai tidak bisa dibagi lagi. Algoritma C4.5 menggunakan konsep *information gain* atau *entropy reduction* untuk memilih pembagian yang optimal (Han, 2006).



**Gambar 2. 2** Contoh Konsep Pohon Keputusan

Pohon keputusan mirip sebuah sebuah struktur pohon dimana terdapat *node internal* (bukan daun) yang mendeskripsikan atribut-atribut, setiap cabangmenggambarkan hasil dari atribut yang diuji, dan setiap daun menggambarkan kelas. Pohon keputusan bekerja mulai dari akar paling atas, jika diberikan sejumlah data uji, misalnya X dimana kelas dari data X belum diketahui, maka pohon keputusan akan menelusuri mulai dari akar sampai *node* dan setiap nilai dari atribut sesuai data X diuji apakah sesuai dengan aturan pohon keputusan, kemudian pohon keputusan akan memprediksi kelas dari tupel X.Perbandingan kinerja antara algoritma lainnya, *decision tree*  yang memakai algoritma C4.5 membuktikan bahwa metode tersebut lebih efisien dan paling sederhana jika dibandingkan dengan algoritma lainnya. Dengan kesederhanaannya algoritma C4.5 memberikan hasil lebih baik untuk klasifikasi. Berdasarkan penelitian yang di lakukan oleh Jyh-jian Sheo, 2008 diperoleh hasil bahwa metode *decision tree C4.5* merupakan metode yang paling baik jika dibandingkan dengan *naive bayes* dan *k-nearst neighboars (KNN).*

Gambar 2.3 menggambarkan pohon keputusan untuk memprediksi apakah seseorang membeli komputer.*Node internal* disimbolkan dengan persegi, cabang disimbolkan dengan garis, dan daun disimbolkan dengan oval.

Ada beberapa tahap dalam membuat sebuah pohon keputusan dengan algoritma C4.5 (Kusrini, 2009), yaitu :

1. Menyiapkan data *training*. Data *training* biasanya diambil dari data histori yang pernah terjadi sebelumnya dan sudah dikelompokkan ke dalam kelas- kelas tertentu.

2. Menentukan akar dari pohon. Akar akan diambil dari atribut yang terpilih, dengan cara menghitung nilai *gain* dari masing-masing atribut, nilai *gain* yang paling tinggi yang akan menjadi akar pertama. Sebelum menghitung nilai *gain* dari atribut, hitung dahulu nilai *entropy*. Untuk menghitung nilai *entropy*digunakan rumus:

...................... 2.1

Keterangan:

S = himpunan kasus n = jumlah partisi S

pi = proporsi Si terhadap S

3. Kemudian hitung nilai *gain* menggunakan rumus:

......................... 2.2

Keterangan:

S = himpunan kasus

A = fitur

n = jumlah partisi atribut A

│Si│ = proporsi Si terhadap S

│S│ = jumlah kasus dalam S

4. Ulangi langkah ke-2 hingga semua *record* terpartisi.

5. Proses partisi pohon keputusan akan berhenti saat :

a. Semua *record* dalam simpul N mendapat kelas yang sama.

b. Tidak ada atribut di dalam *record* yang dipartisi lagi.

c. Tidak ada *record* di dalam cabang yang kosong.

**2.2.3.3 Penerapan Algoritma C4.5**

Berikut ini adalah uraian langkah-langkah dalam algoritma C4.5 untuk menyelesaikan kasus suatu pertandingan tenis akan dilakukan atau tidak, berdasarkan keadaan cuaca, suhu, kelembaban dan angin. (Kusrini&Luthfi,2009).

**Tabel 2.2** *Learning Dataset*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Cuaca** | **Suhu** | **Kelembaban** | **Berangin** | **Main** |
| 1 | Cerah | Panas | Tinggi | Salah | Tidak |
| 2 | Cerah | Panas | Tinggi | Benar | Tidak |
| 3 | Berawan | Panas | Tinggi | Salah | Ya |
| 4 | Hujan | Sejuk | Tinggi | Salah | Ya |
| 5 | Hujan | Dingin | Normal | Salah | Ya |
| 6 | Hujan | Dingin | Normal | Benar | Ya |
| 7 | Berawan | Dingin | Normal | Benar | Ya |
| 8 | Cerah | Sejuk | Tinggi | Salah | Tidak |
| 9 | Cerah | Dingin | Normal | Salah | Ya |
| 10 | Hujan | Sejuk | Normal | Salah | Ya |
| 11 | Cerah | Sejuk | Normal | Benar | Ya |
| 12 | Berawan | Sejuk | Tinggi | Benar | Ya |
| 13 | Berawan | Panas | Normal | Salah | Ya |
| 14 | Hujan | Sejuk | Tinggi | Benar | Tidak |

PadaTabel2.2, atribut atributnya adalah Cuaca, Suhu, Kelembaban, dan Berangin. Setiap atribut memiliki nilai. Sedangkan kelasnya ada pada kolom *Main* yaitu kelas “Tidak” dan kelas “Ya”. Kemudian data tersebut dianalisis; dataset tersebut memiliki 14 kasus yang terdiri 10 “Ya” dan 4 “Tidak” pada kolom *Main* (lihatTabel2.3).

Kemudianhitung entropidenganpersamaan 2.1 sebagaiberikut:

**Tabel 2.3** Hasil Perhitungan pada Dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **Total Kasus** | **Sum(Ya)** | **Sum(Tidak)** | **EntropiTotal** |
| 14 | 10 | 4 | 0.863120569 |

Setelah mendapat kanentropi dari keseluruhan kasus, lakukan analisis pada setiap atribut dan nilai-nilainya dan hitung entropinya seperti yang ditampilkan pada Tabel 2.4

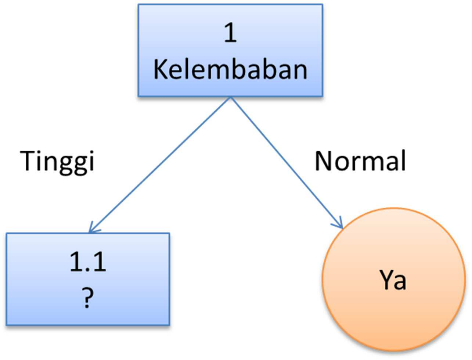
**Tabel 2.4**. Analisis Atribut, Nilai, Banyaknya Kejadian Nilai, Entropi dan *Gain*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Node** | **Atribut** | **Nilai** | **Sum**  **(Nilai)** | **Sum**  **(Ya)** | **Sum**  **(Tidak)** | **Entropi** | **Gain** |
| 1 | Cuaca | Berawan | 4 | 4 | 0 | 0 |  |
|  | Hujan | 5 | 4 | 1 | 0.721928095 |
|  | Cerah | 5 | 2 | 3 | 0.970950594 |
|  |  | | | | | 0.258521037 |
| Suhu | Dingin | 4 | 4 | 0 | 0 |  |
|  | Panas | 4 | 2 | 2 | 1 |
|  | Sejuk | 6 | 4 | 2 | 0.918295834 |
|  |  | | | | | 0.183850925 |
| Kelembaban | Tinggi | 7 | 3 | 4 | 0.985228136 |  |
|  | Normal | 7 | 7 | 0 | 0 |
|  |  | | | | | 0.370506501 |
| Berangin | Salah | 8 | 6 | 2 | 0.811278124 |  |
|  | Benar | 6 | 2 | 4 | 0.918295834 |
|  |  | | | | | 0.005977711 |

Untuk menghitung *gain* setiap atribut menggunakan persamaan 2.2 adalah:

Hitung pula *Gain* (Suhu), *Gain* (Kelembaban), dan *Gain* (Berangin). Hasilnya dapat dilihat pada Tabel 2.4. Karena nilai *gain* terbesar adalah *Gain* (Kelembaban). Maka Kelembaban menjadi *node* akar (*root node*).

Kemudian pada kelembaban normal, memiliki 7 kasus dan semuanya memiliki jawaban Ya (Sum (Total) / Sum (Ya) = 7 / 7 = 1). Dengan demikian kelembaban normal menjadi daun atau *leaf*. Lihat Tabel 2.4 yang selnya berwarna hijau.

Gambar 2.3 Pohon Keputusan

Berdasarkan pembentukan pohon keputusan node 1 (root node), Node1.1 akan dianalisis lebih lanjut. Untuk mempermudah, Tabel 2.7 difilter, dengan mengambil data yang memiliki Kelembaban = Tinggi sehingga jadilah Tabel 2.4

**Tabel 2.5** Data yang Memiliki Kelembaban = Tinggi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Cuaca** | **Suhu** | **Kelembaban** | **Berangin** | **Main** |
| 1 | Cerah | Panas | Tinggi | Salah | Tidak |
| 2 | Cerah | Panas | Tinggi | Benar | Tidak |
| 3 | Berawan | Panas | Tinggi | Salah | Ya |
| 4 | Hujan | Sejuk | Tinggi | Salah | Ya |
| 5 | Cerah | Sejuk | Tinggi | Salah | Tidak |
| 6 | Berawan | Sejuk | Tinggi | Benar | Ya |
| 7 | Hujan | Sejuk | Tinggi | Benar | Tidak |

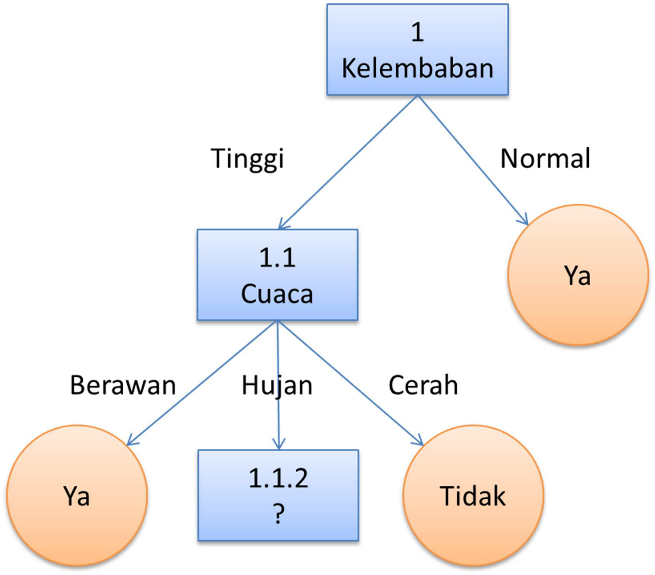
Kemudian data di Tabel 2.4 dianalisis dan dihitung lagi entropi atribut Kelembaban Tinggi dan entropi setiap atribut serta *gain*nya sehingga hasilnya seperti data pada Tabel 2.5 Setelah itu tentukan pilih atribut yang memiliki *gain* tertinggi untuk dibuatkan *node* berikutnya.

**Tabel 2.6** Hasil Analisis *Node* 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| **Kelembaban**  **Tinggi** | **Sum(Ya)** | **Sum(Tidak)** | **Entropi** |
| 7 | 3 | 4 | 0.985228136 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Node** | **Atribut** | **Nilai** | **Sum**  **(Variabel)** | **Sum(Ya)** | **Sum (Tidak)** | **Entropi** | **Gain** |
| 1.1 | Cuaca | Berawan | 2 | 2 | 0 | 0 |  |
|  | Hujan | 2 | 1 | 1 | 1 |
|  | Cerah | 3 | 0 | 3 | 0 |
|  |  | | | | | 0.69951385 |
| Suhu | Dingin | 0 | 0 | 0 | 0 |  |
|  | Panas | 3 | 1 | 2 | 0.918295834 |
|  | Sejuk | 4 | 2 | 2 | 1 |
|  |  | | | | | 0.020244207 |
| Berangin | Salah | 4 | 2 | 2 | 1 |  |
|  | Benar | 3 | 2 | 1 | 0.918295834 |
|  |  | | | | | 0.020244207 |

Dari Tabel 2.6, *gain* tertinggi ada pada atribut Cuaca, dan Nilai yang dijadikan daun atau *leaf* adalah Berawan dan Cerah. Jika divualisasi maka pohon keputusan tampak seperti Gambar 2.4



**Gambar 2.4** Pohon Keputusan Analisis *Node* 1.1

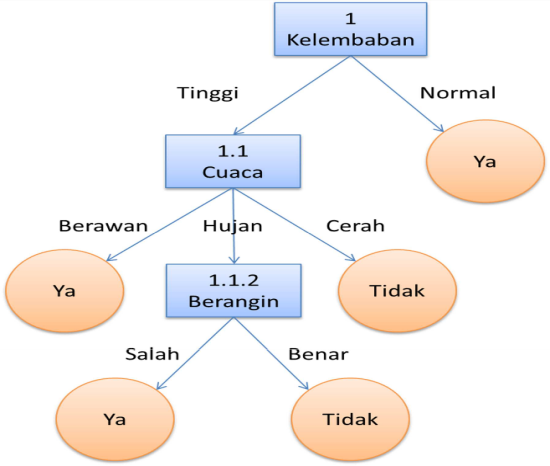
Untuk menganalisis *node* 1.1.2, lakukan lagi langkah-langkah yang sama seperti sebelumnya. Hasilnya ditampilkan pada Tabel 2.7 dan Gambar 2.5

**Tabel 2.7** HasilAnalisis *Node* 1.1.2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Cuaca** | **Suhu** | **Kelembaban** | **Berangin** | **Main** |
| 1 | Hujan | Sejuk | Tinggi | Salah | Ya |
| 2 | Hujan | Sejuk | Tinggi | Benar | Tidak |

|  |  |  |  |
| --- | --- | --- | --- |
| **Kelembaban Tinggi&Hujan** | **Sum(Ya)** | **Sum(Tidak)** | **Entropi** |
| 2 | 1 | 1 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Node** | **Atribut** | **Nilai** | **Sum**  **(Nilai)** | **Sum(Ya)** | **Sum**  **(Tidak)** | **Entropi** | **Gain** |
| 1.1.2 | Suhu | Dingin | 0 | 0 | 0 | 0 |  |
|  | Panas | 0 | 0 | 0 | 0 |
|  | Sejuk | 2 | 1 | 1 | 1 |
|  |  | | | | | 0 |
| Berangin | Salah | 1 | 1 | 0 | 0 |  |
|  | Benar | 1 | 0 | 1 | 0 |
|  |  | | | | | 1 |



**Gambar 2.5**. Pohon Keputusan Akhir

**2.2.5 Metode Evaluasi *Confusion Matrix***

*Confusion Matrix* adalah *tool* yang digunakan untuk evaluasi model klasifikasi untuk memperkirakan objek yang benar atau salah (Gorunescu, 2011). Sebuah *matrix* dari prediksi yang akan dibandingkan dengan kelas yang asli dari inputan atau dengan kata lain berisi informasi nilai *actual* dan prediksi pada klasifikasi.

**Tabel 2.8** *Confusion Matrix*

|  |  |  |
| --- | --- | --- |
| *Clasification* | *Predicted Class* | |
| *Class = Yes* | *Class = No* |
| *Class = Yes* | a (*true positive*-TP) | b (*false negative*-FN) |
| *Class = No* | c (*false positive*-FP) | d (*true negative*-TN) |

Keterangan tabel:

1. *True Positives* merupakan jumlah *record* positif yang diklasifikasikan sebagai positif.
2. *False Positives* merupakan jumlah *record* negatif yang diklasifikasikan sebagai positif.
3. *False negatives* merupakan jumlah *record* positif yang diklasifikasikan sebagai negative.
4. *True negatives* merupakan jumlah *record* negatif yang diklasifikasikan sebagai negative.

Rumus untuk menghitung tingkat akurasi pada matrix adalah:

…………………2.3



**2.5 Siklus Hidup Pengembangan Sistem**

SDLC *(System Development Life Cycle,* Siklus Hidup Pengembangan Sistem) atau *System Life Cycle* (siklus Hidup Sistem), dalam rekayasa sistem dan rekayasa perangkat lunak, adalah proses pembuatan dan pengubahan sistem serta model dan metodologi yang digunakan untuk mengembangkan sistem-sistem tersebut. SDLC *(System Development Life Cycle* juga merupakan pola yang diambil untuk mengembangkan sistem perangkat lunak. (Fairuzelsaid, 2013)

Siklus hidup pengembangan sistem dengan langkah-langkah utamanya yang akan digunakan adalah sebagai berikut :

Gambar 2.6 Siklus Hidup Pengembangan Sistem

**2.3.1 Analisis Sistem**

Model analisis perangkat lunak seperti kita ketahui, sesungguhnya merupakan spesifikasi rinci dari kebutuhan-kebutuhan pengguna dan bekerja sebagai langkah pertama saat kelak akan mengembangkan model perancangan perangkat lunak. (Nugroho, 2010).Setelah proses perencanaan atau *planing* tahapan selanjutnya adalah analisis terhadap data-data mengenai kebutuhan pengguna yang dipadukan dengan hasil studi kelayakan yang meliputi data-data ketersediaan sumber daya yang ada seperti manusia. infrastruktur teknologi yang tersedia dan lain-lain, yang kemudian menetapkan solusi yang tepat untuk diterapkan. Pada tahapan ini seorang analis sistem melakukan pengenalan terhadap segenap permasalahan yang timbul pada penguna melalui dokomposisi dan realisasi permasalahan lebih lanjut, mengenali komponen-komponen sistem/perangkat lunak, objek-objek. Hubungan antar objek dan sebagainya, dimana secara keseluruhan dinotasikan dalam bentuk *use case.*

* + 1. **Desain Sistem**

Dalam desain sistem, dibutuhkan alat bantu desain. Dalam tahapan ini, pengembang sistem bisa menentukan arsitektur sistemnya, merancang gambaran konseptual sistem, merancang *database*, perancangan *interface*, hingga membuat *flowchart* program. Salah satu alat bantu yang bisa digunakan dalam pembuatan sistem bantu keputusan adalah *Data Flow Diagram* (DFD). DFD adalah suatu model logika data atau proses yang dibuat untuk menggambarkan asal data dan tujuan data yang keluar dari sistem, tempat penyimpanan data, proses apa yang menghasilkan data tersebut, serta interaksi antara data yang tersimpan dan proses yang dikenakan pada data tersebut.

Menurut John Burch dan Gary Grudnitski, desain sistem dapat didefinisikan sebagai penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi. (Jogiyanto, 2005 : 196).

Tahap desain sistem mempunyai dua tujuan utama :

1. Untuk memenuhi kebutuhan kepada pemakai sistem.
2. Untuk memberikan gambaran yang jelas dan rancang bangun yang lengkap kepada pemrogram komputer dan ahli-ahli teknik lainnya.

Desain sistem dibagi dalam dua bagian, yaitu desain sistem secara umum *(general systems design)* dan desain sistem terinci *(detailed systems design).*

**2.3.2.1 Desain Sistem Secara Umum (*General Systems Design*)**

Pada tahap desain secara umum, komponen-komponen sistem informasi yang dirancang dengan tujuan dikomunikasikan kepada *user* bukan untuk pemrograman. Komponen sistem informasi yang di desain adalah model, *output, input*, *database*, teknologi, dan kontrol. (Jogiyanto, 2005 : 211).

1. **Desain Model Secara Umum**

Analisis sistem dapat mendesain model dari sistem informasi yang di usulkan dalam bentuk *physical* sistem dan *logical* model. Bagan alir sistem merupakan alat yang tepat digunakan untuk menggambarkan *physical systems, logical* model dapat digambar dengan diagram arus data. (Jogiyanto, 2005 : 211).

Bagan alir sistem merupakan bagan yang menunjukan arus pekerjaan secara keseluruhan dari sistem. Bagan alir sistem digambar dengan simbol-simbol sebagai berikut :

**Tabel 2.9** Daftar Simbol Bagan Alir Dokumen

| No | Nama Simbol | Simbol | Keterangan |
| --- | --- | --- | --- |
| 1. | Terminal |  | Menunjukkan untuk memulai dan mengakhiri  Suatu proses |
| 2. | Dokumen |  | Menunjukkan dokumen input dan *output* baik itu proses manual, mekanik, atau *computer* |
| 3. | Kegiatan Manual |  | Menunjukan pekerjaan manual |
| 4. | Simpanan *Offline* | N  A  C | Menunjukkan *file* non-komputer yang diarsip urut angka (*numerical*), huruf (*alphabetical*), atau tanggal (*chronological*) |
| 5. | Proses |  | Menunjukkan kegiatan proses dari operasi program komputer |
| 6. | Operasi Luar |  | Menunjukkan operasi yang dilakukan diluar operasi *computer* |
| 7. | *Hard Disk* |  | Menunjukkan *input* dan *output* menggunakan *harddisk* |
| 8. | *Keyboard* |  | Menunjukkan *input* yang menggunakan *on-line keyboard* |
| 9. | *Display* |  | Menunjukkan *output* yang ditampilkan di monitor |
| 10. | Hubungan Komunikasi |  | Menunjukkan proses transmisi data melalui *channel* komunikasi |
| 11. | Garis Alir |  | Menunjukkan arus dari proses |
| 12. | Penjelasan |  | Menunjukkan penjelasan dari suatu proses |
| 13. | Penghubung |  | Menunjukkan penghubung ke halaman yang masih sama atau ke halaman yang lain |

(**Sumber**: Jogiyanto HM, 2005 : 802)

Untuk mempermudah penggambaran suatu sistem yang ada atau sistem yang baru yang akan dikembangkan secara logika dan tanpa memperhatikan lingkungan fisik data tersebut mengalir atau lingkungan fisik dimana data tersebut akan disimpan, maka digunakan Diagram Arus Data (DAD) atau *Data Flow Diagram* (DFD).

**Tabel 2.10** Daftar Simbol Diagram Alir Dokumen

| No | Simbol | Keterangan |
| --- | --- | --- |
|  |  | Simbol Proses, Menunjukan informasi dari masukan menjadi keluaran |
|  |  | *Eksternal Entity*, merupakan kesatuan dilingkungan luar *system* yang dapat berupa orang, organisasi atau *system* lain yang berada di lingkungan luarnya yang akan memberikan *input* seta menerima *output* dari *system* |
|  |  | Aliran atau arus data, menggambarkan gerakan paket data atau informasi dari suatu bagian kebagian yang lain, dimana penyimpanan mewakili lokasi penyimpana data |
|  |  | Penyimpanan, digunakan untuk memodelkan kumpulan data atau paket data |

(**Sumber** : Jogiyanto, 2005 : 700-807)

1. **Desain *Output* Secara Umum**

*Output* adalah produk dari sistem informasi yang dapat dilihat. *Output* terdiri dari macam-macam jenis seperti hasil di media kertas, dan hasil di media lunak. Disamping itu *output* dapat berupa hasil dari suatu proses yang akan digunakan oleh proses lain dan tersimpan di suatu media seperti tape, disk, atau kartu. Yang dimaksud dengan *output* pada tahap desain ini adalah *output* yang berupa tampilan di media kertas atau di layar video. (Jogiyanto, 2005 : 213)

**c.** **Desain*Input* Secara Umum**

Alat *input* dapat digolongkan ke dalam 2 golongan, yaitu alat *input* langsung *(online input device)* dan alat *input* tidak langsung *(offline input device).* Alat *input* langsung merupakan alat *input* yang langsung dihubungkan dengan CPU, sedangkan alat *input* tidak langsung adalah alat *input* yang tidak langsung dihubungkan dengan CPU. (Jogiyanto, 2005 : 214)

1. **Desain *Datebase* Secara Umum**

Basis data (*database*) adalah kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan diluar komputer dan digunakan perangkat lunak tertentu untuk memanipulasinya. Sistem basis data adalah suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan satu dengan yang lainnya dan membuatnya tersedia untuk beberapa aplikasi yang bermacam-macam di dalam suatu organisasi. (Jogiyanto, 2005 : 217).

**2.3.2.2 Desain Sistem Secara Terinci (*Detailed systems design***)

**a. Desain *Output* Terinci**

Desain *output* terinci dimaksudkan untuk mengetahui bagaimana dan seperti apa bentuk *output-output* dari sistem yang baru. Desain *output* terinci terbagi atas dua, yaitu desain *output* berbentuk laporan di media kertas dan desain *output* dalam bentuk dialog di layar terminal. (Jogiyanto, 2005 : 362)

1. Desain *output* dalam bentuk laporan : dimaksudkan untuk menghasilkan *output* dalam bentuk laporan dimedia kertas. Bentuk laporan yang paling banyak digunakan adalah dalam bentuk tabel dan berbentuk grafik atau bagan. (Jogiyanto, 2005 : 362)
2. Desain *output* dalam bentuk dialog layar terminal : merupakan rancang bangun dari percakapan antara pemakai sistem atau *user* dengan komputer. Percakapan ini dapat terdiri dari proses memasukkan data ke sistem, menampilkan *output* informasi kepada *user*, atau keduanya.

**b. Desain *Input* Terinci**

Masukan merupakan awal dimulainya proses informasi. Bahan mentah dari informasi adalah data yang terjadi dari transaksi-transaksi yang dilakukan oleh organisasi. Data hasil dari transaksi merupakan masukan untuk sistem informasi. Hasil dari sistem informasi tidak lepas dari data yang dimasukan. Desain *input* terinci dimulai dari desain dokumen dasar sebagai penangkap *input* yang pertama kali. Jika dokumen dasar tidak di desain dengan baik, kemungkinan *input* yang tercatat dapat salah bahkan kurang. (Jogiyanto, 2005 : 375).

Fungsi dokumen dasar dalam penanganan arus data :

1. Dapat menunjukkan macam dari data yang harus dikumpulkan dan ditangkap.
2. Dapat dicatat dengan jelas, konsisten, dan akurat.
3. Dapat mendorong lengkapnya data, disebabkan data yang dibutuhkan disebutkan satu persatu di dalam dokumen dasarnya.

**c. Desain *Database* Terinci**

*Database* merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di simpanan luar komputer dan digunakan perangkat lunak tertentu untuk memanipulasinya. *Database* merupakan salah satu komponen yang penting di dalam sistem informasi, karena berfungsi sebagai basis penyedia informasi bagi para pemakainya. Penerapan *database* dalam sistem informasi disebut *database system.* (Jogiyanto, 2005 : 400).

**2.3.3 Seleksi Sistem**

Tahap ini merupakan tahap untuk memilih perangkat yang akan digunakan untuk sistem informasi. Pengetahuan dibutuhkan oleh pemilih sistem diantaranya adalah pengetahuan tentang siapa yang menyediakan teknologi ini, cara pemilikannya, dan sebagainya. Pemilihan sistem yang harus paham dengan teknik-teknik evaluasi untuk menyelesaikan sistem.

**2.3.4 Implementasi**

Menurut Kusrini (2007 : 43), Implementasi sistem merupakan tahapan untuk meletakkan sistem supaya siap untuk dioperasikan. Pada tahapan ini terdapat banyak aktifitas yang dilakukan, yaitu :

1. Pemrograman dan pengetesan program

Pemrograman merupakan kegiatan menulis program yang akan dieksekusi oleh komputer. Kode program harus berdasarkan dokumentasi yang disediakan oleh analis sistem hasil dari desain sistem.

1. Instalasi perangkat keras dan lunak

Proses pemasangan perangkat keras dan instalasi perangkat lunak yang sudah ada.

1. Pelatihan kepada pemakai

Manusia merupakan faktor yang diperlukan dalam sistem informasi. Jika ingin sukses dalam sistem informasi, maka personil-personil yang terlibat harus diberi pengertian dan pengetahuan tentang sistem informasi dan posisi serta tugas mereka.

1. Pembuatan dokumentasi

Dokumentasi adalah melakukan pencatatan terhadap setiap langkah pekerjaan pembuatan sebuah program yang dilakukan dari awal sampai selesai.

**2.3.4 Perawatan Sistem**

Perawatan sistem informasi adalah suatu upaya untuk memperbaiki, menjaga, menanggulangi, mengembangkan sistem yang ada. Perawatan ini di perlukan untuk meningkatkan efisiensi dan efektivitas kinerja sistem yang ada agar dalam penggunaannya dapat optimal. Beberapa alasan mengapa kita perlu memelihara sistem yang ada yaitu: agar dapat meningkatkan sistem / kinerja sistem, dan menyesuaikan dengan perkembangan, agar sistem yang ada tidak tertinggal.

Aplikasi yang professional dalam SDLC dan teknik maupun perangkat *modeling* yang mendukungnya adalah hal-hal keseluruhan yang terbaik yang dapat seseorang lakukan untuk meningkatkan maintainabilitas sistem.

Jenis – jenis perawatan sistem meliputi :

* 1. Perawatan korektif : adalah pemeliharaan yang mengkoreksi kesalahan – kesalahan yang ditemukan pada sistem, pada saat sistem di jalankan / berjalan.
  2. Pemeliharaan adaptif : yaitu pemelihaaan yang bertujuan untuk menyesuaikan perubahan yang terjadi.
  3. Pemeliharaan perfektif : pemeliharaan ini bertujuan untuk meningkatkan cara kerja suatu sistem.
  4. Pemeliharaan preventif : pemeliharaan ini bertujuan untuk menangani masalah – masalah yang ada.

**2.4 Teknik Pengujian Sistem**

Pengujian sistem adalah elemen kritis dari jaminan kualitas perangkat lunak dan mempresentasikan kajian pokok dari spesifikasi, desain, dan pengkodean. Tujuan dari pengujian ini adalah diharapkan dengan minimal tenaga dan waktu untuk menemukan berbagai potensi kesalahan dan cacat.Harus didasarkan pada kebutuhan berbagai tahap pengembangan, desain dan dokumen lain atau program yang dirancang untuk menguji struktur *internal*, dan menggunakan contoh-contoh ini untuk menjalankan program untuk mendeteksi kesalahan.Pengujian sistem informasi harus mencakup pengujian perangkat lunak, pengujian perangkat keras dan pengujian jaringan.Pengujian *hardware*, jaringan pengujian berdasarkan indikator kinerja spesifik yang akan digunakan di sini pengujian lebih jauh adalah pengujian perangkat lunak.

**2.4.1 *White Box***

Pengujian *white-box (glass box)*, adalah metode desain *test case* yang menggunakan struktur kontrol desain prosedural untuk memperoleh *test case*. Dengan menggunakan metode pengujian *white-box*, perekayasa sistem dapat melakukan *test case* untuk memberikan jaminan bahwa :

1. Semua jalur independen pada suatu modul ditelusuri minimal 1 (satu) kali.
2. Semua jalur keputusan logis *True/False* dilalui.
3. Semua *loop* dieksekusi pada batas yang tercantum dan batas operasionalnya.
4. Struktur data *internal* digunakan agar validitas terjamin.

Pengujian *white-box* bisa dilakukan dengan pengujian *basis path*, metode ini merupakan salah satu teknik pengujian struktur kontrol untuk menjamin semua statemen dalam setiap jalur independen program dieksekusi minimal 1 kali dan tidak menjumpai *error message*. Perhitungan jalur independen dapat dilakukan melalui metrik *Cyclomatic Complexity.* Sebelum menghitung nilai *Cyclomatic Complexity,* harusditerjemahkan desain prosuderal ke grafik alir, kemudian dibuat *flow graphnya*, seperti pada gambar di bawah ini (Roger S. Pressman, 2002 : 536).

2

3

6

5

4

8

7

1

9

10

11

Gambar 2.7 Contoh Bagan Alir

Bagan alir digunakan untuk menggambarkan struktur kontrol program. Dan untuk menggambarkan grafik alir, harus mempertahatikan representasi desain prosedural pada bagan alir. Pada gambar dibawah ini, grafik alir memetakan bagan alir tersebut ke dalam grafik alir yang sesuai (dengan mengasumsikan bahwa tidak ada kondisi senyawa yang didirikan kedalam diamond keputusan dari bagan alir tersebut).Masing-masing lingkaran, yang disebut *simpul* grafik alir, merepresentasikan satu atau lebih statemen prosedural. Urutan kotak proses dan permata keputusan dapat memetakan simpul tunggal. Anak panah tersebut yang disebut *edges* atau *link,* merepresentasikan aliran kontrol dan analog dengan anak panah bagan alir. Edge harus berhenti nada suatu simpul, meskipun bila simpul tersebut tidak mempresentasikan statemen prosedural. (Pressman, 2002).

R1

R4

Gambar 2.8 Contoh Grafik Alir

*Kompleksitas sikiomatis* adalah metriks perangkat lunak yang memberikan pengukuran kuantitatif terhadap kompleksitas logis suatu program. Bila metriks ini digunakan dalam konteks metode pengujian basis path, maka nilai yang terhitung untuk kompleksitas siklomatis menentukan jumlah *jalur independen.* Jalur independen adalah jalur yang melalui program yang mengintroduksi sedikitnya satu rangkaian statemen proses baru atau suatu kondisi baru. Bila dinyatakan dengan terminologi grafik alir, jalur independen harus bergerak sepanjang paling tidak satu edge yang tidak dilewatkan sebelum jalur tersebut ditentukan. Sebagai contoh, serangkaian jalur independen untuk grafik alir yang ditunjukkan pada gambar 2.10 adalah :

Jalur 1 : 1―11

Jalur 2 : 1―2―3―4―5―10―1―11

Jalur 3 : 1―2―3―6―8―9―10―1―11

Jalur 4 : 1―2―3―6―7―9―10―1―11

Jalur 1, 2, 3, dan 4 yang ditentukan diatas terdiri dari sebuah basis set untuk grafik alir pada gambar 2.10. bagaimana kita tahu banyaknya jalur yang dicari?

Komputasi kompleksitas siklomatis memberikan jawaban. Fondasi kompleksitas siklomatis adalah teori grafik, dan memberi kita metriks perangkat lunak yang sangat berguna. Kompleksitas dihitung dalam salah satu dari tiga cara berikut:

1. Jumlah *region* grafik alir sesuai dengan kompleksitas siklomatis.
2. Kompleksitas siklomatis, *V(G),* untuk grafik alir G ditentukan sebagai *V(G) = E +* 2 di mana *E* adalah jumlah *edge grafik alir dan N* adalah jumlah simpul grafik alir.
3. Kompleksitas siklomatis, *V(G)*, untuk grafik alir G juga ditentukan sebagai *V(G)* = P + 1, dimana P adalah jumlah simpul predikat yang diisikan dalam grafik alir G. Pada gambar 2.9 grafik alir, kompleksitas dapat dihitung dengan menggunakan masing-masing dari algoritma yang ditulis di atas:
4. Grafik alir mempunyai 4 region
5. V(G) = 11 edge ― 9 simpul + 2 = 4
6. *V(G)* = 3 simpul yang diperkirakan + 1 = 4

Dengan demikian, kompleksitas siklomatis dan grafik alir pada gambar 2.10 adalah 4. Yang lebih penting, nilai untuk *V(G)* memberikan kita batas atas untuk jumlah jalur independen yang membentuk basis set, dan imlikasinya, batas atas jumlah pengujian yang harus didesain dan dieksekusi untuk menjamin semua statemen program.

**2.4.2 *Black Box***

*Black box aproach* adalah suatu sistem dimana input dan outputnya dapat didefinisikan tetapi prosesnya tidak diketahui atau tidak terdefinisi. Metode ini hanya dapat dimengerti oleh pihak dalam (yang menangani sedangkan pihak luar hanya mengetahui masukan dan hasilnya). Sistem ini terdapat pada sub sistem tingkat terendah.

Metode uji coba *black box* memfokuskan pada keperluan fungsional dan *software.* Karena itu uji coba *black box* memungkinkan pengembang *software* untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba *black box* bukan merupakan alternatif dari uji coba *white box,* tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *whie box.* Ujicoba berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

1. Fungsi-fungsi yang salah atau hilang.
2. Kesalahan *interface.*
3. Kesalahan dalam struktur data atau akses database eksternal.
4. Kesalahan performa dan kesalahan inisialisasi.

Tidak seperti metode *white box* yang dilaksanakandiawal proses, ujicoba *black box* diaplikasikan kebeberapa tahapan berikutnya. Karena ujicoba *black box* dengan sengaja mengabaikan struktur kontrol. Sehingga perhatiannya difokuskan pada informasi *domain*. Ujicoba didesain untuk dapat menjawab pertanyaan pertanyaan berikut:

1. Bagaimana validitas fungsionalnya diuji?
2. Jenis input seperti apa yang akan menghasilkan kasus uji yang baik?
3. Apakah sistem secara khusus sensitive terhadap nilai input tertentu?
4. Bagaimana batasan-batasan kelas data diisolasi?
5. Berapa rasio data dan jumlah data yang dapat ditoleransi oleh sistem?
6. Apa akibat yang akan timbul dan kombinasi spesifik data pada operasi sistem?

Dengan mengaplikasikan ujicoba *black box,* diharapkan dapat menghasilkan sekumpulan kasus uji yang memenuhi kriteria berikut:

1. Kasus uji yang berkurang, jika jumlahnya lebih dari 1, maka jumlah dari uji kasus tambahan harus didesain untuk mencapai ujicoba yang cukup beralasan.
2. Kasus uji yang memberitahukan sesuatu tentang keberadaan atau tidaknya suatu jenis kesalahan, dari pada kesalahan yang terhubung hanya dengan suatu ujicoba yang spesifik.

**2.5 Perangkat Lunak Pendukung**

Tabel 2.11 Perangkat Lunak Pendukung

|  |  |  |
| --- | --- | --- |
| **No.** | **Tools yang digunakan** | **Kegunaan/Fungsi** |
| 1. | Microsoft Visual Basic.Net 2010 | VB .NET adalah salah satu bahasa pemograman tingkat tinggi yang mendekati bahasa manusia. VB.net menyerhanakan bahasa pemograman pada flatform.  Salah satu keunggulan yang dimiliki oleh VB.Net adalah mempunyai fasilitas real time backround Compiler yaitu sebgai penanganan dalam eror atau bug dan Lebih cepat dalam pembuatan aplikasi berbasis dekstop (sumber : Agus Kurniawan Tahun 2007) |
| 2. | Database MySQL | Sebuah perangkat lunak yang digunakan dalam pengoperasian basis data. |
| 3. | Crystall Report For Visual Studio | Digunakan untuk pembuatan laporan. |

**2.6 Kerangka Pikir**



**Gambar 2.9** Kerangka Pikir