BAB II

TINJAUAN PUSTAKA

2.1 Sistem Pakar

Sistem pakar pertama kali dikembangkan oleh komunitas *AI* pada pertengahan tahun 1960. Sistem pakar yang muncul pertama kali adalah *General Purpose Problem Solver* (GPS) yang dikembangkan oleh Newel & Simon (Efraim, 2005)

Beberapa definisi yang ada untuk sistem pakar (Kusumadewi, 2003):

1. Menurut Martin dan Oxman : Sistem pakar adalah sistem berbasis komputer yang menggunakan pengetahuan, fakta, dan teknik penalaran dalam memecahkan masalah, yang biasanya hanya dapat diselesaikan oleh seorang pakar dalam bidang tertentu.
2. Menurut Ignizio : Sistem pakar merupakan bidang yang dicirikan oleh system berbasis pengetahuan (*Knowledge Base System*), memungkinkan adanya komponen untuk berpikir dan mengambil kesimpulan dari sekumpulan kaidah.
3. Menurut Giarratano dan Riley : Sistem pakar adalah salah satu cabang kecerdasan buatan yang menggunakan pengetahuan-pengetahuan khusus yang dimiliki oleh seorang ahli untuk menyelesaikan suatu masalah tertentu.

Secara umum, sistem pakar merupakan sistem yang mengadopsi pengetahuan manusia ke dalam komputer sehingga komputer dapat digunakan untuk menyelesaikan suatu masalah sebagaimana yang dilakukan oleh seorang pakar. Sistem pakar dibuat pada wilayah pengetahuan tertentu dan untuk suatu keahlian tertentu yang mendekati kemampuan manusia di salah satu bidang khusus. Sistem pakar mencoba mencari solusi yang memuaskan sebagaimana yang dilakukan seorang pakar dan dapat memberikan penjelasan terhadap langkah yang diambil serta memberikan alasan atas kesimpulan yang diambil. Dalam penyusunannya, sistem pakar mengkombinasikan kaidah penarikan kesimpulan (*inference rules*)

dengan basis pengetahuan tertentu yang diberikan oleh satu atau lebih pakar dalam bidang tertentu.

2.1.1 Konsep Dasar Sistem Pakar

Konsep dasar dari sistem pakar yaitu meliputi keahlian (*expertise*), ahli (*experts*), pemindahan keahlian (*transfering expertise*), inferensi (*inferencing*), aturan (*rules*) dan kemampuan memberikan penjelasan (*explanation capability*).

Keahlian (*expertise*) adalah pengetahuan yang mendalam tentang suatu masalah tertentu, dimana keahlian bisa diperoleh dari pelatihan/ pendidikan, membaca dan pengalaman dunia nyata. Ada dua macam pengetahuan yaitu pengetahuan dari sumber yang ahli dan pengetahuan dari sumber yang tidak ahli. Pengetahuan dari sumber yang ahli dapat digunakan untuk mengambil keputusan dengan cepat dan tepat.

Ahli (*experts*) adalah seorang yang memiliki keahlian tentang suatu hal dalam tingkatan tertentu, ahli dapat menggunakan suatu permasalahan yang ditetapkan dengan beberapa cara yang berubah- ubah dan merubahnya kedalam bentuk yang dapat dipergunakan oleh dirinya sendiri dengan cepat dan cara pemecahan yang mengesankan. Kemampuan pemecahan masalah adalah penting, tetapi tidak cukup dilakukan sendiri.

Ahli seharusnya dapat untuk menjelaskan hasil yang diperoleh, mempelajari sesuatu yang baru tentang domain masalah, merestrukturisasi pengetahuan kapan saja yang diperlukan dan menentukan apakah keahlian mereka relevan atau saling berhubungan (Kusumadewi, 2003).

2.1.2 Tujuan Sistem Pakar

Tujuan dari sistem pakar adalah untuk memindahkan kemampuan (*transferring expertise*) dari seorang ahli atau sumber keahlian yang lain ke dalam komputer

dan kemudian memindahkannya dari komputer kepada pemakai yang tidak ahli (bukan pakar). Proses ini meliputi empat aktivitas yaitu (Efraim, 2005)

1. Akuisi pengetahuan (*knowledge acquisition*) yaitu kegiatan mencari dan mengumpulkan pengetahuan dari para ahli atau sumber keahlian yang lain.
2. Representasi pengetahuan (*knowledge representation*) adalah kegiatan menyimpan dan mengatur penyimpanan pengetahuan yang diperoleh dalam komputer. Pengetahuan berupa fakta dan aturan disimpan dalam komputer sebagai sebuah komponen yang disebut basis pengetahuan.
3. Inferensi pengetahuan (*knowledge inferencing*) adalah kegiatan melakukan inferensi berdasarkan pengetahuan yang telah disimpan didalam komputer.
4. Pemindahan pengetahuan (*knowledge transfer*) adalah kegiatan pemindahan pengetahuan dari komputer ke pemakai yang tidak ahli.

2.1.3 Ciri-Ciri Sistem Pakar

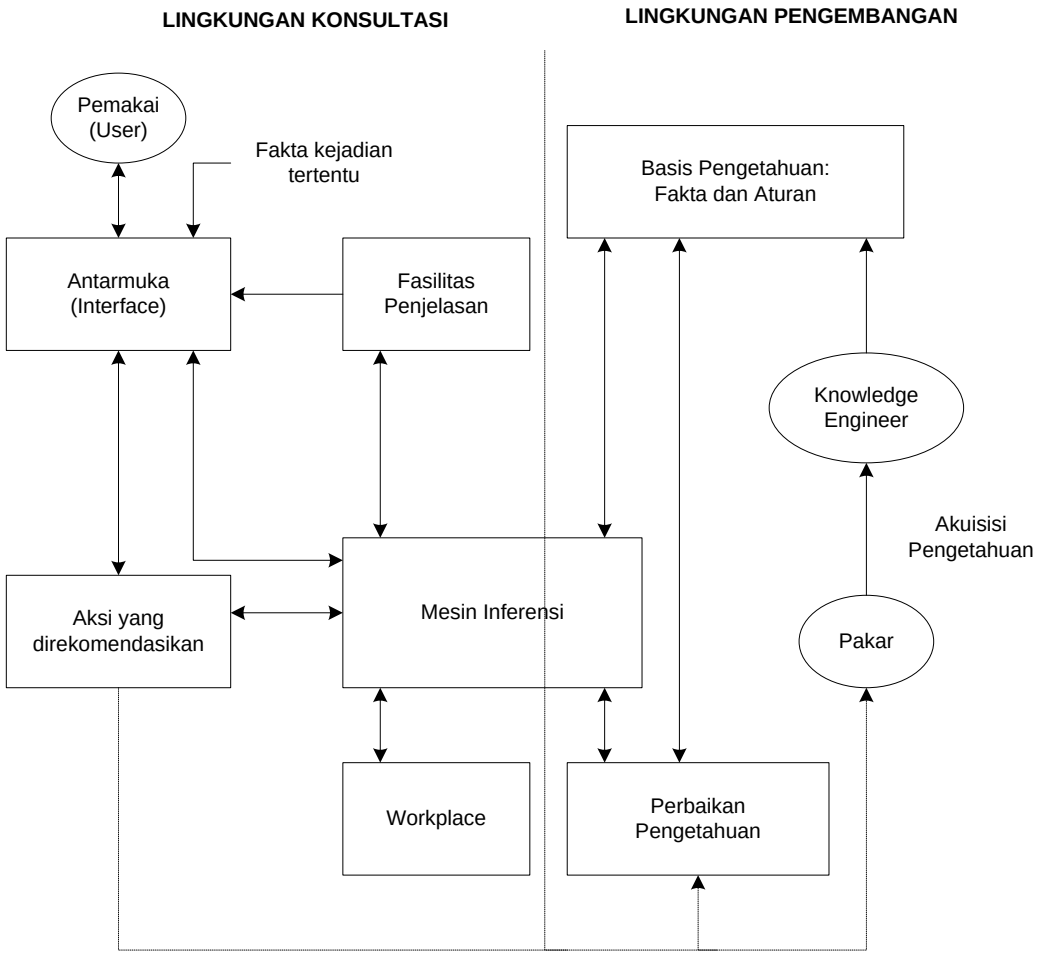
Ciri-ciri dari sistem pakar yaitu (Efraim, 2005) :

1. Terbatas pada bidang yang spesifik
2. Dapat memberikan penalaran untuk data-data yang tidak lengkap atau tidak pasti
3. Dapat mengemukakan rangkaian alasan yang diberikannya dengan cara yang dapat dipahami.
4. Berdasarkan pada *rule* atau kaidah tertentu.
5. Dirancang untuk dapat dikembangkan secara bertahap.
6. *Output* nya bersifat nasihat atau anjuran.
7. *Output* tergantung dari dialog dengan user.
8. *Knowledge base* dan *inference engina* terpisah.

2.1.4 Arsitektur Sistem Pakar

Sistem pakar dapat ditampilkan dengan dua lingkungan, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*). Lingkungan pengembangan digunakan oleh sistem

pakar (ES) *builder* untuk membangun komponen dan memasukkan pengetahuan ke dalam basis pengetahuan. Lingkungan konsultasi digunakan oleh nonpakar untuk memperoleh pengetahuan dan nasihat pakar. Lingkungan ini dapat dipisahkan setelah sistem lengkap. Gambar Struktur Sistem Pakar pada penelitian ini dapat dilihat pada gambar berikut ini (Efraim, 2005)



Gambar 2.1 Arsitektur Sistem Pakar *(Sumber :* Efraim, 2005*)*

1. Pakar

Merupakan orang yang memiliki pengetahuan, penilaian, pengalaman dan metode khusus, serta kemampuan untuk menerapkan bakat ini dalam memberikan nasihat dan memecahkan persoalan.

2. **Akuisisi Pengetahuan**

   Merupakan akumulasi, transfer dan transformasi keahlian pemecahan masalah dari pakar atau sumber pengetahuan terdokumentasi ke program komputer, untuk membangun atau memperluas basis pengetahuan. Sumber pengetahuan potensial antara lain pakar manusia, buku teks dan dokumen multimedia.

3. ***Knowledge Engineer***

   Yaitu seorang spesialis sistem yang menterjemahkan pengetahuan yang dimiliki seorang pakar menjadi pengetahuan yang akan tersimpan dalam basis pengetahuan pada sebuah sistem pakar.

4. **Basis Pengetahuan**

   Berisi pengetahuan relevan yang diperlukan untuk memahami, merumuskan, dan memecahkan persoalan. Basis pengetahuan mencakup dua elemen dasar, yaitu :

   1. Fakta, misalnya situasi persoalan dan teori area persoalan (apa yang diketahuki tentang area domain).
   2. *Rule* atau aturan khusus yang mengarahkan penggunaan pengetahuan untuk memecahkan persoalan khusus dalam domain tertentu (referensi logika, misalnya, antara gejala dan penyebab).

5. **Perbaikan Pengetahuan**

   Pakar manusia memiliki sistem perbaikan pengetahuan, yakni mereka dapat menganalisis pengetahuannya sendiri kegunaannya, belajar darinya, dan meningkatkannya untuk konsultasi mendatang. Serupa pula, evaluasi tersebut diperlukan dalam pembelajaran komputer sehingga program dapat menganalisisk alasan keberhasilan atau kegagalannya. Hal ini dapat mengarah kepkada peningkatan sehingga menghasilkan basis pengetahuan yang lebih akurat dan pertimbangan yang lebih efektif. Dengan komponen ini, pakar mampu menganalisis kinerja dari Sistem pakar, belajar daripadanya, dan meningkatkannya pada konsultasi selanjutnya.

6. Mesin Inferensi

Merupakan otak dari Sistem pakar. Komponen ini sebenarnya adalah program komputer yang menyediakan metodologi untuk *reasoning* (pertimbangan) mengenai informasi dalam basis pengetahuan dan dalam ”*workplace*”, dan digunakan untuk merumuskan kesimpulan.

Mesin Inferensi mempunyai 3 elemen utama, yaitu :

1. *Interpreter* adalah elemen yang mengeksekusi item agenda yang dipilih dengan mengaplikasikannya pada basis pengetahuan *rule* yang berhubungan.
2. *Scheduler* adalah elemen yang menjaga kontrol di sepanjang agenda. Memperkirakan akibat dari pengaplikasian *rule inferensia* yang menampakkan prioritas *item* atau kriteria lain pada agenda.
3. *Consistency enforcer* adalah elemen yang mencoba menjaga konsistensi representasi solusi yang muncul.

7. *Workplace*

Merupakan area kerja memori yang disimpan sebagai *database* untuk deskripsi persoalan terbaru yang ditetapkan oleh data *input*, digunakan juga untuk perekaman hipotesis dan keputusan sementara.

Tiga tipe keputusan dapat direkam dalam *workplace*: rencana (bagaimana mengatasi persoalan), agenda (tindakan potensial sebelum eksekusi), dan solusi (hipotesis kandidat dan arah tindakan alternatif yang telah dihasilkan sistem sampai dengan saat ini).

8. Fasilitas Penjelasan

Ini adalah kemampuan penelusuran kebenaran dari konklusi yang didapat dari sumber-sumbernya. Hal ini krusial untuk transformasi kepakaran dan penyelesaian masalah. Komponen ini mampu menelusuri kebenaran dan untuk menerangkan perilaku Sisem Pakar secara interaktif, menjawab pertanyaan seperti: Mengapa pertanyaan tertentu ditanyakan oleh Sistem pakar? Bagaimana konklusi tertentu dicapai? Mengapa alternatif tertentu ditolak? Rencana apakah yang ada untuk mencapai solusi? Dan apa-apa

saja selanjutnya yang harus dilakukan sebelum diagnosis final dapat ditentukan?

9. **Antarmuka Pemakai (*User Interface*)**

   Sistem pakar berisi *prosessor* bahasa untuk komunikasi berorientasi persoalan yang mudah antara pengguna dan komputer. Komunikasi ini paling baik dilakukan dalam bahasa alami. Dikarenakan batasan teknologi, maka kebanyakan sistem yang ada menggunakan pendekatan pertanyaan dan jawaban untuk berinteraksi dengan pengguna.

10. **Aksi yang direkomendasikan**

    Merupakan saran atau solusi yang direkomendasikan untuk permasalahan yang sedang dihadapi oleh *user*.

11. ***User***

    Umumnya *user* yang dimaksud ini adalah:

    1. *Klien* (yaitu bukan pakar) yang menginginkan *advis*/nasehat. Di sini Sistem pakar bertindak seperti seorang konsultan atau penasehat.
    2. *Learner* (pelajar) untuk mempelajari bagaimana Sistem pakar menyelesaikan permasalahan. Disini Sistem pakar bertindak sebagai seorang instruktur.
    3. Pembangun Sistem pakar yang ingin meningkatkan basis pengetahuannya. Di sini Sistem pakar bertindak sebagai seorang rekan.
    4. Pakar. Di sini Sistem pakar bertindak sebagai seorang kolega atau asisten.

2.1.5 Representasi Pengetahuan

Pengetahuan merupakan kemampuan untuk membentuk model mental yang menggambarkan obyek dengan tepat dan merepresentasikannya dalam aksi yang dilakukan terhadap suatu obyek.

Representasi pengetahuan merupakan metode yang digunakan untuk mengodekan pengetahuan dalam sebuah sistem pakar yang berbasis pengetahuan. Perepresentasian dimaksudkan untuk menangkap sifat-sifat penting problema dan

membuat informasi itu dapat diakses oleh prosedur pemecahan problema (Efraim, 2005)

2.1.6 Keuntungan Sistem Pakar

Banyak manfaat yang dapat diambil dengan adanya sistem pakar, antara lain (Efraim, 2005) :

1. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli.
2. Bisa melakukan proses secara berulang secara otomatis.
3. Menyimpan pengetahuan dan keahlian para pakar.
4. Meningkatkan output dan produktivitas.
5. Meningkatkan kualitas.
6. Mampu mengambil dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka).
7. Mampu beroperasi dalam lingkungan yang berbahaya.
8. Memiliki kemampuan untuk mengakses pengetahuan.
9. Memiliki reliabilitas.
10. Meningkatkan kapabilitas sistem komputer.
11. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian.
12. Sebagai media pelengkap dalam pelatihan.
13. Meningkatkan kapabilitas dalam penyelesaian masalah.
14. Menghemat waktu dalam pengembilan keputusan.

2.1.7 Kelemahan Sistem Pakar

Disamping memiliki keuntungan, sistem pakar juga memiliki beberapa kelemahan, antara lain (Efraim, 2005) :

1. Biaya yang diperlukan untuk membuat dan memeliharanya sangat mahal.
2. Daya kerja dan produktivitas manusia menjadi berkurang karena semuanya dilakukan oleh sistem.
3. Sulit dikembangkan, hal ini tentu saja erat kaitannya dengan ketersediaan pakar di bidangnya.
4. Sistem pakar tidak 100% bernilai benar.

2.2 Metode *Case Based Reasoning* (CBR)

Case Based Reasoning (CBR), adalah proses pemecahan masalah baru berdasarkan solusi dari masalah masa lalu yang sama. *Case Based Reasoning* (CBR) adalah sebuah pendekatan yang menggunakan kasus-kasus lama/ pengalaman untuk memahami dan memecahkan masalah baru. Pendekatan CBR terdiri dari menciptakan pengetahuan dasar (atau *database*) berisi kasus-kasus masa lalu (produk). Mendefinisikan kasus baru (konsep), mengambil kasus serupa dengan kasus baru, dan menyesuaikan solusi dari kasus diambil untuk kasus baru. (Muzid, 2008)

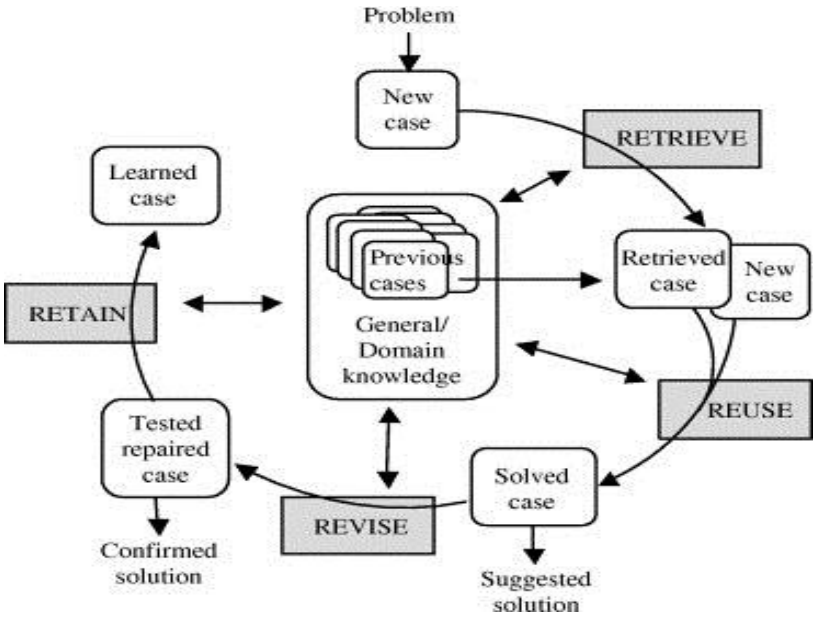
Sedangkan menurut Irawan (2009) *Case Based Reasoning* (CBR) adalah salah satu metode untuk membangun sistem pakar dengan pengambilan keputusan dari kasus yang baru dengan berdasarkan solusi dari kasus-kasus sebelumnya. Konsep dari metode *case based reasoning* ditemukan dari ide untuk menggunakan pengalaman-pengalaman yang terdokumentasi untuk menyelesaikan masalah yang baru. Para *decision maker* kebanyakan menggunakan pengalaman-pengalaman dari *problem solving* terdahulu untuk menyelesaikan masalah yang dihadapi sekarang.

Case Based Reasoning (CBR) merupakan metode yang digunakan untuk mengimplementasikan sistem diagnosa komputer ke dalam aplikasi di dunia nyata. CBR juga dapat digunakan untuk menganalisa suatu masalah sesuai dengan kasus yang dihadapi dan untuk selanjutnya mengklasifikasikan kasus tersebut berdasarkan pada pengalaman masa lalu pengklasifikasian. Kelebihan dari CBR yaitu memungkinkan penggunaan contoh kasus masa lalu untuk mengakuisisi pengetahuan dan akhirnya diketahui pokok permasalahannya. Selain itu CBR juga dapat mencari solusi dari permasalahan tersebut berdasarkan dari pengalaman kasus masa lalu sehingga segala permasalahan dapat diselesaikan untuk selanjutnya kasus serta solusinya disimpan untuk kemudian dapat digunakan kembali untuk memecahkan kasus baru, jika kasus tersebut hampir sama atau mungkin sama dengan kasus terdahulu.

Tahapan CBR, secara keseluruhan model CBR Cycle dapat digambarkan dengan proses sebagai berikut (Yulmaini, Lestari and Antonio, 2020)

a. *Retrieve*, merupakan proses untuk mendapatkan kembali kasus terdahulu yang serupa dengan kasus yang sedang dihadapi.
b. *Reuse*, merupakan proses untuk menggunakan kembali informasi dan pengetahuan dalam kasus terdahulu untuk menyelesaikan masalah yang dihadapi.
c. *Revise*, merupakan proses memperbaiki solusi yang telah ada sebelumnya.
d. *Retain*, merupakan proses penyimpanan kasus baru dan solusinya untuk digunakan dalam menyelesaikan kasus berikutnya.

Keempat proses di atas akan terus dilakukan ketika menghadapi kasus baru. Model CBR tersebut dapat disajikan pada Gambar 2.2.



Gambar 2.2 Tahapan CBR (Sari *et al.*, 2008)

Kelebihan CBR (Yulmaini, Lestari and Antonio, 2020) :

1. Pada CBR tidak memerlukan pemahaman bagaimana menyelesaikan masalah sehingga mengurangi dampak penambahan informasi pengetahuan.
2. Pada CBR pengetahuan diperoleh dengan cara mengumpulkan kejadian-kejadian yang telah terjadi dan tidak memerlukan suatu model yang explisit.
3. Pada CBR memiliki kemampuan untuk belajar, cara yang dilakukan yaitu dengan menambahkan kasus baru seiring waktu berjalan dan tanpa perlu menambahkan aturan baru atau mengubah yang sudah ada
4. Kemampuan untuk mendukung justifikasi dengan menawarkan kasus lampau lebih diutamakan.

Kekurangan CBR (Yulmaini, Lestari and Antonio, 2020):

1. Memerlukan perhitungan yang kompleks dan melakukan pengukuran kesamaan antar dua objek agar dapat menyelesaikan masalah secara keseluruhan.
2. Seringkali menghadapi kesulitan dalam menentukan fitur-fitur dari kasus agar bisa dibandingkan.
3. Semakin banyak kasus-kasus tersimpan dalam *databased*, maka dalam pencarian kasus yang paling mirip dengan kasus baru membutuhkan waktu lebih lama
4. Solusi yang ditawarkan belum tentu yang terbaik, hal ini karena tergantung pada kualitas kasus-kasus sebelumnya yang tersimpan dalam *databased*. Oleh karena itu tahap *Revise* menjadi hal yang penting untuk dilakukan sehingga mengurangi tingkat kesalahan.

2.3 Metode Naive Bayes Classifier

Pendekatan Naive Bayes merupakan sebuah metode klasifikasi yang mengacu pada teorema Bayes. Teorema Bayes digunakan untuk menghitung probabilitas ketidakpastian data (Russell and Norvig, 2010). Proses pendekatan Naïve Bayes Classifier mengasumsikan bahwa ada atau tidaknya suatu fitur pada suatu kelas tidak berhubungan dengan ada atau tidaknya fitur lain di kelas yang sama (Setiawan and Ratnasari, 2014). Pada saat klasifikasi, pendekatan Bayes akan menghasilkan tabel kategori yang paling tinggi nilai probabilitasnya yaitu VMAP (*Maximum* Apriori Probability) dengan atribut inputan G1, G2,… Gn (Aribowo, 2010). Pernyataan tersebut dapat dituliskan dengan formula sebagai berikut :

$$P(wj \mid x) = \frac{P(wj) \times P(x \mid wj)}{P(x)}$$

Perhitungan Naïve Bayes adalah sebagai berikut (Setiawan and Ratnasari, 2014) :

$$P(Gi|Vj) = \frac{nc+m.\ p}{n+m}$$

Dimana :

nc = jumlah *record* pada data *training* yang V = Vj dan G = Gi

p = peluang jenis penyakit (prior)

m = jumlahk gejala

n = jumlah *record* pada kdata *training* yang V = Vj tiap class (penyakit)

Persamaan diatas dapat diselesaikan melalui perhitungan sebagai berikut :

1. Menentukan nilai nc untuk setiap *class*
2. Menentukan nilai P(Vj) atau nilai prior. Pada penelitian ini nilai prior diperoleh dari penyakit yang dipilih dan tidak dipilih jika dipilih berniali 1 jika tidak bernialai 0
3. Menghitung nilai P(Gi|Vj)

$$V_{MAP} = argmax_{vj \in v} P(Vj) \prod i\ P(Gi|Vj)$$

Dimana :

P(Vj) = Peluang jenis penyakit kej nilai prior penyakit ke_j

$$P(Gi|Vj) = \frac{nc+m.\ p}{n+m}$$

4. Menghitung P(Vj) x P(Gi|Vj) untuk tiap V
5. Menentukan hasil klasifikasi yaitu V yang memiliki hasil perkalian yang terbesar

2.4 *Unified Modelling Language* (UML)

UML yang merupakan singkatan dari *Unified Modelling Language* adalah sekumpulan pemodelan konvensi yang digunakan untuk menentukan atau menggambarkan sebuah sistem perangkat lunak dalam kaitannya dengan objek.

UML dapat juga diartikan sebuah bahasa grafik standar yang digunakan untuk memodelkan perangkat lunak berbasis objek. UML pertama kali dikembangkan pada pertengahaan tahun 1990an dengan kerjasama antara James Rumbaugh, Grady Booch dan Ivar Jacobson, yang masing-masing telah mengembangkan notasi mereka sendiri di awal tahun 1990an (Rosa & Salahuddin, 2013)

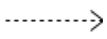
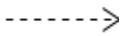
2.4.1 Komponen-komponen UML

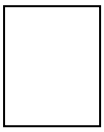
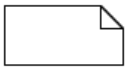
UML mendefinisikan diagram-diagram berikut ini (Rosa & Salahuddin, 2013):

a. *Use case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem.

Tabel 2.1 Simbol *Use Case.*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalizatio n*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.

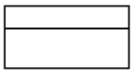
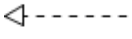
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

Sumber : (Rosa & Salahuddin, 2013)

b. Class Diagram

1) Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).
2) Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain.

Tabel 2.2. Simbol *Class Diagram*.

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

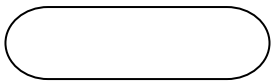
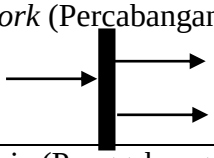
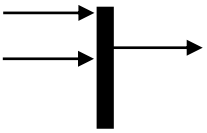
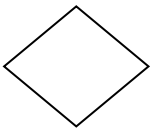
Sumber : (Rosa & Salahuddin, 2013)

c. Activity Diagram

1) *Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beber apa eksekusi.
2) *Activity diagram* merupakan state diagram khusus, yang sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu,

activity diagram tidak menggambarkan perilaku internal sebuah sistem dan interaksi antar subsistem, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Tabel 2.3 Simbol *Activity* Diagram.

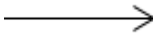
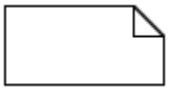
No.	Simbol	Keterangan
1	*Start State* 	*Start state* adalah sebuah kondisi awal sebuah *object* sebelum ada perubahan keadaan. Start state digambarkan dengan sebuah lingkaran solid.
2.	*End State* 	*End state* adalah menggambarkan ketika objek berhenti memberi respon terhadap sebuah event. *End state* digambarkan dengan lingkaran solid di dalam sebuah lingkaran kosong.
3.	*State/Activities* 	*State* atau *activities* menggambarkan kondisi sebuah entitas, dan digambarkan dengan segiempat yang pinggirnya.
4.	*Fork* (Percabangan) 	*Fork* atau percabangan merupakan pemisalah beberapa aliran konkuren dari suatu aliran tunggal.
5.	*Join* (Penggabungan) 	*Join* atau penggabungan merupakan pengabungan beberapa aliran konkuren dalam aliran tunggal.
6.	*Decision* 	*Decision* merupakan suatu logika aliran konkuren yang mempunyai dua cabang aliran konkuren.

Sumber : (Rosa & Salahuddin, 2013)

d. Sequence Diagram

Sequence diagram secara grafis menggambarkan bagaimana objek berinteraksi antara satu sama lain melalui pesan pada sebuah *use case* atau operasi.

Tabel 2.4 Simbol *Sequence Diagram.*

NO	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		*Node*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Sumber : (Rosa & Salahuddin, 2013)

2.5 Basis Data

Database adalah kumpulan file-file yang mempunyai kaitan antara satu file dangan file yang lain sehingga membentuk satu bangunan data untuk menginformasikan satu perusahaan, instansi dalam batasan tertentu (Nugroho, 2011).

Istilah-istilah yang digunakan dalam basis data:

1) *File* : merupakan kumpulan dari atribut *record-record* sejenis yang mempunyai panjang elemen yang sama, atribut yang sama namun berbeda-beda dalam data *value*-nya.

2) *Record* : merupakan kumpulan dari elemen-elemen yang saling berhubungan atau berkaitan menginformasikan tentang *entry* secara lengkap.

3) *Field* : merupakan sekumpulan tanda-tanda yang berbentuk kesatuan tersendiri, merupakan bagian terkecil dari *record* dan bentuknya unik dijadikan *field* kunci yang dapat mewakili *record*-nya.

4) Entity : merupakan tempat kejadian atau konsep yang informasikan direkam.

2.6 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Dimana 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*). SQL adalah sebuah konsep pengoperasian database, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis (Nugroho, 2011).

2.7 Metode Pengumpulan Data

Metode dan instrumen yang sering diartikan sama, padahal berbeda. Metode dan instrumen ini berkenaan dengan cara bagaimana memperoleh data yang diperlukan. Metode lebih menekankan pada strategi, proses dan pendekatan dalam memilih jenis karakteristik serta dimensi ruang dan waktu dari data yang diperlukan. Sedangkan instrumen menekankan kepada alat atau cara untuk menjaring data yang dibutuhkan. Untuk mengumpulkan data dari sampel

penelitian, harus dilakukan dengan metode tertentu sesuai dengan tujuannya. Terdapat 5 metode yang sering digunakan oleh seorang peneliti, yaitu wawancara, pengamatan (observasi), *kuesioner* (angket), *documenter* dan *survey*. Metode yang dipilih untuk setiap variabel tergantung pada berbagai faktor terutama jenis data dan ciri responden. Metode pengumpulan data ini tergantung pada karakteristik data variabel, maka metode yang digunakan tidak selalu sama, untuk di setiap variabelnya. Dalam satu variabel dapat menggunakan dua metode atau lebih, hal ini disebabkan karena yang pertama adalah metode utama dan yang lainnya untuk control silang. Jika satu metode dipandang mencukupi, maka metode lain tidak perlu digunakan dan tidak efisien. Dalam metode pengumpulan data ini terdapat jenis sumber data. Jenis sumber data ialah pengambilan data yang dihimpun langsung oleh peneliti disebut sumber primer, sedangkan apabila melalui tangan kedua disebut sumber sekunder. Data yang dikumpulkan dalam penelitian digunakan untuk menguji hipotesis atau menjawab pertanyaan yang telah dirumuskan, karena data yang diperoleh akan dijadikan landasan dalam mengambil kesimpulan, dan data yang dikumpulkan haruslah data yang benar. Ada beberapa metode yang dapat digunakan dalam pengumpulan data, yang akan dibahas di bawah ini.

2.7.1 Wawancara

Sugiyono (2009) menerangkan bahwa wawancara adalah pertemuan dua orang untuk bertukar informasi dan ide melalui tanya jawab sehingga dapat dikonstruksikan makna dalam suatu topik tertentu dan dengan wawancara, peneliti akan mengetahui hal-hal yang lebih mendalam tentang partisipan dalam menginterprestasikan situasi dan fenomena yang terjadi yang tidak mungkin bisa ditemukan melalui observasi.

2.7.2 Studi Pustaka

Peneliti membaca, mengutip dan membuat catatan yang bersumber pada bahan-bahan pustaka yang mendukung dan berkaitan dengan penelitian ini khususnya dalam pengembangan Sistem. Selanjutnya dengan cara mempelajari dan memahami jurnal dan buku-buku referensi, yang

berhubungan dengan masalah yang akan dibahas dalam karya ilmiah ini. Hal ini dimaksudkan agar penulis memiliki landasan teori yang kuat terkait dengan metode *Case Based Reasoning*.

2.8 Metode Pengembangan Perangkat Lunak Menggunakan Metode *Prototype*

Metode penelitian yang digunakan penulis didasarkan pada metoda pengembangan perangkat lunak prototype yang memiliki lima tahap (Pressman, 2002). Alasan penulis mengambil metode pengembangan sistem prototype karena pengguna dapat dengan mudah beradaptasi dengan sistem yang dikembangkan sesuai dengan kebutuhan pemakai. Selain itu pada saat tahap eveluasi sistem, jika terdapat revisi pada sistem yang dirancang, programmer tidak harus mengulang tahapan dari awal.

Gambar 2.3 Prototyping Model

Sumber : (Pressman, 2002)

Berikut penjelasan tahapan metoda prototype yang digunakan dalam penelitian ini.

1. Communication.
   Penulis menganalisis sistem dengan mewawancarai pihak-pihak fasilitas dan pengguna sebagai pemakai untuk mendapatkan gagasan dari apa yang diinginkan pemakai terhadap sistem yang dibuat.

2. Quick Plan
   Analisis sistem bekerja sama dengan spesialis informasi lain pihak, menggunakan satu atau lebih peralatan prototyping untuk mengembangkan sebuah prototype. (misalnya dengan menambahkan tampilan yang diinginkan pemakai dalam sistem baru seperti menu, interface, fitur, dan database).

3. Modelling Quick Design
   Analisis membuat perancangan sistem untuk mengembangkan prototipe dengan menggunakan UML, sebagai dasar perancangan aplikasi. Dalam hal ini programmer menggunakan PHP sebagai bahasa pemrograman dan MySQL sebagai database.

4. Construction Of Prototype
   Tahapan ini adalah tahapan yang dilakukan setelah kegiatan analisi dan perancangan. Pada bagian ini akan dijelaskan kegiatan-kegiatan yang dilakukan pada tahap pengkodean (coding) sistem operasional, implementasi pembuatan program (programming) dan pengujian (testing).

5. Deployment delevery & feedback
   Penulis yang menguji sistem baru tersebut dan melakukan uji coba terhadap beberapa calon pengguna sehingga penulis dapat menentukan apakah sistem baru dapat diterima. Pada tahap ini pemakai memberi masukan kepada analis apakah sistem dapat diterima. Jika ya sistem baru yang telah diuji dan terima oleh pengguna aplikasi, jika tidak, langkah 4 dan 5 diulangi.

2.8.1 Kelebihan metode prototyping

yang paling utama adalah merupakan salah satu jenis metode pengembangan sistem yang sifatnya sangat cepat dan dapat menghemat waktu. Berbeda dengan pengembangan sistem menggunakan metode waterfall yang membutuhkan banyak biaya dan memakan waktu. Maka bagi user yang membutuhkan sebuah sistem dalam jangka waktu yang sangat singkat, bisa mengandalkan metode pengembangan sistem prototyping ini.

Selain itu, metode prototyping juga memilki beberapa kelebihan lainnya, seperti:

1. Dapat menjalin komunikasi yang baik antar user dan pengembang sistem
2. Setiap perbaikan yang dilakukan pada prototype merupakan hasil masukan dari user yang akan menggunakan sistem tersebut, sehingga lebih reliabel
3. User akan memberikan masukan terhadap sistem sesuai dengan kemauannya
4. Menghemat waktu dalam mengembangkan sebuah sistem
5. Menghemat biaya, terutama pada bagian analisa, karena hanya mencatat poin – point penting saja
6. Cocok digunakan pada sebuah sistem kecil, yang digunakan pada ruang lingkup tertentu, seperti sistem di dalam sebuah kantor
7. Penerapan dari sistem yang menjadi lebih mudah untuk dilakukan.

2.8.2 Kelemahan dari Metode Prototyping

Beberapa kelemahan dan juga kekurangan dari metode prototyping antara lain:

1. Untuk menghemat waktu, biasanya pengembang hanya menggunakan bahasa pemrograman sederhana, yang mungkin rentan dari segi keamanannya
2. Tidak cocok untuk diimplementasikan pada sebuah sistem yang sangat besar dan global, seperti sistem operasi komputer.

2.8.3 Pengimplementasian Sistem Menggunakan Metode Prototyping

Metode prototyping dalam pengembangan sistem cocok untuk digunakan pada sistem yang ingin cepat diselesaikan, dan biasanya berskala kecil, dengan fungsi yang tidak besar. Contoh dari implementasi metode ini adalah pembuatan game quiz yang berisikan banyak pertanyaan untuk user.

Pengembang akan memberikan prototype dari game quiz yang akan dikembangkan, lalu user akan menentukan kekurangan dari game quiz yang menjadi prototype, dan kemudian pengembang akan melakukan perbaikan hingga game tersebut bisa dimainkan oleh semua user, tanpa ada kendala

2.9 Pengujian Sistem

Pada penelitian ini dilakukan pengujian sistem menggunakan *BlackBox Testing*. Menurut (Dondeti, 2012). Tester menggunakan behavioral test (disebut juga Black-Box Tests), sering digunakan untuk menemukan bug dalam high level operations, pada tingkatan fitur, profil operasional dan skenario customer. Tester dapat membuat pengujian fungsional black box berdasarkan pada apa yang harus sistem lakukan. Behavioral testing melibatkan pemahaman rinci mengenai domain aplikasi, masalah bisnis yang dipecahkan oleh sistem dan misi yang dilakukan sistem. Behavioral test paling baik dilakukan oleh penguji yang memahami desain sistem, setidaknya pada tingkat yang tinggi sehingga mereka dapat secara efektif menemukan bug umum untuk jenis desain.

Pengujian menggunakan sekumpulan aktifitas validasi, dengan pendekatan black box testing. Menurut (Rosa & Salahuddin, 2013), black box testing adalah menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian black box testing harus dibuat dengan kasus benar dan kasus salah.

Menurut Pressman (2002) black box testing juga disebut pengujian tingkah laku, memusat pada kebutuhan fungsional perangkat lunak. Teknik pengujian black box memungkinkan memperoleh serangkaian kondisi masukan yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Beberapa jenis kesalahan yang dapat diidentifikasi adalah fungsi tidak benar atau hilang, kesalahan antar muka, kesalahan pada struktur data (pengaksesan basis data), kesalahan performasi, kesalahan inisialisasi dan akhir program.

Tabel 2.5 Rancangan Pengujian Sistem

Test Case Name			Test Date	
Test Case Version		Halaman	Tester	
A.1 Login				
Test ID	Description	Expected Result	Actual Result	Test Result
				() pass () fail

Keterangan Komponen Pengujian:

1. *Test case name*:Nama aplikasi yang akan diuji.
2. *Test Date*:Tanggal saat pengujian.
3. *Test case version*:Jenis pengujian.
4. *Tester*: Siapa yang meguji aplikasi pengujiannya.
5. *Test ID*: Nomor urut saat pengujian.
6. *Description*: Keterangan pada tahap yang akan diuji

7. *Expected Result*: Hasil yang diharapkan pada saat pengujian.
8. *Actual Result*: Realisasi pada hasil pengujian.
9. *Test Result*: Hasil dari pengujianya**.**

2.10 Penelitian Terkait

Berikut ini adalah penelitian terkait yang digunakan pada penelitian, dapat dilihat pada Tabel 2.6 :

Tabel 2.6 Penelitian Terkait

No	Nama (Tahun)	Judul	Metode	Hasil
1	Sabransyah *et al.*, (2017)	Aplikasi Metode Naive Bayes dalam Prediksi Risiko Penyakit Jantung Naive Bayes Method for a Heart Risk Disease Prediction Application	Metode Naive Bayes	Dimana pada penelitian ini membahas ciri-ciri penyakit jantung bisa dikenali sejak dini, tapi banyak orang yang belum dibekali pengetahuan yang cukup mengenai penyakit jantung menyebabkan banyak orang yang terlambat mengetahui jika dirinya terkena penyakit jantung. Penelitian ini menghasilkan Aplikasi Prediksi Risiko Penyakit Jantung Menggunakan Metode Naive Bayes
2	Ma'rifati & Kesuma (2018)	Pengembangan Sistem Pakar Mendeteksi Penyakit Pencernaan Menggunakan Metode Naive Bayes Berbasis Web	Metode *Naive Bayes Berbasis Web*	Dimana pada penelitian ini membahas Penyakit pencernaan adalah penyakit yang menyerang saluran pencernaan. Orang yang terkena penyakit ini akan mengalami gangguan pencernaan seperti diare, maag, dan lain-lain. Penelitian ini menghasilkan website sistem pakar pendeteksi penyakit pencernaan menggunakan Metode Naive Bayes Classifier.
3	Widiyawati & Imron (2018)	Sistem Pakar Diagnosa	*Metode Naive Bayes*	Dimana pada penelitian ini membahas Kucing tidak

		Penyakit Pada Kucing Menggunakan Metode *Naive Bayes Classifier*	*Classifier*	lepas dari virus dan penyakit yang menyerangnya untuk itu, pemilik harus rajin meneliti perkembangan kondisi kucing agar jika terserang suatu penyakit atau diserang virus dapat segera di kenali sedini mungkin, banyak pemelihara yang tidak menyadari bahwa kucing yang mereka miliki mengidap suatu penyakit. Penelitian ini menghasilkan Aplikasi Sistem Pakar Diagnosa Penyakit Pada Kucing Menggunakan Metode Naive Bayes Classifier.
4	Wicaksono *et al*. (2016)	Analisis dan Implementasi Sistem Pendiagnosis Penyakit Tuberculosis Menggunakan Metode Case-Based Reasoning.	Metode *Case-Based Reasoning.*	Dimana pada penelitian yang dilakukan peneliti mengangkat masalah penyakit Tuberculosis atau yang dikenal dengan TB. Penelitian ini mengembangkan suatu aplikasi berbasis pengetahuan yang bertujuan untuk membantu dokter dalam mendiagnosis penyakit TB. Aplikasi atau sistem yang dikembangkan ini menggunakan metode Case-Based Reasoning (CBR) dengan perhitungan similarity antara kasus lama dan baru pada sistem menggunakan probabilitas Bayes.
5	Aribowo (2018)	Pengembangan Sistem Cerdas Menggunakan Penalaran Berbasis Kasus (Case Based Reasoning) Untuk	*Case Based Reasoning*	Dimana pada penelitian yang dilakukan peneliti mengangkat masalah penyakit yang diakibatkan oleh virus *Eksantema* contohnya adalah cacar air, campak, variola, dan sebagainya dengan gejala

		Mendiagnosa Penyakit Akibat Virus Eksantema.		yang hampir mirip satu sama lain. Penelitian ini menghasilkan sistem pakar yang dapat mendiagnosa penyakit yang diakibatkan oleh virus *Eksantema*. Sistem pakar dalam penelitian ini menggunakan pendekatan penalaran berbasis kasus. Jika ada kasus yang mirip maka penalaran untuk menimbang kasus terdekat menggunakan metode Probabilitas Bayes.
6	Harjanto et al. (2018)	Pemanfaatan Aplikasi Sistem Pakar Untuk Diagnosa Penyakit Paru – Paru	Metode *forward chaining*	Dimana pada penelitian yang dilakukan peneliti mengangkat masalah konseling Sistem Pakar Konsultasi Perilaku Siswa ini dengan tujuannya yaitu dapat memudahkan pengguna yaitu guru bimbingan konseling dalam mengkonsultasi perilaku siswa dan memberikan solusi sebagai hasil konsultasi untuk penanganan dari permasalahan siswa. Penelitian ini menghasilkan Rancang bangun yang di fasilitasi sebuah Program sistem pakar bermanfaat untuk mengkonsultasi perilaku pada siswa dan digunakan mengetahui strategi penanganan dalam keputusan yang harus dilakukan oleh guru bimbingan konseling di sekolah guna menangani perilaku siswa. Pembuatan aplikasi ini peneliti menggunakan tools MySQL sebagai database dan untuk Bahasa pemrograman mengunakan

				PHP yang ditulis dengan aplikasi Adobe Dreamweaver. Sistem pakar menggunakan *forward chaining* melalui penyesuain dan dan kebutuhan proses.
7	Yulmaini et al. (2020)	Sistem Pakar Berbasis Web untuk Mendiagnosis Mesin Mobil dengan Metode *Forward Chaining*	Metode Forward Chaining.	Suatu sistem deteksi kerusakan mesin mobil merupakan sebuah sistem yang menggunakan pengetahuan manusia yang telah disimpan didalam komputer untuk menyelesaikan permasalahan yang umumnya memerlukan kepakaran dari seseorang. Sistem yang didesain dengan baik dan meniru pola pikir yang digunakan oleh seseorang pakar untuk menyelesaikan masalah. Dengan menggunakan metode forward chaining dapat mendeteksi sumber masalah dan penyebab kerusakan yang terjadi pada kendaraan khususnya mobil panther Karena metode forward chaining menggunakan pendekatan data-driven, yang proses pencariannya memulai dari premis atau data menuju konklusi.
8	Fitria *et al*., (2018)	Metode *Case Based Reasonig* (Cbr) Pada Sistem Diagnosa Penyakit Kulit.	*Case Based Reasonig* (Cbr)	Penelitian ini menggunakan Metode Case Based Reasoning(CBR), dimana CBR merupakan salah satu metode pendekatan berbasis pengetahuan untuk mempelajari dan memecahkan berdasarkan masalah pengalaman pada masa lalu yang disimpan didalam sistem yang

				dinamakan basis kasus. Pada kasus ini terdapat 44 gejala yang berbeda dan 14 jenis penyakit kulit. Cara pengelolaan data similarity menggunakan nilai atribut dengan nilai kedekatan, nilai – nilai tersebut dihitung dengan menggunakan metode Nearest Neighbor untuk mencari nilai kemiripan yang paling tinggi dari 14 kasus lama yang ada. Berdasarkan hasil input dari contoh kasus yang ada menunjukkan bahwa hasil perhitungan manual dengan perhitungan sistem sama. Sistem ini dapat membantu mendiagnosa jenis penyakit kulit dengan pengambilan kesimpulan yang didapat dari hasil nilai similarity tertinggi dari kasus yang ada berdasarkan gejala-gejaja yang diinputkan oleh user (pasien), dan sistem juga dapat memberikan solusi cara penanganannya.