BAB 2
LANDASAN TEORI

2.1 Pengertian Puskesmas

Puskesmas (Pusat Kesehatan Masyarakat) adalah suatu organisasi kesehatan fungsional yang merupakan pusat pengembangan kesehatan masyarakat yang juga membina peran serta masyarakat di samping memberikan pelayanan secara menyeluruh dan terpadu kepada masyarakat di wilayah kerjanya dalam bentuk kegiatan pokok. Menurut Depkes RI (2004) puskesmas merupakan unit pelaksana teknis dinas kesehatan kabupaten/kota yang bertanggung jawab menyelenggarakan pembangunan kesehatan di wilayah kerja (Effendi, 2009).

Ada 3 (tiga) fungsi puskesmas yaitu: pusat penggerak pembangunan berwawasan kesehatan yang berarti puskesmas selalu berupaya menggerakkan dan memantau penyelenggaraan pembangunan lintas sektor termasuk oleh masyarakat dan dunia usaha di wilayah kerjanya, sehingga berwawasan serta mendukung pembangunan kesehatan. Disamping itu puskesmas aktif memantau dan melaporkan dampak kesehatan dari penyelenggaraan setiap program pembangunan diwilayah kerjanya. Khusus untuk pembangunan kesehatan, upaya yang dilakukan puskesmas adalah mengutamakan pemeliharaan kesehatan dan pencegahan penyakit tanpa mengabaikan penyembuhan penyakit dan pemulihan kesehatan (Trihono, 2005).

2.2 Pengertian EMR

Catatan medis, baik elektronik atau tidak, adalah kumpulan informasi tentang kesehatan pasien yang penting untuk perawatannya sekarang dan masa depan (WHO 2001).Dengan demikian, catatan medis harus berisi informasi yang cukup untuk mengidentifikasi hubungan pasien, serta sebagai informasi yang relevan untuk pelayanan perawatan pasien selama saat ini sampai masa depan, misalnya:

- riwayat medis pasien
- perintah dan hasil dari setiap pemeriksaan fisik atau tes

• informasi yang berkaitan dengan alergi

• faktor lain yang mungkin perlu pertimbangan khusus.

Keamanan dan jaminan akses informasi secara lengkap dikumpulkan dalam rekam medis sangat penting untuk memastikan bahwa profesi kesehatan telah tersedia informasi yang tepat kapan dan di mana mereka membutuhkannya. memaksimalkan kualitas dan efisiensi perawatan kepada pasien.

2.2.1 Sistem EMR

Sebuah sistem EMR terdiri dari informasi klinis dan kemampuan yang dibutuhkan untuk memberikan layanan kesehatan (Gambar 2.1). Sebuah Sistem EMR digunakan oleh penyedia layanan kesehatan untuk dokumentasi, monitoring dan manajemen. Paling tidak sebuah sistem EMR harus sesuai dengan persyaratan legislatif untuk catatan medis.

Sebuah sistem EMR yang dirancang dan diimplementasikan akan memungkinkan akses yang tepat terhadap informasi pasien dan dasar bukti klinis untuk memberikan manfaat, seperti:

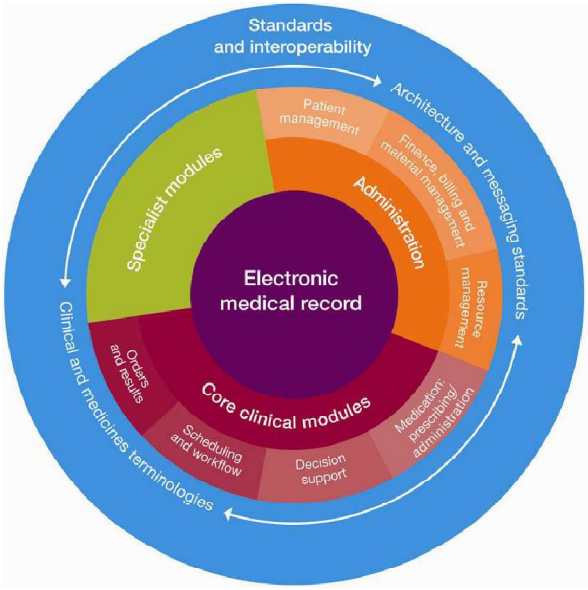
• menyediakan akses ke informasi yang berkualitas tinggi yang sangat penting untuk hasil klinis.

• meningkatkan kemampuan untuk mengembangkan dan menanamkan pedoman terbaik praktek klinis dan jalur klinis untuk mengoptimalkan proses pelayanan kesehatan yang :

- Mengurangi risiko
- Mendukung pengurangan kesalahan medis
- Lebih mendukung pengambilan keputusan klinis
- Meningkatkan hasil kesehatan pasien

• memungkinkan efisiensi biaya administrasi dan untuk meningkatkan kapasitas untuk mengelola permintaan

• berbagi informasi di layanan kesehatan dan perawatan primer untuk mendukung perawatan terkoordinasi konsumen berpusat.



Gambar 2.1 Sistem EMR terdiri EMR dan kemampuan pendukung.

2.3 Data Mining

2.3.1 Pengertian Data Mining

Dengan mudah menyatakan, data mining merujuk pada ekstraksi atau "pertambangan" pengetahuan dari data dalam jumlah besar. Istilah ini sebenarnya keliru. Ingat bahwa penambangan emas dari batu atau pasir yang disebut sebagai tambang emas daripada batu atau penambangan pasir. Dengan demikian, data mining seharusnya lebih tepat bernama "pertambangan pengetahuan dari data," yang sayangnya agak panjang. "pertambangan pengetahuan," istilah yang lebih pendek, mungkin tidak mencerminkan penekanan pada penambangan dari sejumlah besar data. Namun demikian, pertambangan adalah istilah yang jelas karakteristik proses yang menemukan satu set kecil nugget berharga dari banyak bahan baku.Dengan demikian, sehingga kekeliruan kedua kata "data" dan "tambang" menjadi pilihan yang populer. Banyak istilah lainnya membawa makna yang sama atau sedikit berbeda dengan data mining, seperti knowledge mining from data, knowledge extraction,data/pattern analysis, data archaeology, and data dredging (Han & Kamber, 2006).

Banyak orang memperlakukan data mining sebagai sinonim untuk istilah lain populer digunakan, Knowledge Discovery from data, atau KDD. Atau, orang lain melihat data mining hanya sebagai suatu langkah penting dalam proses penemuan pengetahuan (Han & Kamber, 2006).

2.3.2 Proses Data Mining

Data mining dapat digambarkan sebagai suatu proses untuk menemukan suatu informasi baru yang menarik seperti pola, asosiasi, aturan, perubahan, keganjilan, dan struktur penting dari sejumlah besar data yang disimpan pada bank data dan tempat penyimpanan informasi lainnya.

Sebagai suatu rangkaian proses, data mining dapat dibagi menjadi beberapa tahap sebagaimana berikut:

1. Data cleaning

Data cleaning merupakan tahap pembersihan data yang fungsinya untuk membuang data yang tidak konsisten dan noise. Data yang diperoleh, baik dari database suatu perusahaan maupun hasil eksperimen, memiliki isian-isian yang tidak sempurna seperti data yang hilang, data yang tidak valid atau juga hanya sekedar salah ketik. Data-data yang tidak relevan lebih baik dibuang karena keberadaannya bisa mengurangi mutu atau akurasi dari hasil data mining nantinya. Pembersihan data juga akan mempengaruhi performasi dari sistem data mining karena data yang ditangani akan berkurang jumlah dan kompleksitasnya.

2. Data integration

Data yang diperlukan untuk data mining tidak hanya berasal dari satu database tetapi juga berasal dari beberapa database atau file teks. Pada tahap inilah dilakukan penggabungan data dari berbagai database ke dalam satu database yang baru. Integrasi data perlu dilakukan secara cermat karena kesalahan pada integrasi data bisa menghasilkan hasil yang menyimpang dan bahkan menyesatkan pengambilan aksi nantinya. Dalam integrasi data ini juga perlu dilakukan transformasi dan pembersihan data karena seringkali data dari dua database berbeda tidak sama cara penulisannya atau bahkan data yang ada di satu database ternyata tidak ada di database lainnya. Hasil integrasi data sering diwujudkan

dalam sebuah data warehouse karena dengan data warehouse, data dikonsolidasikan dengan struktur khusus yang efisien.

3. Data selection

Pada tahap seleksi, dilakukan penyeleksian data yang sesuai untuk analisis.

4. Data transformation

Transformasi dan pemilihan data ini untuk menentukan kualitas dari hasil data mining, sehingga data diubah dalam bentuk yang tepat atau sesuai untuk diproses dalam data miningdengan cara meringkas atau melakukan operasi.

5. Data mining

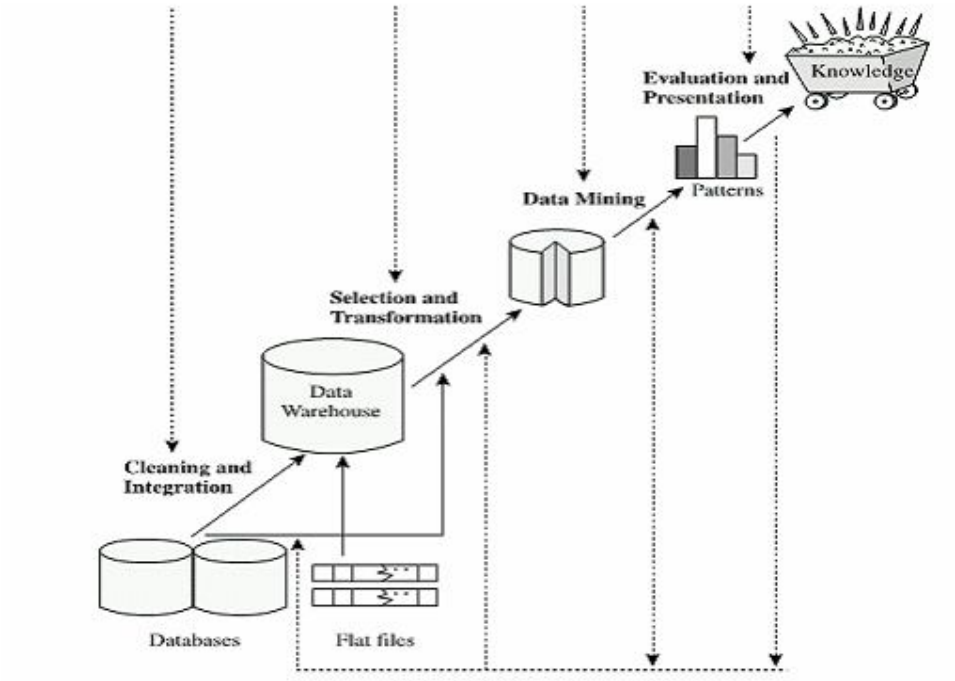
Data mining merupakan proses pokok dimana metode kecerdasan diterapkan dalam ekstraksi informasi atau pola yang penting atau menarik dari data yang ada di database yang besar.

6. Pattern evaluation

Evaluasi pola untuk mengidentifikasi pola-pola menarik untuk di representasikan kedalam knowledge based.

7. Knowledge presentation

Representasi pengetahuan merupakan tahap untuk menunjukkan hasil yang diperoleh kepada pengguna dengan cara visualisasi.



Gambar 2.2 Proses Data Mining

Sumber : Han, Jiawei, dan Kamber, Micheline. (2006). Data Mining: Consepts and Techniques. 2nd Edition. University of Illionis at Urbana-Champaign.

2.3.3 Metode Data Mining

Beberapa metode yang sering disebut-sebut dalam literatur data mining antara lain yaitu association rule mining, clustering, klasifikasi, neural network, genetic algorithm dan lain-lain.

Berikut beberapa jenis teknik data mining yang sering digunakan:

2.3.3.1 Association Rule Mining

Association rule mining adalah metode mining untuk menemukan aturan assosiatif antara suatu kombinasi item. Contoh dari aturan assosiatif dari analisa pembelian di suatu pasar swalayan adalah bisa diketahui berapa besar kemungkinan seorang pelanggan membeli roti bersamaan dengan susu atau selai. Dengan pengetahuan tersebut dapat dilakukan pengaturan penempatan barangnya

atau merancang bentuk pemasaran dengan memakai kupon diskon untuk kombinasi barang tertentu.

Penting tidaknya suatu aturan assosiatif dapat diketahui dengan dua parameter, support yaitu persentase kombinasi item tersebut dalam database dan confidence yaitu kuatnya hubungan antar item dalam aturan assosiatif.

2.3.3.2 Classification

Classification adalah proses untuk menemukan model atau fungsi yang menjelaskan atau membedakan konsep atau kelas data, dengan tujuan untuk dapat memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Teknik ini dapat memberikan klasifikasi pada data baru dengan memanipulasi data yang ada yang telah diklasifikasi dan dengan menggunakan hasilnya untuk memberikan sejumlah aturan. Aturan-aturan tersebut digunakan pada data-data baru untuk diklasifikasi. Teknik ini menggunakan supervised induction, yang memanfaatkan kumpulan pengujian dari record yang terklasifikasi untuk menentukan kelas-kelas tambahan. Sebagai contoh, data transaksi sebuah rumah makan selama periode tertentu dapat digunakan untuk melakukan mining sehingga didapatkan informasi kapan waktu rumah makan tersebut menerima pengunjung paling banyak atau sedikit. Dengan informasi yang telah didapatkan, pemilik rumah makan dapat melakukan promosi-promosi ketika waktu pengunjung sepi, sehingga dapat menarik banyak pengunjung.

2.3.3.3 Clustering

Clustering melakukan pengelompokan data tanpa berdasarkan kelas data tertentu. Bahkan clustering dapat dipakai untuk memberikan label pada kelas data yang belum diketahui Tujuan dari clustering adalah untuk mengelompokkan sejumlah data atau objek kedalam klaster sehingga setiap klaster akan terisi data yang semirip mungkin. Data item dapat dikelompokkan menjadi beberapa grup berdasarkan syarat yang telah ditentukan. Sebagai contoh, pengelompokan konsumen di daerah mana yang mempunyai daya beli tinggi dan daerah mana yang memiliki daya beli rendah.

2.3.3.4 Sequential Pattern Mining

Sequential pattern mining adalah pola yang menggambarkan urutan waktu terjadinya peristiwa (Han & Kamber, 2006). Salah satu contoh dari sequential pattern adalah ketika seorang konsumen membeli sebuah komputer, dapat diperkirakan nantinya akan membeli software-software installer atau printer. Berdasarkan kemungkinan tersebut toko dapat menyediakan barang-barang yang diinginkan.

Dalam sebuah rangkaian database, dimana setiap transaksi merupakan sekumpulan item dengan waktu transaksinya. Sedangkan permasalahannya adalah untuk menemukan sequential pattern dengan minimum support dan periode tertentu yang ditetapkan sendiri oleh pengguna, dimana support dari sebuah pattern merupakan presentase dari data-sequences yang mengandung suatu pola tertentu.

Customer yang pernah melakukan transaksi pemesanan item pada sebuah perusahaan tentunya telah memiliki identitas yang tercatat, transaksi yang dilakukan berulang kali kemungkinan masing-masing data-sequences berhubungan kepada semua pilihan item dari seorang customer, dan masing-masing transaksi kepada item-item yang dipilih oleh customer dalam satu pemesanan.

Dengan metode sequential pattern mining, perusahaan dapat melihat bagaimana pola tiap-tiap customer dalam setiap pemesanan produk. Sequential pattern bekerja dengan cara menganalisa semua sequences yang sering muncul terhadap suatu item yang dipesan oleh customer. Sebagai contoh “25% dari customer yang membeli item CPTO A 31/40 dan 41/50 pada pemesanan pertama, kemudian customer membeli item CPTO A 51/60 pada pemesanan berikutnya”. Dengan begitu akan diperoleh informasi sekiranya item apa saja yang dibeli secara bersamaan dan item apa saja yang dibeli secara berurutan, sehingga menghasilkan relasi antar item.

Penerapan metode *sequential pattern mining* untuk perusahaan mendapatkan pola *customer* dalam setiap melakukan pemesanan *item* dapat dijabarkan sebagai berikut:

1. Membuat *database* yang berisi data transaksi *customer* dengan idcustomer,tanggal dan kode item seperti pada tabel 2.1.

Tabel 2.1 Tabel transaksi diurutkan berdasarkan customer id dan transaction time.

Customer Id	Transaction Time	Items Bought
1	June 25 ’93	30
1	June 30 ‘93	90
2	June 10 ‘93	10,20
2	June 15 ‘93	30
2	June 20 ‘93	40,60,70
3	June 25 ‘93	30,50,70
4	June 25 ‘93	30
4	June 30 ‘93	40,70
4	July 25 ‘93	90
5	June 12 ‘93	90

2. Berdasarkan dari data transaksi diatas, dapat dibuat kemungkinan kemungkinan yang ada dengan metode *sequential pattern mining*. Sebagai contoh, dengan *minimum support* > 25% dari 2 *customer* didapatkan 2 rangkaian yaitu {(30) →(90)} dan {(30)→ (40 ,70)}.

2.4 GSP (*Generalized Sequential Pattern*)

2.4.1 Algoritma

$L1 = \{large\ 1 - sequences\};$

$\boldsymbol{for}\ (k = 2; Lk \neq \emptyset; k++)\boldsymbol{do}$

$\boldsymbol{begin}$

$Ck = New\ candidates\ generated\ from\ Lk - 1$

$\boldsymbol{foreach}\ customer - sequence\ in\ database\ \boldsymbol{do}$

$Increment\ the\ count\ of\ all\ candidate\ in\ C\ that\ are\ contained\ in\ c.$

$Lk = Candidates\ in\ Ck\ with\ minimum\ support$

$\boldsymbol{end}$

Lk adalah himpunan semua *frequent k-sequence*.

Ck adalah set kandidat *k-sequence*.

c adalah *subsequence*.

2.4.2 Teori

Struktur dasar dari algoritma GSP untuk menemukan pola berurutan adalah sebagai berikut. Algoritma membuat beberapa proses atas data. Proses pertama menentukan dukungan (support) dari setiap item, yaitu, jumlah urutan data (data-sequence) yang mencakup item. Pada akhir proses pertama, algoritma mengetahui item mana sering terjadi (frequent), yaitu yang memiliki dukungan (support) minimal. Setiap item tersebut menghasilkan 1-elemen *frequent sequence* yang terdiri dari barang tersebut. Setiap proses berikutnya dimulai dengan satu set *seed* yaitu *frequent sequence* yang ditemukan dalam proses sebelumnya. Satu set *seed* digunakan untuk menghasilkan *frequent sequence* baru yang berpotensi, yang disebut *candidate sequence*. Setiap *candidate sequence* memiliki satu lagi item dari *seed sequence*, sehingga semua *candidate sequence* dalam proses tersebut akan memiliki jumlah item yang sama. *Support* untuk *candidate sequence* ini ditemukan selama proses atas data. Pada akhir proses, algoritma menentukan mana dari *candidate sequence* sebenarnya *frequent*. *Frequent candidate* ini menjadi *seed* untuk proses berikutnya. Algoritma berakhir ketika tidak ada

frequent sequence di akhir proses, atau ketika tidak ada *candidate sequence* yang dihasilkan.

2.4.2.1 Candidate Generation

Bagaimana urutan kandidat yang dihasilkan sebelum proses atas data dimulai.

Mengacu pada *sequence* dengan item k sebagai k-sequence. (Jika item terjadi beberapa kali dalam berbagai elemen *sequence*, setiap kejadian berkontribusi pada nilai k.) Menyatakan Lk sebagai himpunan semua *frequent* k-sequence, dan Ck set kandidat k-sequence. Mengingat L_{k-1}, himpunan *frequent(* k-1) - sequence, kita ingin menghasilkan superset dari himpunan semua *frequent* k-sequence. Pertama-tama mendefinisikan pengertian *subsequence* yang bersebelahan.

Mengingat *sequence* $s=\{s_1,s_2,\ldots s_n\}$ dan *subsequence* c, c adalah subsequence bersebelahan s jika salah satu kondisi berikut berlaku:

1. c berasal dari s dengan menjatuhkan sebuah item dari s_1 atau s_n
2. c berasal dari s dengan menjatuhkan item dari elemen s_i yang memiliki minimal 2 item.
3. c adalah *subsequence* bersebelahan c ', dan c' adalah *subsequence* bersebelahan s.

Sebagai contoh, perhatikan *sequence* s = {(1, 2) →(3, 4) →(5) →(6)}. *Sequence* {(2)→ (3, 4) →(5)}, {(1, 2) →(3) →(5) →(6)} dan {(3) →(5)} adalah beberapa subsequences bersebelahan s. Namun, {(1, 2) →(3, 4) →(6)} dan {(1) →(5) →(6)} tidak.

Candidate dihasilkan dalam dua langkah:

o *Join Phase*.

Menghasilkan *candidate sequence* dengan menggabungkan L_{k-1} dengan L_{k-1}.Sebuah *sequence* s_1 bergabung dengan s_2 jika *subsequence* yang diperoleh dengan menjatuhkan item pertama s_1 sama dengan *subsequence* yang diperoleh dengan menjatuhkan item terakhir dari s_2. *Candidate sequence* yang dihasilkan

oleh penggabungan s_1 dengan s_2 adalah *sequence* s_1 diperpanjang dengan item terakhir di s_2. Item yang ditambahkan menjadi elemen terpisah jika itu adalah elemen yang terpisah di s_2, dan bagian dari elemen terakhir dari s_1 sebaliknya. Ketika penggabungan L_1 dengan L_1, perlu menambahkan item dalam s_2 baik sebagai bagian dari itemset dan sebagai elemen yang terpisah, karena kedua {(x) (y)} dan {(xy)} memberikan *sequence* yang sama {(y)} setelah menghapus item pertama. (Perhatikan bahwa s_1 dan s_2 subsequences berdekatan *candidate sequence* baru.)

o *Prune Phase*.

Menghapus *candidate sequence* yang memiliki bersebelahan (k-1)-*subsequence* yang jumlah *support* kurang dari *minimum support*. Jika disana tidak ada *max-gap constraint*, juga menghapus *candidate sequence* yang memiliki *subsequence* tanpa *minimal support.*

Tabel 2.2 Contoh *Candidate Generation*

Frequent 3-Sequences	*Candidate 4-Sequences*	
	After join	*After pruning*
{(1,2)→(3)}	{(1,2)→(3,4)}	{(1,2)→(3,4)}
{(1,2)→(4)}	((1,2)→(3)→(5)}	
{(1)→(3,4)}		
{(1,3)→(5)}		
{(2)→(3,4)}		
{(2)→(3)→(5)}		

Tabel 2.2 menunjukkan L_3, dan C_4 setelah *Join* dan *Phrune Phase*. Di *Join Phase*, *sequence* {(1, 2)→ (3)} bergabung dengan {(2)→ (3, 4)} untuk menghasilkan {(1, 2) →(3, 4)} dan dengan {(2) (3) →(5)} untuk menghasilkan {(1, 2) (3) →(5)}. *Sequence* yang tersisa tidak bergabung dengan urutan L_3. Misalnya, {(1, 2) →(4)} tidak bergabung dengan urutan apapun karena tidak ada urutan bentuk {(2) →(4 x)} atau {(2) →(4) →(x)}. Pada *prune phase*, {(1, 2)

→(3) →(5)} dijatuhkan sejak itu berdekatan dengan *subsequence* {(1) →(3)→ (5)} yang tidak ada di L_3.

2.4.2.2 Rule Generation

Setelah *frequent sequence* diketahui, mereka dapat digunakan untuk memperoleh *rule* yang menggambarkan hubungan antara *sequence item* yang berbeda.

```
RuleGen(F, min_conf):
  for all frequent sequences β ∈ F do
    for all subsequences α ≺ β do
      conf = fr(β)/fr(α);
      if (conf ≥ min_conf) then
        output the rule α ⇒ β, and conf
```

Gambar 2.3 Algoritma *Rule Generation*

Mengingat minimal *confidence* yang ditentukan pengguna (minconf), dapat menghasilkan semua *rule* yang memenuhi syarat dengan cara algoritma sederhana ditunjukkan pada gambar 2.3.

2.4.3 Contoh Kasus

Tabel 2.3 Tabel Transaksi diurutkan berdasarkan Customer-id dan Transaction-time

Customer-id	Transaction-time	Items
1	10	C D
1	15	A B C
1	20	A B F
1	25	A C D F
2	15	A B F
2	20	E
3	10	A B F
4	10	D G H
4	20	B F
4	25	A G H

Menjadikan Sequence Tabel ditunjukkan di **Tabel 2.4.**

Tabel 2.4 Sequence tabel

S_id	Sequence
1	{(C,D)→(A,B,C)→(A,B,F)→(A,C,D,F)}
2	{(A,B,F)→(E)}
3	{(A,B,F)}
4	{(D,G,H)→(B,F)→(A,G,H)}

Menghitung Support *Candidate 1-Sequence* seperti pada **Tabel 2.5.**

Tabel 2.5 Nilai Support *Candidate 1-sequence*

Item	Support	Item	Support
A	4	E	1
B	4	F	4
C	1	G	1
D	2	H	1

Dengan min-support=2 maka kandidat yang memenuhi seperti pada **Tabel 2.6**.

Tabel 2.6 *Frequent 1-sequence*

Item	Support
A	4
B	4
D	2
F	4

Mengkombinasikan hasil dari *1-sequence* untuk mendapatkan *2-sequence* seperti pada **Tabel 2.7.**

Tabel 2.7 *Candidate generation 2-sequence*

Candidate	Support	Candidate	Support
(A,B)	3	B→D	1
(A,D)	1	B→F	1
(A,F)	3	D→F	2
(B,D)	0	B→A	2
(B,F)	4	D→A	2
(D,F)	1	F→A	2
A→B	1	D→B	2
A→D	1	F→B	0
A→F	1	F→D	1

Dengan min-support=2 maka kandidat yang memenuhi seperti pada **Tabel 2.8.**

Tabel 2.8 *Frequent 2-sequence*

Candidate	Support	Candidate	Support
(A,B)	3	B→A	2
(A,F)	3	D→A	2
(B,F)	4	F→A	2
D→F	2	D→B	2

Mengkombinasikan hasil dari *2-sequence* untuk mendapatkan *3-sequence* seperti pada **Tabel 2.9.**

Tabel 2.9 *Candidate generation 3-sequence*

Candidate	Support	Candidate	Support
(A,B,F)	3	D→(A,B)	1
(A,B)→A	1	D→(A,F)	1
(A,F)→A	1	F→(A,B)	1
(B,F)→A	2	F→(A,F)	1
D→F→A	2	D→(B,F)	2
B→(A,B)	1	D→B→A	2
B→(A,F)	1		

Dengan min-support=2 maka kandidat yang memenuhi seperti pada **Tabel 2.10**.

Tabel 2.10 *Frequent 3-sequence*

Candidate	Support
(A,B,F)	3
(B,F)→A	2
D→F→A	2
D→(B,F)	2
D→B→A	2

Mengkombinasikan hasil dari *3-sequence* untuk mendapatkan *4-sequence* seperti pada **Tabel 2.11.**

Tabel 2.11 *Candidate generation 4-sequence*

Candidate	Support
(A,B,F)→A	1
D→(B,F)→A	2

Dengan min-support=2 maka kandidat yang memenuhi seperti pada **Tabel 2.12.**

Tabel 2.12 *Frequent 4-sequence*

Candidate	Support
D→(B,F)→A	2

Algoritma berhenti jika di hanya ditemukan 1 *frequent-sequence* atau tidak menemukan *candidate-sequence.*

Tabel 2.13 *Frequent k-sequence*

N-Freq Itemset	Frequent Sequence dengan support
L1	{(A)[4],(B)[4],(D)[2],(F)[4]}
L2	{(AB)[3], (AF)[3], (BF)[4], (D→F)[2], (D→A)[2], (B→A)[2], (F→A)[2], (D→B)[2]}
L3	{(ABF)[3],(BF) →A [2],D→F →A[2],D→(BF)[2],D→B→A[2] }
L4	{D→(BF) →A[2]}

Selanjutnya adalah *rule generation*.Berdasarkan **Tabel 2.13** dan dengan algoritma pada gambar 2.3 maka hasil dari rule generation seperti pada **Tabel 2.14**.

Tabel 2.14 *Rule Generation* dengan *confidence*

rule_id	Rule	Confidence
1	A=>(AB)	3 / 4 = 75 %
2	B=>(AB)	3 / 4 = 75 %
3	A=>(AF)	3 / 4 = 75 %
4	F=>(AF)	3 / 4 = 75 %
5	B=>(BF)	4 / 4 = 100 %
6	F=>(BF)	4 / 4 = 100 %
7	D=>D→F	2 / 2 = 100 %
8	D=>D→A	2 / 2 = 100 %
9	B=>B→A	2 / 4 = 50 %
10	F=>F→A	2 / 4 = 50 %
11	D=>D→B	2 / 2 = 100 %
12	A=>(ABF)	3 / 4 = 75 %
13	B=>(ABF)	3 / 4 = 75 %
14	F=>(ABF)	3 / 4 = 75 %
15	AB=>(ABF)	3 / 3 = 100 %
16	AF=>(ABF)	3 / 3 = 100 %
17	BF=>(ABF)	3 / 4 = 75 %
18	B=>(BF) →A	2 / 4 = 50 %
19	F=>(BF) →A	2 / 4 = 50 %
20	BF=>(BF) →A	2 / 4 = 50 %
21	D=>D→F→A	2 / 2 = 100 %
22	D→F=>D→F→A	2 / 2 = 100 %
23	D=>D→(BF)	2 / 2 = 100 %
24	D→B=>D→(BF)	2 / 2 = 100 %
25	D→F=>D→(BF)	2 / 2 = 100 %
26	D=>D→B→A	2 / 2 = 100 %
27	D→B=>D→B→A	2 / 2 = 100 %
28	D=>D→(BF) →A	2 / 2 = 100 %
29	D→B=>D→(BF) →A	2 / 2 = 100 %
30	D→F=>D→(BF) →A	2 / 2 = 100 %
31	D→BF=>D→(BF) →A	2 / 2 = 100 %

Jika minconf=75% maka rule yang terpenuhi seperti pada tabel pada **Tabel 2.15.**

Tabel 2.15 *Rule* yang memenuhi *minconf*

Rule	Confidence
A=>(AB)	3 / 4 = 75 %
B=>(AB)	3 / 4 = 75 %
A=>(AF)	3 / 4 = 75 %
F=>(AF)	3 / 4 = 75 %
B=>(BF)	4 / 4 = 100 %
F=>(BF)	4 / 4 = 100 %
D=>D→F	2 / 2 = 100 %
D=>D→A	2 / 2 = 100 %
D=>D→B	2 / 2 = 100 %
A=>(ABF)	3 / 4 = 75 %
B=>(ABF)	3 / 4 = 75 %
F=>(ABF)	3 / 4 = 75 %
AB=>(ABF)	3 / 3 = 100 %
AF=>(ABF)	3 / 3 = 100 %
BF=>(ABF)	3 / 4 = 75 %
D=>D→F→A	2 / 2 = 100 %
D→F=>D→F→A	2 / 2 = 100 %
D=>D→(BF)	2 / 2 = 100 %
D→B=>D→(BF)	2 / 2 = 100 %
D→F=>D→(BF)	2 / 2 = 100 %
D=>D→B→A	2 / 2 = 100 %
D→B=>D→B→A	2 / 2 = 100 %
D=>D→(BF) →A	2 / 2 = 100 %
D→B=>D→(BF) →A	2 / 2 = 100 %
D→F=>D→(BF) →A	2 / 2 = 100 %
D→BF=>D→(BF) →A	2 / 2 = 100 %

2.5 Pengertian Aplikasi Web

Pada awalnya aplikasi *web* dibangun dengan hanya menggunakan bahasa yang disebut HTML (*HyperText Markup Langauge*). Pada perkembangan berikutnya, sejumlah skrip dan objek dikembangkan untuk memperluas kemampuan HTML seperti PHP dan ASP pada skrip dan Apllet pada objek. Aplikasi *Web* dapat dibagi menjadi dua jenis yaitu aplikasi *web* statis dan dinamis.

Web statis dibentuk dengan menggunakan HTML. Kekurangan aplikasi seperti ini terletak pada keharusan untuk memelihara program secara terus menerus untuk mengikuti setiap perkembangan yang terjadi. Kelemahan ini diatasi oleh model aplikasi *web* dinamis. Pada aplikasi *web* dinamis, perubahan informasi dalam halaman *web* dilakukan tanpa perubahan program tetapi melalui perubahan data. Sebagai implementasi, aplikasi *web* dapat dikoneksikan ke basis data sehingga perubahan informasi dapat dilakukan oleh operator dan tidak menjadi tanggung jawab dari *webmaster*.

Arsitektur aplikasi *web* meliputi klien, *web server*, *middleware* dan basis data. Klien berinteraksi dengan *web server*. Secara internal, *web server* berkomunikasi dengan *middleware* dan *middleware* yang berkomunikasi dengan basis data. Contoh *middleware* adalah PHP dan ASP. Pada mekanisme aplikasi *web* dinamis, terjadi tambahan proses yaitu *server* menerjemahkan kode PHP menjadi kode HTML. Kode PHP yang diterjemahkan oleh mesin PHP yang akan diterima oleh klien (Kadir, 2009).

2.6 Pengenalan Personal Home Page (PHP)

PHP adalah singkatan dari *Personal Home Page* yang merupakan bahasa standar yang digunakan dalam dunia *website*. PHP adalah bahasa pemrograman yang berbentuk *script* yang diletakkan didalam *web server*. Ada beberapa pengertian tentang PHP, akan tetapi PHP dapat diartikan sebagai *Hypertext Preeprocessor*. Ini merupakan bahasa yang hanya dapat berjalan pada *server* yang hasilnya dapat ditampilkan pada klien. *Interpreter* PHP dalam mengeksekusi kode PHP pada sisi *server* disebut *serverside*, berbeda dengan mesin maya Java yang mengeksekusi program pada sisi klien (*client-server*).(Peranginangin, 2009)

2.6.1 Konsep Dasar PHP

Kode PHP diawali dengan tanda lebih kecil (<) dan diakhiri dengan tanda lebih besar (>). Ada beberapa cara untuk menuliskan skrip PHP yaitu:

1. <?

…..skrip PHP

?>

2. <?php
…..skrip PHP
?>
3. <script language=”PHP”>
…..skrip PHP
</script>
4. <%
…..skrip PHP
%>

Pemisah antar instruksi adalah tanda titik koma (;) dan untuk membuat atau menambahkan komentar/standar penulisan adalah: /* komentar */, // komentar, # komentar. Untuk menuliskan skrip PHP ada dua cara yang sering digunakan yaitu *Embedded Script* dan *Non-Embedded Script.*

a. *Embedded Script* adalah script PHP yang disisipkan diantara tag-tag dokumen HTML. Contoh penulisan dari *Embedded Script*:

<html>
<head>
<title>Embedded Script</title>
</head>
<body>
<?pho
echo “Hallo, selamat menggunakan PHP”;
?>
</body>
</html>

b. *Non-Embedded Script* adalah skrip PHP murni, tag HTML yang digunakan untuk membuat dokumen merupakan bagian dari skrip PHP. Contoh penulisan dari *Non-Embedded Script*:

<?php
echo “<html>”;
echo “<head>”;
echo “<title>Mengenal PHP</title>”;

```
echo “</head>”;
echo “<body>”;
echo “<p>PHP itu mudah</p>”;
echo “</body>”;
echo “<html>”;
?>
```

Script yang dibuat dengan PHP disimpan dengan nama *file* dan diikuti dengan ekstensi *.php, misalnya : coba.php. Bila skrip PHP diakses melalui komputer *local* maka *file* PHP disimpan di folder htdocs di *web server*. Sama halnya dengan penaman dokumen HTML, pemberian nama dokumen yang sama tetapi dituliskan dengan *case* yang berbeda akan dianggap sebagai dokumen yang berbeda. Skrip dapat disisipkan di bagian manapun dalam dokumen HTML, begitu pula sebaiknya skrip HTML dapat diletakkan di antara skrip PHP.(Peranginangin, 2009)

2.7 Penelitian Sebelumnya

Telah dilaksanakan beberapa penelitian dengan menggunakan metode GSP (Generalized Sequential Pattern) sebelumnya.

Gregorius Satia Budhi, Andreas Handojo dan Christine Oktavina Wirawan (2009) melakukan penelitian pada perpustakaan UK Petra dengan metode GSP untuk menggali data sekuensial sirkulasi buku. Hasil dari penggalian ini adalah informasi tentang buku-buku yang sering dipinjam secara bersamaan (Association Rules) dan buku-buku yang sering dipinjam secara berurutan oleh peminjam yang sama (Sequential Pattern Rules). Dengan algoritma GSP kedua macam informasi tersebut akan didapat secara bersamaan dalam sekali proses. Dari hasil survei pada para pengambil keputusan di perpustakaan UK Petra tentang kelayakan hasil penelitian ini diaplikasikan, didapat nilai rata-rata sebesar 88.34%.

Dewi Srimurtiningsih (2011) melakukan penelitian terhadap data transaksi penjualan perusahan XYZ kepada kostumernya. Informasi mengenai perilaku customer dalam melakukan transaksi yang sangat dibutuhkan oleh perusahaan sebagai pendukung dalam melakukan pengambilan keputusan produksi. Informasi ini difokuskan pada data sekuensial dari transaksi pemesanan customer untuk menggali pattern asosiasi atau hubungan antar item-item yang dipesan sesuai dengan idcustomer berdasarkan pada periode tertentu. Aplikasi dirancang dan dibuat

berdasarkan data-data transaksi perusahaan yang merupakan data transaksi pemesanan customer pada periode tertentu. Data ini diolah dengan metode sequential pattern mining dan menggunakan algoritma Generalized Sequential Pattern (GSP).

Dengan mempelajari penelitian tersebut, maka penulis ingin meneliti kasus lain dengan menggunakan metode yang sama yaitu Generalized Sequential Pattern.