BAB 2
LANDASAN TEORI

2.1 Data Mining

Data mining adalah proses menemukan pola yang menarik dan pengetahuan dari data dalam jumlah besar [9]. *Data mining* memiliki 2 buah fungsi, yaitu fungsi deskriptif dan fungsi prediktif. Fungsi deskriptif adalah fungsi untuk memahami data yang diamati. Dilakukan sebuah proses untuk mengetahui perilaku dari sebuah data. Dengan fungsi ini, maka bisa menemukan pola tertentu dalam sebuah data [10].

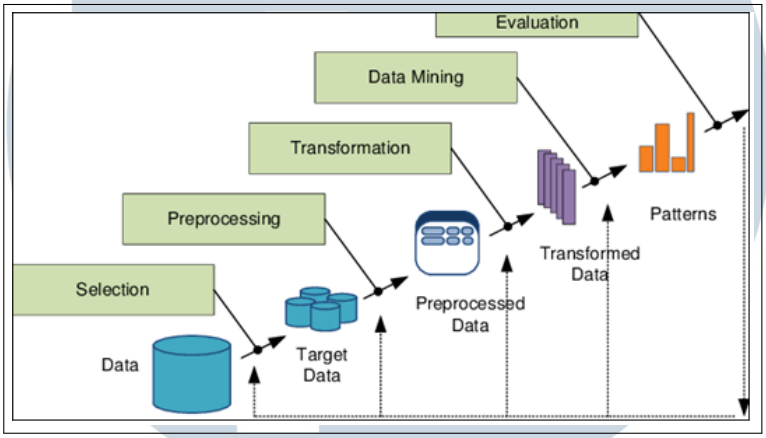
Fungsi prediktif adalah fungsi bagaimana sebuah proses menemukan pola tertentu dari suatu data. Pola tersebut dapat diketahui dari berbagai variabel pada data. Jika pola telah ditemukan, maka pola dapat digunakan untuk memprediksi variabel lain yang belum diketahui nilai atau jenisnya [10].

Dalam *data mining*, *association rules* berguna untuk *analyzing and predicting customer behavior*. *Association rules* memainkan peran penting dalam *customer analytics, market basket analysis, product clustering, catalog design and store layout* [11]. Terdapat berbagai macam metode yang biasa dipakai dalam *data mining*. Metode tersebut ada: *classification, association, clustering, regression, forecasting, sequencing, dan descriptive*. Berikut penjelasan dari beberapa metode *data mining* [12].

1. *Association rule*, mengidentifikasi produk yang sering dibeli bersamaan oleh pelanggan.
2. *Classification*, memperkirakan kelas dari suatu objek yang labelnya belum diketahui.
3. *Regression*, mencari pola numerik. Hasilnya berupa fungsi sebagai penentu hasil yang didasarkan dari nilai input.
4. *Cluster analysis*, mengelompokan suatu kelas ke dalam beberapa segmen berdasarkan atribut yang ditentukan.
5. *Forecasting*, memprediksi nilai yang dicapai pada satu periode.
6. *Descriptive*, memahami lebih dalam mengenai data yang masuk dalam pengamatan.

2.2 Knowledge Discovery in Database (KDD)

Knowledge Discovery in Database merupakan proses menemukan pengetahuan yang berguna dari data [13]. Sistem ini terdiri dari kumpulan komponen yang bersama-sama dapat secara efisien mengidentifikasi dan mengekstrak pola baru yang menarik dan berguna dari data yang disimpan dalam *real-world databases* [14].



Gambar 2.1. Tahapan KDD
sumber: [13]

Berikut penjelasan tahapan KDD dari Gambar 2.1 [9].

1. *Data Selection*, proses pengambilan data yang relevan dengan tugas analisis *data set*. Penyeleksian data yang dipakai terhadap perhitungan yang dilakukan dalam suatu penelitian.
2. *Preprocessing*, langkah menghasilkan *data set* yang bersih sehingga dapat digunakan dalam tahap berikutnya. Data yang telah diseleksi dan memiliki nilai *missing value* dapat dilakukan pengisian dengan memberi nilai rata-rata dari data tersebut.
3. *Transformation*, data diubah dan digabungkan ke dalam bentuk yang sesuai untuk *mining* dengan melakukan operasi ringkasan atau agregasi.
4. *Data mining*, proses dan metode untuk mengekstrak pola data dengan teknik/metode tertentu diterapkan.

5. *Interpretation/Evaluation*, proses mengidentifikasi pola yang mewakili pengetahuan berdasarkan ukuran ketertarikan. Mengidentifikasi pola sesuai atau bertentangan dengan fakta/hipotesa sebelumnya.

2.3 Association Rule Mining

Association rules digunakan untuk menemukan korelasi dan kejadian bersama antara kumpulan data. Idealnya digunakan untuk menjelaskan pola dalam data dari *repository* informasi yang tampak independen, seperti *relational databases* dan *transactional databases* [11]. *Association Rule Mining* adalah teknik *data mining* yang menemukan pola dalam data [15]. *Association rule* mempunyai 2 bagian, yaitu: *antecedent (if) & consequent (then)* [16]. Dari kedua bagian tersebut, terdapat 3 relasi penting yang harus diperhatikan:

1. *Support*: menunjukan seberapa sering hubungan *if or then* muncul dalam *data set* [9, 17, 18].

$$sup(A) = \frac{frq(A)}{N} \tag{2.1}$$

Keterangan:

frq(A) = Jumlah transaksi yang mengandung A.

N = Jumlah keseluruhan transaksi.

$$sup(A \Rightarrow B) = \frac{frq(B,A)}{N} \tag{2.2}$$

Keterangan:

frq(B,A) = Jumlah transaksi yang mengandung *item* A dan B.

N = Jumlah keseluruhan transaksi.

2. *Confidence*: menunjukan berapa kali hubungan ini terbukti benar [9, 17].

$$conf(A \Rightarrow B) = \frac{frq(B,A)}{frq(A)} \tag{2.3}$$

Keterangan:

frq(B,A) = Jumlah transaksi dari A yang mengandung B.

frq(A) = Jumlah transaksi yang mengandung A.

3. *Lift*: alat ukur untuk melihat seberapa baik *rule* yang dihasilkan [9].

$$lift = \frac{sup(A \Rightarrow B)}{sup(B) \cdot sup(A)} \tag{2.4}$$

Keterangan:

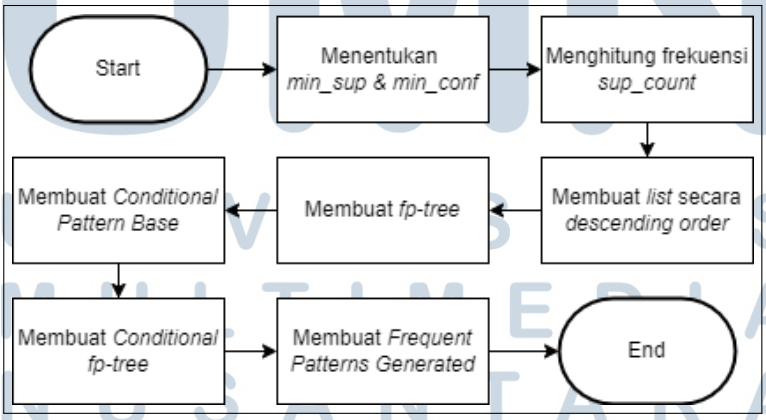
sup(A⇒B) = Nilai *support* dari persamaan 2.2.

sup(B) · *sup(A)* = Nilai *support item* B dikali *support item* A.

Nilai *lift* sama dengan 1, maka produk A dan B tidak memiliki keterkaitan. Apabila nilai *lift ratio* lebih besar dari 1, maka produk A dan B berkorelasi positif. Sedangkan jika nilai *lift ratio* lebih kecil dari 1, maka produk A dan B berkorelasi negatif. *Rules* yang memenuhi batas *minimum support* dan *minimum confidence* dapat dinyatakan sebagai sebuah *strong association rules*. *Rules* di bawah *threshold* dapat dianggap tidak menarik dan nilainya kurang [9].

2.4 Metode FP-Growth

Algoritma *fp-growth (Frequent Pattern Growth)* digunakan untuk menemukan *frequent itemsets* dalam suatu *data set* dan lebih cepat daripada algoritma *apriori* [19]. Algoritma *fp-growth* adalah algoritma klasik dalam *association rules mining* [20]. *Frequent patterns* merupakan pola yang sering muncul dalam data. Algoritma ini mengubah data menjadi pohon daripada *sets*. Struktur data pohon ini memungkinkan pemindaian yang lebih cepat [9].



Gambar 2.2. Tahapan fp-growth

Berikut adalah penjelasan dari Gambar 2.2 [9]:

1. Menentukan *minimum support* dan *minimum confidence*.

2. Melakukan perhitungan frekuensi kemunculan setiap *itemset* yang ada pada setiap data transaksi lalu membuat *list* dari *itemset* tersebut beserta dengan setiap kemunculannya (*sup_count*).

3. Mengurutkan *list* yang telah dibuat dengan menggunakan *sup_count* secara *descending*. *List itemset* dalam setiap transaksi juga diurutkan secara *descending*.

4. Membuat *fp-tree* dengan melakukan:

    - Membuat *null* sebagai akar dari *tree* atau sebagai *node root*.
    - Melakukan *scan* pada data transaksi.
    - Membuat *branch* dengan memasukkan *itemset* dari setiap transaksi dengan mengikuti urutan *sup_count* yang telah dilakukan pengurutan sebelumnya.

5. Membuat tabel *conditional pattern base, conditional fp-tree* dan *frequent patterns generated*.