BAB II

LANDASAN TEORI

2.1 Sistem Rekomendasi

Sistem rekomendasi adalah sistem yang mengidentifikasi produk sesuai dengan kebutuhan dan keinginan user. Sistem tersebut akan membimbing user menuju produk yang paling relevan dari banyaknya pilihan produk (Prasetya, 2017). Menurut Sebastia, L(2009) sistem rekomendasi adalah sebuah alat personalisasi yang menyediakan pengguna sebuah informasi dari barang-barang yang sesuai dengan keinginan pengguna. Sistem rekomendasi menganalisis ketersediaan data, informasi, dan lingkungan dari pengguna, dan menyimpulkannya ke dalam sebuah preferensi. Maka dari itu, sistem rekomendasi menawarkan kemungkinan dari pengolahan informasi personal sehingga preferensi yang disediakan sesuai dengan kebutuhan dari pengguna itu sendiri yang ditampilkan pada sistem menggunakan suatu teknik atau model rekomendasi (Sebastia, L et al., 2009).

Menurut Tang dkk(2013), sistem rekomendasi pertama kali ditemukan pada tahun 1990, banyak penelitian terkait dengan sistem rekomendasi. Sistem rekomendasi diaplikasikan ke banyak bidang. Adapun tiga metode yang dipakai adalah *Content Based, Collaborative Filtering,* dan *Hybrid. Content Based* menggunakan persamaan dari suatu produk untuk direkomendasikan kepada pengguna. Namun *Content Based* memiliki kekurangan, yaitu jika konten yang tersedia terbatas, maka akurasi dari rekomendasi akan sangat rendah(Yuan dkk., 2014). *Collaborative Filtering* merupakan metode yang paling sering digunakan, karena bergantung pada riwayat pemilihan atau penilaian (Su dan Khosgoftaar,

2009). Adapun *Hybrid* adalah metode yang menggabungkan kedua metode di atas guna menghasilkan luaran yang lebih baik (Tang dkk., 2013).

2.2 Algoritma Apriori

Algoritma Apriori pertama kali diusulkan untuk menilai *frequent itemset* dalam aturan asosiasi *boolean* pada 1994 oleh Agrawal dan Srikant (Han, Kamber, dan Pei, 2011). Nama Apriori diambil berdasarkan fakta bahwa algoritma ini menggunakan data dari *frequent itemset* yang sudah ada sebelumnya (*priori knowledge*). Algoritma Apriori adalah salah satu algoritma klasik data mining dan digunakan agar komputer dapat mempelajari aturan asosiasi untuk melakukan pencarian *frequent itemset* atau kombinasi *itemset* (Erwin, 2009).

Apriori memiliki dua tahap dalam metodologi dasar analisis asosiasi. Tahap tersebut adalah:

1. Analisa pola frekuensi tinggi

   Tahap ini mencari kombinasi item yang memenuhi syarat minimum dari nilai *support* dalam database. Nilai *support* sebuah item diperoleh dengan memakai rumus berikut (Larose, 2005).

$$Support(A) = \frac{Jumlah\ Transaksi\ Mengandung\ A}{Total\ Transaksi} \quad \ldots(2.1)$$

   Sedangkan nilai dari *support* dua *item* diperoleh dari rumus berikut:

$$Support(A, B) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Total\ Transaksi} \quad \ldots(2.2)$$

2. Pembentukan Aturan Asosiatif

   Setelah pola frekuensi tinggi ditemukan, tahap selanjutnya adalah mencari aturan asosiatif yang memenuhi syarat minimum untuk *confidence* dengan

menghitung *confidence* dari aturan " *jika* A *maka* B" dengan rumus berikut (Larose, 2005).

$$Confidence = P(B|A) = \frac{Jumlah\ Transaksi\ Mengandung\ A\ dan\ B}{Jumlah\ Transaksi\ Mengandung\ A} \quad \ldots(2.3)$$

Nilai *Support* dan nilai *Confidence* berperan dalam menunjukkan penting tidaknya suatu aturan asosiatif. Nilai *Support* merupakan nilai yang menunjukkan persentase kombinasi item dalam database, sedangkan nilai *Confidence* merupakan nilai yang menunjukkan kekuatan hubungan antar *item* dalam aturan asosiatif (Saragih, 2013).

Proses utama dalam Algoritma Apriori adalah mendapatkan *frequent itemset,* dengan dua proses, sebagai berikut (Erwin, 2009).

1. *Join* (Penggabungan)

   Join dilakukan dengan menkombinasikan *item* dengan *item* sampai tidak ada kombinasi lain lagi.

2. *Prune* (Pemangkasan)

   Prune dilakukan dengan memangkas item yang telah dikombinasikan dengan menggunakan *minimum support* yang ditentukan oleh pengguna.

2.3 Lift Ratio

Lift Ratio dipakai sebagai bentuk evaluasi untuk mengukur kekuatan dari sebuah aturan asosiasi. Nilai *support* dan *confidence* yang menunjukkan keabsahan dari hasil informasi juga diukur (Santosa, 2007). Untuk mendapatkan nilai *benchmark confidence* dapat dihitung menggunakan rumus sebagai berikut (Fauzy, dkk., 2016):

$$Benchmark\ Confidence = \frac{Jumlah\ transaksi\ consequent\ item}{Jumlah\ transaksi\ basis\ data} \quad \ldots(2.4)$$

Sedangkan untuk menghitung nilai *lift ratio* digunakan rumus sebagai berikut (Fauzy, dkk, 2016):

$$Lift\ Ratio = \frac{Confidence\ (A,B)}{Benchmark\ Confidence\ (A,B)} \quad \ldots(2.5)$$

Lift Ratio merupakan nilai yang menunjukkan ke-*valid*-an proses transaksi dan memberikan informasi bahwa *item* A benar dibeli bersamaan dengan *item* B dan sebuah transaksi dikatakan *valid* jika mempunyai nilai *Lift Ratio* lebih dari satu (Widiati, 2014). Dengan kata lain, *Lift Ratio* dapat mengukur tingkat ke-*valid*-an dari transaksi-transaksi yang dihitung menggunakan algoritma Apriori.

2.4 White-box Testing

White box testing adalah salah satu teknik yang paling penting untuk menguji software, yang mana sangat efektif dalam memvalidasi desain, keputusan, asumsi dan menemukan kesalahan dalam program dan kesalahan dalam implementasi pada suatu sistem (Khan, 2011).

Kelebihan dari teknik pengujian White box adalah dapat mengatasi beberapa kesalahan sebagai berikut (Nidhra and Dondetti, 2012).

1. Kesalahan Logika

   Kesalahan Logika adalah kesalahan yang terjadi pada waktu menggunakan logika pengulangan dan *if*. Pengujian ini akan mendeteksi kondisi yang dianggap tidak sesuai dan mencari kapan pengulangan akan berakhir.

2. Ketidaksesuaian Asumsi

   Menampilkan dan melacak asumsi yang dianggap tidak sesuai dengan ekspektasi, lalu dianalisa dan dilakukan perbaikan.

3. Kesalahan Pengetikan

   Mendeteksi serta menemukan kode yang memiliki ketentuan tertentu(missal: *case sensitive)* pada program.

Salah satu jenis dari white box testing adalah Unit Testing. Unit Testing adalah pengujian yang dilakukan dengan menguji tiap baris pada kumpulan kode yang dibuat (Sommerville, 2011). Pengujian Unit Testing terdiri dari beberapa teknik, salah satunya adalah Path Coverage. Path Coverage memastikan alur program yang terdapat pada sistem dapat dilalui setidaknya satu kali(Syaikhuddin, dkk, 2018).