# BAB II TINJAUAN PUSTAKA

## Perpustakaan Digital

Kemajuan teknologi komunikasi informasi, internet, dan digitalisasi buku, surat kabar, majalah, jurnal, video ceramah, dan sebagian besar buku referensi mengubah konsep perpustakaan konvensional dengan model perpustakaan baru yang disebut *resource center*. Sumber daya digital ini disimpan menggunakan teknologi penyimpanan elektronik atau berlangganan dari penyedia layanan tersebut untuk mengurangi penyimpanan buku, jurnal, majalah berbasis kertas dan kemudahan dalam pencarian *database*, akses fasilitas, dan menekan biaya pemeliharaan perpustakaan (Aithal, 2016).

Perpustakaan digital adalah bidang penelitian yang terus berkembang. Perpustakaan digital didefinisikan sebagai kumpulan informasi elektronik yang berisi *repository* objek digital yang besar dan beragam, dan dapat diakses oleh sejumlah besar *user* yang tersebar secara geografis. Objek digital yang dimaksud seperti teks, gambar, peta, suara, video, katalog, dan jurnal ilmiah (Khiste, et al., 2018). Setiap *user* dapat mengakses sumber daya hingga informasi yang tersedia di perpustakaan dengan menggunakan komputer dan koneksi ke jaringan perpustakaan tanpa harus hadir secara fisik di perpustakaan (Nahak & Padhi, 2019).

Sejak 1990, sistem rekomendasi telah menjadi bidang yang dipelajari secara ekstensif, karena telah menjadi teknologi kunci dalam aplikasi *e-commerce*. Sistem rekomendasi dalam perpustakaan digital merupakan permintaan yang spesifik, karena luasnya koleksi digital dan kebutuhan informasi yang beragam membuat *rating* dari *user* diperlukan. Berbagai sistem rekomendasi telah dikembangkan seperti *content-based*, *collaborative filtering*, dan *hybrid systems*. Sistem ini bekerja berdasarkan histori *user*, seperti catatan pinjaman, *session* data atau *rating* dari *user*. Namun, pemberian *rating* pada perpustakaan digital jarang dilakukan oleh *user* sehingga mempengaruhi kualitas rekomendasi (Wenige & Ruhland, 2018).

### Jenis Koleksi Digital

Perubahan perpustakaan konvensional menjadi perpustakaan digital telah mengubah operasi, fungsi dan layanan perpustakaan (Kapoor & Tiwari, 2020). Perpustakaan digital kini berperan sebagai media informasi yang menyediakan jenis koleksi digital seperti:

1. E-book

*E-book* adalah publikasi buku dalam bentuk digital yang terdiri dari teks, gambar, atau keduanya dan dapat dibaca di komputer, *e-readers* atau perangkat elektronik lainnya. *E-book* disukai oleh *user* karena fitur-fitur yang ada di dalamnya, seperti ukuran font yang dapat diubah, membuat kutipan, tautan ke situs lain yang relevan, pencarian, *sharing*, dll (Kapoor & Tiwari, 2020).

1. E-Journal

Seperti makalah penelitian, komunikasi ilmiah, terbitan berkala ilmiah dalam bentuk elektronik merupakan *tools* yang berguna bagi para peneliti. *E-journal* berdampak tidak hanya pada perpustakaan tetapi juga pada penulis dan penerbit. Oleh karena itu, mayoritas *user* sekarang mengharapkan informasi terkini dan tepat waktu yang dapat diperoleh dengan mudah, cepat, tepat dan dari jarak jauh. Perpustakaan akademik dan perpustakaan khusus biasanya tidak dapat menolak *e-journal* dalam koleksinya. Terdapat jenis jurnal online lainnya seperti artikel akses terbuka yang tersedia di web untuk dilihat dan di *download* tanpa biaya (Kapoor & Tiwari, 2020).

1. E-Thesis

*E-Thesis* merupakan layanan yang sangat berguna bagi *user* atau sebagian besar peneliti dalam mengumpulkan data yang besar untuk mata pelajaran tertentu. Pengadaan dan pengaksesan *e-thesis* dapat mengurangi duplikasi pekerjaan penelitian dan memberikan wilayah penelitian kepada para *user* (Kapoor & Tiwari, 2020).

### Format File Digital

Terdapat banyak format *file* digital, di antaranya sebagai berikut:

1. PDF (*Portable Document Format*)

PDF adalah salah satu format *e-book* paling populer, karena format PDF dikembangkan untuk menyediakan sarana yang tidak bergantung platform untuk menampilkan dan bertukar dokumen dengan tata letak tetap. *File* PDF didukung oleh hampir semua perangkat *e-reader*, seperti tablet, PC, laptop, dan *smartphone*. Hal tersebut membuat PDF dapat diterima secara luas sebagai format pengarsipan digital. Namun, beberapa penelitian juga menunjukkan bahwa PDF bukanlah format yang ideal untuk membaca di layar perangkat. Karena format PDF yang dirancang untuk mereproduksi halaman dengan tata letak tetap, sehingga aliran ulang teks agar sesuai dengan perangkat seluler dan layar pembaca *e-book* bisa menjadi masalah. Mallett mencatat bahwa perangkat *Sony Reader* dan *iPod Touch* melakukan pekerjaan yang buruk dalam mendukung PDF, dokumen ditampilkan dengan ukuran yang sangat kecil dan kata-katanya terkadang campur aduk (Zeng, et al., 2016).

1. EPUB (*Electronic Publication*)

EPUB merupakan standar *e-book* gratis dan terbuka yang dibuat oleh *International Digital Publishing Forum* (IDPF). EPUB dikembangkan dengan menggunakan format teks mengalir secara *real-time* sehingga orang dapat membacanya menggunakan *e-reader* mana pun. Hal tersebut membuat EPUB menjadi format *file* yang disukai dalam hal keterbacaan informasi secara tekstual dan lebih cocok untuk persyaratan ilmiah daripada format PDF. Meskipun EPUB adalah standar industri, EPUB belum sepenuhnya didukung oleh semua perangkat *e-reader* (Zeng, et al., 2016)*.*

1. HTML (*Hyper Text Markup Language*)

HTML merupakan tulang punggung dari *world wide web*. Ada banyak teks yang didistribusikan dalam format ini karena dapat digunakan pada berbagai platform komputer dengan sistem operasi yang berbeda-beda. HTML memiliki *style* utama CSS (*cascading style sheet*) dan dapat disisipi bahasa pemrograman lainnya seperti *PHP (Hypertext Preprocessor)*, *Perl*, *JavaScript*, *VBScript*, dll. Namun, HTML tidak bisa berdiri sendiri sebagai bahasa pemrograman web. HTML juga memiliki penggunaan *tag* yang banyak dan versi HTML yang baru belum tentu bisa dijalankan pada semua *browser* (Fitri & Syah, 2020).

1. AZW (*Amazon World*)

Format AZW merupakan format *file* milik amazon yang menyerupai format MOBI dengan menyertakan DRM (*digital rights management*) yang di khususkan untuk *kindle* *amazon* yang tidak di dukung secara luas di *e-reader* (Fitri & Syah, 2020)*.*

### Digitalisasi File Digital

Digitalisasi didefinisikan sebagai proses pengambilan bahan pustaka tradisional berupa buku dan kertas dan mengubahnya menjadi bentuk elektronik sehingga dapat disimpan dan dimanipulasi oleh komputer (Singh, 2017). Terdapat enam tahapan digitalisasi *file* digital yaitu:

1. *Registering*

Sebelum memindai dokumen perlu untuk mendaftarkannya terlebih dahulu dan menggunakan sistem pengarsipan untuk melacaknya. Ini berfungsi untuk menghindari risiko kehilangan *hardcopy*, kehilangan *file*, melewatkan langkah-langkah dalam proses atau menggandakan pekerjaan yang mungkin tanpa disadari. Ada juga risiko kehilangan *file* versi elektronik karena salah nama atau disimpan di *subdirectory* yang salah.

1. *Scanning* Dokumen

Sebelum melakukan pemindaian dokumen, bersihkan terlebih dahulu, pastikan semua halaman ada dan terurut, jika terdapat dokumen dalam kondisi buruk carilah salinan baru. Lakukan pemindaian sesuai dengan *tools* dan teknik pemindaian yang ada.

1. *Optical Character Recognition* (OCR)

OCR merupakan perangkat lunak yang berfungsi mengubah gambar yang dipindai menjadi *file* teks yang dapat dibaca oleh *software* pengolah kata. OCR akan memecah blok teks menjadi beberapa baris atau menjadi karakter individu dan mencocokkan gambar dari setiap huruf dengan pola yang dikenali sebagai "a", "b", dll. Penggunaan OCR yang dikhususkan untuk menghindari masalah penggunaan bahasa skrip latin dengan karakter beraksen.

1. *Proofreading*

*Proofreading* merupakan tahapan koreksi teks dan tata letak dokumen. Ini dapat dilakukan dengan dua cara yaitu:

1. Membandingkan teks yang dipindai pada layar dengan *hardcopy* dan memasukkan koreksi langsung ke komputer. Pemeriksa ejaan pengolah kata akan membantu dalam kesalahan ejaan dengan cepat.
2. Mencetak teks pindaian dan membandingkannya dengan salinan aslinya. Tandai koreksi apapun pada hasil cetak, lalu masukkan ke dalam komputer. Ini adalah metode yang lebih lambat, tetapi mungkin merupakan opsi terbaik jika tidak ada cukup komputer untuk setiap korektor.
3. *Reformatting*

OCR dapat menghasilkan dokumen yang terdiri dari teks lurus, tanpa kolom, *header* dan *footer*. Untuk itu diperlukan pembaruan untuk mengoreksi dan memasukkannya ulang dalam halaman, atau juga adanya pembaharuan jenis huruf, *style* judul, dan sebagainya, untuk membuat dokumen lebih menarik dan mudah dibaca. Sebagai alternatif, sesuaikan pengaturan program OCR untuk mempertahankan tata letak halaman.

1. Versi Final

Kebanyakan dokumen memerlukan beberapa informasi tambahan agar pembaca dapat mengidentifikasinya dengan mudah. Contohnya, untuk sebuah buku diperlukan penambahan judul buku, penulis atau editor, penerbit, dan tanggal penerbitan pada dokumen terkait. Untuk artikel jurnal perlu mencantumkan judul jurnal, tanggal, volume dan nomor terbitan, judul artikel dan pengarang serta nomor halaman pada jurnal asli yang dicetak. Dengan kata lain ada kebutuhan untuk menambahkan metadata untuk mendeskripsikan setiap dokumen.

### Klasifikasi Item Perpustakaan

Klasifikasi merupakan salah satu *tools* tertua yang dikembangkan oleh ahli pustakawan dan dianggap sebagai dasar dari kepustakawanan karena fungsinya yang serbaguna. Tujuan tradisional klasifikasi untuk menempatkan objek fisik di rak telah kehilangan fungsinya dalam kasus sumber daya elektronik jarak jauh. Namun, klasifikasi memiliki fungsi lain dalam kategorisasi yang lebih berguna dalam kasus sumber daya elektronik, karena berperan dalam pengorganisasian subjek (Kumbhar, 2012). Berikut beberapa sistem klasifikasi yang digunakan sebagai pedoman dalam perpustakaan, yaitu:

1. *Dewey Decimal Classification*

*Dewey decimal classification* atau disingkat DDC merupakan sistem yang mengorganisasi pengetahuan secara umum, yang terus menerus direvisi untuk mengikuti perkembangan pengetahuan. DDC diterbitkan pertama kali pada tahun 1873 oleh Melvin Dewey. Edisi termutakhir dari DDC adalah DDC 23 yang diterbitkan pada tahun 2011 oleh Joan S. Mitchell dengan hak cipta pada *Online Computer Library Centre, Inc* (OCLC) (Mallawa, 2012). Kelas umum *dewey decimal classification* 23 dapat dilihat pada tabel 2.1.

Tabel 2. Kelas Umum *Dewey Decimal Classification* 23

Sumber (Suresha & Narayanaswamy, 2016)

|  |  |  |
| --- | --- | --- |
| No | Kode | Subjek |
| 1 | 000 | *Computers, Information and General Reference* |
| 2 | 100 | *Philosophy and Psychology* |
| 3 | 200 | *Religion* |
| 4 | 300 | *Social Sciences* |
| 5 | 400 | *Language* |
| 6 | 500 | *Science and Mathematics* |
| 7 | 600 | *Technology* |
| 8 | 700 | *Arts and Recreation* |
| 9 | 800 | *Literature* |
| 10 | 900 | *History and Geography* |

1. *Universal Decimal Classification*

*Universal decimal classification* atau disingkat UDC merupakan klasifikasi internasional pertama yang dirancang untuk informasi bibliografi dan pekerjaan dokumentasi yang disebut perluasan dari *dewey decimal classification* (Satija, 2008). UDC mencakup seluruh domain pengetahuan, dengan 10 kelas utama dan kelas utama ke-4 yang masih kosong, dapat dilihat pada tabel 2.2.

Tabel 2. Kelas Utama *Universal Decimal Classification*

Sumber (Satija, 2008)

|  |  |  |
| --- | --- | --- |
| No | Kode | Subjek |
| 1 | 0 | *Generalities* |
| 2 | 1 | *Philosophy; Psychology* |
| 3 | 2 | *Religion, Theology* |
| 4 | 3 | *Social Sciences* |
| 5 | 4 | *[Vacant]* |
| 6 | 5 | *Natural Sciences; Mathematics* |
| 7 | 6 | *Technology* |
| 8 | 7 | *The Arts* |
| 9 | 8 | *Language; Linguistics; Literature* |
| 10 | 9 | *Geography; Biography; History* |

Dari ke-10 kelas utama tersebut dapat diperluas lagi dengan notasi dari tabel bantu yang terdiri dari dua jenis, yaitu alat bantu umum yang dapat diterapkan secara universal dan alat bantu khusus yang dapat diterapkan secara terbatas atau secara lokal (Satija, 2008).

## Sistem Rekomendasi

Sistem rekomendasi merupakan fitur yang berfungsi untuk membantu *user* untuk menemukan saran tentang *item* apa yang sebaiknya digunakan atau dipilih. Secara umum sistem rekomendasi ditujukan untuk individu yang kekurangan pengalaman atau kompetensi yang cukup untuk mengevaluasi banyaknya jumlah alternatif *item* yang ada pada suatu kasus tertentu (Elahi, et al., 2016). Saat ini sistem rekomendasi telah diterapkan di berbagai domain seperti musik, film, buku, dan produk. Fungsi tersebut tidak hanya dapat merekomendasikan produk yang disukai *user* tetapi juga dapat secara aktif mempromosikan pemasaran produk. Dasar dari sistem rekomendasi dibangun dengan asumsi bahwa riwayat *user* internet dapat sepenuhnya mewakili preferensi *user*, dan asumsi ini berarti bahwa preferensi *user* dapat diperoleh dengan *data mining* dan analisis riwayat *user* (Zhang & Yang, 2019)*.*

Penggunaan teknik rekomendasi yang akurat dan efisien sangat penting bagi suatu sistem untuk memberikan hasil yang baik dan bermanfaat kepada setiap penggunaannya. Salah satu masalah yang menyebabkan sistem rekomendasi tidak akurat adalah karena terjadinya *sparsity*. *Sparsity* merupakan masalah yang terjadi akibat kurangnya informasi yang cukup, yaitu hanya sedikit dari jumlah *item* yang tersedia dalam *database* yang dinilai oleh *user.* Akibatnya, *user* tidak mendapatkan hasil rekomendasi yang baik dan bermanfaat (Isinkaye, et al., 2015).

Dalam melakukan pengumpulan data *user* untuk sistem rekomendasi, terdapat dua cara (Andini & Zulkarnain, 2013) yaitu dengan cara eksplisit dan implisit. Metode eksplisit merupakan pengumpulan data yang diperoleh berdasarkan *feedback* dari *user* secara langsung, data dapat diperoleh dengan meminta pendapat langsung dari *user*, bisa berupa *rating*, *likes/dislikes*, maupun kata pencarian (*keyword*). Metode implisit merupakan pengumpulan data berdasarkan pengamatan pola kecenderungan *user*, seperti kecenderungan *user* terhadap suatu *item* dengan kriteria tertentu yang dimiliki *item* tersebut. Metode pengumpulan data pada teknik implisit tidak melibatkan kontribusi *user* melainkan dengan cara pendekatan dari perilaku *user* dan menyimpulkan data-data yang sudah dikumpulkan oleh sistem. Berdasarkan pendekatan yang umum dalam sistem rekomendasi dibagi menjadi tiga klasifikasi yaitu*: content-based recommendation, collaborative filtering recommendation, dan hybrid recommendation* (Lü, et al., 2012).

### Content-based Recommendation

*Content-based recommendation* menggunakan ketersediaan konten (sering juga disebut dengan fitur, atribut atau karakteristik) sebuah *item* sebagai basis dalam pemberian rekomendasi (Ricci, et al., 2015). Sebagai contoh, sebuah film mempunyai konten seperti genre, *author*, tahun rilis, dan lain-lain, atau sebuah *file* dokumen memiliki konten berupa tulisan yang ada di dalamnya.

*Content-based recommendation* mencoba untuk melakukan mencocokkan (*matching*) antara profil *user* (*user* *profile*) dengan konten *item* (*item* *content*). Misalnya, seorang *user* telah memberi *rating* kepada lima buah *item* 𝑖1, 𝑖2, 𝑖3, 𝑖4, 𝑖5 yang memiliki empat buah fitur 𝑗1, 𝑗2, 𝑗3, 𝑗4. Angka 1 menandakan bahwa *item* tersebut memiliki atribut yang bersangkutan, sedangkan angka 0 menunjukkan bahwa *item* tersebut tidak memiliki atribut tersebut. Contoh matriks *rating* *content-based recommendation* dapat dilihat pada tabel 2.3.

Tabel 2. Contoh Matriks *Rating* *Content-based Recommendation*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # |  |  |  |  | *Rating* |
|  | 1 | 1 | 1 | 1 | 4 |
|  | 1 | 0 | 1 | 1 | 3 |
|  | 1 | 1 | 0 | 0 | 2 |
|  | 1 | 0 | 1 | 1 | 3 |
|  | 1 | 1 | 0 | 1 | ? |

Untuk menghitung bobot dari masing-masing atribut dapat dengan menggunakan rumus berikut (Uluyagmur, et al., 2012).

(1)

Di mana:

𝑤 (𝑢, 𝑗𝑘) merupakan bobot yang dimiliki oleh *user* u terhadap fitur 𝑗𝑘.

𝐼𝑢 merupakan satu set *item* yang telah di *rating* oleh *user* u.

𝑥 (𝑖, 𝑗) merupakan nilai kehadiran (angka 1 atau 0) sebuah fitur di dalam sebuah *item*.

𝑟 (𝑢, 𝑖) merupakan *rating* yang diberikan *user* u terhadap *item* i.

Sedangkan untuk menghitung prediksi *rating* yang akan diberikan seorang *user* terhadap sebuah *item* dapat menggunakan formula berikut ini.

(2)

Di mana:

merupakan prediksi *rating* *user* u terhadap *item* i*.*

𝐷𝑖 merupakan fitur yang muncul di dalam *item* i.

*Content-based recommendation* memiliki beberapa kelebihan (Ricci, et al., 2015), yaitu:

1. *Content-based recommendation* hanya memanfaatkan *rating* yang diberikan oleh *user* aktif.
2. *Content-based recommendation* dapat memberikan rekomendasi dari *user* lain yang menyukai *item* serupa.
3. *Content-based recommendation* dapat memberikan rekomendasi *item* yang belum dinilai *user* mana pun.

Namun, *content-based recommendation* juga memiliki beberapa kelemahan (Ricci, et al., 2015) , yaitu:

1. *Content-based recommendation* memiliki batasan dengan jenis fitur dengan objek yang direkomendasikan.
2. *Content-based recommendation* tidak dapat menemukan sesuatu yang tidak terduga, sehingga sistem akan merekomendasikan *item* yang serupa dengan yang sudah di *rating* sebelumnya.
3. Sistem rekomendasi harus memiliki *rating* agar dapat memberikan rekomendasi yang akurat, oleh karena itu hanya sedikit *item* yang dapat direkomendasikan sistem.

### Collaborative Filtering Recommendation

Sistem rekomendasi *collaborative filtering* memberikan rekomendasi *item* berdasarkan kumpulan *user* dengan preferensi yang sama atau *similarity* dari setiap *user*. Proses *collaborative* *filtering* dilakukan dengan melihat persamaan *rating* pada matriks *rating* dengan metode *user to user*. *Similarity* antara *user* dapat dihitung berdasarkan *rating* yang diberikan oleh *user*. Metode *collaborative* *filtering* juga sering disebut sebagai “*people-to-people correlation*” (Rahmawati, et al., 2018). Pendekatan *collaborative filtering* dibagi menjadi dua kelompok: Pendekatan berbasis memori dan model. Pendekatan berbasis memori (berbasis *heuristic*) membuat prediksi berdasarkan kesamaan antara *user* dan *item*. Pendekatan berbasis model berusaha membuat model prediksi melalui *machine learning*. Secara khusus, model berbasis matriks *factorization* telah memperoleh popularitas karena memiliki akurasi dan skalabilitas yang relatif tinggi. Meski demikian, *collaborative filtering* masih memiliki tantangan terhadap *sparsity* matriks *rating* (Yang, et al., 2017).

*Collaborative filtering* bertujuan untuk merekomendasikan *item* baru atau memperkirakan kegunaan *item* tertentu terhadap *user* tertentu berdasarkan kesukaan *user* di masa lalu dan pandangan dari *user* lain yang berpikiran sama. Terdapat dua tugas yang dapat dilakukan oleh *collaborative filtering* yakni (Nilashi, et al., 2013):

1. *Rating* *Prediction* yaitu memprediksi *rating* yang akan dimiliki oleh *item* yang tak terlihat kepada *user* target.
2. *Recommendation Task* yaitu menyediakan daftar rekomendasi *top*-N dari *item* relevan yang tak terlihat kepada *user* target.

*Collaborative filtering* beroperasi di dalam sebuah matriks *rating*. *Rating* yang diberikan oleh *user* terhadap *item,* direpresentasikan sebagai R dan nilai *rating* tersebut merupakan bilangan bulat tidak negatif atau bilangan real dengan jarak tertentu. *Collaborative filtering* mencoba memprediksi *rating* yang akan diberikan oleh *user* terhadap suatu *item* yang belum pernah diberi *rating* sebelumnya.

Misalnya, terdapat lima *user* 𝑢1,u2,..𝑢5 dan lima *item* 𝑖1,*i*2,..𝑖5. Sistem rekomendasi akan memprediksi berapa *rating* yang akan diberikan oleh u1 kepada 𝑖5 seperti pada tabel 2.4.

Tabel 2. Contoh Matriks *User* x *Item* Sistem Rekomendasi *Collaborative Filtering*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *Item* | *Item* | *Item* | *Item* | *Item* |
| *User* | 5 | 7 | 5 | 7 | ? |
| *User* | 5 | 7 | 5 | 7 | 9 |
| *User* | 5 | 7 | 5 | 7 | 9 |
| *User* | 6 | 6 | 6 | 6 | 5 |
| *User* | 6 | 6 | 6 | 6 | 5 |

Proses pada *collaborative filtering* dimulai dengan menghitung *similarity* antar *user* yang memiliki ketertarikan yang sama dengan menggunakan persamaan (Rahmawati, et al., 2018):

(3)

Keterangan:

adalah *similarity* antara *user* u dan *user* j

adalah interaksi *user* u pada pekerjaan x

adalah rata-rata interaksi dari *user* u

adalah interaksi dari *user* j pada pekerjaan x

adalah rata-rata interaksi dari pengguna j

Kemudian dihitung prediksi interaksi menggunakan persamaan (Rahmawati, et al., 2018):

(4)

Keterangan:

U’x adalah prediksi interaksi *user* u pada pekerjaan x

adalah rata-rata interaksi dari *user* u

adalah interaksi dari *user* j pada pekerjaan x

adalah *similarity* antara *user* u dan *user* j

Pada praktiknya sistem rekomendasi *collaborative filtering* mengalami berbagai macam masalah. Salah satu masalahnya yaitu *sparsity* matriks *rating*. Di sebagian besar sistem rekomendasi, banyak *user* hanya menilai sejumlah kecil *item* sehingga membuat ukuran kesamaan memiliki akurasi yang sangat rendah (Saeed & Mansoori, 2017)*. Sparsity* identik dengan data yang jarang atau banyaknya data yang masih kosong. Hal ini diakibatkan oleh adanya *item* baru yang muncul, *item* yang baru muncul bisa jadi kurang sesuai dengan keinginan *user* sehingga *user* baru membiarkan begitu saja *item* yang ditampilkan dan tidak memberikan *rating-*nya pada suatu *item* karena kurangnya referensi. *Rating* *user* merupakan bagian penting untuk menghasilkan rekomendasi yang andal, *item* yang tidak mendapatkan *rating* akan sulit untuk ditemukan kaitannya dengan *item* yang lain. Banyak lagi faktor yang menyebabkan data *sparsity*, sehingga menjadi tantangan terbesar dalam *collaborative filtering* yang paling sulit untuk diatasi (Li, et al., 2012). Sistem rekomendasi *collaborative filtering* memiliki beberapa kekurangan, diantaranya adalah (Lucas, et al., 2013):

1. Hasil rekomendasi yang tidak andal yang dikarenakan matriks *rating* yang jarang (*sparsity problem*).
2. Jika terdapat *item* baru di dalam sistem tersebut, maka sistem tidak dapat merekomendasikan *item* tersebut sampai *user* lain berminat pada *item* tersebut (*first rater problem*)
3. Hasil rekomendasi tidak selamanya andal, terdapat kemungkinan adanya sebuah *user* yang hanya memiliki sedikit *neighbor* dengan tingkat kesamaan (*similarity*) yang kecil (*grey sheep problem*)

### Hybrid Approaches

Masing-masing teknik sistem rekomendasi memiliki kelebihan dan kelemahannya tersendiri. Sehingga algoritma *hybrid recommendation* *system* yang menggabungkan kekuatan dari model dan algoritma yang berbeda untuk mengatasi dan menyelesaikan masalah-masalah yang ada pada *content-based* dan *collaborative filtering recommendation*, dan menjadi target penelitian yang sedang marak dikembangkan (Lü, et al., 2012). Terdapat beberapa cara penggabungan yang dapat dilakukan dalam metode *hybrid* (Li & Kim, 2003) yaitu:

1. Penggabungan Linear (*linear combination*)

Penggabungan ini menggabungkan hasil prediksi (*rating*) dari metode *content-based* dan *collaborative*. Penggabungan ini dilakukan dengan cara pemberian *ranking* atau *voting.*

1. Penggabungan Secara Sekuensial (*Sequential Combination*)

Penggabungan ini adalah melakukan perhitungan pada salah satu pendekatan *filtering* (misalkan *content-based*) kemudian hasilnya digabungkan dengan pendekatan lainnya (misalkan *collaborative*).

1. Penggabungan secara *Item-based Clustering Hybrid Method* (ICHM)

Penggabungan ini menggabungkan informasi *item* dan *rating* *user* untuk menghitung kemiripan *item*. Metode *Item-based* *Clustering* *Hybrid Method (ICHM)* merupakan sebuah metode yang menerapkan penggabungan *hybrid recommender* *system* dengan tujuan untuk meningkatkan akurasi prediksi pendekatan *collaborative filtering* dan menangani masalah *item* baru yang belum di *rating* (*cold-start problem*).

## Deep Learning

*Deep learning* adalah bentuk *machine learning* yang memungkinkan komputer untuk belajar dari pengalaman dan memahami dunia dalam hal hierarki konsep. Karena komputer mengumpulkan pengetahuan dari pengalaman, sehingga operator komputer manusia tidak diperlukan untuk menentukan semua pengetahuan yang dibutuhkan oleh komputer. Hierarki konsep memungkinkan komputer untuk mempelajari konsep yang rumit dengan membangunnya menjadi konsep yang lebih sederhana. Grafik hierarki ini akan memiliki banyak lapisan (Kwanggi, 2016).

*Deep learning* telah berhasil diterapkan di banyak domain penelitian berbeda, seperti *computer vision*, *speech recognition*, *natural language processing* dan lain sebagainya.Pada bidang ini, dibandingkan dengan pendekatan tradisional, pendekatan *deep learning* meningkatkan kinerja secara luar biasa. Karena keberhasilan besar *deep learning*, beberapa peneliti mencoba menggunakannya dalam sistem rekomendasi dan berharap dapat meningkatkan kinerja sistem rekomendasi, seperti halnya penerapan model *deep learning* di bidang penelitian lain (Liu & Wu, 2017). Dalam satu dekade terakhir, banyak sekali penelitian yang mencoba untuk memperkenalkan metode *deep learning* ke dalam sistem rekomendasi untuk meningkatkan kinerja, hasilnya metode ini dapat memperoleh hasil rekomendasi yang memuaskan berbeda dengan sistem rekomendasi tradisional. Misalnya, *Restricted Boltzmann Machines* (RBM) yang terdiri dari *hidden layer* dan *visible layer* (Zhou, et al., 2018)*.*

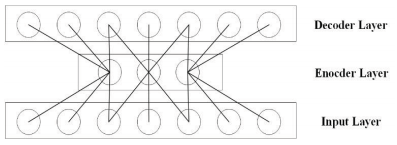
Salah satu teknik yang digunakan pada *deep learning* adalah *auto-encoder. Auto-encoder* merupakan metode *unsupervised learning* yang mencobamerekonstruksi masukan data di lapisankeluaran secara umum, lapisan *bottleneck* (lapisan paling tengah) digunakan sebagairepresentasi fitur menonjol dari datamasukan. Ada banyak varian dari *auto-encoders* seperti *denoising auto-encoder, marginalized denoising auto-encoder, sparse auto-encoder, contractive auto-encoder* dan *variational auto-encode*r (VAE). Teknik *deep learning* memungkinkan model untuk secara otomatis mempelajari fitur bagi *user* dan *item* dari sumber daya yang berbeda. Fitur ini digeneralisasi dengan baik dan dapat digunakan secara efektif untuk meningkatkan kualitas rekomendasi. Dengan menggunakan *deep learning* dalam pemodelan berbagai tipe data, sistem rekomendasi akan dapat lebih memahami apa yang dibutuhkan *user* dan hal tersebut akan meningkatkan hasil rekomendasi (Ikasari, et al., 2018). Terdapat beberapa teknik *deep learning* yang digunakan pada sistem rekomendasi (Zayyad & Kurniawardhani, 2021) yaitu:

### Multilayer Perceptron

*Multilayer Perceptron (*MLP*)* adalah bentuk khusus dari *Neural Network* (NN), MPL merupakan model komputasi yang sering digunakan untuk memecahkan masalah pembelajaran. Ini terdiri dari grafik terarah berbobot dari satu set neuron yang saling berhubungan yang diatur dalam lapisan terpisah: lapisan *input*, lapisan tersembunyi, dan lapisan *output*. Setiap lapisan memiliki satu atau lebih neuron. Setiap neuron *i* terhubung ke neuron *j* dari lapisan berikutnya dengan satu set link berbobot yang dilambangkan dengan *w1i,...,wji*. Pada lapisan input, {*a1, a2,...,am*} mewakili *m* sinyal *input* yang terkait dengan m atribut. Pada lapisan tersembunyi dan keluaran, setiap neuron j menerima sinyal masukan sebagai kombinasi linier dari keluaran yang diberikan oleh: . Kombinasi linier ditransformasikan menjadi sinyal keluaran menggunakan fungsi aktivasi (*vj*). Sinyal-sinyal ini dikirim dalam arah maju lapis demi lapis ke lapisan keluaran yang memberikan keluaran *yj* untuk setiap neuron keluaran j. Dalam klasifikasi, setiap kelas dikaitkan dengan neuron keluaran dan prediksi biasanya diberikan oleh yang memiliki tingkat aktivasi tertinggi (Ribeiro, et al., 2012).

### Autoencoder Based Recommendation

*Autoencoder* adalah jenis jaringan saraf yang digunakan untuk mengekstraksi fitur dari data, atau merekonstruksi data *input* dari data yang rusak, yang mirip dengan tugas rekomendasi. Jaringan *autoencoder* klasik memiliki lapisan *input* dan *output* yang terhubung penuh dengan ukuran yang sama, dan lapisan tersembunyi yang terhubung penuh dengan ukuran yang lebih kecil. Saat dilatih, model menggunakan data *input* dan data *output* yang sama. Gambar 2.1 adalah struktur *autoencoder* yang menunjukkan bahwa *autoencoder* dapat digunakan dalam sistem rekomendasi. Sebagai contoh, setelah melatih jaringan dengan kode vektor dalam basis data kode, jaringan dapat melakukan pekerjaan rekomendasi dengan cara berikut: Diberikan *input* vektor kode 0-1, *output* jaringan dapat dianggap sebagai *recommendation score vector*. Dengan kata lain, nilai setiap dimensi dari vektor keluaran adalah skor rekomendasi dari API yang bersangkutan. API diberi peringkat berdasarkan skor, dan API dengan peringkat tinggi akan menjadi hasil rekomendasi (Liu, et al., 2019). Berikut struktur dari *autoencoder*:

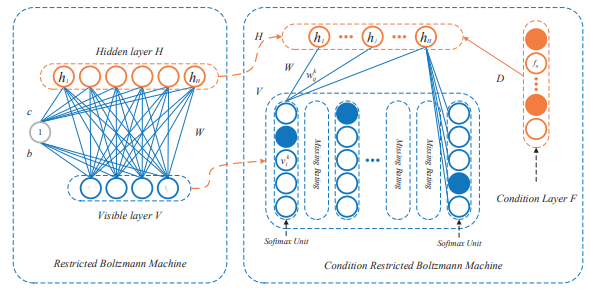


Gambar 2. Struktur *Autoencoder*

Sumber (Liu, et al., 2019)

### Restricted Boltzmann Machine based Recommendation

*Restricted Boltzmann Machine* (RBM) adalah model graf generatif tak berarah dua lapis, yang terdiri dari *m* unit tampak *V = (v1, ..., vm )* mewakili data yang dapat diamati dan *n* unit tersembunyi *H = (h1, ..., hn )* menangkap dependensi antara variabel yang dapat diamati yang ditunjukkan seperti pada gambar 2.2. Ketika model RBM dasar digunakan untuk CF, setiap *user* atau *item* dapat dianggap sebagai kasus pelatihan tunggal untuk RBM. Setiap RBM mungkin memiliki jumlah unit *softmax* yang terlihat berbeda tetapi memiliki jumlah unit tersembunyi yang sama, yaitu, semua bobot dan bias yang terhubung dibagi di antara model RBM (Xie, et al., 2016). Berikut model dari RBM:



Gambar 2. *Restricted Boltzmann Machine* dengan unit tersembunyi biner dan unit terlihat *softmax*

Sumber (Xie, et al., 2016)

## Deep Collaborative Filtering

Algoritma *deep collaborative filtering* adalah algoritma yang dibuat untuk mengatasi masalah *sparsity* matriks *rating* pada algoritma *collaborative filtering.* Penyelesaian permasalahan tersebut dilakukan dengan mempelajari *side information* yang diperoleh dari profil *user* / *item* seperti demografi *user*, genre *item*, dll. *Deep collaborative filtering* yang menjembatanimatriks *factorization* dan *deep feature learning* merupakan *hybrid collaborative filtering model*. *Deep collaborative filtering* menggabungkan *probabilistic matrix factorization* dan *marginalized denoising auto-encoders. Probabilistic matrix factorization* adalah pendekatan *collaborative filtering* yang diterapkan secara luas dengan kinerja yang sangat baik, dan *marginalized denoising auto-encoders* adalah metode yang andal dalam mengekstraksi fitur tingkat tinggi dari *input*. *Marginalized denoising auto-encoders* merupakan varian dari *auto-encoder* yang merupakan metode *unsupervised learning.* Kombinasi keduanya memanfaatkan fungsinya masing-masing untuk mempelajari model yang lebih kompleks (Li, et al., 2015). Notasi yang digunakan pada *deep collaborative filtering* dapat dilihat pada tabel 2.5 dan ilustrasi penerapan *deep collaborative filtering* dapat dilihat pada gambar 2.3.

Tabel 2. Ringkasan Notasi

Sumber (Li, et al., 2015)

|  |  |
| --- | --- |
| Notasi | Deskripsi |
| m | Jumlah *user* |
| n | Jumlah *item* |
| d | Dimensi *latent factor* |
| p | Dimensi *user feature* |
| q | Dimensi *item feature* |
| *R* € | Matriks *Rating* |
| *U* € | *Latent factor user* |
| *V* € | *Latent factor item* |
| *X* € | *Side information of user* |
| *Y* € | *Side information of item* |
| *W1*  € | *Mapping Function* untuk X di *auto-encoder* |
| *P1* € | Matriks *Projection* untuk *U* |



Gambar 2. Ilustrasi Penerapan *Deep Collaborative Filtering*

Sumber (Li, et al., 2015)

### Matriks Factorization

Matriks *factorization* adalah pendekatan *collaborative filtering* yang paling efektif. Ini memungkinkan kita untuk menemukan *latent factor* interaksi *user-item* dengan memfaktorkan matriks interaksi ke dalam *latent space* *user-item feature*. Matriks *factorization* menguraikan matriks *rating* *R* € yang terdiri dari *rating yang diberikan* oleh *m* *user* untuk *n* *item* menjadi dua *low-rank* *U* € dan *V* € yang terdiri dari fitur *user* dan *item*. Contoh matriks *factorization* dapat dilihat pada gambar 2.4.



Gambar 2. Contoh *Matriks Factorization*

(Sumber: <https://developers.google.com/machinelearning/recommendation/collaborative/matrix>)

Salah satu metode matriks *factorization* klasik adalah *probabilistic matrix factorization* (PMF). Asumsi yang mendasari metode ini adalah bahwa distribusi probabilitas sebelumnya dari *latent factor* dan probabilitas *rating* yang diamati mengikuti *Gaussian distribution*. Banyak algoritma yang telah dikembangkan untuk meningkatkan kinerja PMF, salah satunya dengan memasukkan *side information* seperti hubungan sosial. Tetapi metode matriks *factorization* masih mengalami masalah *cold-start*, yaitu rekomendasi apa yang harus dibuat ketika *user* / *item* baru tiba di sistem. Masalah lain yang sering muncul di banyak aplikasi dunia nyata adalah *sparsity* data. Memasukkan *side information* telah menunjukkan kinerja yang menjanjikan dalam *collaborative filtering*. Tetapi akan bermasalah jika *side information* tidak lengkap. Oleh karena itu, menggabungkan *deap learning* dengan matriks *factorization* akan memberikan hasil yang lebih baik. Model pada penelitian ini didasarkan pada pendekatan *probabilistic matrix factorization* karena telah terbukti memiliki kinerja yang sangat baik pada beberapa domain.

Sistem mempelajari *latent factor* dengan meminimalkan fungsi objektif berikut:

(5)

Di mana adalah *loss* *function* untuk memprediksi *rating* menggunakan *latent factor* U dan V dan dua istilah terakhir adalah regularisasi yang digunakan untuk menghindari *overfitting*. merupakan *frobenius norm* dan . C merupakan bobot matriks (Li, et al., 2015).

### Marginalized Denoising Auto-encoder

Sebagai bentuk dari jaringan neural, *auto-encoder* mengambil suatu masukan dan memetakannya (menyandikan) ke representasi tersembunyi melalui pemetaan deterministik. *Denoising* *auto-encoders* merekonstruksi masukan dari versi data yang rusak dengan mempelajari pemetaan yang lebih kuat dari data. Berbagai jenis *auto-encoder* telah dikembangkan dalam literatur dan telah menunjukkan hasil yang menjanjikan di beberapa domain. Selain itu, *denoising auto-encoder* dapat ditumpuk untuk membangun *deep network* yang juga dikenal sebagai *stacked denoising autoencoder* (SDA) yang memungkinkan pembelajaran representasi tingkat yang lebih tinggi. Salah satu kelemahan utama SDA adalah biaya pelatihan komputasi yang tinggi, karena SDA mengandalkan teknik pengoptimalan berulang dan numerik untuk mempelajari sejumlah besar parameter model.

*Marginalized denoising auto-encoder* (mDA) adalah varian dari SDA yang menghindari biaya komputasi yang tinggi dengan meminggirkan kerusakan fitur secara acak dan dengan demikian memiliki solusi *closed-form* untuk mempelajari parameter model. Oleh karena itu, mDA sangat skalabel dan lebih cepat dari SDA.

Diberikan sampel set mDA mempertimbangkan beberapa lintasan (misalnya, c-*times*) dari *corruptions* acak untuk mendapatkan . Kemudian merekonstruksi *input* dengan pemetaan W yang meminimalkan *squared loss* sebagai berikut:

, (6)

Di mana merepresentasikan *corrupted version* ke- dari *input original* dan W merepresentasikan pemetaan yang diharapkan dapat meminimalkan *loss function*.

Objektif di atas dapat ditulis ulang dalam bentuk matriks sebagai

, (7)

Di mana merupakan *c-times repeated version* dari X, dan merupakan *corrupted version*. , di mana dan (Li, et al., 2015).

### Normalisasi Data Pada Deep Collaborative Filtering

Normalisasi data dilakukan pada matriks *rating* yang mengindikasikan *rating* oleh *user* terhadap *item*. Normalisasi dilakukan dengan menggunakan *zero-mean normalization*. Nilai *rating* dari dataset dalam skala 1 – 10 akan dinormalisasikan menjadi *zero-mean* (Li, et al., 2015). Berikut rumus untuk metode *zero-mean normalization* (Henderi, et al., 2021).

(8)

Di mana:

adalah nilai hasil normalisasi

adalah nilai *rating* yang lama

adalah rata-rata

adalah nilai *standard deviation*

U1-U10 adalah *user* dan J1-J10 adalah *item* yang di *rating* ataupun yang tidak di *rating* oleh *user*. Data yang belum dinormalisasi dapat dilihat pada tabel 2.6 dan data yang sudah dinormalisasi dapat dilihat pada tabel 2.7.

Tabel 2. Data Sebelum Normalisasi

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** | **J7** | **J8** | **J9** | **J10** |
| **U1** | 8 | 4 | 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| **U2** | 0 | 1 | 0 | 1 | 2 | 1 | 0 | 2 | 0 | 5 |
| **U3** | 5 | 1 | 1 | 0 | 1 | 3 | 0 | 1 | 4 | 0 |
| **U4** | 0 | 0 | 3 | 4 | 0 | 7 | 0 | 6 | 0 | 6 |
| **U5** | 0 | 0 | 1 | 0 | 0 | 5 | 2 | 1 | 3 | 0 |
| **U6** | 0 | 1 | 0 | 5 | 3 | 4 | 0 | 4 | 8 | 2 |
| **U7** | 1 | 2 | 0 | 0 | 0 | 4 | 3 | 3 | 4 | 5 |
| **U8** | 0 | 4 | 3 | 2 | 3 | 0 | 2 | 0 | 0 | 0 |
| **U9** | 0 | 3 | 4 | 2 | 0 | 1 | 2 | 2 | 1 | 6 |
| **U10** | 1 | 4 | 0 | 3 | 1 | 3 | 0 | 3 | 5 | 0 |

Tabel 2. Data Setelah Normalisasi

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | **J1** | **J2** | **J3** | **J4** | **J5** | **J6** | **J7** | **J8** | **J9** | **J10** |
| **U1** | 2,989 | 1,042 | 0,555 | -0,905 | -0,905 | -0,419 | -0,905 | -0,905 | -0,905 | -0,905 |
| **U2** | -0,905 | -0,419 | -0,905 | -0,419 | 0,068 | -0,419 | -0,905 | 0,068 | -0,905 | 1,528 |
| **U3** | 1,528 | -0,419 | -0,419 | -0,905 | -0,419 | 0,555 | -0,905 | -0,419 | 1,042 | -0,905 |
| **U4** | -0,905 | -0,905 | 0,555 | 1,042 | -0,905 | 2,502 | -0,905 | 2,015 | -0,905 | 2,015 |
| **U5** | -0,905 | -0,905 | -0,419 | -0,905 | -0,905 | 1,528 | 0,068 | -0,419 | 0,555 | -0,905 |
| **U6** | -0,905 | -0,419 | -0,905 | 1,528 | 0,555 | 1,042 | -0,905 | 1,042 | 2,989 | 0,068 |
| **U7** | -0,419 | 0,068 | -0,905 | -0,905 | -0,905 | 1,042 | 0,555 | 0,555 | 1,042 | 1,528 |
| **U8** | -0,905 | 1,042 | 0,555 | 0,068 | 0,555 | -0,905 | 0,068 | -0,905 | -0,905 | -0,905 |
| **U9** | -0,905 | 0,555 | 1,042 | 0,068 | -0,905 | -0,419 | 0,068 | 0,068 | -0,419 | 2,015 |
| **U10** | -0,419 | 1,042 | -0,905 | 0,555 | -0,419 | 0,555 | -0,905 | 0,555 | 1,528 | -0,905 |

### Menghitung Kesalahan Model (*Loss Function*)

Inti dari *machine learning* adalah model yang dilatih dengan data pelatihan, dan metode untuk menyesuaikan parameter model adalah berdasarkan *loss function*. Tujuan dari pelatihan ini adalah untuk meminimalkan *loss* akibat salah prediksi rata-rata model. *Loss function* diterapkan untuk menilai kualitas dan kapabilitas model pada pelatihan model *machine learning* (Chen, et al., 2018). *Deep collaborative filtering* memanfaatkan *loss function* dari *probabilistic matrix factorization* untuk menguraikan matriks *rating* R yaitu, . Berikut fungsi objektif yang digunakan pada *deep collaborative filtering*:

(9)

Di mana:

A adalah matriks indikator yang menunjukkan entri yang tidak kosong di matriks *rating* dan melambangkan produk *hadamard* atau *pointwise product*. adalah pemetaan rekonstruktif, adalah matriks *Projection*, adalah parameter *tradeoff*. Istilah pertama menunjukkan proses pembelajaran *marginalized denoising auto encoders.* Ini mengukur kesalahan rekonstruksi antara input fitur *user* dan fitur yang dipetakan dari *input* yang rusak yakni, . adalah *learned mapping* yang diharapkan untuk meminimalkan *loss data.* Istilah kedua berfungsi untuk menghubungkan *hidden layer feature* dan *latent factor* U (Li, et al., 2015).

### Tahapan Rekomendasi Pada Deep Collaborative Filtering

* + - 1. Mencari *latent factor user* dan *item*

*Latent factor* *model* telah menjadi teknik yang sukses digunakan untuk membangun sistem rekomendasi. Sebagian besar penelitian difokuskan untuk mempelajari l*atent factor* dalam situasi data *sparsity* (Rao, et al., 2017). Model *latent factor* mencoba menemukan perkiraan *rating* di matriks *user-item*, di mana *rating* digunakan untuk mengisi entri yang hilang (Sammut & Webb, 2011). Pada proses untuk mendapatkan *latent factor* akan terdapat *noisy data* yang tak terhingga jumlahnya sehingga perlu dilakukan *mapping function* untuk *side information user* dan *item* (Li, et al., 2015). Dilakukan *mapping function* W1 untuk *side information* *user* dan W2 untuk *side information item*.

(10)

(11)

,

.

Pada dan Misalkan = Z, dan p = *corrupt\_ratio*, untuk mendapatkan nilai Z setiap nilai yang bukan di diagonal akan dikalikan dengan (1 - p)2 dan yang berada di diagonal akan dikalikan dengan (1 - p) (Chen, et al., 2012).

(12)

Demikian pula dengan dan , setiap elemen pada akan dikalikan dengan (1 - p).

(13)

Lalu diperlukan suatu matriks *projection* untuk memetakan *latent factor user* dan *item* ke *feature space.* *Feature space* merupakan kumpulan fitur yang digunakan untuk mengkarakterisasi data yang akan dilatih (Mohamed, 2017). Berikut rumus matriks *projection*:

(14)

(15)

Di mana untuk matriks *projection* *latent factor user* dan untuk matriks *projection* *latent factor item*.

Untuk menyelesaikan permasalahan *latent factor* U dan V, digunakan algoritma *stochastic gradient descent (SGD).* SGD adalah metode yang befungsi untuk mengoptimalkan fungsi *objective* *machine-learning* (Li & Orabona, 2019). Berikut rumus untuk mencari SGD:

(16)

(17)

Di mana η adalah *learning rate*, dan turunan dari rumus diatas didefinisikan sebagai berikut:

(18)

(19)

**Algoritma DCF untuk menemukan *latent factor* *user* dan *item***

**Input:** Matriks *rating* *R*, *user feature* *X, item feature* *Y,* *Parameters*

**Output :** *Latent factor* *U,V*

1: *Initialize* *U,V,*and

2: ***while*** *validation error decreases*, **do**

3: *Update* *using* (10);

4: *Update* *using* (11);

5: *Update* *using* (14);

6: *Update* *using* (15);

7: ***for*** *each* *observed* , ***do***

8: *Update* *using* (16);

9: *Update* *using* (17);

10: ***end for***

11: ***end while***

* + - 1. Menghasilkan rekomendasi *set*

Setelah *latent factor user* dan *item* ditemukan, dilakukan perkalian matriks untuk kedua *latent factor* seperti pada gambar 2.5, sehingga didapatkan *user-item* matriks *rating* yang baru. Matriks *rating* yang baru dapat dilihat pada tabel 2.8.



Gambar 2. Perkalian Matriks *Latent Factor User* dan *Item*

Tabel 2. Matriks *Rating* Hasil Prediksi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Item1 | Item2 | Item3 | Item4 |
| User1 | 2,36704 | 3,55596 | 1,68560 | 2,30801 |
| User2 | 3,61134 | 4,85569 | 3,06585 | 3,81974 |
| User3 | 3,13898 | 3,82720 | 2,66200 | 3,31269 |
| User4 | 2,88840 | 3,23815 | 2,30140 | 3,13117 |

Matriks *rating* yang didapatkan akan dilakukan normalisasi kembali menggunakan metode *min-max normalization* sehingga nilai *rating* yang didapat berada di dalam skala 1-10. Berikut rumus metode *min-max normalization* (Gajera, et al., 2016).

(20)

Di mana:

adalah nilai hasil normalisasi

adalah nilai *rating* yang lama

adalah nilai minimal pada matriks *rating*

adalah nilai maksimal pada matriks *rating*

adalah nilai minimal yang baru

adalah nilai maksimal yang baru

Normalisasi matriks *rating* menggunakan *min-max normalization* dapat dilihat pada tabel 2.9.

Tabel 2. Matriks *Rating* Prediksi Setelah Normalisasi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Item1 | Item2 | Item3 | Item4 |
| User1 | 2,934633 | 6,31002 | 1 | 2,767044 |
| User2 | 6,467245 | 10 | 4,91858 | 7,058901 |
| User3 | 5,126198 | 7,08008 | 3,772035 | 5,619367 |
| User4 | 4,414793 | 5,407746 | 2,748278 | 5,104025 |

Jadi, setiap *user* akan mendapat rekomendasi *item* berdasarkan prediksi *rating user* terhadap *item* di mulai dari *item* dengan *rating* tertinggi sampai yang terendah.

## Pengujian Perangkat Lunak dan Pengujian Keakuratan Hasil Rekomendasi

Perangkat lunak memiliki *bug* dan hal itu tidak bisa untuk dihindari. Kode ditulis oleh manusia, dan manusia dapat membuat kesalahan. Kebutuhan dapat ambigu atau salah, Kebutuhan dapat disalahpahami, komponen perangkat lunak dapat disalahgunakan, pengembang dapat membuat kesalahan saat menulis kode, dan bahkan kode yang pernah berfungsi mungkin tidak lagi benar ketika asumsi yang sebelumnya valid menjadi tidak lagi berlaku setelah perubahan. Pengujian perangkat lunak adalah respons intuitif untuk masalah ini. Setelah membangun dan menjalankan sistem, dilakukan pemeriksaan apakah sistem berfungsi seperti yang diharapkan (Fraser & Rojas, 2019). Terdapat 2 pengujian yang akan dilakukan antara lain:

### Pengujian Perangkat Lunak dengan Black Box Testing

*Black box testing* memainkan peran penting dalam pengujian perangkat lunak, ini membantu dalam validasi fungsionalitas keseluruhan sistem. *Black box testing* dilakukan berdasarkan kebutuhan pelanggan sehingga setiap persyaratan yang tidak lengkap atau tidak terduga dapat dengan mudah diidentifikasi dan diatasi. *Black box testing* dilakukan berdasarkan perspektif *user* akhir. Pentingnya *black box testing* adalah menangani masukan yang valid dan tidak valid dari sudut pandang pelanggan (Nidhra & Dondeti, 2012).

Keuntungan utama dari *black box testing* adalah, penguji tidak perlu memiliki pengetahuan tentang bahasa pemrograman tertentu, tidak hanya bahasa pemrograman tetapi juga pengetahuan tentang implementasi. Dalam *black box testing*, baik pemrogram maupun penguji tidak bergantung satu sama lain. Keuntungan lainnya adalah pengujian dilakukan dari sudut pandang *user*. Keuntungan signifikan dari *black box testing* adalah membantu mengungkap ambiguitas atau ketidakkonsistenan dalam spesifikasi persyaratan (Nidhra & Dondeti, 2012).

### Pengujian Keakuratan Hasil Rekomendasi

Sistem rekomendasi diimplementasikan untuk membantu *user* dalam mengenali informasi yang diinginkan. Akurasi adalah salah satu metrik kinerja yang banyak digunakan untuk rekomendasi untuk mengukur tingkat *error* antara *rating* aktual dan prediksi. Ini juga merupakan kriteria paling umum yang digunakan untuk mengevaluasi keberhasilan sistem pemberi rekomendasi baik yang terkait dengan prediksi *rating* maupun rekomendasi. Dalam sistem rekomendasi, hasil rekomendasi yang diberikan perlu dilakukan pengukuran untuk mengukur tingkat akurasi hasil rekomendasi dapat dilakukan dengan melihat nilai *error* pada hasil rekomendasi yang diberikan. Metode perhitungan *error* yang digunakan yaitu *Mean Absolute Error (MAE)* dan *Root Mean Square Error* (RMSE). MAE dan RMSE disebut sebagai metrik akurasi prediksi untuk mengevaluasi kesesuaian dengan *rating* *user* sebenarnya dari prediksi sistem pemberi rekomendasi (Nilashi, et al., 2013).

1. *Mean Absolute Error* (MAE)

MAE adalah metrik yang umumnya digunakan untuk pengukuran akurasi prediksi dan jauh lebih banyak digunakan daripada metrik lainnya. MAE ditentukan sebagai deviasi absolut rata-rata antara *rating* yang diprediksi dan *rating* sebenarnya. Persamaan MAE yaitu sebagai berikut (Nilashi, et al., 2013):

(21)

Di mana*,*

*pi* = Nilai prediksi *item* ke i

*qi* = Nilai *rating* sebenarnya *item* ke i

N = jumlah *item* yang dihitung

Semakin rendah nilai MAE yang didapat, maka sistem semakin akurat dalam memprediksi.

1. *Root Mean Square Error* (RMSE)

RMSE adalah metrik lain dalam akurasi prediksi, yaitu menghitung statistik akurasi. Secara konsep ini mirip dengan MAE, perbedaannya RMSE mendapatkan hasil dengan mengkuadratkan *error* (Nilashi, et al., 2013). Berikut rumus persamaan RMSE:

(22)

Di mana,

𝑓𝑡 = Permintaan aktual periode *t*

= Ramalan permintaan periode *t*

m = Jumlah periode peramalan