BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1. Tinjauan Pustaka

Pada penelitian sebelumnya yang dilakukan oleh Zhang, L. (Zhang, L., 2014), ia memanfaatkan algoritma *User-Based Collaborative Filtering* (UCF) untuk mencari rekomendasi yang fokus pada penyajian *item* baru atau disebut *novelty*. Ia mengembangkan algoritma UCF dengan menambahkan faktor *dissimilarity* (*item* yang berbeda dari profil pengguna aktif) dan faktor tingkat kepopuleran *item* dengan tujuan penambahan faktor-faktor tersebut dapat membantu pengguna aktif mendapatkan *item* yang bersifat "baru" (*novelty*). Kata "baru" atau *novelty* dalam penelitian ini merujuk pada *item* yang di *rating* oleh pengguna aktif, namun *item* tersebut sama sekali tidak memiliki kesamaan profil dengan daftar *rating* pengguna. Manfaat yang didapat dari penelitian ini adalah supaya rekomendasi yang ditawarkan tidak bersifat kaku dan tidak sempit lingkupnya.

Dalam perkembangannya, para peneliti mulai mengembangkan algoritma-algoritma rekomendasi yang sudah ada dengan tujuan mendapatkan akurasi dan presisi rekomendasi yang lebih baik dari sebelumnya. Selain menambahkan variabel-variabel baru, beberapa peneliti memanfaatkan metode *clustering* seperti K-Means untuk membagi data *rating* menjadi beberapa *cluster*. Inilah yang dilakukan oleh Zhang (Zhang, L., 2014) dalam penelitiannya dengan fokus pengembangan algoritma *User-Based Clustering Collaborative Filtering*. Hasil

dari penelitian ini menunjukkan bahwa dengan memanfaatkan *clustering* dalam algoritma rekomendasi dapat mengurangi masalah ketersebaran data dan menambah kemampuan sistem rekomendasi dalam mencari *neighbor* yang baik.

Tahap selanjutnya dalam perkembangan algoritma sistem rekomendasi adalah menggabungkan dua atau lebih algoritma rekomendasi dengan tujuan supaya dapat saling menutupi kelemahan algoritma satu sama lain. Penelitian yang dilakukan oleh Shinde (Shinde, S. K., 2011) dengan menggabungkan *User-Based* dan *Item-Based Collaborative Filtering* serta memanfaatkan K-*Medoids clustering* dalam pembagian *neighbor*-nya, menunjukkan bahwa dengan memanfaatkan *clustering* dalam proses pencarian *neighbor* dapat menambah kemampuan rekomendasi sistem.

Selain berfokus pada penggabungan algoritma rekomendasi ataupun *clustering*, De Campos (De Campos et al., 2010) melakukan penelitian dengan melibatkan "model" jaringan (*network*) pada sistem rekomendasi yang bersifat *Hybrid*. Tujuan pemodelan ini adalah untuk mengatur peran dari algoritma rekomendasi gabungan yang diterapkan dalam sistem. Ia memanfaatkan *Bayesian Network Model* supaya peran algoritma rekomendasi gabungannya (*Hybrid*) dapat menghasilkan rekomendasi yang akurat.

Beberapa peneliti lain yang seperti Formoso (Formoso et al., 2013) lebih memfokuskan penelitiannya pada efisiensi data. Pada penelitiannya, ia menekankan bahwa dalam penerapannya di dunia nyata (*online*), faktor ukuran data dan kecepatan penyajian rekomendasi sangatlah penting. Dengan cara

mengkompresi ukuran matrix *user-item rating* hingga 75% dari ukuran aslinya, Formoso menunjukkan bahwa kecepatan rekomendasi naik dua kali lipat dibandingkan dengan rekomendasi yang masih menggunakan ukuran matrix aslinya.

Beberapa penelitian diatas memiliki fokusnya masing-masing, mulai dari penggabungan algoritma, *clustering*, hingga pemodelan jaringan dan mempercepat kecepatan rekomendasi. Dalam penelitian kali ini, peneliti ingin lebih fokus pada peningkatan kualitas pencarian *neighbor* dengan memanfaatkan *Pearson-Correlation* untuk menghitung seberapa dekat pengguna aktif dengan pengguna lainnya. Semakin besar nilai yang dihasilkan dari pengukuran kedekatan tersebut, maka semakin besar kemungkinannya pengguna tersebut adalah sumber rekomendasi yang baik bagi pengguna aktif.

Baik atau tidaknya rekomendasi yang diberikan, nantinya akan terlihat pada bagian evaluasi rekomendasi. Pada bagian evaluasi, sistem akan mencocokan seberapa banyak rekomendasi yang sudah di *rating* oleh pengguna. Daftar tersebutlah yang disebut dengan rekomendasi yang akurat.

Penelitian ini mencoba mencari tau seberapa akurat rekomendasi sistem yang menerapkan *User-Based Collaborative Filtering* dan memanfaatkan *Pearson-Correlation* sebagai dasar pencarian *neighbor*-nya. Peneliti memiliki asumsi bahwa pemilihan *neighbor* sangatlah berperan penting dalam keakuratan rekomendasi. Untuk itu peneliti hanya akan mengambil semua *neighbor* yang memiliki koefisien relasi lebih dari 0,5 (untuk nilai 1 yang berarti *neighbor*

memiliki relasi kuat dengan pengguna aktif) dengan pengguna aktif. Relasi dibawah batas tersebut dianggap relasi yang lemah dan tidak layak menjadi sumber rekomendasi bagi pengguna aktif.

Tabel 2.1 Tabel Rangkuman Tinjauan Pustaka

No	Nama Peneliti, Tahun	Judul Penelitian	Pokok Bahasan
1	Zhang, L., Peng, Li F., Phelan, C.A., 2014	Novel Recommendation of User-Based Collaborative Filtering	Penelitian ini membahas tentang pengembangan dari algoritma rekomendasi User-Based Collaborative Filtering dengan melibatkan faktor "pembaruan" (novelty) pada item yang akan direkomendasikan.
2	Zhang, L., Qin, Tao, Teng, PiQiang, 2014	An Improved Collaborative Filtering Algorithm Based on User Interest	Pada penelitian ini, algoritma Collaborative Filtering dikembangkan dengan mengelompokkan para pengguna sistem terlebih dahulu melalui clustering yang didasari dari ketertarikan tiap pengguna. Metode pencarian neighbor juga dikembangkan untuk mengatasi ketersebaran data dalam sistem.
3	Shinde, S. K., Kulkarni, U. V., 2011	Hybrid Personalized Recommender System Using Fast K-medoids Clustering Algorithm	Penelitian ini menggabungkan User-Based dan Item-Based Collaborative Filtering dan mengelompokkan pengguna dengan K-Medoids Clustering Algorithm dengan tujuan untuk mencari rekomendasi yang berkualitas dan akurat bagi pengguna aktif.
4	Formoso, Vreixo, Fernandez, D., Cacheda, F., Carneiro, V., 2013	Using Rating Matrix Compression Techniques to Speed Up Collaborative Recommendations	Penelitian ini fokus pada cara untuk mengkompresi tabel rating menjadi sebuah matrix dengan tujuan untuk mempercepat pencarian rekomendasi dan menghemat ukuran data.
5	Beel, J., Langer, S., Genzmehr,	A Comparative Analysis of	Pada penelitian ini membuktikan bahwa ada perbedaan hasil

	M., Gipp, B. and Nürnberger, A. 2013	Offline and Online Evaluations and Discussion of Research Paper Recommender System Evaluation	evaluasi rekomendasi yang didapat pada lingkungan sistem yang online dengan lingkungan sistem yang offline. Fokus pada penelitian ini adalah membandingkan hasil evaluasi pada sistem online dan offline.
6	De Campos,L. M., Fernandez-Luna,J. M., Huete,F. J., Rueda-Morales,M. A., 2010	Combining Content-Based And Collaborative Recommendations: A Hybrid Approach Based On Bayesian Networks	Penelitian ini menggabungkan Content-Based Filtering dengan Collaborative Filtering dan memodelkannya menggunakan pendekatan Bayesian Networks. Tujuan penggabungan kedua algoritma tersebut adalah untuk meningkatkan kualitas rekomendasi sistem.

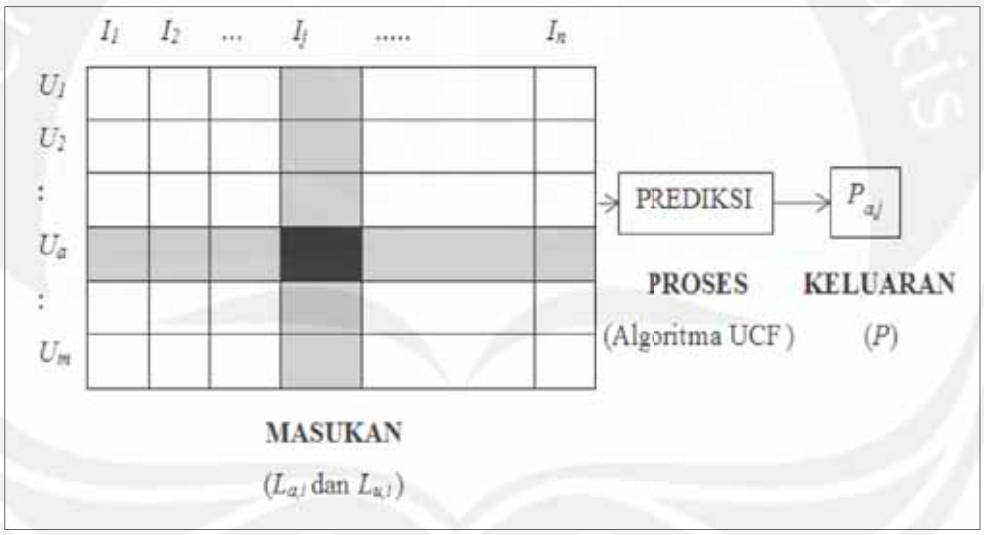
2.2. Tinjauan Teoritis

2.2.1. User-Based Collaborative Filtering

Pada UCF, ada banyak teknik yang digunakan untuk mencari nilai prediksi untuk pengguna aktif ($P_{a,i}$), dimulai dari cara sederhana dengan menjumlahkan semua *rating* dari pengguna lain yang memiliki selera sama dengan pengguna aktif (U_a) dan kemudian dibagi dengan jumlah pengguna tersebut, hingga menggunakan cara pembobotan. Seperti yang sudah dibahas sebelumnya, pengguna lain yang memiliki selera sama dengan U_a disebut dengan *neighbor*. Untuk mencari *neighbor* tersebut banyak cara yang ditawarkan, ada yang memanfaatkan teknik *K-nearest neighbor* (KNN) ada juga yang mencarinya secara manual dengan mencari irisan dari semua daftar *item* yang sudah di *rating* oleh pengguna aktif ($L_{a,i}$) dengan semua daftar *item* yang sudah di *rating* oleh pengguna lain ($L_{u,i}$) satu-persatu (Shinde, S. K. & Kulkarni, U. V., 2011) dan (Ghauth, K. I. & Abdullah, N. A., 2011). Hal terpenting yang perlu diperhatikan dalam mencari *neighbor* adalah calon *neighbor* setidaknya harus memiliki beberapa variabel yang tidak kosong (*null*) dari irisan $L_{u,i}$ dengan $L_{a,i}$ supaya nantinya dapat dilakukan pengukuran kedekatan.

Secara garis besar, dalam mencari nilai prediksi (P) dan daftar rekomendasi (L) untuk pengguna aktif, UCF dapat dibagi menjadi tiga tahap yaitu : masukan (*input*), proses (algoritma UCF) dan keluaran (P dan L). *Input* yang dimaksud adalah berupa nilai dari irisan $L_{a,i}$ dengan $L_{u,i}$ dan kemudian nilai

tersebut akan diproses menggunakan algoritma UCF yang menghasilkan nilai $P_{a,i}$. Lihat Rumus (3) untuk skema tahapan UCF, dimana I_j adalah *item* yang akan dicari nilai *P*-nya. Dalam algoritma UCF, sebelum mencari nilai $P_{a,j}$ perlu dicari terlebih dahulu *neighbor* dari U_a, setelah mendapatkan *neighbor* yang cukup, maka kemudian diambil $L_{u,j}$ dengan *u* adalah semua *neighbor* dari pengguna aktif (u_1 s/d u_m). Dari titik ini, $L_{u,j}$ bisa diproses dengan banyak jenis algoritma, mulai dari yang paling sederhana, pembobotan ringan dan pembobotan rata-rata *rating U*, untuk lebih jelasnya dapat dilihat pada Rumus (1).



Gambar 2.1 Skema Tahapan UCF

$$P_{a,j} = \frac{\sum_{U=1}^{m} R_{u,j}}{m} \qquad \text{.....(1)}$$

Pada Rumus (1), variabel *m* mewakili jumlah *neighbor* yang sudah terpilih dengan pendekatan tertentu. Algoritma ini adalah termasuk yang paling

sederhana dalam mencari nilai *P*, disebut sederhana karena dalam perhitungannya tidak melibatkan variabel yang menunjukan bahwa seorang *neighbor* berbeda dari *neighbor* lainnya. Algoritma sederhana ini dapat dikembangkan lagi menjadi sedikit lebih spesifik untuk membedakan setiap *neighbor*, caranya dengan memberi bobot pada masing-masing nilai $R_{u,j}$ *neighbor*.

$$P_{a,j} = \frac{\sum_{U=1}^{m} R_{u,j} \cdot W_u}{\sum W_u} \quad \text{.....(2)}$$

Pada Rumus (2), variabel *W* adalah bobot yang diberikan untuk pengguna U_m , nilai *W* bisa ditentukan secara manual oleh U_a atau pembuat sistem, bahkan dapat juga dicari menggunakan algoritma tertentu. Selain dengan memberi bobot untuk setiap *neighbor*, cara lain untuk mengembangkan rumus (2), yaitu dengan menambahkan variabel rata-rata *rating* dari U_a dan U_m. Lebih jelasnya lihat Rumus (3).

$$P_{a,j} = \bar{R}_a + \frac{\sum_{U=1}^{m} \left(R_{u,j} - \bar{R}_u\right)}{m} \quad \text{.....(3)}$$

Variabel $\bar{R}_a$ didapat dari rata-rata $R_{a,j}$ pada $L_{a,j}$ sedangkan $\bar{R}_u$ didapat dari rata-rata $R_{u,j}$ pada $L_{u,i}$ dengan asumsi rata-rata *rating* menunjukan "sifat" dasar seorang pengguna. Kata "sifat" menunjuk pada aktivitas lampau seorang pengguna, apakah dia seorang *hater* yang selalu memberi *rating* rendah, ataukah ia adalah seorang *lover* yang selalu memberi *rating* tinggi pada setiap *item* dalam

L yang dimilikinya. Semakin rendah nilai $\bar{R}$ pada seorang *neighboor* maka semakin besar kontribusi nilainya terhadap $P_{a,j}$ ketika suatu saat ia memberi nilai tinggi pada sebuah *item j*, tetapi jika ia memberi nilai rendah terhadap *item j* maka kontribusi nilainya kecil terhadap pencarian $P_{a,j}$. Sebaliknya, jika seorang *neighbor* memiliki nilai $\bar{R}$ yang tinggi, maka semakin kecil kontribusinya terhadap pencarian $P_{a,j}$ ketika ia memberi *rating* tinggi pada *item j*, tetapi jika ia memberi nilai rendah terhadap *item j* maka kontribusi nilainya sangat besar.

Algoritma pembobotan dan rata-rata *rating* pengguna juga dapat digabungkan supaya *P* yang didapat lebih spesifik. Caranya, pada Rumus (3) akan ditambahkan variabel bobot untuk setiap *neighbor*-nya, sekali lagi bobot dapat dicari dengan menggunakan berbagai cara tergantung dari asumsi pembuat sistem, lihat Rumus (4).

$$P_{a,j} = \bar{R}_a + \frac{\sum_{U=1}^{m} \left(R_{u,j} - \bar{R}_u\right) \cdot W_u}{\sum W_u} \quad \text{.....(4)}$$

Dengan menggabungkan pembobotan dan rata-rata *rating* pengguna, nilai $P_{a,j}$ yang didapat akan lebih bersifat pribadi karena melibatkan dua variabel yang berasal dari kegiatan lampau pengguna aktif, maupun *neighbor*.

Pada penelitian kali ini, peneliti menggunakan Rumus (4) untuk mencari prediksi, karena variabel yang digunakan sangatlah spesifik dan melibatkan lebih banyak variabel (rata-rata *rating* pengguna aktif, *rating* pengguna aktif terhadap *item* aktif, rata-rata *rating neighbor* hingga nilai bobot kedekatan antara *neighbor*

dengan pengguna aktif) yang ada dalam sistem rekomendasi. Peneliti juga memanfaatkan metode pengukuran *Pearson-Correlation Coefficient* untuk mengukur seberapa "dekat" pengguna aktif dengan pengguna lainnya. Semakin dekat pengguna tersebut dengan pengguna aktif, maka ia akan menjadi kandidat sumber rekomendasi (*neighbor*) dan nilai *rating*-nya akan mempengaruhi semua nilai prediksi yang akan dihitung nantinya.

2.2.2. Pearson-Correlation Coefficient

Korelasi adalah sebuah teknik pengukuran yang menentukan seberapa dekat relasi antar dua himpunan bilangan yang berbeda. Dengan syarat himpunan bilangan tersebut harus memiliki urutan yang tetap dan berpasangan satu dengan lainnya antar kedua himpunan. Hasil pengukuran dapat berupa relasi positif ataupun relasi negatif. Relasi positif menunjukkan bahwa kedua himpunan memiliki kecenderungan kenaikan atau penambahan nilai yang sejajar. Sedangkan relasi negatif menunjukkan kedua himpunan memiliki kecenderungan penurunan atau pengurangan nilai yang sejajar. Sejajar dalam konteks ini berarti penurunan atau kenaikan nilai yang saling mengikuti antar kedua variabel tersebut. Salah satu teknik pengukuran korelasi adalah *Pearson Product Moment Correlation* atau biasa disingkat menjadi *Pearson Correlation*.

Rumus yang digunakan pada teknik pengukuran ini bisa dilihat pada rumus (5) dibawah ini:

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2]\,[n\sum y^2 - (\sum y)^2]}} \quad \text{.....(5)}$$

Variabel *r* menunjukkan nilai korelasi yang didapat dari dua himpunan bilangan *x* dan *y* yang memiliki urutan dan berpasangan. Nilai *r* memiliki batas atas yaitu 1 dan batas bawah yaitu -1. Jika *r* bernilai 0 berarti kedua himpunan tidak memiliki relasi sama sekali. Jika *r* bernilai 1 berarti kedua himpunan memiliki penambahan nilai yang searah (sejajar), sedangkan jika *r* bernilai -1 maka kedua himpunan memiliki penurunan yang searah. Untuk *r* yang bernilai 1 atau -1 bisa disimpulkan bahwa kedua himpunan memiliki relasi atau kedekatan. Lihat pada Gambar 2.2 untuk contoh dua himpunan bilangan yang memiliki korelasi negatif dan positif.

Correlation = 0,91

	I1	I2
U1	5	4
U2	4	3
U3	2	1
U4	4	3
U5	1	1
U6	5	4
U7	5	5
U8	1	1
U9	3	2
U10	4	2

Correlation = -1

	I1	I2
U1	5	1
U2	5	1
U3	5	1
U4	5	1
U5	5	1
U6	1	5
U7	1	5
U8	1	5
U9	1	5
U10	1	5

Gambar 2.2 Contoh himpunan bilangan yang memiliki korelasi yang positif dan negatif

Variabel I1 dan I2 adalah dua himpunan bilangan yang berurutan dan berpasangan dari U1 – U10.

2.2.3. Evaluasi Sistem Rekomendasi

Dalam sistem rekomendasi, terutama yang memanfaatkan perhitungan prediksi, evaluasi keakuratan daftar rekomendasi dan prediksi sangatlah penting. Dengan memanfaatkan teknik evaluasi yang sudah tersedia, peneliti dapat mengetahui seberapa akurat teknik rekomendasi dan prediksi yang dikembangkan.

Pada bagian ini, akan dibahas tiga jenis evaluasi yaitu : *Mean Absolute Error* (MAE), *Precision*, dan *Recall Value*. MAE bertujuan menghitung seberapa besar rata-rata selisih nilai prediksi *rating* yang dihasilkan oleh peneliti dengan nilai *rating* yang diberikan oleh pengguna, sedangkan *Precision* dan *Recall* bertujuan untuk menghitung seberapa banyak persentase *item* yang direkomendasikan dan di beri *rating* oleh pengguna ataupun *item* yang sudah di *rating* tetapi tidak direkomendasikan oleh sistem.

Evaluasi MAE memanfaatkan teknik perhitungan yang sangat sederhana, yaitu dengan mencari selisih dari semua *item* yang sudah diberi *rating* oleh pengguna dan memiliki nilai *prediksi*. Nantinya selisih tersebut akan di absolute-kan (nilai positif) dan akhirnya dirata-rata. Dari hasil MAE, dapat terlihat jelas seberapa “jauh” selisih nilai prediksi *rating* yang diberikan oleh sistem dengan nilai *rating* yang diberikan oleh pengguna. Semakin besar nilai yang dihasilkan oleh MAE maka dapat diartikan bahwa nilai prediksi yang dihasilkan semakin tidak akurat, sebaliknya jika nilai MAE yang dihasilkan mendekati 0 maka prediksi yang dihasilkan sistem semakin mendekati akurat.

Rumus yang digunakan dalam evaluasi MAE dapat dilihat dibawah ini :

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|P_{a,i} - R_{a,i}\right| \quad \text{.....(6)}$$

Untuk n adalah jumlah semua *item* pada daftar rekomendasi pengguna aktif yang memiliki nilai prediksi. $P_{a,i}$ adalah nilai prediksi *item* ke-i milik pengguna aktif dan $R_{a,i}$ adalah nilai *rating* yang diberikan oleh pengguna aktif untuk *item* ke-i.

Selanjutnya, untuk mengevaluasi hasil rekomendasi yang dihasilkan oleh sistem melalui teknik peyaringan yang dibuat oleh peneliti, *Precision* dan *Recall Value* dimanfaatkan untuk melihat seberapa akurat rekomendasi *item* yang dihasilkan untuk tiap pengguna sistem.

Dalam memberikan rekomendasi, ada beberapa kemungkinan yang akan dihasilkan, antara lain : (1) *item* yang sudah diberi *rating* oleh pengguna, juga direkomendasikan oleh sistem (*True-Positive*); (2) *item* yang di rekomendasikan, tetapi tidak di *rating* oleh pengguna; (3) *item* yang sudah di beri *rating* tetapi tidak direkomendasikan. Beberapa kemungkinan tersebut bisa diringkas menjadi tabel dibawah ini.

Tabel 2.2 Klasifikasi Dari Beberapa Kemungkinan Hasil Rekomendasi

	Rekomendasi	Tidak Direkomendasikan
Sudah Diberi *Rating*	*True-Positive* (TP)	*False-Negative* (FN)
Tidak Diberi *Rating*	*False-Positive* (FP)	*True-Negative* (TN)

Hasil dari klasifikasi diatas dapat dimanfaatkan untuk mencari nilai *Precision* dan *Recall*. *Precision* dapat diasumsikan sebagai sebuah cara untuk melihat seberapa banyak *item* yang "tepat" direkomendasikan untuk pengguna sistem, sedangkan *Recall* dapat diasumsikan sebagai sebuah cara untuk mengetahui seberapa banyak *item* yang sudah di *rating* oleh pengguna dan direkomendasikan untuknya. Untuk itu, rumus untuk mencari nilai *Precision* dan *Recall* adalah :

$$Precision = \frac{\sum TP}{\sum TP + \sum FP} \quad \text{.....(7)}$$

$$Recall = \frac{\sum TP}{\sum TP + \sum FN} \quad \text{.....(8)}$$

Jika nilai *Precision* mendekati 1 maka berarti rekomendasi yang diberikan kepada pengguna banyak yang digunakan (di beri *rating*) oleh pengguna, sebaliknya jika semakin mendekati 0 maka rekomendasi yang diberikan banyak yang tidak tepat untuk pengguna sistem.