BAB II
TINJAUAN PUSTAKA

2.1 Sistem Rekomendasi

Sistem rekomendasi merupakan model aplikasi dari hasil observasi terhadap dan keinginan pengguna. Oleh sebab itu sistem rekomendasi memerlukan model rekomendasi yang tepat agar yang direkomendasikan sesuai dengan keinginan pengguna serta mempermudah pengguna mengambil keputusan yang tetap dalam menentukan produk yang akan digunakannya (Sharda, 2010).

Sistem rekomendasi merupakan sebuah alat atau *web* personalisasi yang menyediakan *user* sebuah informasi daftar *item-item* yang sesuai dengan keinginan masing- masing *user* (Onandia, & Sebastia, 2009). Sistem rekomendasi menyimpulkan preferensi *user* dengan menganalisis ketersediaan data *user*, informasi tentang *user* dan lingkungannya. Oleh karena itu, sistem rekomendasi akan menawarkan kemungkinan dan dari penyaringan informasi personal sehingga hanya informasi yang sesuai dengan kebutuhan dan preferensi *user* yang akan ditampilkan di sistem dengan menggunakan sebuah teknik atau model rekomendasi.

2.2 Data Clustering

Clustering atau klasterisasi adalah metode pengelompokan data (Tan, 2006). *Clustering* adalah sebuah proses untuk mengelompokan data ke dalam beberapa *cluster* atau kelompok sehingga data dalam satu *cluster* memiliki tingkat kemiripan yang maksimum dan data antar *cluster* memiliki kemiripan yang minimum.

Clustering merupakan proses partisi satu *set* objek data ke dalam himpunan bagian yang disebut dengan *cluster* (Tan, 2006). Objek yang di dalam *cluster* memiliki kemiripan karakteristik antar satu sama lainnya dan berbeda dengan *cluster* yang lain. Partisi tidak dilakukan secara manual melainkan dengan suatu algoritma *clustering*. Oleh karena itu, *clustering* sangat berguna dan bisa menemukan *group* atau kelompok yang tidak dikenal dalam

data. *Clustering* banyak digunakan dalam berbagai aplikasi seperti misalnya pada *business inteligence,* pengenalan pola citra*, web search,* bidang ilmu biologi, dan untuk keamanan (*security*). Di dalam *business inteligence*, *clustering* bisa mengatur banyak *customer* ke dalam banyaknya kelompok. Contohnya mengelompokan *customer* ke dalam beberapa *cluster* dengan kesamaan karakteristik yang kuat. *Clustering* juga dikenal sebagai data segmentasi karena *clustering* mempartisi banyak *dataset* ke dalam banyak *group* berdasarkan kesamaannya. Selain itu *clustering* juga bisa sebagai *outlier detection*. Manfaat dari *Clustering* ini sendiri, yaitu :

1. *Clustering* merupakan metode segmentasi data yang sangat berguna dalam prediksi dan analisa masalah bisnis tertentu. Misalnya Segmentasi pasar, marketing dan pemetaan zonasi wilayah.
2. Identifikasi obyek dalam bidang berbagai bidang seperti *computer vision* dan *image processing*.

2.2.1 Konsep dasar Clustering

Hasil *clustering* yang baik akan menghasilkan tingkat kesamaan yang tinggi dalam satu kelas dan tingkat kesamaan yang rendah antar kelas (Han, J. et al., 2012). Kesamaan yang dimaksud merupakan pengukuran secara *numeric* terhadap dua buah objek. Nilai kesamaan antar kedua objek akan semakin tinggi jika kedua objek yang dibandingkan memiliki kemiripan yang tinggi. Begitu juga dengan sebaliknya. Kualitas hasil *clustering* sangat bergantung pada metode yang dipakai. Dalam *clustering* dikenal empat tipe data, keempat tipe data pada tersebut ialah:

1. Variabel berskala interval
2. Variabel biner
3. Variabel nominal, ordinal, dan rasio
4. Variabel dengan tipe lainnya.

Metode *clustering* juga harus dapat mengukur kemampuannya sendiri dalam usaha untuk menemukan suatu pola tersembunyi pada data yang sedang diteliti. Terdapat berbagai metode yang dapat digunakan untuk mengukur nilai kesamaan antar objek-objek yang dibandingkan. Salah satunya ialah dengan *weighted*

Euclidean Distance. *Euclidean distance* menghitung jarak dua buah point dengan mengetahui nilai dari masing-masing atribut pada kedua poin tersebut. Berikut formula yang digunakan untuk menghitung jarak dengan *Euclidean distance*:

$$Distance\ (p,q) = \frac{(\sum_k^n \mu k\ |Pk - qk\ |\mathrm{r}) * 1}{r} \quad (2.1)$$

Keterangan:

N = Jumlah *record* data

K= Urutan *field* data

r= 2

μk= Bobot *field* yang diberikan *user*

Jarak adalah pendekatan yang umum dipakai untuk menentukan kesamaan atau ketidaksamaan dua vektor fitur yang dinyatakan dengan *Rank*ing. Apabila nilai *Ranking* yang dihasilkan semakin kecil nilainya maka semakin dekat/tinggi kesamaan antara kedua vektor tersebut. Teknik pengukuran jarak dengan metode *Euclidean* menjadi salah satu metode yang paling umum digunakan. Pengukuran jarak dengan metode *euclidean* dapat dituliskan dengan persamaan berikut:

$$J(V_1, V_2) = \sqrt{\sum_{k=1}^{n} (v_1(k) - v_2(k))^2} \quad (2.2)$$

dimana v1 dan v2 adalah dua vektor yang jaraknya akan dihitung dan N menyatakan panjang vektor.

2.2.2 Syarat *Clustering*

Syarat sekaligus tantangan yang harus dipenuhi oleh suatu algoritma *clustering* (Han, J. & Kamber, M., 2012) adalah:

1. Skalabilitas

   Suatu metode *clustering* harus mampu menangani data dalam jumlah yang besar. Saat ini data dalam jumlah besar sudah sangat umum digunakan dalam berbagai bidang misalnya saja suatu database. Tidak hanya berisi ratusan

objek, suatu database dengan ukuran besar bahkan berisi lebih dari jutaan objek.

2. Kemampuan analisa beragam bentuk data
3. Algortima klasteriasi harus mampu dimplementasikan pada berbagai macam bentuk data seperti data nominal, ordinal maupun gabungannya
4. Menemukan *cluster* dengan bentuk yang tidak terduga

   Banyak algoritma clustering yang menggunakan metode *Euclidean* atau *Manhattan* yang hasilnya berbentuk bulat. Padahal hasil *clustering* dapat berbentuk aneh dan tidak sama antara satu dengan yang lain. Karenanya dibutuhkan kemampuan untuk menganalisa *cluster* dengan bentuk apapun pada suatu algoritma *clustering*.
5. Kemampuan untuk dapat menangani *noise*

   Data tidak selalu dalam keadaan baik. Ada kalanya terdapat data yang rusak, tidak dimengerti atau hilang. Karena sistem inilah, suatu algortima *clustering* dituntut untuk mampu menangani data yang rusak.
6. Sensitifitas terhadap perubahan *input*

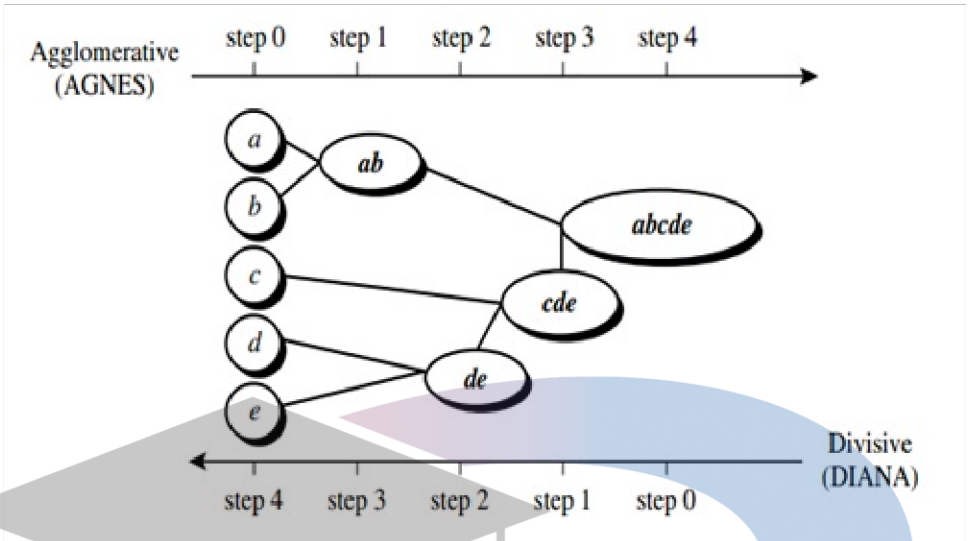
   Perubahan atau penambahan data pada *input* dapat menyebabkan terjadi perubahan pada *cluster* yang telah ada bahkan bisa menyebabkan perubahan yang mencolok apabila menggunakan algoritma *clustering* yang memiliki tingkat sensitifitas rendah.
7. Mampu melakukan *clustering* untuk data dimensi tinggi

   Suatu kelompok data dapat berisi banyak dimensi ataupun atribut. Untuk itu diperlukan algoritma *clustering* yang mampu menangani data dengan dimensi yang jumlahnya tidak sedikit.
8. Interpresasi dan kegunaan

   Hasil dari *clustering* harus dapat diinterpretasikan dan berguna.

2.2.3 *Hierarchical Clustering*

Pada *hierarchical clustering* data dikelompokkan melalui suatu bagan yang berupa hirarki, dimana terdapat penggabungan dua grup yang terdekat disetiap iterasinya ataupun pembagian dari seluruh *set* data kedalam *cluster*.



Gambar 2. 1 *Hierarchical Clustering*

Sumber: (Han dkk, 2012)

Langkah melakukan Hierarchical clustering:

1 Identifikasi *item* dengan jarak terdekat

2. Gabungkan *item* itu kedalam satu *cluster*

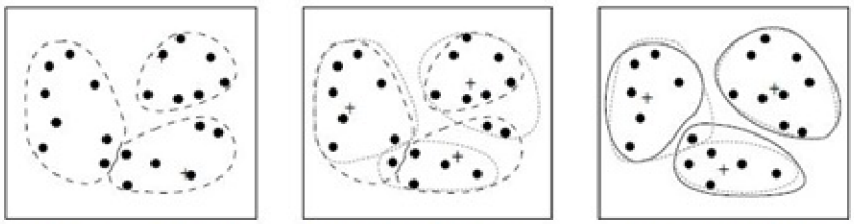
3. Hitung jarak antar *cluster*

4. Ulangi dari awal sampai semua terhubung

Contoh metode *hierarchy clustering*: *Single Linkage, Complete Linkage, Average Linkage, Average Group Linkage.*

2.2.4 *Partitional Clustering*

Partitional clustering yaitu data dikelompokkan ke dalam sejumlah *cluster* tanpa adanya struktur hirarki antara satu dengan yang lainnya. Pada metode *partitional clustering* setiap *cluster* memiliki titik pusat *cluster (centroid)* dan secara umum metode ini memiliki fungsi tujuan yaitu meminimumkan jarak (*dissimilarity*) dari seluruh data ke pusat *cluster* masing-masing. Contoh metode *partitional clustering: K-Means, Fuzzy K-means* dan *Mixture Modelling*.



Gambar 2. 2 Proses *Clustering* Objek Menggunakan metode *K-Means*

Metode *K-means* merupakan metode *clustering* yang paling sederhana dan umum. Hal ini dikarenakan *K-means* mempunyai kemampuan mengelompokkan data dalam jumlah yang cukup besar dengan waktu komputasi yang cepat dan efisien. *K-Means* merupakan salah satu algoritma klastering dengan metode partisi (*partitioning method*) yang berbasis titik pusat (*centroid*) selain algoritma *k-Medoids* yang berbasis objek. Algoritma ini pertama kali diusulkan oleh MacQueen (1967) dan dikembangkan oleh Hartigan dan Wong tahun 1975 dengan tujuan untuk dapat membagi M data *point* dalam N dimensi kedalam sejumlah k *cluster* dimana proses klastering dilakukan dengan meminimalkan jarak *sum squares* antara data dengan masing masing pusat *cluster (centroid-based).* Algoritma *k-Means* dalam penerapannya memerlukan tiga parameter yang seluruhnya ditentukan pengguna yaitu jumlah *cluster* k, inisialisasi klaster, dan jarak sistem, Biasanya, k-Means dijalankan secara independen dengan inisialisasi yang berbeda menghasilkan *cluster* akhir yang berbeda karena algoritma ini secara prinsip hanya mengelompokan data menuju *local minimal*. Salah satu cara untuk mengatasi *local minimal* adalah dengan mengimplementasikan algoritma *k-Means*, untuk K yang diberikan, dengan beberapa nilai initial partisi yang berbeda dan selanjutnya dipilih partisi dengan kesalahan kuadrat terkecil (Jain, 2009).

K-Means adalah teknik yang cukup sederhana dan cepat dalam proses clustering objek (*clustering*). Algoritma *K-means* mendefinisikan *centroid* atau pusat *cluster* dari *cluster* menjadi rata-rata point dari *cluster* tersebut.Dalam penerapan algoritma *k-Means*, jika diberikan sekumpulan data X = {x1, x2, …,xn} dimana xi = (xi1, xi2, …, xin) adalah sistem dalam ruang *real Rn*, maka algoritma

k-Means akan menyusun partisi X dalam sejumlah k *cluster* (a priori). Setiap *cluster* memiliki titik tengah (*centroid*) yang merupakan nilai rata rata (*mean*) dari data-data dalam *cluster* tersebut. Tahapan awal, algoritma *k-Means* adalah memilih secara acak k buah obyek sebagai *centroid* dalam data. Kemudian, jarak antara objek dan *centroid* dihitung menggunakan *Euclidian distance*. Algoritma *k-Means* secara *iterative* meningkatkan variasi nilai dalam dalam tiap tiap *cluster* dimana objek selanjutnya ditempatkan dalam kelompok yang terdekat, dihitung dari titik tengah klaster. Titik tengah baru ditentukan bila semua data telah ditempatkan dalam *cluster* terdekat. Proses penentuan titik tengah dan penempatan data dalam cluster diulangi sampai nilai titik tengah dari semua *cluster* yang terbentuk tidak berubah lagi (Han dkk, 2012).

Algoritma *k-means*:

Langkah 1: Tentukan berapa banyak *cluster* k dari *dataset* yang akan dibagi.

Langkah 2: Tetapkan secara acak data k menjadi pusat awal lokasi klaster.

Langkah 3: Untuk masing-masing data, temukan pusat *cluster* terdekat. Dengan demikian berarti masing-masing pusat *cluster* memiliki sebuah *subset* dari *dataset*, sehingga mewakili bagian dari *dataset*. Oleh karena itu, telah terbentuk *cluster* k: C1, C2, C3, …, Ck .

Langkah 4: Untuk masing-masing *cluster* k, temukan pusat luasan klaster, dan perbarui lokasi dari masing-masing pusat *cluster* ke nilai baru dari pusat luasan.

Langkah 5: Ulangi langkah ke-3 dan ke-5 hingga data-data pada tiap *cluster* menjadi terpusat atau selesai.

2.3 *Collaborative Pairwise Learning* to *Rank*

Collaborative Pairwise Learning to *Rank*(CPLR) merupakan sebuah Algoritma yang digunakan untuk mengelompokkan data (*clustering*) dengan cara menjumlahkan *rating* atau *feedback* yang diberikan pengguna pada suatu *item* dan menghasilkan suatu rekomendasi baru berdasarkan perbandingan antar pola pengguna, biasanya nilai dari *rating* dapat berupa *binary* "suka/tidak suka" (Liu, H. et al., 2018). Metode CPLR merekomendasikan suatu *item* dengan cara membagi *set item* yang sudah ditetapkan menjadi 3 *subset*: *set* positif, yaitu *item* yang telah diberikan *feedback* oleh *user*. kolaborative *set*, yaitu *item* yang telah diberikan

feedback oleh *user* lain (*neighbor*) tetapi tidak dengan *user* itu sendiri. *set negative*, yaitu *item* yang tidak diberikan *feedback* oleh *user* maupun *user* lain. Adapun tahapan untuk proses perhitungan dari metode tersebut:

1. Definisi Masalah

   Dengan variabel u dan i yang menunjukan pengguna (u) dan kriteria(i) kita dapat mendefinisikan pengguna dengan implisit matriks umpan balik R sebagai berikut :

$$Rui = \begin{cases} 1, & \frac{Jika\ (u,i) Interaksi\ telah\ diamati}{Jika\ Tidak} \\ 0, & \end{cases} \quad (2.3)$$

   Pengguna(u) akan mengatur kriteria (i) yang ditetapkan $P_u \subseteq_i$ Menandakan kriteria untuk pengguna (u) yang telah memberikan umpan balik positif yaitu $Pu = \{ i \mid Rui = 1 \}$ diprediksi preferensi pengguna u pada kriteria i dilambangkan sebagai R_{ui} berguna untuk menunjukan urutan preferensi relatif, misalnya relatif agar preferensi *(u,i)(u,j)* berarti bahwa pengguna u lebih suka pada i kriteria j.

   Hal ini bertujuan untuk memberikan setiap pengguna u sebuah personalitasi daftar peringkat kriteria dari I\P$u$ hanya menggunakan data umpan balik implisit yaitu, asumsi yang hanya mengetahui positif umpan balik Pu untuk setiap pengguna(u) memprediksi benar-benar suka/tidak suka yang mencoba untuk memprediksi preferensi relatif dari setiap pengguna pada kriteria.

2. *Bayesian Personalized Ranking (BPR)*

   Tujuan utama dari BPR ini adalah untuk mengambil umpan balik implisit sebagai preferensi relatif yang menyatakan apakah pengguna benar-benar suka atau tidak. Ini mengasumsikan pengguna kepada kriteria j, dari pada kriteria i. Kemungkinan preferensi relatif antar semua kriteria didapatkan dalam contoh berikut :

$$BPR(u) = \prod_{i \in P_u} \prod_{j \in I \setminus P_u} P(r_{ui} > r_{uj})[1 - P(r_{uj} > r_{ui})]. \quad (2.4)$$

   P(·) adalah fungsi probalitas yang relatif didekati dengan fungsi *sigmoid* logistik σ (x) = 1/(1 + e−x). (2.5)

Berdasarkan asumsi bahwa semua pengguna independen satu sama lain. Ada kemungkinan gabungan dari preferensi relatif dari dua pengguna yang berbeda misalkan u dan w dapat disederhanakan BPR(u, w) = BPR(u)BPR(w). Dengan demikian untuk semua pengguna dapat dihitung sebagai berikut.

$$BPR = \prod_{u \in U} BPR(u). \tag{2.6}$$

Tujuan dari algoritma BPR ini adalah untuk pencarian yang optimal dari parameter model untuk memaksimalkan probabilitas. Fungsi dari BPR dapat didefinisikan sebagai berikut,

$$\begin{aligned} BPR - OPT &:= \ln BPR - \lambda_\Theta \parallel \Theta \parallel^2 \\ &= \sum_{u \in U} \sum_{i \in P_u} \sum_{j \in I \setminus P_u} \ln \sigma(r_{ui} - r_{uj}) - \lambda_\Theta \parallel \Theta \parallel^2. \end{aligned} \tag{2.7}$$

3. *Collaborative filtering*

Collaborative filtering merupakan teknik paling populer yang digunakan pada sistem rekomendasi. *Collaborative filtering* berfungsi untuk memprediksi dan merekomendasi berdasarkan peringkat atau perilaku pengguna didalam sistem. Implementasi dari pendekatan ini disebut *user*-based *Collaborative filtering* (*User*-CF), yakni membuat rekomendasi kepada pengguna yang aktif berdasarkan *item* atau kriteria yang disukai oleh pengguna lain, metode ini disebut juga metode berbasis lingkukan atau *neighborhood-based method* yang akan membuat prediksi atau rekomendasi berdasarkan informasi lingkungan sistem. Metode berbasis lingkungan ini juga disebut sebagai algoritma *memory based*, karena *rating* matriks asli diadakan di memori dan digunakan secara langsung untuk menghasilkan rekomendasi.

4. *Learning to Rank*

Pada umumnya, teknik pembelajaran mesin atau *machine Learning* berfungsi untuk memberikan nilai atau *rating* pada peringkat dari sebuah *item* atau kriteria. Pembelajaran tersebut dapat dikelompokan menjadi tiga kategori yakni secara berurutan/*pointwise*, berpasangan/*Pairwise*, dan sesuai urutan/*listwise*. Metode *pointwise* ini akan mengurangi peringkat regresi atau kualifikasi urutan. Sedangkan metode *listwise* berfungsi untuk meminimalkan kesalahan pada urutan.

5. *Personalized recommendation from implicit feedback*

Luasnya penelitian tentang *personalized recommendation* telah difokuskan pada umpan balik eksplisit. Teknik umpan balik eksplisit dapat dibandingkan dengan umpan balik implisit yang memiliki keuntungan dari pengalaman pengguna yang lebih baik dan aplikasi yang lebih luas.

Pada data umpan balik implisit, semua contoh negatif dan positif dicampur secara bersamaan. Untuk mengatasi masalah ini, diusulkan untuk memberikan bobot yang berbeda dengan kesalahan pada contoh negatif dan positif yang akan dimasukkan kedalam matriks kerangka faktorisasi. Metode ini mengasumsikan semua umpan balik tidak mengamati contoh negatif dan memerlukan bobot relatif positif.

6. *Collaborative Pairwise Learning* to *Rank*

Seperti yang disebutkan diatas, BPR adalah pembelajaran secara berpasangan untuk mendapatkan peringkat yang memperlakukan pengguna sebagai preferensi relatif yang benar-benar suka atau tidak. BPR berkinerja lebih baik untuk merekomendasikan umpan balik implisit dibandingkan metode yang ada, yakni (1) memperlakukan semua umpan balik yang tidak teramati sama dengan umpan balik negatif misalkan tidak suka, (2) memperlakukan semua umpan balik yang diamati sebagai hal yang sama, dan (3) mengabaikan pengaruh antar pengguna. Namun, umpan balik *item*/kriteria yang diamati dapat disebabkan karena tidak terlihat dan bukannya tidak suka. Untuk mengatasi masalah ini diusulkan untuk menggunakan algoritma *Collaborative Pairwise Learning* to *Rank* (CPLR), yang mempertimbangkan pengaruh antar pengguna pada preferensi untuk *item*/kriteria dengan mengamati umpan balik.

7. Objective function

Sama seperti *Collaborative Filtering*, yang mengasumsikan bahwa pengguna yang disepakati dikemarin hari, cenderung akan menyetujui lagi dihari yang akan datang. Seorang pengguna akan cenderung memilih *item* yang mana telah mendapatkan umpan balik positif dari tetangga dengan preferensi yang serupa. Untuk umpan balik dari tetangga pengguna, kita akan membagi

item/kriteria yang tidak teramati menjadi dua *subset*. Untuk setiap pengguna, seluruh *item*/kriteria akan dibagi menjadi tiga himpunan yakni P_u , C_u, dan L_u. $P_u$menunjukan *set* positif, C_u menunjukkan *set* kolaboratif, dan L_u menunjukkan *set* yang ditinggalkan. Ketiga himbunan tersebut dapat dirumuskan sebagai berikut :

$$\begin{cases} P_u = \{i \mid R_{ui} = 1\}, & \text{Positive set} \\ C_u = \{t \mid \exists w \in N_u,\ t \in P_w \,\&\, t \notin P_u\}, & \text{Collaborative set} \\ L_u = I - P_u - C_u, & \text{Left-out set} \end{cases} \tag{2.8}$$

Dimana N_u menunjukkan tetangga dari pengguna.

Dapat diasusmsikan bahwa pengguna lebih menyukai *item* yang telah diberikan umpan balik positif untuk barang-barang lainnya dan pengguna lebih suka barang yang tetangga-tetangganya telah memberikan umpan balik positif, kita bisa mendapatkannya dengan preferensi sebagai berikut:

(u, i) > (u,t), (u, i) > (u, j), (u,t) > (u, j) *(2.9)*

where $i \in P_u$, $t \in C_u$ and $j \in L_u$

Berbeda dari BPR, yang mencoba untuk mengoptimalkan AUC standar yang dirancang untuk klasifikasi biner, dan kami akan mencoba mengoptimalkan generalisasi AUC

$$GAUC = \frac{1}{\sum_{ck<cl} nknl} \sum_{yi<yj} \delta(f(xi) < f(xj)) \tag{2.10}$$

Dimana C_k dan Y_i adalah label kelas ordinal, N_k adalah jumlah dari C_k, f(xi) adalah nilai prediksi dan δ() adalah fungsi indikator biner. Pengukuran ini mirip dengan *C-index* (*cordodance index*) sebagai ukuran kebaikan yang digunakan dalam statistik. *C-index* adalah *estimator* dari *probabilitas cordodance* dari model regresi ordinal dengan menghitung jumlah (kelas bawah, kelas lebih tinggi) objek pasangan itu diberi peringkat dengan benar oleh model. Sama seperti AUC standar juga menggunakan kerugian yang tidak terdiferensiasi δ (x> 0) yang identik dengan fungsi *Heaviside* dibawah ini:

$$\delta(x > 0) = H(x) = \begin{cases} 1, x > 0 \\ 0, otherwise. \end{cases} \tag{2.11}$$

Didalam tulisan ini, kami menggunakan pengganti fungsi kerugian ln σ (x) untuk menggantikan fungsi step loss δ (x > 0) seperti pada BPR. Oleh karena itu, kami mendapatkan perkiraan GAUC didalam persamaan berikut ini:

$$\frac{1}{N} = \sum_{u \in U}\left[\sum_{i \in Pu}\sum_{t \in Cu} \ln(\sigma r_{ui} - r_{ut})) + \sum_{t \in Cu}\sum_{t \in Lu} \ln(\sigma(r_{ut} - r_{uj})) + \sum_{i \epsilon Pu}\sum_{j \epsilon Lu} \ln(\sigma(r_{ui} - r_{uj}))\right] \quad (2.12)$$

Dimana $N = \sum_{u \in U}(|p_u|.|c_u| + |c_u|.|l_u| + |p_u|.|l_u)$

Mengasumsikan semua umpan balik yang telah diamati dari beberapa data. Kemudian mencoba untuk menyesuaikan data yang berpasangan dalam persamaan preferensi menggunakan informasi lingkungan pengguna / *user neighborhood information*

$$\text{CPLR} - \text{OPT} = \sum_{u \in u}\left[\propto \sum_{i \in p_u}\sum_{t \in C_u} \ln(\sigma(c_{uit}(r_{ui} - r_{ut}))) + \beta \sum_{t \in Cu}\sum_{j \in lu} \ln(\sigma(c_{utj}(c_{ut} - r_{uj}))) + \gamma \sum_{i \epsilon p_u}\sum_{j \epsilon l_u} ln(\sigma\left(c_{uij}(r_{ui} - r_{uj})\right))\right] - \frac{1}{2}\lambda ||||^2 \quad (2.13)$$

Dimana C_{uit}, C_{utj}, dan C_{uij} adalah koefisien, α, β and γ adalah koefisien kontrol, dan $\lambda\theta$ adalah koefisien regulasi. Koefisien C_{uit}, C_{utj}, dan C_{uij} digunakan untuk menyesuaikan koefesien preferensi secara berpasangan. Istilah regulasi λ 2 digunakan untuk menghindari *overfitting* dalam proses pembelajaran mesin. Koefisien, α, β and γ digunakan untuk mengontrol jenis preferensi relatif yang akan digunakan bobotnya dimodel. Model ini bertujuan secara langsung untuk mengoptimalkan pendekatan dari GAUC. Ketika α ditetapkan sebagai pada nilai 0, dapat diasumsikan *set item* positif dan *set item* kolaboratif tidak bisa ditandingi, ketika β diset menjadi 0 diasumsikan *item* positif dengan umpan balik juga tidak bisa ditandingi.

8. *Model Learning*

Pada *model Learning* ini akan menggunakan *Stochastic Gradient Descent* yang berfungsi untuk mengoptimalkan fungsi pada objektif dalam persamaan, untuk menyederhanakannya dapat didefinisi

$r_{uij} = c_{uij}(r_{ui} - r_{uj})$. gradien parameter dalam fungsi dapat dihitung sebagai berikut :

$$\partial \frac{CPLR-OPT}{\partial} = \sum_{u\epsilon u}\left[\propto \sum_{i\epsilon Pu}\sum_{t\epsilon Cu}\frac{\partial}{\partial}\ln\sigma\left(r_{uit}\right) + \beta \sum_{t\epsilon Cu}\sum_{j\epsilon Cu}\frac{\partial}{\partial}\ln\sigma\left(r_{utj}\right) + \gamma \sum_{i\epsilon pu}\sum_{j\epsilon lu}\frac{\partial}{\partial}\ln\sigma\left(r_{utj}\right)\right] - \frac{1}{2}\lambda\frac{\partial}{\partial}||||| =$$
$$\sum_{u\epsilon U}\left[\alpha \sum_{i\epsilon Pu}\sum_{t\epsilon Lu}\frac{e^{-rTuit}}{1+e^{Tuit}}\frac{\partial}{\partial}r_{ui} + \beta \sum_{t\epsilon Cu}\sum_{j\epsilon lu}\frac{e^{-rTuit}}{1+e^{Tuit}}\frac{\partial}{\partial}r_{ui} + \gamma \sum_{i\epsilon pu}\sum_{j\epsilon lu}\frac{e^{-rTuit}}{1+e^{Tuit}}\frac{\partial}{\partial}r_{ui}\right] - \lambda \quad (2.14)$$

Fungsi preferensi dapat dimodelkan dengan matriks faktorisasi

$$r_{ui=\mathrm{Wu}}V_i^T + \mathrm{bi} = \sum_{f=1}^{d} W_{uf}\ V_{if} + b_i \quad (2.15)$$

Dimana u ∈U, U, i ∈ I, b ∈ R|I| , W ∈ R|U| × d, V ∈ Rd × |I| , dan d adalah angka untuk faktor laten. Parameter pengguna khusus W_{uf} dapat dihitung sebagai berikut:

$$\frac{\partial \mathrm{CPLR-OPT}}{\partial \mathrm{W_{uf}}} = \frac{\propto .C_{uit}}{1+e^{ruit}}\left(V_{if} - V_{jf}\right) + \frac{\boldsymbol{\beta}.C_{uit}}{1+e^{ruit}}\left(V_{if} - V_{jf}\right) + \frac{\boldsymbol{\gamma}.C_{uit}}{1+e^{ruit}}\left(V_{if} - V_{jf}\right) - \lambda w W_{uf} \quad (2.16)$$

Demikian pula, kita bisa mendapatkan gradien parameter spesifik *item*/kriteria

$$\frac{\partial \mathrm{CPLR-OPT}}{\partial\, \partial_{\mathrm{if}}} = \frac{\propto .C_{uit}}{1+e^{ruit}}W_{uf} + \frac{\gamma.C_{uit}}{1+e^{ruit}}W_{uf} - \lambda v V_{if} \quad (2.17)$$

$$\frac{\partial CPLR-OPT}{\partial Vtf} = \frac{\propto .C_{uit}}{1+e^{ruit}}\left(-W_{uf}\right) + \frac{\boldsymbol{\beta}.C_{uit}}{1+e^{ruit}}W_{uf} - \lambda v V_{tf} \quad (2.18)$$

$$\frac{\partial \mathrm{CPLR-OPT}}{\partial\, \partial_{\mathrm{jf}}} = \frac{\boldsymbol{\beta}.C_{uitj}}{1+e^{ruitj}}\left(-W_{uf}\right) + \frac{\gamma.C_{uij}}{1+e^{ruij}}\left(-W_{uf}\right) - \lambda v V_{jf} \quad (2.19)$$

$$\frac{\partial \mathrm{CPLR-OPT}}{\partial bt} = \frac{\propto .C_{uit}}{1+e^{ruit}} + \frac{\gamma.C_{uij}}{1+e^{ruij}} - \lambda v V_{if} \quad (2.20)$$

$$\frac{\partial \mathrm{CPLR-OPT}}{\partial bt} = \frac{\propto .C_{uit}}{1+e^{ruit}}(-1) + \frac{\beta.C_{uij}}{1+e^{ruij}} - \lambda_b b_t \quad (2.21)$$

$$\frac{\partial \mathrm{CPLR-OPT}}{\partial bj} = \frac{\boldsymbol{\beta}.C_{utj}}{1+e^{rutj}}(-1) + \frac{\gamma.C_{uij}}{1+e^{ruij}}(-1) - \lambda_b b_j \quad (2.22)$$

Dimana i ∈Pu,t∈ Cu dan j ∈ Lu.

9. *The CPLR algorithm*

Pseudocode dari algoritma CPLR ditunjukan dalam algoritma berikut, yang terdiri dari langkah inisialisasi dan proses *Training* yang berulang.

Input **:** *Feedback* positif yang diamati F = {(u,i)}, ukuran dari *user neighborhood* = k, waktu pengambilan sample = T, jumlah *latentfactors* = d.

Output : jumlah parameter model $= \{W_{\in} R^{|u| x d}, V_{\in} R^{d x |I|}, b_{\in} R^{|I|}\}$

Inisialisasi : Mengacak parameter inisialisasi W, V dan b.

for each u ϵ U **do**

Memilih *k nearest neighboors* dari u untuk membangun *neighborhood Nu*.

Memecah kumpulan *item* I kedalam tiga bagian P, C, L.

end for

Training **:**

for s = 0; s < T; s++ **do**

Memilih secara acak seorang pengguna u ϵ U.

Memilih secara acak sebuah *item* i ϵ P.

Memilih secara acak sebuah *item* t ϵ C.

Memilih secara acak sebuah *item* j ϵ L.

Menghitung Gradients dari hubungan parameter menurut .

Memperbaharui parameter Wu, V_i, V_t, V_j, b_i, b_t dan b_j.

end for

Langkah inisialisasi melibatkan pengaturan parameter model secara acak dan membagi item set I menjadi tiga bagian P_u, C_u dan L_u untuk setiap pengguna sesuai dengan perilaku pengguna u dan tetangganya. Tetangga k-pengguna terdekat dapat dipilih sesuai kriteria yang berbeda. makalah ini, kami mengadopsi cosine similarity yang umum digunakan antara dua pengguna *u* dan *w*.

$$sim(u,w) = \frac{(|Pu| \cap |Pw|)}{(|Pu| * |Pw|)} \qquad (2.23)$$

Di setiap iterasi, kami secara acak mengambil sampel pengguna u dan tiga item set {i.t.j}. Di mana i∈P_u, t∈C_u, j∈L_u. Kemudian kita menghitung gradien parameter dalam fungsi objektif sesuai dengan persamaan (2.16 sampai 2.22). Koefisien kepercayaan Cuit, Cutj, dan Cuij didefinisikan sebagai berikut :

$$c_{uit} = \frac{1+s_{ui}}{1+s_{ut}},\ c_{utj} = 1 + s_{ut}\ ,\quad c_{uij} = 1 + s_{ui} \tag{2.24}$$

Koefisien s_{ui} untuk item $i \in P_u \cup C_u$ didefinisikan sebagai berikut:

$$s_{ui} = \sum_{w \in Nu} sim(u,w) * \partial(i \in Pw) \tag{2.25}$$

2.4 *MAE (Mean Absolute Error)*

Mean absolute error (MAE) adalah ukuran perbedaan antara dua variabel kontinyu Asumsikan X dan Y adalah variabel dan penghematan pasangan yang mengekpresikan fenomena yang sama. Contoh Y dan X mencakup perbandingan prediksi versus Penghematan, waktu berikutnya dan waktu awal, satu teknik pengukuran dan teknik pengukuran alternatif

Pertimbangan sebidang titik n , dimana titik saya memiliki koordinat (xi, yi). *Mean Absolute Error (MAE)* adalah jarak vertikal rata-rata antara masing-masing titik dan garis Y= X, yang juga dikenal sebagai jalur *One-to-One*.

MAE juga merupakan jarak horizontal rata-rata antara masing-masing titik dan garis Y= X. Persamaan MAE digunakan untuk mengevaluasi kualitas dari sistem dan perhitungan yang paling sering digunakan. Semakin kecil nilai *MAE,*maka nilai prediksi yang dihasilkan semakin baik(Julia, 2017).

$$MAE = \frac{\sum_{i=1}^{n} |fi - hi|}{n} \tag{2.26}$$

Keterangan :

fi = Nilai titik lokasi sebenarnya.

hi = Nilai prediksi titik lokasi.

n = Banyak rute yang akan diuji tingkat kesalahan antara nilai titik lokasi sebenarnya dan Nilai prediksi titik lokasi.

2.5 Anime

Anime adalah sebuah animasi khas jepang. Kata *anime* itu sendiri berasal dari kata jepang yaitu "*Anime*shon" yang diterjemahkan ke bahasa inggris yaitu "*Anime*". *Anime* ini mencakup semua judul animasi termasuk fitur film dan video original animasi (Brenner, R. E., 2007). *Anime* diproduksi pertama kali pada tahun 1917, namun masih sebatas film animasi pendek berdurasi dua hingga lima menit yang sebagian besar bercerita tentang *folk tales* masa itu. Karakter utama dalam *anime* biasanya robot, *monster*, *superhero*. Kriteria *anime* seperti *genre* secara umum ada lima, aksi/petualangan berfokus pada pertempuran, perang dan persaingan fisik (yang mencakup seni bela diri, pertempuran senjata atau materi berorientasi aksi lainnya). Drama biasanya menampilkan pengembangan karakter dan tema emosional yang tinggi serta melibatkan komplikasi hubungan. *Horor* menggunakan tema gelap dan supranatural. Fiksi ilmiah bergantung pada unsur-unsur futuristik, khususnya *sains* dan teknologi masa depan. Dan *anime* progresif cenderung lebih *stylish*/bergaya (Aeschliman, L. et al., 2007).

Anime tidak hanya terkenal dijepang tetapi di berbagai negara, dan tidak hanya *anime* yang asli tetapi yang *dubbed* atau terjemahan (Gravett, 2004). *Anime* saat ini telah menjadi komoditas internasional, semakin menarik perhatian banyak akademis maupun praktisi dari berbagai bidang maupun negara. *Anime* berperan penting dalam pembentukan media *scope global*, baik cetak maupun elektronik (MacWilliams, 2011).