SISTEM REKOMENDASI PADA BUKU DENGAN MENGGUNAKAN TAGS AND LATENT FACTORS

BOOK RECOMMENDATION SYSTEM USING TAGS AND LATENT FACTORS

Shendy Krishnamurty[#1], Dade Nurjanah[#2], Rita Rismala[#3]

#School of Computing, Telkom University

Jl. Telekomunikasi No.01, Terusan Buah Batu, Bandung, Jawa Barat, Indonesia

[1]krishnashendy@gmail.com

[2]dadenurjanah@gmail.com

[3]rismala.rita@gmail.com

Abstrak

Sistem Rekomendasi atau *Recommender System*, bertujuan untuk membantu *user* dengan cara memberikan rekomendasi kepada *user* ketika dihadapkan dengan jumlah informasi yang besar. Rekomendasi yang diberikan diharapkan dapat membantu *user* dalam proses pengambilan keputusan, seperti buku apa yang akan dibaca. Dengan adanya rekomendasi yang tepat, *user* akan memiliki preferensi tambahan ketika mencari buku. Dalam pengerjaan Tugas Akhir ini, penulis menggunakan *Tags and Latent Factors* yang merupakan algoritma tambahan pada metode *Matrix Factorization* yang membantu sistem rekomendasi dalam meningkatkan akurasi prediksi sehingga mampu menebak minat *user* dan mampu memberikan rekomendasi. *Tags* berfungsi sebagai jembatan yang memungkinkan *user* untuk lebih memahami hubungan yang tidak diketahui antara *item* dan *user* itu sendiri, sedangkan *latent factors* bertujuan untuk membentuk kesamaan antara *user* dan *item* dimana kesamaan ini berupa antar *user* yang tertarik atau menyukai *item* yang sama. Pada Tugas akhir ini akan dilakukan skenario pengujian dimana nilai dari hasil pengujian akan menjadi parameter akurat atau tidaknya sebuah sistem dengan menggunakan *Mean Absolute Error* (MAE) . Diberikan 3 buah skenario pengujian dengan menggunakan 3 jenis dan jumlah data berbeda dan didapatkan 3 nilai hasil pengujian akhir dengan nilai masing-masing 0.41, 0.38, dan 0.38 dimana 3 nilai tersebut memiliki selisih yang tidak terlalu besar. Dan dapat disimpulkan bahwa *Tags and Latent Factors* dapat digunakan untuk memberikan rekomendasi pada buku.

Kata kunci: *recommender system, matrix factorization, tags and tatent factors*

1. Pendahuluan

Sistem rekomendasi merupakan teknik yang menyediakan saran terkait suatu hal yang dapat dimanfaatkan oleh *user,* salah satunya pada sistem rekomendasi buku. Tanpa adanya sistem rekomendasi, *user* hanya berfokus pada satu hasil pencarian saja sementara mereka masih bisa mendapatkan rekomendasi yang lebih baik tentang buku lain yang bisa saja akan menarik minat mereka. Sistem rekomendasi memberikan manfaat kepada *user* terkait dalam masalah pencarian buku. Adanya sistem rekomendasi akan membantu merekomendasikan buku-buku untuk pembeli yang sesuai dengan minat mereka [1].

Dalam pengimplementasiannya, sistem rekomendasi mempunyai beberapa metode yang sering digunakan dan salah satunya adalah *Matrix Factorization* (MF). Metode ini memiliki kemampuan untuk memprediksi *rating* untuk menghasilkan rekomendasi kepada *user*. Metode ini dikembangkan pada kasus [2] dengan menambahkan *tags and latent factors* untuk meningkatkan akurasi dalam memprediksi *item* sehingga dapat memberikan rekomendasi yang sesuai dengan minat *user*. *Tagging* atau memberi label pada data menawarkan *user* sebuah cara alternatif untuk memberi rekomendasi dan memprediksi minat *user*. Karena *tag*s mudah dipahami oleh *user*, *tags* berfungsi sebagai jembatan yang memungkinkan *user* untuk lebih memahami hubungan yang tidak diketahui antara *item* dan *user* itu sendiri [3]. Berdasarkan percobaan pada [2], diduga dengan adanya *tags* yang diberikan kepada *item* berkolerasi dengan *rating*, sehingga dapat digunakan untuk memprediksi minat *user* dengan lebih baik. Sedangkan *latent factors* merupakan bagian dari *matrix factorization*, dimana *user* dan *item* dimodelkan oleh

factor models yang memodelkan *user* dan *item* yang bertujuan untuk membentuk kesamaan antara *user* dan *item*. Kesamaan ini berupa antar *user* yang tertarik atau menyukai *item* yang sama. Kesamaan dan hubungan antara *user* dan *item* akan dieksploitasi untuk memprediksi nilai *rating* sehingga mampu memberikan rekomendasi [2].

2. Tinjauan Pustaka

2.1 Sistem Rekomendasi

Sistem rekomendasi didefinisikan sebagai program yang mencoba untuk merekomendasikan *item* yang paling cocok (produk atau jasa) untuk *user* tertentu (individu atau bisnis) dengan memprediksi minat *user* pada *item* berdasarkan informasi terkait tentang *item*, *user* dan interaksi antara *item* dan *user* [4]. Sistem rekomendasi pada buku juga mempertimbangkan banyak parameter seperti *genre* pada buku, kualitas buku dengan cara memberikan *rating* yang diberi oleh *user* lain. Tujuan dari sistem rekomendasi buku ini adalah untuk merekomendasikan buku-buku untuk *user* yang sesuai dengan minat mereka [1].

Sistem rekomendasi dibangun dengan menggunakan teknik-teknik yang sudah ada, antara lain *Content-based Filtering* dan *Collaborative Filtering.* Kedua teknik tersebut merupakan teknik yang paling lama muncul dan banyak digunakan. Pada sub-bab berikut akan dideskripsikan pengertian dari kedua teknik tersebut dan telah diimplementasikan pada kasus apa saja termasuk sistem rekomendasi pada buku.

2.1.1 Content-based Filtering

Teknik rekomendasi *Content-based Filtering* (CB) merekomendasikan artikel atau komoditas yang mirip dengan *item* yang sebelumnya disukai oleh *user* tertentu. Prinsip-prinsip dasar dari sistem rekomendasi CB adalah: 1) Untuk menganalisis deskripsi *item* yang disukai oleh *user* tertentu untuk menentukan atribut umum pokok (preferensi) yang dapat digunakan untuk membedakan barang-barang tersebut. Preferensi ini disimpan dalam profil *user*. 2) Untuk membandingkan atribut masing-masing *item* dengan profil *user* sehingga hanya *item* yang memiliki tingkat kesamaan yang tinggi dengan profil *user* yang akan direkomendasikan [4].

CB telah diimplementasikan pada banyak kasus, berikut contoh kasus serta bidang penerapannya:

Tabel 1 Contoh Penerapan CB

No.	Judul	Tahun	Metode
1.	*Recommending Twitter Users to Follow Using Content and Collaborative Filtering Approaches* [5]	2010	*Content-Based Filtering*
2.	*Twitter Content-Based Spam Filtering* [6]	2014	*Content-Based Spam Filtering*
3.	*Content-Based Book Recommending Using Learning for Text Categorization* [7]	1999	*Content-Based using Learning*
4.	*Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining* [1]	2014	*Collaborative Filtering*

Kelebihan dan kekurangan yang ada pada CB:

- Kelebihan: Pendekatan CB tidak memerlukan data dari *user* lain dan memiliki kapabilitas untuk merekomendasikan *item* kepada *user* dengan selera yang unik.
- Kekurangan: Dalam CB *item* terbatas pada deskripsi awal atau fitur *item* [8].

2.1.2 Collaborative Filtering

Collaborative Filtering (CF) sistem bekerja dengan mengumpulkan *feedback user* dalam bentuk *rating* untuk *item* dalam domain tertentu dan memanfaatkan kesamaan dalam perilaku memberi *rating* di antara beberapa *user* dalam menentukan bagaimana untuk merekomendasikan *item* [9].

CF telah diimplementasikan pada banyak kasus, berikut contoh kasus serta bidang penerapannya:

Tabel 2 Contoh Penerapan CF

No.	Judul	Tahun	Metode
1.	*Amazon.com Recommendations: Item-to-Item Collaborative Filtering* [10]	2003	*Item-to-item Collaborative Filtering*

2.	*Explaining Collaborative Filtering Recommendations* [11]	2000	*Collaborative Filtering*
3.	*Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining* [1]	2014	*Collaborative Filtering*

Kelebihan dan kekurangan yang ada pada CF:

- Kelebihan: Pendekatan CF tidak membutuhkan representasi dari *item* dalam hal fitur tetapi hanya didasarkan pada penilaian dari komunitas *user* yang berpartisipasi.
- Kekurangan: *Item* tidak dapat direkomendasikan untuk setiap pengguna sampai dan kecuali *item* telah di *rating* oleh *user* lain atau berkorelasi dengan *item* sejenis lainnya [8].

2.2 Matrix Factorization

Matrix Factorization (MF) merupakan salah satu teknik yang populer untuk memprediksi *rating*. Pada dasarnya, dalam MF diberikan sebuah matriks dari peringkat *rating* (contoh: antara nilai 1 dan 5) dari *user* untuk *item* $\{r_{ui}\}_{m \times n}$ (m untuk *user* dan *n* untuk *item*) diuraikan menjadi matriks *two lower dimensional*, sehingga prediksi *rating* dari semua entitas yang tidak dikenal pada matriks original $\{r_{ui}\}$ dapat dihitung. Dalam MF, *user* dan *item* dimodelkan dengan vektor dari *abstract factor* yang dipelajari oleh *mining rating* yang tersedia. Karena alasan itu algoritma MF juga dapat disebut dengan *factor models*. Dalam *factor models*, sangat mungkin untuk membangun kesamaan dari *user* dan *item* karena mereka berbagi representasi yang umum. Persamaan dan hubungan antara *user* dan *item* dimanfaatkan untuk memprediksi *rating* yang hilang dan menghasilkan rekomendasi [2].

Pada MF, diimplementasikan sebuah algoritma yang mampu mampu mengeksploitasi informasi *tagging* pada proses memprediksi *rating* dimana user dan *item* dimodelkan sebagai berikut, $p_u \in \mathbb{R}^k$ dan $q_m \in \mathbb{R}^k$, dimana p_u = parameter vektor p (*user*), q_m = parameter vektor q (*item*), $\mathbb{R}^k$ = *latent features* yang telah dimodelkan dari *user* dan *item*. *Rating* diperkirakan dengan menghitung *dot product* pada vektor: $\check{r}_{um} = p_u^T q_m$ dimana *T* nantinya didefinisikan sebagai *tags* [2].

2.2.1 Tags and Latent Factors

Tagging atau memberi label pada data menawarkan *user* sebuah cara alternatif untuk memberi rekomendasi dan memprediksi minat *user*. Karena *tag*s mudah dipahami oleh *user*, *tags* berfungsi sebagai jembatan yang memungkinkan *user* untuk lebih memahami hubungan yang tidak diketahui antara *item* dan *user* itu sendiri [3]. Sedangkan *latent factors* merupakan bagian dari *matrix factorization*, dimana *user* dan *item* dimodelkan oleh *factor models* yang memodelkan *user* dan *item* yang bertujuan untuk membentuk kesamaan antara *user* dan *item*. Kesamaan dan hubungan antara *user* dan *item* akan dieksploitasi untuk memprediksi nilai *rating* yang hilang sehingga mampu memberikan rekomendasi [2].

Dalam menentukan parameter model, diperlukan rumus yang mampu mengatasi masalah minimisasi, yaitu sebagai berikut [2]:

$$min_{pu,qm,xt,ys} \sum_{(u,m)\in R} (rum - \hat{r}um)^2 + \lambda \left(||pu||^2 + ||qm||^2 + \sum_{t \epsilon Tu} ||xt||^2 + \sum_{s \epsilon Tm} ||ys||^2 \right) \quad (2\text{-}1)$$

R diatas didefinisikan sebagai *training set* dan *testing set*. Parameter $\lambda \in R$ bertugas untuk mengontrol kompleksitas pada model, yang berguna untuk mencegah *overfitting*. Parameter p_u, q_m, x_t, y_s secara otomatis dipelajari selama fase *training dan testing* algoritma. Masalah minimasi sebelumnya dapat secara efisien diselesaikan menggunakan *stochastic gradient descent*, yaitu pada algoritma berikut [2]:

for $(u,m) \in R$ **do**

$eum \leftarrow rum \leftarrow \hat{r}um \Rightarrow Compute\ \hat{r}um\ using\ eq.1$

⇒ Simultaneously update the parameter vectors:

$$pu \leftarrow pu - \alpha \left[\lambda pu - eum \left(qm + |Tm|^{-1} \sum_{s \in Tm} ys \right) \right]$$

$$qm \leftarrow qm - \alpha \left[\lambda qm - eum \left(pu + |Tu|^{-1} \sum_{t \in Tu} xt \right) \right]$$

for all $t \in Tu$ **do**

$$xt \leftarrow xt - \alpha \left[\lambda xt - \frac{eum}{|Tu|} \left(qm + |Tm|^{-1} \sum_{s \in Tm} ys \right) \right]$$

end for

for all $s \in Tm$ **do**

$$ys \leftarrow ys - \alpha \left[\lambda ys - \frac{eum}{|Tm|} \left(pu + |Tu|^{-1} \sum_{t \in Tu} xt \right) \right]$$

end for
end for
end procedure (2-2)

Parameter α mengontrol sejauh mana parameter diperbarui dalam tiap *step* pada algoritma diatas, nilai yang terlalu kecil akan membuat proses *learning* sangat lambat sedangkan dengan nilai yang terlalu besar algoritma bisa saja gagal untuk bersatu [2].

MF yang menggunakan *tags*, dimodelkan sebagai berikut, $x_t \in \mathbb{R}^k$ dan $y_s \in \mathbb{R}^k$, dimana [2]: x_t = parameter vektor x (*user tags*), y_s = parameter vektor y (*item tags*), $\mathbb{R}^k$ = *latent features* yang telah dimodelkan dari *user* dan *item.* Untuk mendapatkan nilai *rating* akan digunakan rumus sebagai berikut [2]:

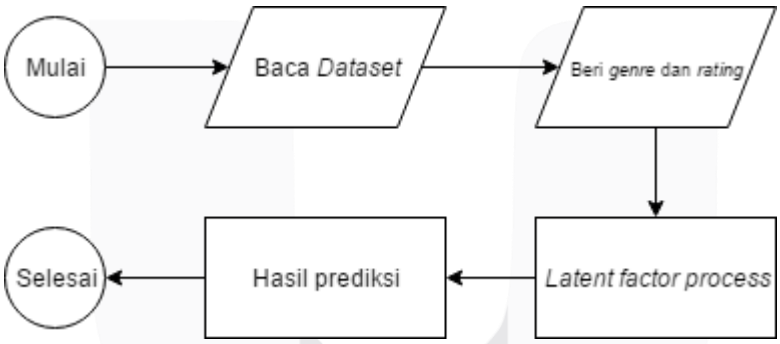
$$\hat{r}um = \left(pu + \frac{1}{|Tu|} \sum_{t \in Tu} xt \right)^T \left(qm + \frac{1}{|Tm|} \sum_{s \in Tm} ys \right) \quad (2\text{-}3)$$

Keterangan:
T_u = *set* dari *tags* yang ditentukan oleh *user* u untuk tiap *item*
T_m = *set* dari *tags* yang ditentukan pada *item* m oleh tiap *user*

3. Desain Sistem

3.1 Gambaran Umum Sistem

Sistem dengan dibangun sesuai dengan alur sistem rekomendasi berikut:



Gambar 1 Gambaran Umum Sistem

Keterangan:

- Baca *dataset*: Sistem membaca *dataset* buku yang nantinya akan keluar data buku berjumlah 100 buah.
- Beri *genre* dan *rating*: Disini *user* dapat memberikan *genre* dan nilai *rating* pada buku yang nantinya akan menjadi informasi berguna untuk memberikan rekomendasi pada *user* lain.
- *Latent factor process*: *User* yang telah memberikan *genre* dan *rating* pada buku kemudian akan diolah untuk memberikan hasil dalam bentuk rekomendasi buku.
  Hasil prediksi: Disini sistem menampilkan *genre* dan nilai *rating* dari buku yang menjadi rekomendasi untuk tiap *user*

3.2 Baca Dataset

Pada baca *dataset* ini, proses yang berjalan hanya menampilkan buku yang berjumlah 100 buah. Dengan adanya tampilan buku tersebut, seterusnya *user* akan melanjutkan proses dimana memberikan *tagging* berupa *genre* pada buku dan juga memberikan nilai *rating*. Buku yang sudah diberikan *tagging genre* dan *rating* oleh *user* tadi akan tersimpan pada *database* yang nantinya akan digunakan sebagai data untuk menghasilkan nilai *rating* sebenarnya sehingga dapat memberikan rekomendasi nantinya.

Tabel 3 Jumlah data pada training dan testing

	Jumlah data yang digunakan
Dataset Training	100

Dataset Testing	51

Dataset yang digunakan pada Tugas Akhir ini adalah data buku *Book-Crossing* milik *Institut für Informatik, Universität Freiburg* yang menyediakan *dataset* tentang buku yang mengandung 278,858 *user* (tidak dikenal tetapi memiliki informasi demografi) menyediakan 1,149,780 *rating* (eksplisit maupun implisit) tentang 271,379 buku (http://www2.informatik.uni-freiburg.de/~cziegler/BX/).

Pada *dataset* ini hanya mengambil data 100 buku saja. Buku yang digunakan tidak berdasarkan kategori tertentu, hanya diambil secara acak. Pengambilan data sebanyak 100 buku dikarenakan untuk penghematan waktu dalam membaca *dataset*. Untuk data pengguna sendiri, penulis menggunakan data milik sendiri dimana jumlah pengguna yang akan digunakan dalam pengujian Tugas Akhir ini sebanyak 5 orang. Begitu juga dengan nilai *rating*, nilai *rating* yang digunakan berasal dari masukan yang dilakukan oleh 5 pengguna tadi. Berikut tampilan dari *dataset Book-Crossing*:

Tabel 4 Dataset Book-Crossing

ISBN;"Book-Title";"Book-Author";"Year-Of-Publication";"Publisher";"Image-URL-S";"Image-URL-M";"Image-URL-L"
0399135782;"The Kitchen God's Wife";"Amy Tan";"1991";"Putnam Pub Group";"http://images.amazon.com/images/P/0399135782.01.THUMBZZZ.jpg";"http://images.amazon.com/images/P/0399135782.01.MZZZZZZZ.jpg";"http://images.amazon.com/images/P/0399135782.01.LZZZZZZZ.jpg"
0609804618;"Our Dumb Century: The Onion Presents 100 Years of Headlines from America's Finest News Source";"The Onion";"1999";"Three Rivers Press";"http://images.amazon.com/images/P/0609804618.01.THUMBZZZ.jpg";"http://images.amazon.com/images/P/0609804618.01.MZZZZZZZ.jpg";"http://images.amazon.com/images/P/0609804618.01.LZZZZZZZ.jpg"

3.3 Beri Genre dan Rating

Dalam memberikan *tagging* dan *rating* pada buku diperlukan *user*, nantinya pada saat menjalankan program akan berlaku skenario dimana *user* nantinya akan terbagi menjadi dua. Untuk *user* yang memberikan *tagging* dan *rating* tidak berasal dari golongan tertentu dengan kata lain semua *user* adalah sederajat. *User* yang pertama berperan sebagai *user* lama dimana bertugas memberikan *tagging* dan *rating* pada buku. Maksud dari *user* lama disini adalah *user* yang sudah melakukan *review* terhadap buku baik dari memberikan *genre* maupun *rating* yang dimana nantinya *genre* dan *rating* tersebut akan mempengaruhi minat *user* yang lain untuk mencari buku yang mereka inginkan. Untuk *genre* buku yang digunakan dalam pengerjaan Tugas Akhir ini penulis hanya menggunakan 6(enam) saja, yaitu:

1. *Romance*
2. *Mystery/Thriller*
3. *Fantasy*
4. *Young Adult*
5. *Science Fiction*
6. *Children*

Sedangkan untuk nilai *rating* pada buku, penulis mengambil rentang nilai 1-5 dimana nilai 5 merupakan nilai rekomendasi terbaik.

User yang kedua berperan sebagai *user* yang diberikan rekomendasi buku dimana buku yang telah diberikan *tagging genre* dan *rating* oleh *user* lama akan diolah dengan *latent factors* yang nantinya akan memberikan rekomendasi buku terbaik berdasarkan *genre* buku yang diinginkan.

3.4 Latent Factor Process

Pada proses ini, buku yang sudah diberikan *genre* dan *rating* oleh *user* pertama akan tersimpan dalam *database* sebagai *data training* lalu akan diuji sebagai *data testing*. *Data training* dan *testing* ini kemudian akan diolah menggunakan rumus dari persamaan (2-1) berikut:

$$min_{pu,qm,xt,ys} \sum_{(u,m)\in R} (rum - \hat{r}um)^2$$
$$+\lambda \left(||pu||^2 + ||qm||^2 + \sum_{t \epsilon Tu} ||xt||^2 + \sum_{s \epsilon Tm} ||ys||^2 \right)$$

Keterangan:

R : *data training* dan *testing set*

p_u : parameter model *user*

q_m : parameter model *item*

T_u : *set* dari *tags* yang ditentukan oleh *user* u untuk tiap *item*
T_m : *set* dari *tags* yang ditentukan pada *item* m oleh tiap *user*
x_t : parameter vektor x (*user tags*)
y_s : parameter vektor y (*item tags*)

3.5 Nilai Prediksi

Proses ini akan mengeluarkan nilai prediksi baik *genre* dan juga *rating* dari buku. Untuk mengeluarkan *genre* dan *rating*, rumus dari persamaan (2-3) yang dipakai sebagai berikut:

$$\hat{r}um = (pu + \frac{1}{|Tu|} \sum_{t \in Tu} xt)^T \ (qm + \frac{1}{|Tm|} \sum_{s \in Tm} ys)$$

Keterangan:
p_u : parameter model *user*
q_m : parameter model *item*
T_u : *set* dari *tags* yang ditentukan oleh *user* u untuk tiap *item*
T_m : *set* dari *tags* yang ditentukan pada *item* m oleh tiap *user*

4. Pengujian dan Analisis

4.1 Strategi pengujian

Tujuan dari sistem yang dikerjakan yaitu memberikan rekomendasi buku kepada *user* berdasarkan nilai *rating* tertinggi. Sebuah buku dianggap menarik apabila hasil prediksi *rating* bernilai tinggi berdasarkan metode yang penulis gunakan. Pengujian sistem akan dilakukan dengan menggunakan *evaluation metrics* yaitu *Mean Absolute Error* (MAE) dan akan dibagi menjadi tiga skenario pengujian.

4.2 Skenario pengujian

$$MAE = \frac{1}{n} \sum_{i=1}^{n} | f_i - y_i | \quad (4\text{-}1)$$

Berdasarkan rumus MAE dibawah ini akan dilakukan skenario pengujian dilakukan sebagai berikut:
Keterangan:
f_i : nilai hasil ramalan
y_i : nilai sebenarnya
n : jumlah data

Tabel 5 Skenario Pengujian

No	Pengujian	Keterangan
1.	MAE dengan data *rating Latent Factor* dan data *rating* awal	Data *rating* Latent Factor: *fi* Data *rating* awal: *yi* Jumlah buku yang digunakan: 51
2.	MAE dengan data *rating Latent Factor* dan data *rating* awal	Data *rating* Latent Factor: *fi* Data *rating* awal: *yi* Jumlah buku yang digunakan: 35
3.	MAE dengan data *rating Latent Factor* dan data *rating* awal	Data *rating* Latent Factor: *fi* Data *rating* awal: *yi* Jumlah buku yang digunakan: 25

4.3 Analisis hasil pengujian

Dari tabel skenario pengujian diatas, akan ditampilkan hasil dari pengujian yang dilakukan sebagai berikut:

Tabel 6 Hasil Pengujian

No	Total Data Rating Latent Factor	Total Data Rating	Jumlah Buku	Jumlah User	MAE
1.	169	177,85	51	5	0,41
2.	121	125,86	35	5	0,38
3.	86	89	25	5	0,38

Pada tabel diatas, telah dilakukan 3 skenario pengujian dengan menggunakan jumlah data yang berbeda dengan jumlah masing-masing 51, 35 dan 25 buku. Disini digunakan jumlah maksimal 51 buku karena itu jumlah yang sama dari buku yang telah diberikan nilai *rating* oleh *user*. Dan skenario yang menggunakan buku sebanyak 35 dan 25 buah karena ingin membandingkan dengan penggunaan jumlah buku yang lebih sedikit. Dapat dilihat

dari hasil pengujian 1 dengan buku yang digunakan sebanyak 51 buah mendapatkan skor MAE sebesar 0,41. Untuk pengujian 2 dan 3, dengan jumlah masing-masing buku yang digunakan sebanyak 35 dan 25, didapatkan skor MAE yang sama sebesar 0,38. Dari hasil pengujian ini dapat dilihat bahwa jumlah buku yang digunakan tidak berpengaruh besar terhadap nilai pengujian karena selisih MAE dari 3 skenario pengujian tidak memiliki selisih yang cukup jauh.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Berdasarkan hasil skenario pengujian dan analisis yang telah dilakukan, dapat ditarik kesimpulan, yaitu Sistem rekomendasi pada buku yang menggunakan *Tags and Latent Factors* dapat memberikan rekomendasi yang akurat dimana nilai pengujian yang didapat dari 3 skenario pengujian memiliki nilai masing-masing 0.41, 0.38, dan 0.38 dimana nilai pengujian mendekati angka 0 maka memiliki prediksi yang benar.

5.2 Saran

Terdapat beberapa saran yang dapat dijadikan pertimbangan untuk pengembangan dari Tugas Akhir ini untuk kedepannya, yaitu dalam pengerjaan Tugas Akhir ini, ada parameter yang harus diubah agar metode ini dapat dimaksimalkan dengan baik dan diantaranya adalah penggunaan jumlah buku yang digunakan harus ditambah dan jumlah *user* yang berpartisipasi dalam melakukan *tagging* dan memberikan *rating* juga harus ditambah.

Daftar Pustaka

[1] A. S. Tewari, A. Kumar dan A. G. Barman, “Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining,” pp. 1-4, 2014.

[2] M. Ge, M. Elahi, I. Fernandes-Tobias, F. Ricci dan D. Massimo, “Using Tags and Latent Factors in a Food Recomender System,” pp. 1-7, 2015.

[3] S. Sen, J. Vig dan J. Riedl, “Tagommenders: Connection Users to Items through Tags,” pp. 1-10, 2009.

[4] J. Lu, D. Wu, M. Mao, W. Wang dan G. Zhang, “Recommender System Application Developments: A Survey,” pp. 1-30, 2015.

[5] J. Hannon, M. Bennett dan B. Smyth, “Recommending Twitter Users to Follow Using Content and Collaborative Filtering Approaches,” pp. 1-8, 2010.

[6] I. Santos, I. Minambres-Marcos, C. Laorden, P. Galan-Garcia, A. Santamaria-Ibirika dan P. G. Bringas, “Twitter Content-Based Spam Filtering,” pp. 1-8, 2014.

[7] R. J. Mooney dan L. Roy, “Content-Based Book Recommending Using Learning for Text Categorization,” pp. 1-7, 1999.

[8] D. S. Jain, A. Grover, P. S. Thakur dan S. K. Choudhary, “Trends, Problems And Solutions of Recommender System,” pp. 1-4, 2015.

[9] P. Melville dan V. Sindhwani, “Recommender Systems,” pp. 1-14, 2010.

[10] G. Linden, B. Smith dan J. York, “Amazon.com Recommendations: Item-to-Item Collaborative Filtering,” pp. 1-5, 2003.

[11] J. L. Herlocker, J. A. Konstan dan J. Riedl, “Explaining Collaborative Filtering Recommendations,” pp. 1-10, 2000.