Jurnal Elektronik Ilmu Komputer Udayana
Volume 13, No 2. November 2024

p-ISSN: 2301-5373
e-ISSN: 2654-5101

Analisis Algoritma ALS-MF (*Alternating Least Square Matrix Factorization*) dengan SVD (*Singular Value Decomposition*) pada Metode *Collaborative Filtering*

Ngakan Putu Widyasprana[a1], I Made Widhi Wirawan[a2], I Gede Santi Astawa[a3], I Dewa Made Bayu Atmaja Darmawan[a4]

[a] Program Studi Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Udayana, Kuta Selatan, Badung Bali, Indonesia
[1]ngakanputu39@email.com
[2] made_widhi@unud.ac.id
[3] santi.astawa@unud.ac.id
[4] dewabayu@unud.ac.id

Abstraksi

Perkembangan industri game hingga mencapai $175,8 miliar di tahun 2021, memicu kebutuhan akan sistem rekomendasi yang presisi di berbagai platform seperti console, mobile, dan tren yang sedang berkembang seperti e-sport ataupun VR. Collaborative filtering, salah satu metode yang sering digunakan, memiliki kelemahan dalam menangani item baru atau yang belum diberi rating. Penelitian sebelumnya menunjukkan bahwa algoritma Alternating Least Squares (ALS) dapat meningkatkan akurasi sistem rekomendasi pada dataset Goodreads dengan nilai RMSE 0,86-0,88. Selain itu, ALS terbukti lebih unggul dalam perbandingan sistem rekomendasi dengan nilai RMSE 0,641. Berdasarkan hal tersebut, penelitian mengusung metode collaborative filtering dengan algoritma ALS untuk mengatasi kelemahannya, serta meningkatkan akurasi rekomendasi game. Algortima ALS juga dibandingkan dengan sistem dengan menggunakan SVD untuk memberikan komparasi antar algortima. Hasil akhir pada model didapatkan model ALS optimal dengan nilai hyperparameter rank 1, regularisasi 0.1, dan 5 iterasi, metode sklearn split untuk pembagian data, dan skor persentil untuk metode konversi rating. Selain itu ALS juga unggul dibandingkan dengan menggunakan SVD dengan berhasil mencapai nilai Root Mean Squared Error (RMSE) sebesar 1.1142 untuk data uji dan 0.8873 untuk data latih.

Kata Kunci: *Sistem Rekomendasi, ALS, SVD, RMSE, Hybrid Systems, Machine Learning, RMSE*

1. Pendahuluan

Perkembangan teknologi yang cepat pada periode tahun ini membut banyak sekali inovasi-inovasi yang diluncurkan untuk memenuhi kebutuhan akan perkembangan itu. Salah satunya adalah sistem rekomendasi, sebagai contohnya, sistem rekomendasi game [1]. Dengan perkembangan yang masif ini sangat diperlukannya sistem rekomendasi yang presisi agar mampu mengatasi hal ini sehingga pengguna tidak perlu lagi bingung dalam memutuskan video game yang ingin dimainkan [2]. Beberapa contoh metode yang biasa digunakan dalam sistem rekomendasi yakni *content-base filtering* dan *collaborative filtering* [3]. *Content-base filtering* adalah metode rekomendasi sistem yang membutuhkan sejumlah informasi tentang fitur item seperti genre, tahun, sutradara, aktor dan lainnya. Sedangkan *collaborative filtering* adalah metode rekomendasi sistem yang membutuhkan hanya informasi preferensi historis, tidak menyangkut fitur item. Hal ini berdasarkan pengguna yang setuju di masa lalu cenderung akan setuju di masa mendatang.

Mengacu pada hal tersebut pada penelitian sebelumnya, melakukan komparasi antara *content-base filtering* dengan *collaborative filtering*. Dari penelitian ini didapat bahwa *content-base filtering* memiliki keunggulan dalam merekomendasikan item kepada pengguna berdasarkan fitur item dan profil pengguna, tetapi kekurangannya adalah adalah tidak merekomendasikan berdasarkan kualitas dari item tersebut. Sedangkan untuk, *collaborative filtering* memiliki keunggulan dalam hasil prediksi yang berdasarkan preferensi pengguna lainnya, seperti rating dari pengguna dan

kualitas dari rekomendasi. Akan tetapi kekurangannya adalah kebutuhan akan data yang begitu besar sehingga rekomendasi baru dapat dilakukan [4]. Pada penelitian lainnya, dalam penerapan metode *content-based filtering* pada sistem rekomendasi aplikasi MANGAN mendapati kemiripan item profile antara satu toko dengan toko lainnya terlihat dengan jelas, dan akurasi yang di dapat dari rekomendasi sistem ini adalah 0.511. Dikatakan bahwa, rendahnya nilai akurasi ini karena kurangnya data dan metode pengujian yang kurang maksimal [5]. Akan tetapi dalam penerapannya metode *collaborative filtering* memiliki kelemahan yakni sulit untuk melakukan rekomendasi pada data yang sifatnya baru atau data yang belum diberikan rating sama sekali oleh satu-pun pengguna [6] .

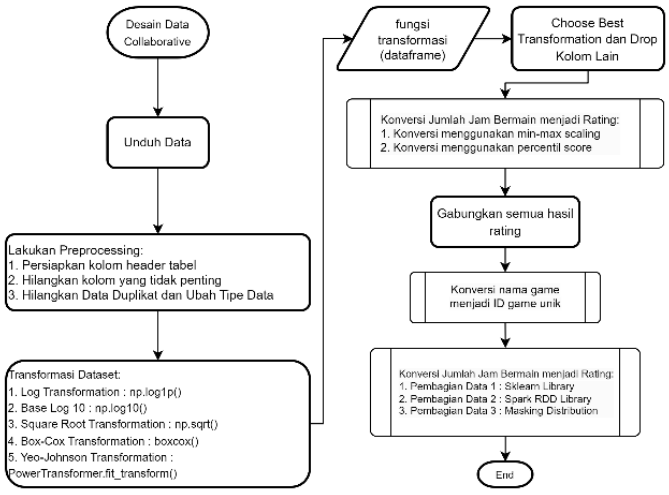
Untuk mengatasi masalah ini adapun penelitian sebelumnya pada dataset Goodreads, menggunakan algoritma ALS sebagai dasar algortima, kinerja dalam memprediksi data dari Goodreads sejumlah 981.756 mendapati nilai MAE dan RMSE di angka 0.86 dan 0.89 pada data latih dan data uji [7]. Adapun penelitian lain yakni melakukan analisis membandingkan penerapan algortima faktorisasi matriks dengan KNN (*K-Nearest Neighbor*), dan didapatkan hasil prediksi rating menggunakan faktorisasi matriks lebih sesuai dan nilai pengujian MAE dan RMSE lebih rendah dibandingkan dengan algortima KNN [8]. Lalu, ada penelitian lainnya juga yang melakukan reduksi kesalahan pada sistem rekomendasi menggunakan pendekatan *collaborative filtering* berbasis model faktorisasi matrik. Dari penelitian ini didapat bahwa, hasil reduksi kesalahan menjadi lebih kecil yakni sejumlah 0.6814 dibandingkan dengan menggunakan model berbasis memori yakni sebesar 2.984 [9].

Berdasarkan dengan pemaparan permasalahan serta beberapa penelitian terkait mengenai metode pada sistem rekomendasi, pada penelitian saat ini akan menggunakan sistem hybrid yang menggunakan konsep *collaborative filtering* pada dataset dan melakukan penerapan algortima *machine learning* ALS-MF (*Alternating Least Square Matrix Factorization*) dengan dibandingkan menggunakan algoritma SVD (*Singular Value Decomposition*). Untuk proses implementasi model akan menggunakan Apache Spark sebagai baseline pembuatan model. Dalam mengatasi perbedaan dalam interpretasi rating pada data dilakukan beberapa skenario uji yang nanti akan dibandingkan dan dicari skenario uji terbaik untuk dataset yang digunakan.

2. Metode Penelitian

2.1 Desain Sistem

Pada penelitian kali ini membagi metode penelitian menjadi 3 bagian, bagian pertama adalah bagian persiapan data *metode collaborative*, untuk bagian kedua adalah bagian implementasi model *machine learning* ALS-MF dengan Apache Spar, untuk bagian ketiga adalah bagian implementasi model *machine learning* SVD dan metode terakhir adalah desain dalam melakukan skenario uji sistem. Berikut adalah rincian metodenya:



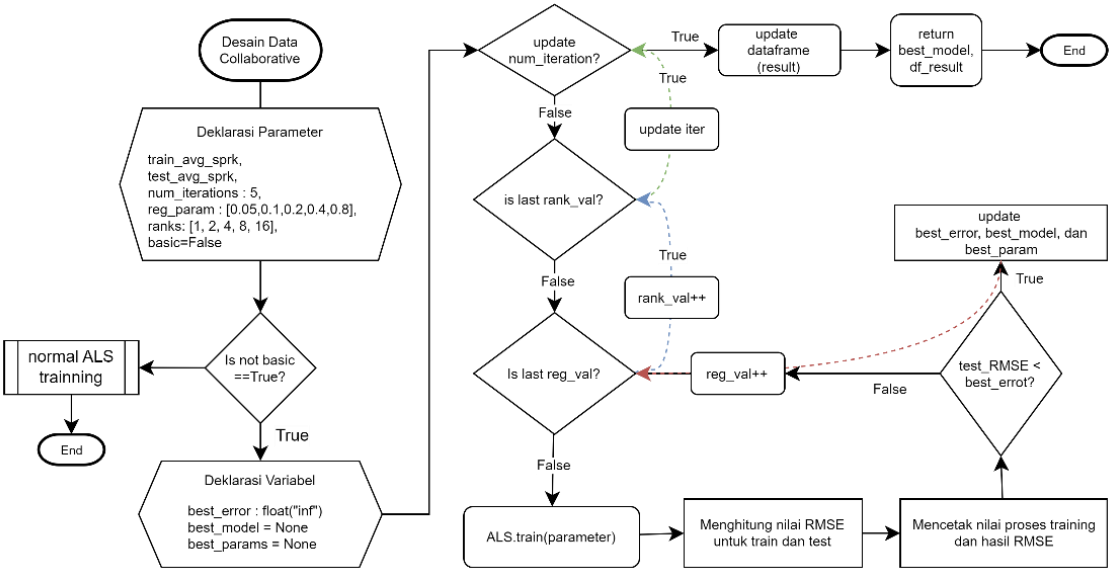
Gambar 1. Desain Metode Collaborative

Berdasarkan dengan gambar 1 diatas berikut adalah proses tahapan dalam melakukan desain data metode *collaborative filtering*:

p-ISSN: 2301-5373
e-ISSN: 2654-5101

1. Tahap pertama, melakukan proses pengunduhan data dan melakukan pengubahan jenis datanya menjadi sebuah pandas *dataframe*.
2. Tahap kedua, melakukan *preprocessing* data dengan mempersiapkan *header* data sebagai nama kolom terlebih dahulu, lalu menghilangkan kolom data yang tidak penting, dan menghilangkan data duplikat dan mengubah tipe data beberapa kolom agar bisa digunakan untuk tahap selanjutnya.
3. Tahap ketiga, melakukan transformasi dataset sesuai dengan jenis kecondongan data, tingkat *skewness* data dengan beberapa metode, seperti transformasi logaritma natural, transformasi kuadrat, transformasi box-cox dan transofrmasi yeo-johnson.
4. Tahap keempat, memproses *dataframe* dengan fungsi transformasi dan mendapatkan hasil dari transformasi baik dalam bentuk grafik dan juga dalam bentuk statistika.
5. Tahap kelima, memilih hasil dari transformasi data terbaik mengacu pada nilai *skewness* yang paling kecil dari original-nya. Jika sudah memilih, maka semua data kolom transformasinya dihapus.
6. Tahap keenam, terdapat bagian proses untuk melakukan konversi kolom game_hourtime menjadi sebuah rating diskrit dengan nilai [1, 2, 3, 4, 5].
7. Tahap ketujuh, setelah semua pendekatan konversi rating didapat, maka gabungkan semua konversi rating menjadi satu dalam *dataframe* asli.
8. Tahap kedelapan, terdapat bagian proses untuk mengubah game_name yang berupa *string* menjadi sebuah id *game* unik yang mengkodekan setiap nama game menjadi unik.
9. Tahap kesembilan, setelah dataset siap, selanjutnya ada subproses melakukan pembagian data dengan beberapa pendekatan untuk memastikan pembagian data cukup beragam untuk bisa dilakukan analisis.

Setelah menyiapkan data menjadi bentuk *collaborative filtering*, tahap selanjutnya adalah melakukan persiapan model menggunakan algoritma ALS-MF dengan *Library* PySpark dari *engine* Apache Spark, seperti pada gambar 2 berikut:



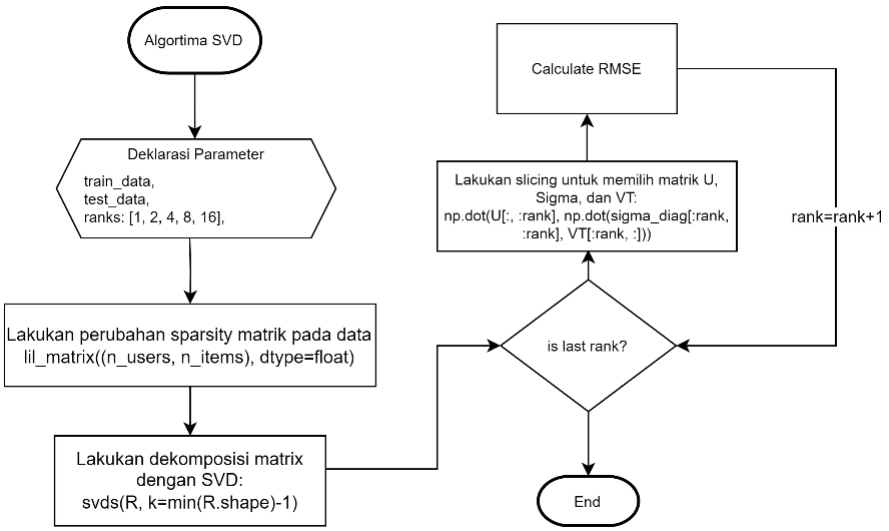
Gambar 2. Desain ALS-MF dengan *Library* PySpark

Berdasarkan dengan gambar 2 diatas, setelah data sudah dalam format *collaborative filtering*, selanjutnya akan melakukan proses latih model data dengan ALS-MF yang dibuat dengan Apache Spark. Berikut adalah tahapan prosesnya:

1. Tahap pertama, melakukan inisialisasi parameter yang akan digunakan untuk tahap latihan model ALS nanti.
2. Tahap kedua, jika variabel basic adalah *true* maka akan menuju subproses normal latihan model dengan ALS. Jika tidak maka akan melanjutkan ke proses selanjutnya yakni proses *hyperparameter tuning* menggunakan metode *grid search*.
3. Tahap ketiga, melakukan definisi variabel yang digunakan untuk proses *hyperparameter tuning*.

4. Tahap keempat, melakukan proses perulangan dimulai dari num_iteration, rank_val, dan reg_val. Perulangan akan terus berjalan hingga seluruh kombinasi 3 komponen terpenuhi dengan diakhiri hingga melakukan perubahan pada num_iteration.
5. Tahap kelima, pada proses ini saat melakukan latihan model menggunakan model ALS langsung menggunakan inisialisasi estimator langsung ALS. Parameter yang parameter yang digunakan sesuai yang telah dideklarasikan.
6. Tahap keenam, menghitung nilai data latih dan data uji RMSE dengan hasil model setelah *trainning*.
7. Tahap ketujuh, tahap ini melakukan cetak nilai parameter dengan nilai RMSE nya untuk mengetahui proggress dari *hyperparameter tuning*.
8. Tahap kedelapan, melakukan cek apakah nilai tes RMSE lebih kecil dari nilai best_error, jika iya lakukan proses prubahan parameter best_error, best_params, dan best_model.
9. Tahap kesembilan, ulangi semua proses hingga perulangan berakhir dan melakukan perubahan pada *dataframe* untuk keseluruhan hasil. Dan mengembalikkan keseluruhan parameter terbaik.

Setelah implementasi sistem ALS-MF, adapun implementasi sistem menggunakan algortima SVD dengan *Library* Scipy, implementasi pada gambar 3 dibawah:



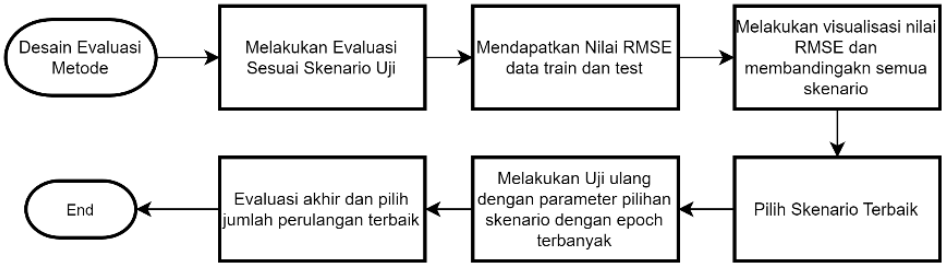
Gambar 3. Desain SVD dengan *Library* Scipy

Berdasarkan gambar 3 diatas, format data yang masuk masih sama dengan model dengan ALS-MF sebelumnya. Berikut adalah penjelasan tahapan dalam melakukan implementasi model:

1. Tahap pertama, lakukan deklarasi parameter dan *hyperparameter* yakni data latih dan data uji, lalu untuk *hyperparameter* adalah rank.
2. Tahapa kedua, ubah data menjadi sebuah *sparsity matrix* yang meminiliki ukuran (user_id, game_id), dan lakukan dekomposisi dengan algortima SVD untuk mendapatkan nilai U, Σ, dan V^T.
3. Tahap ketiga, lakukan perulangan untuk setiap kemungkinan rank dan lakukan proses pememilihan fitur U, Σ, dan V^T berdasarkan dengan nilai rank.
4. Tahap keempat, lakukan perthitungan dan cetak nilai RMSE hingga seluruhnya dan program berakhir.

Setelah implementasi sistem selesai, tahap selanjutnya melakukan proses latihan model dengan skenario uji yang telah dibuat dan melakukan analisis terhadap keseluruhan skenario uji lalu mendapatkan hasil akhir sistem ALS-MF dan SVD. Untuk rinciannya dapat dilihat pada gambar 4 dibawah :

p-ISSN: 2301-5373
e-ISSN: 2654-5101



Gambar 4. Desain Evaluasi Metode ALS-MF

Berdasarkan dengan gambar 4 diatas, berikut adalah rincian dalam melakukan skenario uji sistem:

1. Tahap pertama, melakukan evaluasi terhadap seluruh skenario yag telah dibuat dan mendapatkan nilai RMSE baik data latih dan data uji untuk tahap selanjutnya nanti.
2. Tahap kedua, setelah melakukan pengujian, dapatkan nilai RMSE baik data latih atau data uji yang nantinya akan digunakan untuk proses pembandingan.
3. Tahap ketiga, melakukan visualisasi dari nilai RMSE data latih dan data uji untuk dilihat pengaruh dari skenario uji dengan nilai RMSE pada model ALS dan SVD.
4. Tahap keempat, melakukan sortir dan pemilihan skenario terbaik.
5. Tahap kelima, melakukan uji ulang dengan skenario terbaik dengan jumlah perulangan yang lebih dari yang digunakan di skenario untuk mengetahui penurunan nilai RMSE pada model.
6. Tahap keenam, memilih jumlah perulangan terbaik yang memiliki tingkat penurunan nilai RMSE yang signifikan.

2.2 Evaluasi Sistem

Untuk mengukur seberapa baik sistem yang telah dibuat dengan skenario uji yang dijalankan, maka matrik pengukuran yang digunakan adalah RMSE (*Root Mean Square Error*). Untuk proses kalkulasi dari RMSE sebagai berikut:

$$RMSE = \left(\frac{\sum(y_i - \hat{y}_i)}{n}\right)^{1/2} \quad (1)$$

Dapat dilihat pada persamaan (1), dalam menghitung nilai *error* sistem akan menggunakan penjumlahan data hasil observasi (y_i) dikurangi data hasil prediksi ($\hat{y}_i$) yang dibagi dengan jumlah data dan dikuadratkan. Sehingga hasil akhirnya berupa nilai RMSE yang semakin mendekati 0 maka semakin bagus. Untuk menunjang dalam melakukan evaluasi sistem, akan menggunakan skenario uji pada tabel 1 berikut:

Tabel 1. Skenario Uji Sistem ALS-MF

Metode Tuning	Metode Pembagian Data	Metode Konversi Rating	Hyperparameter	
Skenario Uji 1				
Grid Search	Sklearn Split	Min-Max Scaling	Ranks	[1,2,4,8,16]
		Percentil Score	Lamda	[0.05,0.1,0.2,0.4,0.8]
Skenario Uji 2				
Grid Search	Spark Split	Min-Max Scaling	Ranks	[1,2,4,8,16]
		Percentil Score	Lamda	[0.05,0.1,0.2,0.4,0.8]
Skenario Uji 3				
Grid Search	Masking Distribution	Min-Max Scaling	Ranks	[1,2,4,8,16]
		Percentil Score	Lamda	[0.05,0.1,0.2,0.4,0.8]

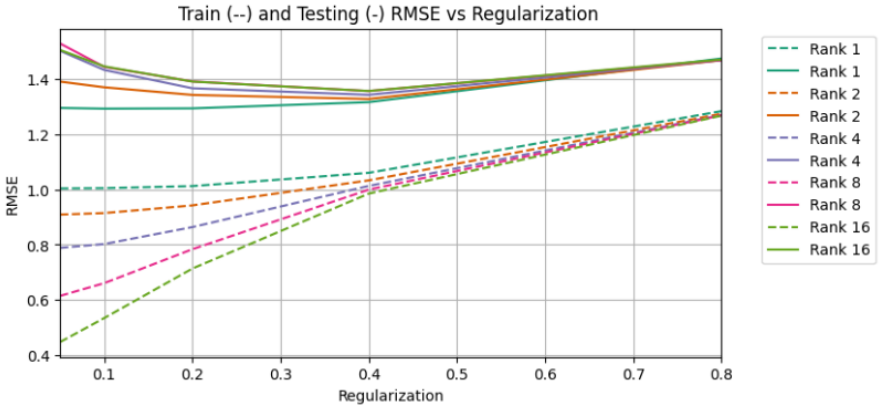
Sedangkan untuk skenario uji sistem SVD hanya menggunakan nilai rank saja dengan rentang 1 sampai 1000.

3. Result and Discussion

Setelah melakukan perancangan, hasil evaluasi akan dilakukan 4 tahap pembandingan dan 1 hasil akhir, rinciannya sebagai berikut: 1. Pembandingan antar *hyperparameter*, 2. Pembandingan antar metode konversi rating, 3. Pembandingan antar pembagian data, 4. Pembandingan antar algortima model, dan 5. Pemiihan hasil akhir model. Untuk penjelasan lebih lanjut sebagai berikut:

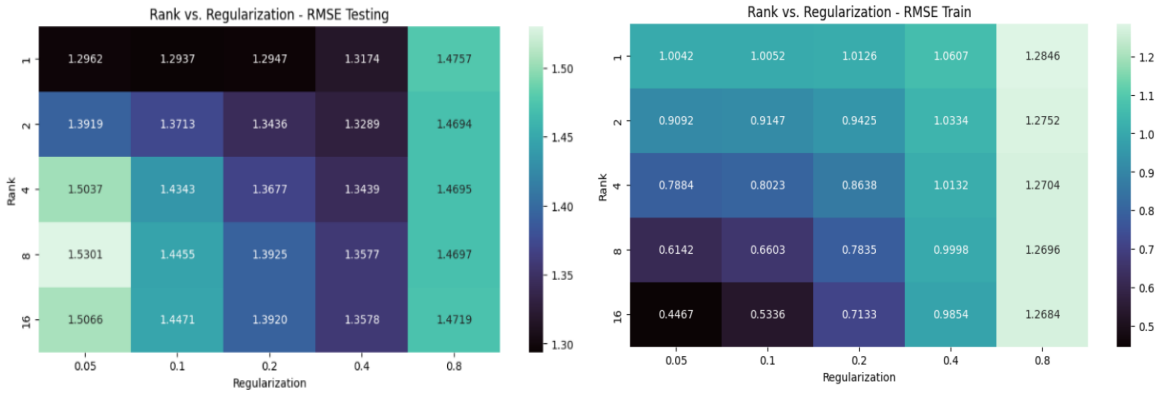
3.1. Perbandingan Hyperparameter

Secara garis besar semakin tinggi rank yang ditetapkan maka nilai RMSE semakin rendah untuk data latih sedangkan sebaliknya untuk data uji. Terlihat pada gambar 4 (hasil skenario 1) dibawah, hasil visualisasi rank dengan warna yang berbeda-beda dan juga dibedakan dengan garis putus-putus dengan yang normal. Untuk yang garis putus-putus menandakan hasil RMSE untuk data latih sedangkan untuk garis normal menandakan hasil RMSE untuk data uji. Jika melihat dari tingkat posisi garis dan nilai RMSE, maka rank paling rendah untuk data latih adalah rank 16 sedangkan untuk data uji yang paling rendah adalah rank 1. Ada juga nilai RMSE data uji yang terendah kedua yakni rank 8 tetapi memiliki nilai RMSE data latih yang sangat tinggi.



Gambar 5. Visualisasi Hasil Hyperparameter

Selain pada gambar 5 diatas yang merupakan hasil skenario-1, pada hasil skenario-2 dan skenario-3 juga menunjukkan hal serupa, hal ini menandakan semakin tinggi rank maka semakin rendah nilai *error* pada data latih, sedangkan pada data uji nilai *error* akan semakin meningkat. Berbanding terbalik dengan nilai regularisasi, pada semua skenario menunjukkan semakin tinggi nilai regularisasi maka semakin tinggi juga nilai RMSE baik pada data latih dan data uji. Dapat juga dilihat pada salah satu hasil visualisasi skenario pada heatmap gambar 6 berikut:



Gambar 6. Visualisasi Heatmap Skenario Uji

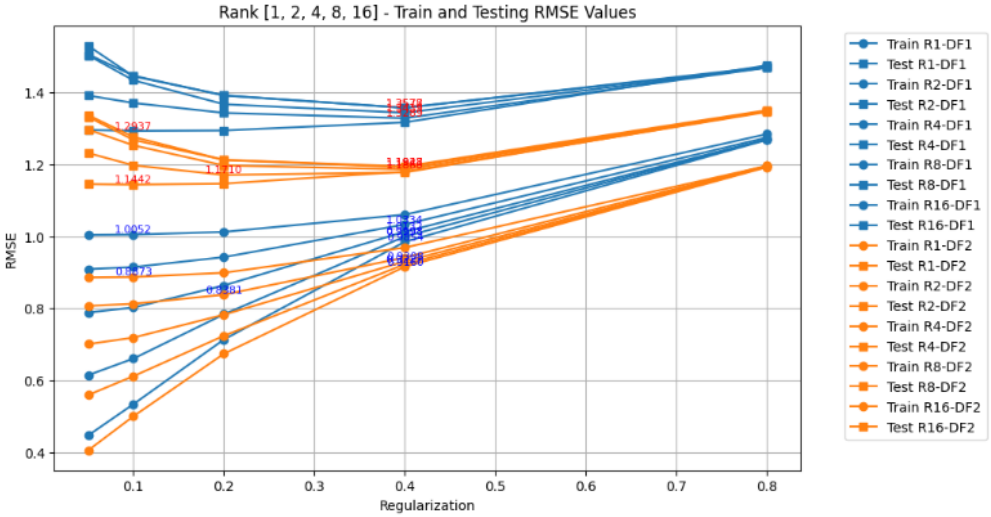
Dapat dilihat juga pada gambar 6 diatas, pada data uji (gambar kiri) dan pada data latih (gambar kanan) menunjukkan bahwa untuk nilai parameter terbaik dengan nilai RMSE 1.2937 ada pada nilai regularisasi 0.1 dengan rank 1, sedangkan pada data latih menunjukkan hasil terbaik pada nilai RMSE 0.4417 ada pada nilai regularisasi 0.05 dan rank 16. Untuk memilih *hyperparameter* rentangan nilai antara data latih dan data uji disarankan tidak boleh terlampau jauh, hal ini

p-ISSN: 2301-5373
e-ISSN: 2654-5101

dikarenakan sistem mengalami *overvitting* pada salah satu data sehingga tidak akan bisa mencerminkan data yang sebenarnya. Untuk itu dipilih hasil yang rentangan *error* antara latih dan uji yang tidak begitu besar. Sehingga pemilihan *hyperparameter* untuk rank adalah 1 dan untuk nilai regularisasi adalah 0.1.

3.2. Perbandingan Metode Konversi Rating

Pada perbandingan metode konversi rating antara *min-max scaling* dengan *percentil score* didapatkan bahwa rata-rata nilai RMSE pada data *test* metode *percentil score* jauh lebih unggul dibandingkan dengan metode *min-max scaling*. Dapat dilihat pada gambar 7 berikut:

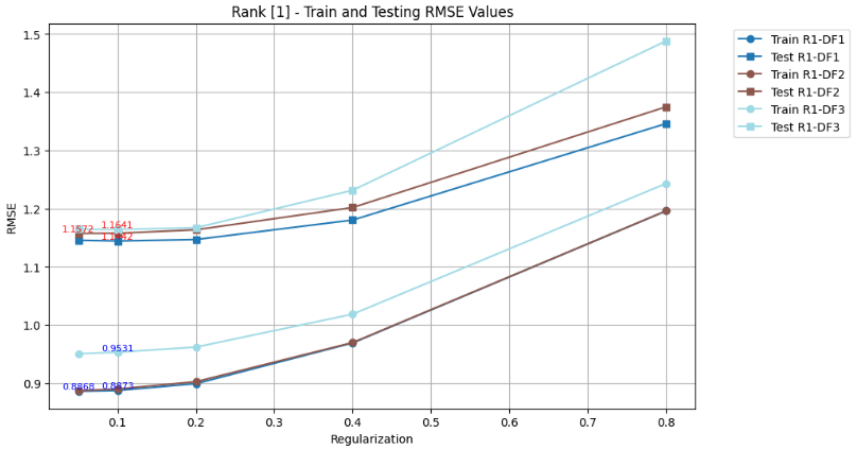


Gambar 7. Perbandingan Metode Konversi Rating *Percentil Score* dengan *Min-Max Scaling*

Dapat dilihat pada gambar 7 diatas untuk garis biru menandakan metode konversi rating *min-max scaling* sedangkan untuk garis oranye metode konversi rating dengan *percentil score*. Dilihat dari gambar untuk data *test* (garis dengan titik kotak) pada metode *percentil score* (warna oranye) rata-rata lebih unggul dibandingkan dengan metode *min-max scaling*, jika mengacu pada data *train* (garis dengan titik bulat) pada metode *percentil score* tetap menjadi yang paling rendah dalam nilai RMSE walau saling berhimpitan dengan metode *min-max scaling.* Jadi secara keseluruhan, metode konversi rating dengan metode *percentil score* jauh lebih baik dibandingkan dengan metode *min-max scaling*.

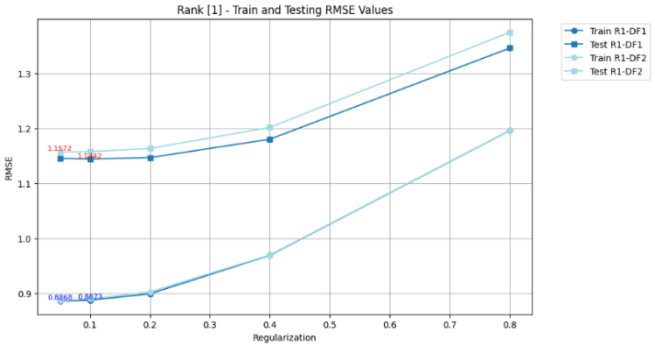
3.3. Perbandingan Pembagian Data

Perbandingan antara skenario akan membandingkan 3 skenario pembagian data dengan menggunakan *hyperparameter* terbaik yakni nilai rank=1 dan nilai regularisasi adalah 0.1. Lalu menggunakan metode konversi rating yang terbaik juga menggunakan metode *percentil score*. Dengan semua pilihan tersebut, berikut pada gambar 8 adalah hasil perbandingan 3 skenario:



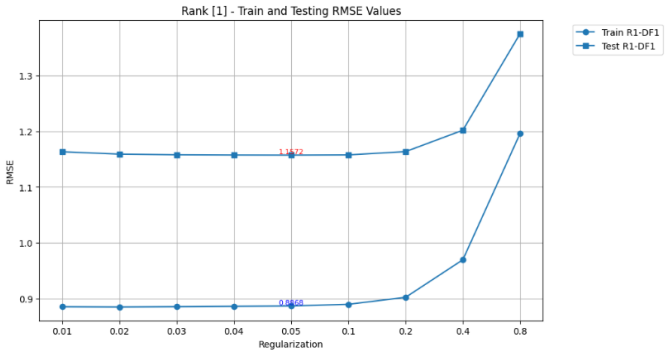
Gambar 8. Perbandingan Antara Pembagian Data

Dapat dilihat pada gambar 8 diatas, skenario yang memiliki nilai RMSE untuk data *test* dan *train* yang tinggi adalah skenario ketiga dengan menggunakan *masking distribution*, sedangkan skenario yang memiliki nilai RMSE data *test* dan *train* terendah adalah skenario pertama dengan metode sklearn *split*. Jika menelaah lebih dalam untuk skenario pertama dan kedua, maka dapat dilihat pada gambar 9 berikut:



Gambar 9. Perbandingan Sklearn *Split* (skenario 1) dan Pyspark *Split* (skenario 2)

Dapat dilihat pada gambar 9 diatas, untuk skenario 1 masih unggul untuk nilai RMSE pada data *test* yang berada pada nilai 1.1142 dibandingkan dengan nilai RMSE *test* skenario 2 yakni pada nilai 1.1572, akan tetapi untuk data train skenario 2 lebih unggul 0.005 dari skenario 1. Untuk skenario kedua, karena nilai regularisasi berada pada titik terendah, ada kemungkinan bahwa skenario 2 masih bisa mendapatkan nilai yang lebih rendah lagi, untuk itu, dilakukan proses trainning ulang dengan nilai regularisasi [0.01, 0.02, 0.03, 0.04, 0.05], berikut adalah hasilnya:



Gambar 10. Pelebaran Nilai Regularisasi dengan PySpark Split (skenario 2)

Dapat dilihat pada gambar 10 diatas, walau sudah dilakukan pelebaran nilai RMSE perbedaan nilai RMSE sangat kecil dengan nilai regularisasi 0.05, dengan ini nilai regularisasi 0.05 tetap menjadi yang paling rendah dari semua nilai regularisasi, jadi pemilihan nilai regularisasi untuk skenario 2 akan tetap 0.05. Dari seluruh perbandingan skenario jika dicari nilai selisih data *train* dan *test*, skenario pertama memiliki nilai 0.2269 sedangkan skenario kedua memiliki nilai 0.2704. Jadi secara selisih antara train dan test skenario 1 jauh lebih unggul sebesar 0.05.

3.4. Perbandingan Antara Algortima

Untuk melakukan perbandingan antara algortima ALS-MF dan SVD, berikut adalah rekapitulasi keseluruhan *error* model ALS pada tabel 2 berikut:

Tabel 2. Rekapitulasi RMSE ALS-MF

ALS							
Skenario		Rata-Rata		Minimum		Maximum	
		Train	Test	Train	Test	Train	Test
Min-Max Scaling	1	0.93	1.40	0.44	1.29	1.28	1.53
	2	0.93	1.41	0.44	1.30	1.28	1.54
	3	1.03	1.39	0.47	1.19	1.37	1.68

p-ISSN: 2301-5373
e-ISSN: 2654-5101

Percentil Score	1	0.85	1.24	0.40	1.14	1.19	1.35
	2	0.86	1.26	0.40	1.15	1.20	1.38
	3	0.91	1.36	0.41	1.16	1.24	1.57

Dapat dilhat pada tabel diatas, adalah keseluruhan rekapitulasi nilai RMSE pada model ALS-MF dengan nilai rata-rata, nilai minimum, dan nilai maksimum pada data uji dan data latih. Sedangkan untuk hasil rekapitulasi keseluruhan *error* model algortima SVD pada tabel 3 berikut:

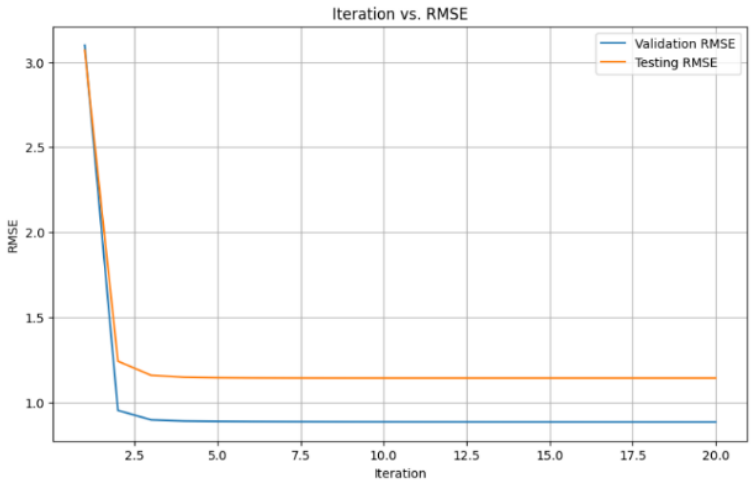
Tabel 3. Rekapitulasi RMSE SVD

SVD							
Skenario		Rata-Rata		Minimum		Maximum	
		Train	Test	Train	Test	Train	Test
Min-Max Scaling	1	3.81	4.25	6e-5	6e-5	4.23	4.54
	2	3.82	4.26	125e-6	8e-5	4.22	4.56
	3	3.62	4.47	3e-8	3e-15	4.14	4.66
Percentil Score	1	2.96	3.23	6e-5	3.20	3.23	3.46
	2	2.95	3.25	8e-5	3.23	3.22	3.48
	3	2.74	3.50	3e-15	3.49	3.07	3.65

Pemilihan model menjadi krusial untuk melihat model mana yang memiliki tingkat *error* terendah untuk dataset penelitian. Untuk perbandingan pertama, secara rata-rata keseluruhan baik metode *percentil score* maupun *min-max scaling* nilai RMSE pada model ALS mengungguli keseluruhan skenario 1, 2, dan 3 pada model SVD. Nilai rata-rata keseluruhan skenario dari semua metode *min-max scaling* dan *percentil score* didepatkan nilai RMSE ALS adalah 0.91 untuk data *train* dan 1.32 untuk data *test*. Sedangkan untuk model SVD nilai rata-rata keseluruhan skenario adalah 3.31 untuk data *train* dan 3.82 untuk data *test*. Jika melihat nilai *minimum* pada tabel 2 dan 3, secara keseluruhan nilai RMSE minimum model SVD unggul bahkan mendekati nilai 0 dibandingkan dengan model ALS, akan tetapi nilai RMSE untuk *maximum* nilai pada data *train* maupun *test* jauh lebih kecil model ALS dibandingkan model SVD. Jadi secara keseluruhan, model ALS lebih unggul dalam memproses data sehingga jarak antara data *train* dan *test* tidak terlalu besar jika dibandingkan dengan model SVD yang sangat *overviting* pada data *test* sehingga nilai RMSE pada data *test* sangat buruk.

3.4 Pemilihan Hasil Akhir Model

Setelah melakukan analisis perbandingan antara *hyperparameter*, metode konversi rating, metode pembagian data, dan perbandingan antara algoritm. Semua skenario diatas masih dilakukan pada iterasi dengan jumlah 5. Untuk menentukan jumlah iterasi lanjutan, maka akan dilakukan proses trainning lagi dengan model terbaik dan melihat jumlah iterasi yang cocok untuk model. Hasil dapat dilihat pada gambar 11 berikut:



Gambar 11. Pemilihan Model Terbaik terhadap Pertambahan Perulangan

Dapat dilihat pada gambar 11 diatas, untuk iterasi lebih dari 3 pengurangan nilai RMSE tidak terlalu signifikan sehingga penambahan jumlah iterasi mulai kelima dan selanjutnya tidak cukup berpengaruh dalam menurunkan nilai RMSE model. Dengan itu pemilihan jumlah iterasi terbaik masih tetap akan menggunakan 5 iterasi.

4. Kesimpuan

Berdasarkan hasil pemaparan penelitian yang telah dilakukan oleh penulis diatas, dapat diambil beberapa kesimpulan, sebagai berikut:

1. Dari hasil penelitian terutama dalam pengimplementasian model ALS dan model SVD, didapatkan bahwa dengan model faktorisasi matrik biasa dengan SVD mendapatkan nilai RMSE yang terbaik pada data *train* dan data *test* sebesar 2.95 dan 3.25. Sedangkan dengan melakukan implementasi ALS pada faktorisasi matrik dapat mengurangi atau mereduksi nilai RMSE yang terbaik pada data *train* dan *test* sebesar 0.85 dan 1.24. Selain itu, baik pada selisih nilai *minimum* dan maksimum keseluruhan RMSE didapatkan bahwa rentangan terkecil RMSE berada pada model ALS dengan nilai rentangannya konsisten pada data *train* dan *test* sebesar 0.8 dan 0.2, sedangkan untuk model SVD selisih nilai *minimum* dan maksimumnya pada data *train* dan *test* sebesar 3.3 dan 2.1.
2. Dari hasil penelitian yang dilakukan dengan menggunakan 3 skenario berbeda, didapat bahwa semakin tinggi rank maka nilai RMSE pada data *train* dan data *test* condong akan semakin meningkat, nilai terendah RMSE berada pada nilai 0.40 dan nilai tertingginya pada 1.19. Sedangkan untuk nilai regularisasi juga menunjukkan hal serupa. Lalu, untuk pemilihan jumlah iterasi, semakin tinggi jumlah iterasi tidak menghasilkan perbedaan yang signifikan setelah iterasi ke-5, dengan perbedaan rata-rata setelah iterasi kelima sebesar 0.0001. Jadi untuk hyperparameter akhir yang didapat adalah rank=1, regularisasi=0.1, dan iterasi=5. Dengan hasil akhir terbaik adalah nilai RMSE (*Root Mean Squared Error*) sebesar 1.1142 untuk data uji dan 0.8873 untuk data latih.

Referensi

[1] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep Learning based Recommender System: A Survey and New Perspectives” *ACM Computing Surveys*, vol. 52, no. 1, p. 1-38, 2019.

[2] R. Robert, “Video Game Recommendation System: Using Steam’s Data to Find and Recommend Similar Games”, 10 May 2021. [Online]. Available: https://medium.com/web-mining-is688-spring-2021/video-game-recommendation-system-b9bcb306bf16. [10 June 2024].

[3] S. Lou, “Introduction to Recommender System: Approaches of Collaborative Filtering: Nearest Neighborhood and Matrix Factorization”, 10 Desember 2018. [Online]. Available: https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26. [11 Juni 2024].

[4] A. Aziz, M. Fayyaz, “Comparison of Content Based and Collaborative Filtering in Recommendation Systems” in *Conference: International Conference on Multimedia Information Technology and Applications*, Vietnam, 2021, pp. 428-431.

[5] R.H. Mondi, A. Wijayanto, and Winarno, “Recommendation System With Content-Based Filtering Method For Culinary Tourism In Mangan Application” *ITSMART: Jurnal Ilmiah Teknologi dan Informasi*, vol. 8, no. 2, p. 65-72, 2019.

[6] Y.I. Lubis, D.J. Napitulu, and A.S. Dharma, “Implementation of Hybrid Filtering (Collaborative and Content-based) Methods for the Tourism Recommendation System” in *The 12th National Conference on Information Technology and Electrical Engineering (CITEE)*, Yogyakarta, 2020, pp. 28-35.

[7] S. Martin, J.I. Sitohang, and B. Jonathan, “Mesin Rekomendasi Menggunakan Algoritma Alternating Least Square (ALS) pada Goodreads” *Jurnal CoreIT*, vol. 6, no. 2, p. 79-84, 2020.

[8] J.E. Prayogo, A. Suharso, and A. Rizal, “Analisis Perbandingan Model Matrix Factorization dan K-Nearest Neighbor dalam Mesin Rekomendasi Collaborative Berbasis Prediksi Rating” *Jurnal Informatika Universitas Pamulang*, vol. 5, no. 4, p. 506-514, 2020.

[9] A.A. Amin, “Mereduksi Error Prediksi Pada Sistem Rekomendasi Menggunakan Pendekatan Colalborative Filtering Berbasis Model Matrix Factorization” *EXPLORE*, vol. 11, no. 2, p. 8-14, 2021.