Berkala Ilmu Perpustakaan dan Informasi, Vol. 14, No. 1, Juni 2018, Hal. 11-25
DOI: 10.22146/bip.32208
ISSN 1693-7740 (Print), ISSN 2477-0361 (Online)
Tersedia online di https://jurnal.ugm.ac.id/bip

Big data analytic untuk pembuatan rekomendasi koleksi film personal menggunakan *Mlib. Apache Spark*

Indah SurvyanaWahyudi[1]
[1]Sekolah Tinggi Energi dan Mineral-Akamigas
Email: indahsurvyana@gmail.com

Naskah diterima: 11 Januari 2018, direvisi: 24 Maret 2018, disetujui: 15 Mei 2018

ABSTRAK

Pendahuluan. Era digital ditandai ledakan informasi digital yang menimbulkan masalah dalam temu kembali informasi. Mesin pencari mempunyai keterbatasan pada kata kunci/*query* yang diingat pengguna. Sistem rekomendasi sebagai solusi memberikan informasi yang bersifat personal dengan mempelajari penilaian pengguna terhadap item yang pernah dipilihnya.
Metode penelitian. Pada kajian ini peneliti memberikan model mesin rekomendasi yang pengujiannya menggunakan dataset dari movielends.org dengan dua tahap *filtering*.
Data analisis. *Alternating Least Square-Weigh Regulation* (ALS-WR) sebagai algoritma *big data analytic* dalam memprediksi *ratting* dan meranking film berdasarkan *rating* yang telah diberikan pengguna dan *rating* seluruh pengguna (*collaborative filtering*) dan *Cosine Similiarity* sebagai *filter* kedua untuk mendekatkan item berdasarkan kemiripan *genre*.
Hasil dan Pembahasan. Hasilnya pada dataset 100K *Root Mean Squared Error* (RMSE) prediksi adalah 0.96 (validasi) sementara 0.94 (test). Dataset 1M, RMSE 0.86 (validasi) dan 0.96 (test). Dataset 10M nilai RMSE 0.81 (validasi) dan 0.81 (test). Hasil dari *cosine similarity* didapatkan nilai 1 untuk kemiripan 100%, nilai itu akan berkurang berdasarkan tingkat kemiripan suatu item. Uji penerimaan user didapatkan 28% hasil rekomendasi pertama dapat diterima *user*, nilai ini meningkat menjadi 62% tingkat penerimaan user terhadap rekomendasi kedua.
Kesimpulan dan Saran. Hasil akhir ternyata 75% responden lebih menyukai rekomendasi kedua yaitu hasil dari *filtering* dua tahap dibandingkan hanya satu tahap.

Kata Kunci: Mesin Rekomendasi; *Big Data Analytic*; ALS-WR; *Cosine Similarity*; *Data Mining*

ABSTRACT

Introduction. *The digital age is characterized by the explosion of digital information that creates problems in information retrieval. Search engines have a weakness in the keywords/queries that users can remember. Recommendations arise as solutions to provide personal information.*
Data Collection Method*. In this paper, the researcher presented a recommendation engine model using dataset from movielends.org.*
Analysis Data. *Alternating Least Square-Weight Regulation (ALS-WR) was used as a big data analytic algorithm in rating prediction and Cosine Similiarity as the second filter to bring items closer to the genre.*
Results and Discussions*.The results of Root Mean Squared Error (RMSE) from 100K datasets were 0.96 (validation) and 0.94 (test). The results RMSE from 1M dataset were 0.86 (validation) and 0.96 (test). The results RMSE from 10M dataset were 0.81 (validation) and 0.81 (test). The result cosine similarity was 1 for 100% resemblance and it decreased based on the similarity level. The user acceptance test was 28% user accepts the result of first recommendation, this value increased to 62% acceptance level of the user against the second recommendation.*

***Conclusions**. The final results show that 75% of respondents prefer the second recommendation from two-stage filtering than just collaborative filtering.*

***Keywords:** Recommendation Engine; Big Data Analytic; ALS-WR; Cosine Similarity; Data Mining*

A. PENDAHULUAN

Film adalah gambar hidup (KBBI-Online, 2018). Film disebut juga sinema yang bersumber dari kata *kinematic* atau gerak (Joseph, 2011). Dalam perpustakaan film digolongkan ke dalam GMD (*General Material Designation) motion picture* atau gambar bergerak. Dalam sejarahnya, film menjadi media yang sangat berpengaruh jika dibandingkan dengan media-media yang lain karena unsur-unsur film dalam bentuk audio dan visual dapat memberikan kekuatan universal komunikasi. Pesan-pesan komunikasi terwujud dalam cerita dan misi yang dibawa dalam bentuk drama, *action*, komedi, dan horor. Besarnya kekuatan film kemudian melahirkan industri film yang menjadikan film sebagai komuditas yang mempunyai harga jual dan konsumen (pengguna film). Usaha peredaran film untuk mencapai konsumennya mengalami berbagai perubahan menyesuaikan perkembangan teknologi komunikasi dan informasi dalam menyimpan dan menyebarluaskan informasi. Era informasi membawa film disimpan dalam tempat-tempat yang lebih praktis dalam bentuk kepingan CD dan DVD sehingga pengguna dapat dengan mudah membawanya.

Di era digital ini, paradigma dan kebutuhan masyarakat dunia mulai bergeser dari informasi yang dibatasi oleh lembaran kertas, kepingan DVD atau CD kini menjadi informasi dalam bentuk digital. Internet menghidupkan sumber informasi digital dengan kemudahan akses dari berbagai perangkat teknologi informasi dan komunikasi secara cerdas, praktis dan terintegrasi. Terjadilah ledakan data dan informasi digital di mana sosial media, proses bisnis, publikasi-publikasi digital berperan meningkatkan volume dan pertumbuhan informasi digital. Pada bisnis film sendiri, tingginya produksi film secara global dan tingginya minat masyarakat akan film yang mendasari dibuatnya *Internet Movie Database (IMDb)*. *Database* ini meliputi katalog perfilman yang memuat semua informasi mengenai film yang dimaksud, seperti judul, sutradara, aktor, *genre*, sinopsis dan *rating*. Pada IMDb, masyarakat diberikan kemudahan untuk dapat mencari film berdasarkan kata kunci yang diinputkan oleh pengguna dengan mudah tanpa harus mencari secara manual layaknya di media cetak. Selain itu, masyarakat dapat melihat berbagai film baik terbaru maupun lama sewaktu-waktu dan dimanapun dengan mengakses internet. Namun, pertumbuhan jumlah film di seluruh dunia yang begitu cepat memicu pertumbuhan database yang semakin cepat. Database kemudian menjadi *big data*. Sebuah istilah yang cukup sering terdengar dan menjadi masalah di era digital ini.

Big data mempunyai karakteristik 3V yaitu yaitu *Volume*, *Variety* dan *Velocity*. *Volume* menandakan bahwa ukuran dan kapasitas data akan terus tumbuh dan bertambah seiring dengan pertambahan waktu. *Variety* adalah keragaman, yang berarti bahwa kategori yang dimiliki *big data* dan juga fakta yang sangat penting untuk dianalisis. *Velocity* adalah kecepatan, yang mengacu pada kecepatan data atau seberapa cepat data dihasilkan dan diproses untuk memenuhi tuntutan dan tantangan pertumbuhan dan perkembangan. Keterbatasan *big data* karena *set* data yang besar. *Data set big data* tumbuh dengan cepat dan saling terkait dan menjadi masalah untuk menemukan apa yang sebenarnya dicari pengguna. Mesin pencari memecahkan sebagian masalah itu, namun personalisasi informasi tidak diberikan, sementara sistem *recommender* adalah alat untuk penyaringan dan pemilahan item dan informasi yang didasarkan perilaku pengguna, profil pengguna atau *preference* pendapat dari komunitas pengguna untuk membantu individu dalam mengidentifikasi konten yang menarik dan berpotensi besar untuk dipilih, dibeli atau digunakan (Asnov, 2011). Disinilah era digital

membawa era *big data* meninggalkan popularitas mesin pencari menuju mesin rekomendasi (Anderson, 2008).

Kemampuan analisis *big data* diyakini mampu membantu untuk mengatasi ragam persoalan, salah satunya adalah bagaimana menghadirkan rekomendasi yang sesuai dengan karakter dan minat pelanggan (Kadam, 2017). *E-commerce* dan penyedia layanan jasa berbasis internet berlomba-lomba memperbaharui dan meningkatkan kualitas mesin rekomendasinya agar dapat menghasilkan rekomendasi terbaik bagi pengguna dan dapat mempengaruhi pengguna untuk membeli dan mengambil produk tersebut. Netflix saalah satu perusahaan rental film terbesar di dunia (Price, 2015). Fakta mengatakan 75% *subscriber* Netflix didapatkan dari *recommendation engine* (Team, 2012), 89% pendapatan google diperoleh dari personal *advertise* yang diperoleh dari *recomendation engine* dan pendapatan Amazon naik 29% berkat *recomendation engine* (Morrison, 2016). Netflix mempekerjakan 300 orang dan menghabiskan 150 juta dolar pertahun khusus untuk memelihara dan meningkatkan kualitas mesin rekomendasi (Roettgers, 2014).

Dari beberapa metode populer yang digunakan dalam membuat personal rekomendasi yaitu *content based filtering*, *collaborative filtering* dan *hybrid*. *Content based filtering* memanfaatkan interaksi antara konten–item dengan profil pengguna (Ricci, 2011), di mana yang termasuk *content* item *di sini* seperti *genre*, *author*, dll. *Collaboratif filtering* (CF) bekerja dengan membangun *database* preferensi untuk produk oleh konsumen. Seperti promosi dari mulut ke mulut, *collaborative filtering* memberikan prediksi *rating* dan personal rekomendasi berdasarkan yang disukai pengguna lain (tetangga) yang mempunyai selera yang sama (Walker, 2003). Sementara *hybrid* merupakan penggabungan *content based filtering* dan *collaborative filtering*.

Di antara ketiga metode di atas, perusahaan-perusahaan *e-commerce* raksasa seperti Amazon, Netflix, Pandora dan beberapa *e-commerce* menggunakan metode *collaborative filtering* sebagai representasi dari *wisdow of crowd* (Levitin, 2015). Bahkan Netflix mempekerjakan 300 orang dan menghabiskan 150 juta dolar per tahun khusus untuk memelihara dan meningkatkan kualitas mesin rekomendasi mereka (J.Roettgers, 2014). Namun ada tiga permasalahan yang besar dari *collaboratif filtering*, masalah yang pertama adalah data *sparsity* di mana banyak sistem *recommender* komersial didasarkan pada *dataset* besar, akibatnya, matriks *user*-item yang digunakan untuk penyaringan kolaboratif bisa sangat besar dan jarang, ini menjadi tantangan dalam hasil rekomendasi dan ini berakibat pada jumlah *error* dalam perhitungan matriks. Masalah yang kedua adalah skalabilitas yaitu besarnya data yang akan berpengaruh pada proses jalannya perhitungan untuk menghasilkan rekomendasi. Masalah ketiga adalah *cold start problem* di mana pengguna baru dan item baru mulai memasuki sistem, karena *collaborative filtering* bekerja berdasarkan interaksi antara pengguna dan item (Khoshgoftar, 2009).

Hasil kajian menjelaskan sebuah model mesin rekomendasi film untuk pengguna baru yang dapat mengatasi ketiga masalah di atas dengan menggunakan algoritma *alternating least square-weight regularization* untuk metode *collaborative filtering* sebagai *big data analytic* dalam memprediksi *rating* dan membangun personal rekomendasi serta menggunakan mesin pembelajaran *big data* yaitu *M.lib apache spark*. Pada penelitian ini penulis menggunakan *M.lib apache spark single alone* tanpa *hadoop*, berbeda dengan penelitian sebelumnya yang menggunakan *hadoop* yang membutuhkan kapasitas penyimpanan yang besar dan *memory* yang juga besar untuk menjalankan mesin. Selain itu pada penelitian ini walaupun pemrograman dalam *spark* menggunakan bahasa *phyton*, namun *interface* dibuat di PHP biasa sehingga dapat diaplikasikan di web yang berbasis PHP. *Apache Spark* diketahui sebagai *tools big data* yang mempunyai kecepatan 1000 kali lebih cepat daripada *hadoop* (Jonnalagadda, 2016) dan *Mlib. Spark* adalah mesin pembelajaran dari *Apache Spark* dengan *library* algoritma yang cukup lengkap untuk proses *analytic*. Penelitian

sebelumnya mengenai *collaborative filtering* menggunakan algoritma *pearson correlation* (Bobadilla, 2010), k-Nearest Neighbors (k-NN) atau menggunakan *Singular Value Decomposition* SVD (Pr¨ugel-Bennett, 2010) untuk faktorisasi matriks, akan tetapi *Root Mean Square Error* (RMSE) masih tinggi, namun telah terdapat penelitian yang membandingkan skalabilitas antara kNN dan *Alternating Least Square* (ALS) dengan menggunakan bahasa dan mesin pembelajaran *Scala*, hasilnya tingkat akurasi dalam memprediksi dan *runtime* ALS jauh lebih baik dari pada k-NN (Ondra, 2014). Untuk meningkatkan kualitas *collaborative filtering* dan untuk mengatasi *cold start problem* terutama pada item baru digunakan metode *content boost filtering* (Melville, 2002), pada penelitian lainnya menggunakan *content similiarity based on collaborative filtering* (Phuong, 2014). Oleh karena itu peneliti mengunakan pendapat (Phuong, 2014) namun dengan algoritma yang berbeda di mana pada penelitian sebelumnya menggunakan KNN, pada penelitian ini menggunakan algoritma *analytic* ALS dengan *Wheight Regularization* (ALS-WR), kemudian hasil dari ALS-WR di filter kembali dengan *Cosine Similiarity*.

B. TINJAUAN PUSTAKA

Collaborative filtering bekerja berdasarkan interaksi antara pengguna dan item di mana interaksi keduanya kemudian menghasilkan sebuah profil yang dapat diolah untuk menghasilkan personal rekomendasi untuk pengguna tersebut. Terdapat dua pendekatan dalam pengambilan data, yaitu pendekatan *implicit* dan *explicit*. Pendekatan *implicit*, artinya, sistem yang menyimpan dan mempelajari perilaku pengguna terhadap item, contohnya item apa yang pernah dibeli/dipinjam pengguna, berapa kali penggula melihat barang/mengklik item tersebut, dsb. Sementara pendekatan *explicit*, yaitu dengan menanyakan kepada pengguna secara langsung deskripsi item yang bagaimana yang ia sukai/minati, contoh keluarannya berupa rating atau kuesioner (Gawesh Jawaheer, 2010).

Penelitian ini menggunakan metode *collaborative filtering* mengambil data eksplisit dari *rating*. *Rating* yang dimaksud *di sini* adalah penilaian yang diberikan pengguna terhadap item yang telah dibeli atau dilihat, lalu kemudian mesin pembelajaran dengan algoritmanya akan mengolah rating untuk membangun sebuah rekomendasi untuk *user* tertentu terutama untuk pengguna baru yang belum memasukkan banyak penilaian. Untuk itu diperlukan model korelasi antar item untuk memprediksi peringkat item seperti pada konsep *collaborative filtering* (*word to mouth*), di mana item yang memperoleh banyak penilaian akan menjadi rekomendasi utama, namun *Alternating Least Square –Weight Regularization* (ALS-WR) mencoba bersikap adil dengan memprediksi rating untuk semua item meskipun tidak semua pengguna memberikan penilaian untuk seluruh item. Akan tetapi masalah lain muncul dari hasil *collaborative filtering* ini, bahwa terkadang pengguna masih menginginkan item berdasarkan kontennya terutama dari *genre* atau subjek. Oleh karena itu, untuk meningkatkan kualitas penyaringan kolaboratif, korelasi antara konten masih diperlukan misalnya film memiliki informasi *genre* yang diberikan oleh para ilmuwan dan sutradara. Terkadang, kategori informasi ini lebih disukai daripada rekomendasi berdasarkan penilaian kolaboratif (Sang-Min Choi, 2012) (Song, 2009).

Tahap pertama pengguna baru memberikan *rating*, kemudian proses *collaborative filtering* kemudian secara *temporary* rating baru tersebut tersimpan di *database*, kemudian mesin *learning* mengolah *preference* tersebut dengan membandingkan dengan *database* yang telah ada. Tahap pertama selesai menghasilkan daftar rekomendasi, kemudian daftar rekomendasi tersebut tersimpan secara *temporary* untuk pembuatan rekomendasi pada tahap berikutnya. Di halaman *interface*, pengguna yang telah mendapatkan hasil rekomendasi kemudian dapat memilih/mengklik item yang disukainya untuk melihat detail item tersebut, dari proses ini menghasilkan data implisit yang menjadi dasar pembuatan rekomendasi tahap kedua. Proses yang terjadi pada tahap kedua ini yaitu daftar rekomendasi dari proses *collaborative filtering* tadi kemudian di-*filter*

kembali berdasarkan *genre*-nya dengan *prefference* baru yaitu item yang baru saja dipilih. Melalui proses *genre similiarity*, pengguna mendapatkan rekomendasi kedua.

Pada Gambar 1. evaluasi yang dilakukan terdiri dari 3 yaitu Evaluasi Model dari algoritma ALS-WR yang menguji tingkat *error* atau akurasi dalam memprediksi *rating*. Evaluasi yang ke dua adalah Evaluasi Presisi yang menguji kemiripan *genre* berdasarkan item film yang dipilih pengguna dan evaluasi terakhir adalah Evaluasi Penerimaan Pengguna (*User Acceptance Test*) untuk menguji kualitas dari model mesin rekomendasi ini.

Pada tahap *collaboration filtering*, algoritma yang digunakan adalah ALS-WR salah satu jenis algoritma untuk faktorisasi dalam matriks yang memperbaiki algoritma *Singular Value Decomposition* (SVD), *K-Nearst Neightboard* (KNN) dan *stochastic gradient descent* (SGD). Sementara untuk rekomendasi kedua menggunakan Tf-Idf dan *Cosine Similiarity* untuk mencari kemiripan *genre* dari film yang dipilih pengguna.

1. Algoritma Mesin Rekomendasi *Alternating Least Square - Weight Regularization (ALS-WR)* untuk *Collaborative Filtering*

Konsep dasar ALS-WR adalah matriks besar dari interaksi pengguna dan item dan mengetahui fitur laten (atau disembunyikan) yang berhubungan antara pengguna dan item. Berdasarkan Gambar 2, matriks besar tersebut kemudian dipecah ke dalam matriks jauh lebih kecil. Matriks preferensi R diasumsikan dapat difaktorisasi (dipecah) menjadi 2 matriks yang lebih kecil. Jika dimensi matriks preferensi = X × Y, di mana X a$X \in \mathbb{R}^{m x f}$an Y adalah ⊣item maka h$Y \in \mathbb{R}^{f x n}$sasinya adalah matriks X dengan sebutan b dan matriks Y dengan di mana *f* adalah *rank* atau faktor laten dari faktorisasi ini yang mempengaruhi X dan Y. Hasil kali baris X dan kolom Y menyatakan kesesuaian pengguna X dengan konten Y. Namun, pada kebanyakan *database ecommerce*, rental film atau perpustakaan, beberapa pengguna cenderung hanya memberikan rating pada item yang sudah dibelinya, dipinjam atau dilihatnya, akibatnya matriks Q tidak seluruhnya terisi, di sini ALS bekerja yaitu memprediksi faktor laten tersebut dengan W atau *weight* untuk identifikasi dengan cara :

$$w_{ui} = \begin{cases} 0 & \text{if } q_{ui} = 0 \\ 1 & \text{else} \end{cases}$$

(Nee, 2016)

Karena pada prinsipnya algoritma ini hanya memprediksi nilai untuk item yang telah dinilai oleh pengguna sebelumnya, tidak pada item yang belum dinilai sama sekali, sesuai dengan prinsip *collaborative filtering* adalah *wisdom crowd* atau penilaian yang bergantung pada penilaian publik, seperti sistem pemasaran dari mulut ke mulut (*word to mouth*) yang telah dijelaskan pada bagian pendahulan. Amazon, yahoo maupun netflix mempunyai trik tersendiri untuk mengatasi masalah item baru (*cold start problem*) yaitu dengan men-*display* item baru di halaman depan web, berharap pengguna dapat segera melihat, membeli dan memberikan penilain terhadap item tersebut.

Lebih lanjut, selain memiliki *weight*, algoritma ini juga memiliki lamda (λ) yang berfungsi sebagai paramater regulasi agar rekomendasi yang dihasilkan mengarah kepada item yang bernilai baik sesuai dengan preferensi pengguna. Maka dari penjelasan dan logika di atas didapatkan fungsi sebagai berikut:

$$J(x_u) = (q_u - x_u Y) W_u (q_u - x_u Y)^T + \lambda x_u x_u^T$$

$$J(y_i) = (q_i - X y_i) W_i (q_i - X y_i)^T + \lambda y_i y_i^T$$

(Nee, 2016)

Untuk mendapatkan hasil dari persamaan di atas, maka paramater regulasi harus disetel menggunakan *cross-validation* agar algoritma dapat di-*generate* lebih baik. Solusinya untuk nilai yang hilang adalah dengan menyelesaikan fungsi di bawah ini:

$$x_u = (Y W_u Y^T + \lambda I)^{-1} Y W_u q_u$$

$$y_i = (X^T W i X + \lambda I)^{-1} X^T W_i q_i$$

(Nee, 2016)

Melalui iterasi pada mesin pembelajaran, proses tersebut dilakukan berulang-ulang hingga mencapai *error* terkecil atau titik konvergen. Hasil dari iterasi kemudian menjadi regulasi yang akan digunakan untuk mengisi data yang hilang, kemudian *rating* dapat diurutkan untuk menghasilkan Top-N yang menjadi dasar pembuatan rekomendasi personal. Setelah Top-N dihasilkan selanjutnya adalah pembuatan rekomendasi personal yaitu rekomendasi yang disusun berdasarkan personal pengguna, dalam penelitian ini adalah pengguna baru yang profilnya belum terdapat di *datasets*. Mesin pembelajaran kemudian akan menambahkan rating yang diberikan oleh pengguna baru tersebut untuk kemudian dibuatkan prediksi kembali dengan menggunakan *best* model hasil *training* Top-N, kemudian hasilnya diurutkan kembali menjadi Top-N yang sifatnya personal (*personal* Top-N *Recommendation*).

Cosine Similiarity untuk Kemiripan *Genre*

Algoritma yang digunakan untuk rekomendasi ke dua adalah *Cosine Similiary* di mana algoritma menghitung kemiripan antara suatu *Query* (Q) dengan daftar dokumen (dengan semua dokumen). Kemudian dilakukan pengurutan dan dikembalikan kepada pengguna. Kemiripan antar dokumen *Cosine Similarity* rumusnya adalah sebagai berikut (ˆ) :

$$Sim\left(\underset{d_j}{\rightarrow}.\underset{q}{\rightarrow}\right) = \frac{\sum_{i=1}^{t}(w_{ij} \times w_{iq})}{\sqrt{\sum_{i=1}^{t}(w_{ij})^2} \times \sqrt{\sum_{i=1}^{t}(w_{iq})^2}}$$

(Garcia, 2016)

Langkah dalam menghitung rumus *Cosine Similarity* adalah (a) hitung hasil perkalian *scalar* antara Q dan data uji, hasilnya perkalian dari setiap dokumen dengan Q dijumlahkan (sesuai pembilang pada rumus di atas). (b) hitung panjang setiap dokumen, termasuk Q. Caranya dengan mengkuadratkan bobot setiap *term* dalam setiap dokumen, jumlahkan nilai kuadrat dan terakhir diakarkan. Untuk menghitung bobot digunakan *Term Frequency-Inverse Document Frequency* (Tf-Idf) dengan tahapan penghitungan sebagai berikut :

1. menghitung *Term Frequency* (tf), dengan persamaan
   tf = tfij (Garcia, 2016)
2. menghitung *Inverse Document Frequency* (idf), dengan persamaan
   idfi =log (N/dfj) (Garcia, 2016)
3. menghitung bobot tiap term, dengan persamaan
   $Wij = tfij.\log\frac{N}{dfi}$
   (Garcia, 2016)

Pada penelitian ini *Query* yang dituju adalah *genre* item yang di klik oleh pengguna, misalnya pengguna mengklik film *American Beauty* yang ber-*genre Romance* and *commedy*, maka sistem akan mencari film dengan *genre* serupa kemudian dilakukan pembobotan untuk pembuatan peringkatnya, kemudian 10 daftar dari proses tersebut kemudian direkomendasikan ke pengguna menjadi Rekomendasi ke-2 dalam sistem ini.

C. METODE PENELITIAN

Dataset yang diambil untuk penelitian ini berasal dari *dataset movielends*. Peneliti mengambil 3 *dataset* berukuran 100K (10.681 *movies*, 10.004 *ratings*, 671 *user*), 1M (3952 *movies*, 1.000.209 *rating*, 6040 *user*) dan 10M (10.681 *movies*, 71.567 *user*, 10.000.054 *rating*) di mana masing-masing *dataset* memiliki format '.dat' artinya data dipisahkan dengan dua titik (::) dengan struktur sebagai berikut :

a. movie.dat :MovieID::Title::Genres
b. rating.dat:UserID::MovieID::Tag::Timestamp.
c. User.dat:UserID::Gender::Age::Occupation::Zip-code

Recommender engine dibangun menggunakan mesin *learning MŁLib Spark* dari *Apache Spark*. *Spark* sendiri merupakan salah satu proyek *Apache* yang disebut dengan "*Lightning Fast Cluster Computing*". Kecepatan dalam meng-*cluster* yang bisa dihasilkan adalah 100 kali lebih cepat dari pada *hadoop* dan 10 kalilebih cepat dari pada *Mapreduce* pada memori (McDonald, 2015). Keunggulan *Spark* dalam perhitungan berulang ini yang menyebabkan *MLlib Spark* berlari cepat. *MLlib* berisi algoritma berkualitas tinggi yang memanfaatkan iterasi, dan dapat

menghasilkan hasil yang lebih baik daripada *MapReduce*. Desain dan cara kerja *MLlib* yang sederhana memungkinkan menjalankan berbagai algoritma pada *Resilent Distributed Dataset* (RDD. Setiap dataset di RDD dibagi menjadi partisi logis, yang dapat dihitung pada *node* yang berbeda dari *cluster*. RDD dapat berisi jenis *Python*, *Java*, atau *Scala* benda, termasuk kelas yang ditetapkan pengguna. Untuk menggunakan *MLib* dimulai dengan RDD *string* yang mewakili pesan, kemudian jalankan algoritma yang terdapat dalam fitur *MLib*, setelah mendapatkan hasil kemudian evaluasi model pada dataset uji menggunakan salah satu fungsi evaluasi/validasi *MLib* ini.

Spark dalam menjalankan algoritmanya, mendistribusikan proses kerja secara pararel. Terdapat *Driver* Program, *Cluter Manager* dan *workers*. *Spark* menjalankan aplikasi secara independen untuk memproses di sebuah kluster yang dikoordinasikan oleh *driver program*. Kemudian *driver program* yang terkoneksi dengan beberapa *cluster manager* akan mengalokasikan sumber daya ke seluruh aplikasi. Setelah terhubung, *Spark* membutuhkan eksekutor yaitu *workers* pada *node* yang akan memproses jalannya komputasi (menggunakan bahasa *phyton*, *java*, *scala*) secara pararel dan menyimpan data.

Model mesin rekomendasi pada penelitian ini dibangun dalam mesin pembelajaran *MLib Spark* berdasarkan data eksplisit dan *feedback* implisit yang diberikan *user* film yang telah ditontonnya. Dalam penelitian ini yang menggunakan metode *collaborative filtering* mengambil data eksplisit dari *rating*. *Rating* yang dimaksud di sini adalah penilaian yang diberikan *user* terhadap item yang telah dibeli atau dilihat, lalu kemudian mesin pembelajaran dengan algoritma ALS-WR akan melalukan *analytic* dengan mengolah *rating* tersebut untuk mencari hubungan antar item berdasarkan tabel *rating* untuk membentuk sebuah rekomendasi. Proses *analytic* dalam *MLib Spark* dengan menggunakan algoritma ALS WR dan *cosine similiarity* terdapat dalam Gambar 3.

Pada Gambar 3. menunjukkan diagram alir pembangunan mesin rekomendasi dalam menghasilkan rekomendasi menggunakan algoritma ALS-WR. Pada proses awal *collaborative filtering*, yaitu input data ke *Spark*, kemudian data akan masuk ke RDD.

RDD berfungsi sebagai media pennyimpanan dan pembelajaran pada mesin *learning spark*. Pada RDD, *manager* konteks diciptakan, hal tersebut berguna agar proses kerja dapat dilaksanakan secara pararel sehingga waktu yang dibutuhkan relatif cepat karena dalam penciptaan model nantinya iterasi akan dilakukan berulang-ulang untuk mencapai titik konvergen. Setelah manajer konteks diciptakan, *dataset* dibagi 3 (*spare set*) untuk 60% data untuk *train*, 20% *dataset* untuk validasi dan dan 20% *dataset* untuk *test*. Pembagian *dataset* tersebut secara random. Kemudian dilakukan proses *training* dilakukan yaitu pertama dengan memetakan data set ke dalam matriks *user*-item seperti pada Gambar 2. Beberapa paramater dimasukkan yaitu beberapa paramater *rank* (w) dan lambda (λ). Dalam algoritma ALS-WR, matriks *user* item dipecah menjadi dua yaitu matriks X yang berukuran *mxf* dan matriks Y yang berukuran *fxn*.

Dengan menggunakan persamaan *least square* akan membuat *multiple* kombinasi dengan beberapa parameter tersebut *rank* dan lambda. Hasilnya kemudian di validasi dengan data validasi untuk menghasilkan RMSE terkecil. *Best model* yang didapatkan kemudian di teskan kembali ke *data set* untuk membuktikan tidak ada *overfitting* dari proses tersebut. *Best model* kemudian digunakan untuk menghasilkan prediksi rekomendasi. Prediksi ini menghasilkan prediksi *rating* untuk setiap item untuk mengisi sparsitas pada *dataset* dengan demikian semua item mendapatkan *rating* dari semua user. Kemudian setiap item total jumlah ratingnya dan diurutkan/diranking dari item yang memiliki jumlah *rating* terbesar hingga terkecil. Daftar rekomendasi ini kemudian menghasilkan daftar rekomendasi film berdasarkan *rating* terbanyak atau yang disebut TopN. Setelah TopN dihasilkan selanjutnya adalah pembuatan rekomendasi personal yaitu rekomendasi yang disusun berdasar *personal user* dalam penelitian ini adalah *user* baru yang profilnya belum terdapat di *datasets*. Mesin pembelajaran kemudian akan

menambahkan *rating* yang diberikan oleh *user* baru tersebut untuk kemudian dibuatkan prediksi kembali dengan menggunakan *best model* hasil training tadi lalu hasilnya diurutkan kembali menjadi Top-N yang sifatnya personal (*personal Top-N Recommendation*). Setelah hasil rekomendasi pertama berdasarkan *collaborative filtering* disimpan, di halaman *interface*, *user* yang telah mendapatkan hasil rekomendasi kemudian dapat memilih/mengklik item yang disukainya untuk melihat detail item tersebut, dari proses ini menghasilkan data implisit yang menjadi dasar pembuatan rekomendasi tahap kedua.

Pada pembuatan rekomendasi tahap ke-2, di dalam *MLib Spark* yang terjadi sebelum pencarian kemiripan terlebih dahulu dilakukan *pre prosesing* yaitu seluruh data akan diubah semua tulisan menjadi huruf kecil yang disebut dengan istilah *case folding*. Setelah proses *case folding* yaitu *tokenizing*, pada proses ini dihapus semua karakter seperti angka, tanda baca, *symbol* dan memisahkan kalimat menjadi kata atau disebut juga dengan *parsing*.

Setelah melakukan tahap *pre-processing* adalah proses pembobotan dengan menghitung frekuensi kemunculan *genre* dalam satu film dan seluruh seluruh daftar film yang dijadikan data uji serta data itu sendiri yang akan dklasifikasikan. Film diibaratkan sebagai dokumen, satu film adalah 1 dokumen, dengan demikian *pre-processing* adalah proses pembobotan dengan menghitung frekuensi kemunculan *genre* dalam satu dokumen dan seluruh daftar dokumen yang dijadikan data uji serta data itu sendiri yang akan dklasifikasikan. Rumus dari proses ini adalah "*Wd*t = *tf.idf*", di mana "*idf=Log(n/df)*", n adalah jumlah data uji ditambah data Q, tf adalah kata dasar yang muncul dalam satu data, dan *df* adalah jumlah kata dasar yang muncul dalam satu . Pada contoh ini *Query* adalah genre dari film *jurastic park* yaitu *Adventure* dan *Action*.

Contoh perhitungan ini jika *query* adalah genre dari film *jurastic park* yaitu *Adventure* dan *Action*.

Dokumen 1 (D1) *Star Trek*. *Genre* : *war*, *action*, *science-fi*, *adventure*

Tf untuk *term Action* =1

Df yaitu kemunculan kata *action* untuk seluruh dokumen adalah 5 dari n yaitu total seluruh term dari seluruh dokumen.

Nilai rata frekuensi untuk kata *action* (n/df) dengan demikian adalah: 7/5=1.40. Terkadang suatu term muncul di hampir sebagian besar dokumen mengakibatkan proses pencarian *term* unik terganggu. Idf berfungsi mengurangi bobot suatu *term* jika kemunculannya banyak tersebar di seluruh koleksi dokumen kita. Rumusnya adalah dengan *inverse document frequency*. *Document frequency* adalah seberapa banyak suatu *term* muncul di seluruh *document* yang diselidiki. Rumus Idf adalah Logn/df dengan demikian cara menghitungnya adalah Log 7/5 =0.146. Log atau logarritmik diperlukan untuk mengurangi besarnya bilangan, di mana logaritmik suatu bilangan akan mengurangi digit jumlah, contoh 1000 dengan log (1000) hanya menghasilkan angka tiga. Pembobotan Tf-Idf (Wdt) adalah perkalian Tf x Idf. Untuk Dokumen 1 nilai Wdt nya adalah 0.14 x 1 = 0.146. Dengan demikian bobot tTf-Idf kata *action*.

Cosine Similarity adalah menghitung tingkat kemiripan *vector* data Q dengan data uji. Langkah dalam menghitung rumus *Cosine Similarity* adalah dengan menghitung hasil perkalian *scalar* antara Q dan data uji. Hasilnya perkalian dari setiap film dengan Q dijumlahkan (sesuai pembilang pada rumus *cosine similarity*) hitung panjang setiap dokumen, termasuk Q. Caranya kuadratkan bobot setiap *term* dalam setiap dokumen, jumlahkan nilai kuadrat dan terakhir akarkan.

Pembuatan *Interface*

Dalam penelitian ini, *Spark* dan *Mlib Spark* 2.1.0 di instal di ubuntu 16.10 pada komputer dengan prosessor *core* i3 yang memiliki RAM 8GB. Algoritma ditulis pada *phycharm* dengan bahasa *phyton*.

Terlebih dahulu *engine rekomender* yang dibuat yaitu untuk menjalankan ALS-WR. *Spark* dan *MLib* spark terlebih dahulu dijalankan pertama kali di terminal, setelah Top N *recommendation* berhasil didapatkan kemudian disimpan di SQL, karena *Spark* sendiri memiliki API yang dapat terhubung

dengan SQL. Setelah itu *interface* diciptakan agar pengguna bisa melakukan *rating*. Dengan menggunakan php shell, php dapat terhubung dengan *Mlib Spark*, jadi setelah pengguna selesai me-*rating* dan mengklik tombol "*run*" secara otomatis spark berjalan untuk memproses rekomendasi I dengan algoritma ALS-WR. Setelah daftar rekomendasi I keluar, pengguna dapat meng-klik salah satu judul film kemudian proses *cosine similiarity* berjalan kemudian daftar rekomendasi II dapat keluar.

Uji Validasi, Uji Presisi dan Uji Penerimaan User.

Uji validasi bertujuan untuk menemukan parameter terbaik dari suatu model yang dilakukan dengan cara menguji besarnya *error* pada data *testing*. Hasil dari validasi menunjukkan tingkat akurasi sebuah model. Uji validasi ini untuk mengitung *error* yang dihasilkan dari proses ALS-WR.

Root Mean Squared Error (RMSE) menjadi alternatif yang lebih intuitif dibandingkan *Mean Absolute Error* (MSE) karena memiliki skala pengukuran yang sama dengan data yang sedang dievaluasi. RMSE didapatkan rumus:

$$\text{RMSE}= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(fi - yi)^2} \qquad \text{(Nee, 2016)}$$

Di mana *fi* adalah nilai hasil peramalan, yi adalah nilai sebenarnya, dan n adalah jumlah data. RMSE memberikan bobot yang lebih besar jika dibandingkan dengan MSE, yakni nilai akar kuadratik dari *error*. Sebagai contoh, dua kali nilai RMSE artinya model memiliki *error* dua kali lebih besar dari sebelumnya. Sedangkan dua kali nilai MSE tidak berarti demikian. Jika MSE dapat dianalogikan sebagai varian, maka RMSE dapat dianalogikan sebagai standar deviasi.

Presisi merupakan salah satu pengujian dasar dan paling sering digunakan dalam penentuan efektifitas *information retrival system* maupun *recommendation siystem*. Presisi juga dapat digunakan untuk evaluasi kualitas hasil dari rekomendasi jika hasil prediksi diberikan *filter* baru dengan metode klasifikasi atau kemiripan dengan suatu konten. Untuk mengetahui kualitas hasil rekomendasi, dapat menggunakan rumus relevansi presisi yang membandingkan antara item yang relevan dengan total item yang dihasilkan atau yang direkomendasikan kepada *user*.

$$\textit{Precission}= \frac{\textit{relevant item retrieved}}{\textit{retrieved item}} \qquad \text{(Garcia, 2016)}$$

Di mana *relevant item retrieved* adalah jumlah item relevan yang terpanggil, dan *retrieved item* adalah jumlah seluruh item relevan yang yang ditampilkan sebagai hasil rekomendasi. Suatu item dikatakan sangat presisi jika nilainya adalah 1 dan tidak presisi apabila nilainya menjauhi angka 1. *Threshold value* atau ambang batas menjadi penentu dari jumlah item yang ditemukan, semakin tinggi *threshold* maka jumlah item yang ditemukan sedikit. Semakin rendah *threshold* maka semakin banyak item yang ditemukan namun memiliki presisi yang rendah.

Uji Penerimaan Pengguna atau *User Acceptance Test* (UAT) untuk menguji tingkat penerimaan pengguna terhadap hasil rekomendasi. Pengguna yang mencoba *engine* ini berjumlah 20 orang. Pengguna yang mengisi kuesioner ini dipilih secara acak dan tidak ada kriteria khusus, responden berkisar usia 17-32 tahun yang kebetulan hampir seluruhnya adalah mahasiswa Teknik Elektro dari ITS. Pemilihan jumlah responden 20 orang agar diperoleh distribusi nilai hasil pengukuran mendekati normal (Jogiyanto, 2008). Pada percobaan *engine*, di-*seting data set* 1M dengan paramater *best model* dengan rank 10, lambda 0,8 dan untuk *cosine similarity* diberikan batas *threshold* 50% atau 0,5. Kemudian hasil rekomendasi dari parameter yang telah di set ini diberikan kepada 20 orang pengguna. Pengguna mencoba memberikan *rating* untuk item film yang pernah dia sukai atau pernah ditonton, kemudian diberikan kuesioner yang terdiri dari 5 pertanyaan sebagai berikut :

1. Dari daftar rekomendasi pertama, apakah memberikan hasil yang tidak terduga (mengejutkan) yang anda tidak menyangka itu ada ternyata film tersebut tersedia ?
2. Dari 10 daftar rekomendasi I, berapa jumlah rekomendasi yang relevan menurut selera anda ?

3. Klik salah satu dari 10 daftar yang dihasilkan dari rekomendasi metode I, kemudian lihat 10 daftar rekomendasi selanjutnya. Apakah memberikan hasil yang tidak terduga ?
4. Dari 10 daftar rekomendasi yang tersedia pada lembar rekomendasi II, berapa jumlah rekomendasi yang relevan menurut selera Anda ?
5. Rekomendasi yang manakah yang lebih baik menurut selera Anda diantara kedua hasil rekomendasi tersebut ?

Kemudian hasil dari kuesioner di kumpulkan untuk menjadi tambahan bahan analisis kelayakan *engine*.

D. HASIL DAN PEMBAHASAN

ALS-WR adalah algoritma matrik faktorisasi dengan beberapa kali iterasi, sehingga mendapatkan beberapa nilai alternatif. Model terbaik didapatkan pada saat titik konvergen telah tercapai dan error menjadi minimal. Sebelum mencapai model terbaik, terlebih dahulu t*raining* atau pelatihan perlu dilakukan pada mesin pembelajaran *Mlib Apache Sp*ark. Dalam proses *training* ini, data dibagi 3 secara random, terpisah dan tidak tumpang tindih. Proses *training* ini meliputi *trai*n, *tes*t dan *validation*. Pada proses awal, *MLib apache spark* akan melatih *multiple model* berdasarkan *training set*, memilih model terbaik pada *validation set* dan pada akhirnya mengevaluasi model terbaik pada *test se*t. *Output*-nya adalah RMSE yaitu kuadrat rata-rata *error* terkecil yang dihasilkan dari prediksi. *Mlib spark* juga menambahkan rating baru ke *training set* untuk membuat *personal* rekomendasi. Kemudian hasil dari proses iterasi ini akan disimpan dalam *memory* dengan *calling cache* karena ini akan diperlukan dalam proses pembuatan rekomendasi untuk hasil yang paling baik.

Peneliti menggunakan 3 *dataset* dengan parameter lamda yang berbeda-beda, beberapa kali iterasi, rank 6 dan 12 serta partisi sebanyak 3 untuk melatih mesin pembalajaran. Hasil dari proses *train* ini menghasilkan model terbaik pada Gambar 4.

Terlihat pada Gambar 5, nilai RMSE terkecil dari 3 *dataset*, semakin besar *dataset*, *error* yang dihasilkan kecil, namun paramater dan jumlah iterasi juga turut mempengaruhi hasil. Inilah yang menjadi kelebihan dari algoritma ALS-WR ini bahwa Algoritma ini tergolong adaptif. Itu berarti hasil dapat disetel, dirubah dan dikontrol melalui paramater yang ditetapkan. Jika terdapat perubahan data, maka algoritma ini dengan cepat akan belajar untuk menyesuaikannya, men-*training* dirinya kembali hingga mendapatkan model terbaik. Semakin banyak data, justru *error* yang dihasilkan semakin kecil, dengan demikian algoritma ini cocok digunakan untuk mengatasi skalabilitas pada data yang terus bertumbuh sebagai salah satu ciri *big data*. Hasil rekomendasi yang didapatkan juga mampu memberikan hasil yang tidak terduga, ini dapat dilihat pada Gambar 5.

Genre dari hasil rekomendasi yang pertama cenderung random, tidak mengikuti *genre movie* yang diinputkan. Oleh karena itu pendekatan kemiripan diperlukan agar presisi model ini menjadi tinggi. Hal ini mengakomodasi pengguna yang lebih tertarik *movie* berdasarkan g*enre* dibandingkan saran dari pengguna lain yang memiliki selera yang sama.

Untuk itu pada rekomendasi ke dua, hasil dari algoritma ALS-WR yang telah diurutkan, diambil 5000 urutan teratas untuk didekatkan kembali menurut *genre*-nya dengan menggunakan algoritma Tf-Idf dan *cosine similiarity*. Hasil dari *cosine similiarity* dapat dilihat pada Gambar 6.

Berdasarkan Gambar 6 dapat dilihat bahwa peneliti hanya menampilkan 10 *movie* di halaman *interface* pengguna, dari *query* yang diinputkan di mana yang diambil adalah *genre*-nya saja. Terlihat 10 *movie* yang terbaik bernilai 1 (satu) itu berarti ke sepuluh *movie* tadi mempunyai tingkat kemiripan 100% dengan *query*. Semakin dinaikkan *threshold*-nya, maka *movie* yang ditemukan akan semakin sedikit, dan nilai presisinya juga akan semakin sedikit. Namun dengan membatasi jumlah *movie* yang ditampilkan hanya 10 *movie*, ini akan

memberikan rekomendasi terbaik untuk pengguna bahwa hanya *movie* yang memiliki tingkat kemiripan tertinggi saja yang diberikan kepada pengguna.

Dengan demikian, menggunakan 2 algoritma ALS-WR dan *cosine similiarity* tadi pengguna bisa mendapatkan hasil terbaik dengan *error* kecil dan presisi yang tinggi. Dengan menggunakan metode 2 tahap *filtering* ini item dapat diperingkat menurut prediksi *rating* yang dihasilkan dari faktorisasi keseluruhan *rating* namun tetap dapat didekatkan dengan *genre* yang dipilih pengguna. Untuk penilaian penerimaan pengguna terhadap hasil rekomendasi yang diberikan melalui metode 2 tahap ini, tingkat penerimaan terhadap item film yang direkomendasikan meningkat dari rata-rata 28% untuk rekomendasi dengan *collaborative filtering* dan menjadi 62% untuk rekomendasi dengan metode kemiripan *genre* berbasis *collaborative filtering* (2 tahap *filtering*). Hasil akhir ternyata 75% responden lebih menyukai rekomendasi ke dua yaitu hasil dari dua tahap *filtering* dibandingkan hanya *collaborative filtering* saja.

E. KESIMPULAN

Dengan menggunakan algoritma ALS-WR, dapat dihindari terjadinya *overfiting* yang ditunjukkan pada *dataset* 100K. Hasil validasi menghasilkan RMSE 0.96 sementara hasil *test* adalah 0.94. Pada *dataset* 1M nilai RMSE pada data validasi adalah 0.86 sementara RMSE pada *dataset* adalah 0.96. Pada *dataset* 10M nilai RMSE data validasi adalah 0.81 sementara RMSE pada *data test* diperoleh 0.81. Dari data tersebut dapat dilihat bahwa algoritma ALS-WR dapat menanggulangi *overfitting,* terbukti dari RMSE pada *data set* dan data validasi pada saat *training* hampir sama. Hasil *di atas* juga menggambarkan bahwa ALS-WR dapat mengatasi masalah skalabilitas untuk data yang terus tumbuh. Hal tersebut ditunjukkan dengan semakin besar data, tingkat *error* justru semakin kecil.

Dengan menggunakan metode 2 tahap *filtering* ini item dapat diperingkat menurut prediksi *rating* yang dihasilkan dari faktorisasi keseluruhan *rating*, namun tetap dapat didekatkan dengan *genre* yang dipilih user. Untuk penilaian penerimaan *user* terhadap hasil rekomendasi yang diberikan melalui metode 2 tahap ini, tingkat penerimaan terhadap item film yang direkomendasikan meningkat dari rata-rata 28% untuk rekomendasi dengan *collaborative filtering* menjadi 62% untuk rekomendasi dengan metode kemiripan *genre* berbasis *collaborative filtering* (2 tahap *filtering*). Hasil akhir ternyata 75% responden lebih menyukai rekomendasi ke dua yaitu hasil dari dua tahap *filtering* dibandingkan hanya *collaborative filtering* saja.

DAFTAR PUSTAKA

KBBI-Online. (2018). diakses tanggal 3 Maret, 2018, dari Kamus Besar Bahasa Indonesia Online: https://kbbi.web.id/film

Anderson, C. (2008). *The long tail: Why the future of business is selling less of more.* New York: Hachette Books.

Asnov, D. (2011). *Algorithms and methods in recommenders systems.* Berlin, Germany: Berlin Institute of Technology.

Bobadilla, J. (2010). A new collaborative filtering metric that improves the behavior of recommender systems. *Knowledge-Based Systems Journal*, 23 (6),520-528.

Garcia, E. (2016, Maret 10). *Cosine Similiarity Tutorial*. diakses tanggal 3 Maret, 2018, dari minerazzi.com: http://www.minerazzi.com/tutorials/cosine-similarity-tutorial.pdf

Gawesh Jawaheer, M. S. (2010). Comparison of implicit and explicit feedback from an online music recommendation service. *International Workshop on Information Heterogeneity and Fusion in Recommender Systems.* Barcelona: ACM.

J.Roettgers. (2014). *Netflix spends $150 million on content recommendations every year*. diakses tanggal 3 Maret, 2018, dari gigaom.com: https://gigaom.com/2014/10/09/netflix-spends-150-million-on-content-recommendations-every-year/

Jogiyanto. (2008). *Metodologi penelitian sistem informasi.* Yogyakarta: Andi.

Jonnalagadda, V. S. (2016). A review study of apache spark in big data processing. *International Journal of Computer Science Trends and Technology (IJCST),*, 4 (3), 93-98.

Joseph, D. (2011, Desember 16). Landasan konseptual perencanaan dan perancangan pusat apresiasi film di Yogyakarta. Yogyakarta, DIY Yogyakarta, Indonesia: Universitas Atma Jaya Yogyakarta.

Kadam, S. D. (2017). Big data analytics-recommendation system with Hadoop Framework. *Inventive Computation Technologies (ICICT), International Conference on* (pp. 1-5). Coimbatore: IEEE.

Khoshgoftar. (2009). A Survey of collaborative filtering techniques. *Artivicial Inteliigence Journal*, Vol.2009, 1-19.

Levitin, D. J. (2015). *The organized mind: Thinking straight in the age of information overload.* New York: Dutton.

Melville, P. (2002). Content-boosted collaborative filtering for improved recommendations. *Artificial intelligence* (pp. 187-192). Menlo Park, CA, USA.: American Association for Artificial Intelligence.

Morrison, B. (2016). *What do Google, Netflix, Amazon and best buy have in common?* diakses tanggal 3 Maret, 2018, dari nectarom: https://www.nectarom.com/google-netflix-amazon-best-buy-common/

Nee, D. (2016, Desember 17). *Collaborative Filtering Using Alternating Least Square*. diakses tanggal 3 Maret, 2018, dari danielnee.com: danielnee.com/2016/collaborative-filtering-using-alternating-least-square/

Ondra, F. (2014). *Machine learning at Scale*. Retrieved Maret 6, 2018, from github: https://github.com/OndraFiedler/spark-recommender/blob/master/reportAndDocumentation.pdf

Phuong. (2014). Collaborative filtering with a graph-based similarity measure. *International Conference on Computing, Management and Telecommunications.* Da Nang, Vietnam: IEEE.

Prugel-Bennett, M. A. (2010). An improved switching hybrid recommender system using naive bayes classifier and collaborative filtering. *International Multi Conferenceof Engineers and Computer Science.* Hongkong: iaeng.org.

Price, D. (2015). *Suprising facts and stats about The Big Data industry*. diakses tanggal 3 Maret, 2018, dari cloudtweaks.com: http://cloudtweaks.com/2015/03/surprising-facts-and-stats-about-the-big-data-industry/

Ricci, F. (2011). *Recommender systems handbook.* Springer US.

Roettgers, J. (2014). *Netflix spends $150 million on content recommendations every year*. diakses tanggal 3 Maret, 2018, dari gigaom.com: https://gigaom.com/2014/10/09/netflix-spends-150-million-on-content-recommendations-every-year/

Sang-Min Choi, Y.-S. H. (2012). A movie recommendation algorithm based on genre correlations. *International Journal Expert Systems with Applications* (pp. 8079-8085). New York: Pergamon Press.

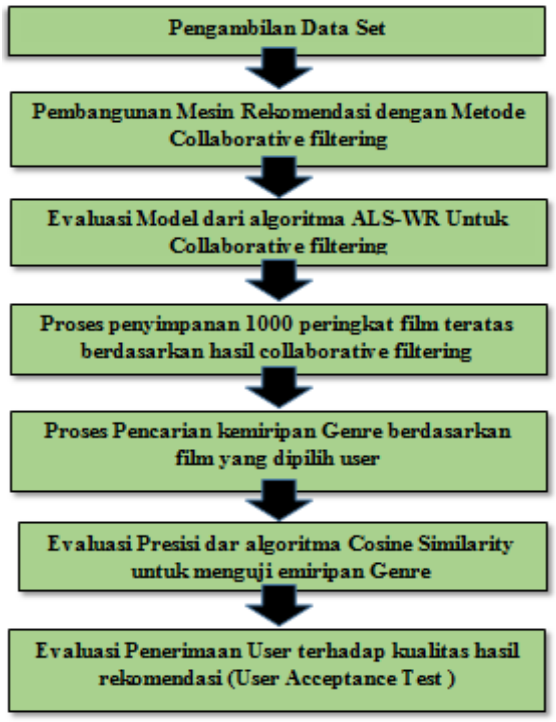
Song. (2009). A collaborative filtering recommendation algorithm based on item genre and rating similarity. *International Conference on Computational Intelligence and Natural Computing.* Wuhan, China: IEEE.

Team, T. (2012). *netflixs yields 131 value with user recommendation tools*. diakses tanggal 3 Maret, 2018, dari Forbes: http://www.forbes.com/sites/greatspeculations/2012/04/17/netflixs-yields-131-value-with-user-recommendation-tools/#5b5a177f199a

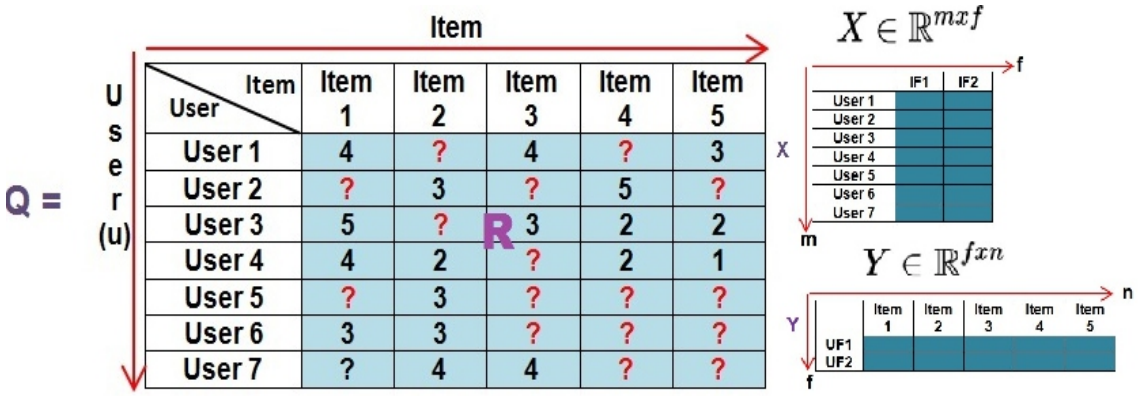
Tikk, G. T. (2012). Alternating Least Square for Personalized Ranking. *ACM.*

Walker, A. (2003). Supporting word of mouth social networks through collaborative filtering. *Journal of Interactive Learning Research*, 14 (1), 78-79.

DAFTAR GAMBAR

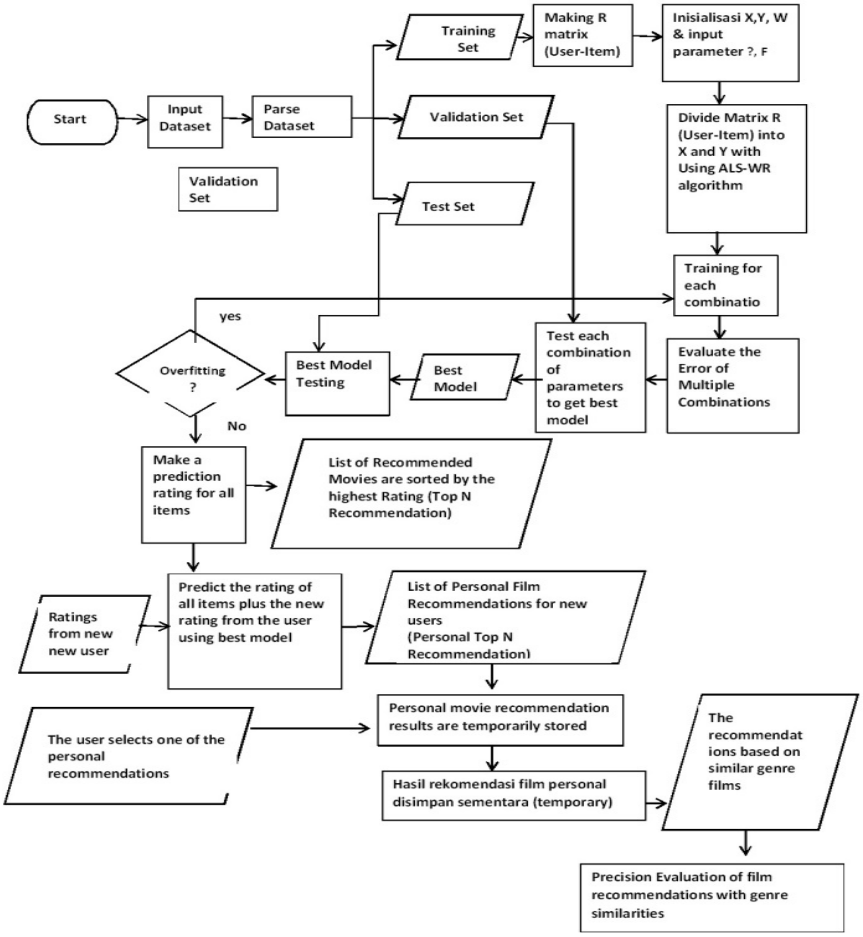


Gambar 1. Diagram Alir penelitian Mesin Rekomendasi

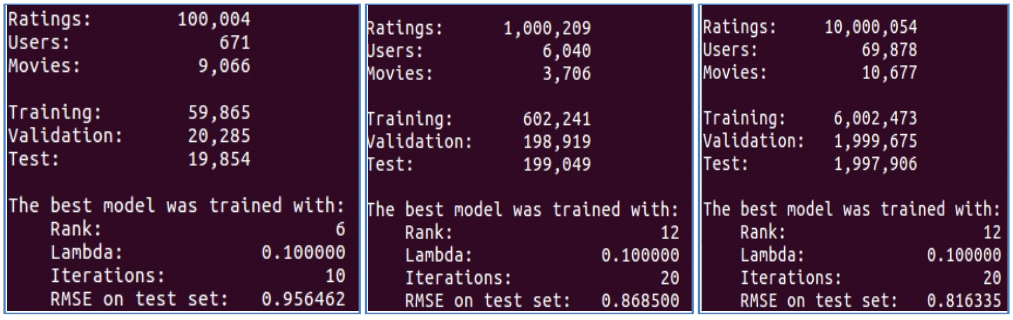


User \ Item	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	4	?	4	?	3
User 2	?	3	?	5	?
User 3	5	?	3	2	2
User 4	4	2	?	2	1
User 5	?	3	?	?	?
User 6	3	3	?	?	?
User 7	?	4	4	?	?

Gambar 2. Logika dasar algoritma ALS-WR



Gambar 3. Digram alir Pembangunan Mesin Rekomendasi dalam *MLib Spark*



Gambar 4. Best Model from 3 dataset

ALS-WR					
Input			Output	Runtime Process	Validation
DataSet	Title Movie	Rating	Title Movie		RMSE
100K	Titanic	4	Fearless	Recommend Process: 29 second, Training Process : 1 Minutes, 16 second	Best Model = λ =0.1, Rank =10 15 Iteration RMSE = 0.957
	Jurastic Park	2	Mr. Wonderful		
	The Matrix	2	Firm, The		
	Toy Story	2	Color of Night		
	Home Alone	3	Endless Summer 2		
	City of Angles	4	With Honors		
	Breaveheart	3	It Takes Two		
	Starwars	2	Bio-Dome		
	Something to talk about	3	Persuasion		
	Miracle on 34th street	3	Low Down Dirty Shame, A		
1M	Titanic	4	New Jersey Drive	Recommend Process: 1 minute, 12 second, Training Process: 2 minutes 32 second	Best Model = λ =0.1, Rank =10 15 Iteration RMSE = 0.868
	Jurastic Park	2	Breakfast at Tiffanys		
	The Matrix	2	His Girl Friday		
	Toy Story	2	Just the Ticket		
	Home Alone	3	I'll Be Home For Christmas		
	City of Angles	4	Halloween 5: The Revenge of Michael Myers		
	Breaveheart	3	For the Moment		
	Starwars	2	Goya In Bordeaux (Goya en Bodeps)		
	Something to talk about	3	Message In a Bottle		
	Miracle on 34th street	3	Thomas and the Magic Railroad		
10M	Titanic	4	From Beyond	Recommend Process: 9 minute, 20 second, Training Process : 22 minutes, 28 second	Best Model = λ =0.1, Rank =10 20 Iteration RMSE = 0.818
	Jurastic Park	2	Toy Story 2		
	The Matrix	2	Making the Grade		
	Toy Story	2	Honeymooners, The		
	Home Alone	3	Layer Cake		
	City of Angles	4	Unforgotten: Twenty-Five Years After Willowbrook		
	Breaveheart	3	Red Corner		
	Starwars	2	Surviving Picasso		
	Something to talk about	3	Living Sea, The		
	Miracle on 34th street	3	Dresser, The		

Gambar 5. Hasil Collaboration Filtering

Tf-Idf & Cosine Similiarity					
Input	Output				Validation
Title & Genre	Title Movie	Cosine Value	Threshold	Relevant Movie	Precision
(Data set 100K) Low Down Dirty Shame, A : Action Comedy	Low Down Dirty Shame	1	10%	285	0.95
	D E B S	1	20%	285	0.95
	Tuxedo	1	30%	285	0.94
	Hard Way	1	40%	283	0.77
	The Pacifier	1	50%	233	0.54
	Mr.Nice Guy	1	60%	163	0.41
	I love Trouble	1	70%	126	0.41
	Taxi 3	1	80%	16	0.05
	Talladega Nights: The Ballad of Ricky Roby	1	90%	13	0.04
	Three Fugitives	1	100%	13	0.04
(Data set 1M) New Jersey Drive: Crime,Drama	New Jersey Drive	1	10%	285	0.94
	Piece of the Action	1	20%	285	0.94
	Dead man walking	1	30%	284	0.94
	Best Laid Plans	1	40%	261	0.86
	Insquistor, the	1	50%	222	0.73
	Carlitas Way	1	60%	166	0.55
	Drighstore Cowboy	1	70%	166	0.55
	Hoax	1	80%	20	0.06
	City by the sea	1	90%	17	0.05
	Baby Boy	1	100%	17	0.05
(Data set 10M) Red Corner : Crime, Thriller	Red corner	1	10%	278	0.92
	Bank Job,the	1	20%	278	0.92
	16 Blocks	1	30%	277	0.92
	Murder by number	1	40%	245	0.81
	Night Visitor	1	50%	159	0.52
	Postman Always Rings Twice, the	1	60%	72	0.23
	Day of Jackal, The	1	70%	71	0.23
	Night Moves	1	80%	39	0.12
	Nightcap	1	90%	12	0.03
	Internal Affairs	1	100%	12	0.03

Gambar 6. Hasil penyaringan kedua dengan metode kemiripan genre