

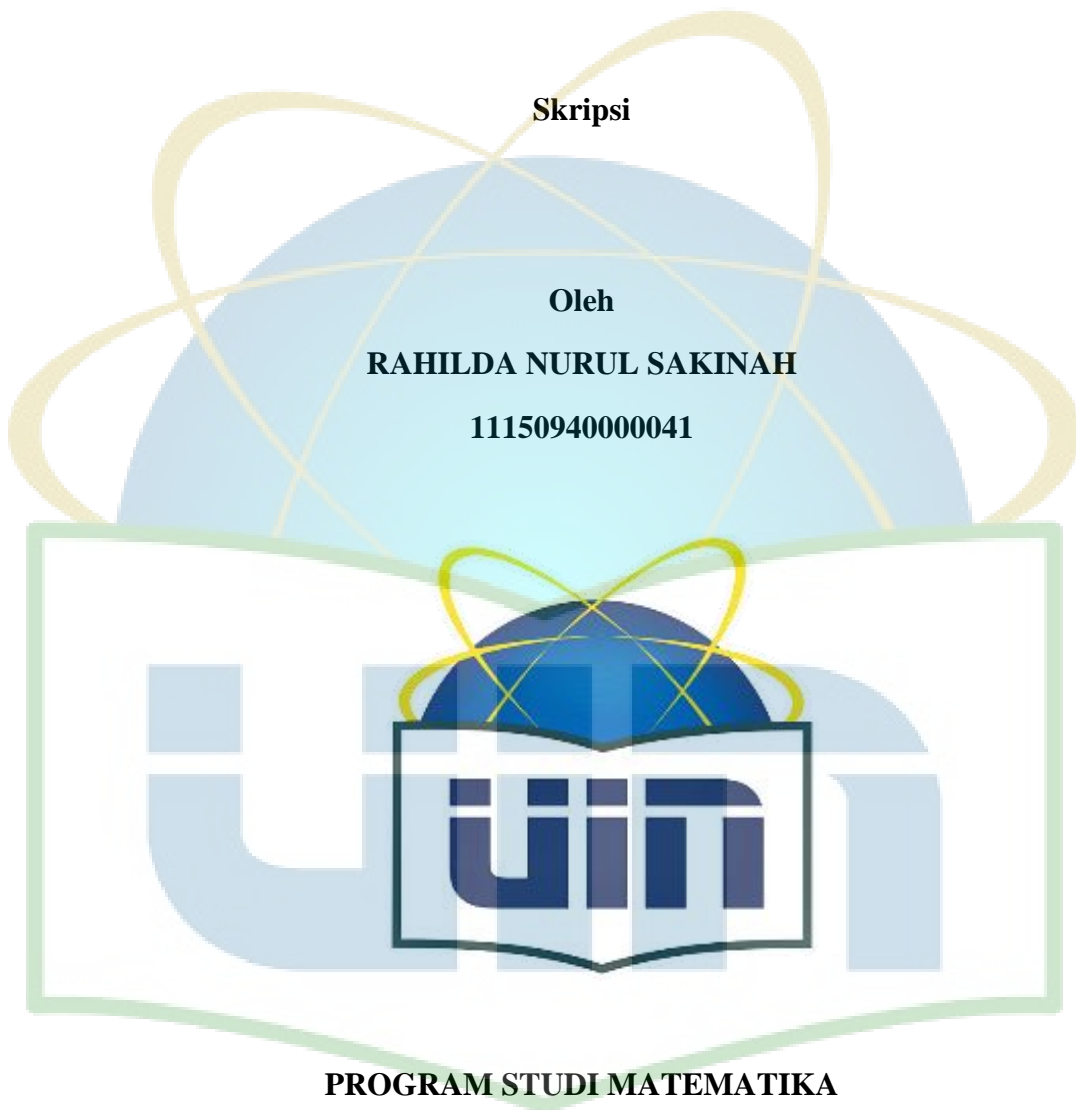
**ANALISIS SISTEM REKOMENDASI DATA RATING AIRBNB
MENGUNAKAN INISIALISASI NON-NEGATIVE DOUBLE
SINGULAR VALUE DECOMPOSITION PADA METODE NON-
NEGATIVE MATRIX FACTORIZATION**

Skripsi

Oleh

RAHILDA NURUL SAKINAH

11150940000041



PROGRAM STUDI MATEMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH

JAKARTA

2020 M / 1441 H

**ANALISIS SISTEM REKOMENDASI DATA RATING AIRBNB
MENGUNAKAN INISIALISASI NON-NEGATIVE DOUBLE
SINGULAR VALUE DECOMPOSITION PADA METODE
NON-NEGATIVE MATRIX FACTORIZATION**

Skripsi

Diajukan kepada

Universitas Islam Negeri Syarif Hidayatullah Jakarta

Fakultas Sains dan Teknologi

**Untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)**

Oleh:

Rahilda Nurul Sakinah

11150940000041

**PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UIN SYARIF HIDAYATULLAH JAKARTA
2020 M / 1440 H**

PERNYATAAN

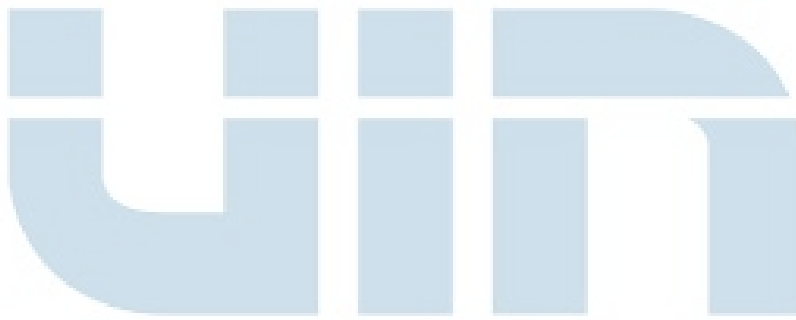
DENGAN INI SAYA MENYATAKAN BAHWA SKRIPSI INI BENAR-BENAR HASIL KARYA SENDIRI YANG BELUM PERNAH DIAJUKAN SEBAGAI SKRIPSI ATAU KARYA ILMIAH PADA PERGURUAN TINGGI ATAU LEMBAGA MANAPUN.

Jakarta, Oktober 2020



Rahilda Nurul Sakinah

11150940000041



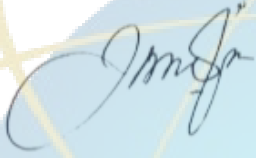
LEMBAR PENGESAHAN

Skripsi berjudul “Analisis Sistem Rekomendasi Data Rating Airbnb Menggunakan Inisialisasi Non-Negative Double Singular Value Decomposition Pada Metode Non-Negative Matrix Factorization” yang ditulis oleh **Rahilda Nurul Sakinah, NIM 11150940000041** telah diuji dan dinyatakan lulus dalam sidang Munaqosyah Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta pada hari Jum’at, 2 Oktober 2020. Skripsi ini telah diterima sebagai salah satu syarat untuk memperoleh gelar sarjana strata satu (S1) Program Studi Matematika.

Menyetujui,

Pembimbing I


Pembimbing II



Yanne Irene, M.Si
NIP. 197412312005012018


M. Irvan Septiar Musti, M.Si
NUP. 9920113224

Penguji I

Penguji II


Dr. Taufik Edy Sutanto, M.Sc.Tech
NIP. 197905302006041002



Dr. Suma'inna, M.Si
NIP. 197912082007012015


Mengetahui,

Dekan Fakultas Sains dan Teknologi

Ketua program Studi Matematika




Prof. Dr. Lily S Eka P, M.Env.Stud
NIP. 196904042005012005


Dr. Suma'inna, M.Si
NIP. 197912082007012015

LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Yang bertanda tangan di bawah ini:

Nama : Rahilda Nurul Sakinah

NIM : 11150940000041

Program Studi : Matematika Fakultas Sains dan Teknologi

Demi pengembangan ilmu pengetahuan, saya menyetujui untuk memberikan **Hak Bebas Royalti Non-Eksklusif** (*Non-Exclusive-Free Right*) kepada Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta atas karya ilmiah saya yang berjudul:

“Analisis Sistem Rekomendasi Data Rating Airbnb Menggunakan Inisialisasi Non-Negative Double Singular Value Decomposition Pada Metode Non-Negative Matrix Factorization”

beserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-Eksklusif ini, Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta berhak menyimpan, mengalihmedia/formatkan, mengelolanya dalam bentuk pangkalan data (*database*), mendistribusikannya, dan menampilkan/mempublikasikannya di internet dan media lain untuk kepentingan akademis tanpa perlu meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta. Segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta karya ilmiah ini menjadi tanggungjawab saya sebagai penulis.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Dibuat di Jakarta

Pada tanggal: 2 Oktober 2020

Yang membuat pernyataan



(Rahilda Nurul Sakinah)

PERSEMBAHAN DAN MOTTO

Skripsi ini kupersembahkan kepada
kedua orang tua yang telah sabar menunggu anak bungsunya lulus.

MOTTO

“Tidak perlu risau, setiap manusia punya lintasan waktu masing-masing.”



ABSTRAK

Rahilda Nurul Sakinah, Analisis Sistem Rekomendasi Data Rating Airbnb Menggunakan Inisialisasi Non-Negative Double Singular Value Decomposition Pada Metode Non-Negative Matrix Factorization, di bawah bimbingan **Yanne Irene, M.Si** dan **M. Irvan Septiar Musti, M.Si.**

Pada tahun 2019, sebanyak 150 juta penduduk Indonesia mengeluarkan sebanyak 9,3 miliar dollar amerika dalam bidang travel termasuk akomodasi. Banyaknya dana yang digelontorkan menyebabkan banyak bermunculan aplikasi dibidang *Online Travel Agent* (OTA) salah satunya Airbnb yang memiliki 800.995 data rating. Kumpulan data Airbnb yang besar inilah yang akan digunakan untuk merancang sebuah sistem rekomendasi. Sistem rekomendasi dapat membantu pengguna dalam menentukan penginapan apa yang sesuai dengan kesukaan dan kebutuhan mereka. Karenanya peneliti mencoba mengusulkan penelitian dengan teknik *Collaborative filtering* yang didasarkan pada preferensi dari pengguna lain yang serupa menggunakan metode *Non-Negative Matrix Factorization* (NMF) dengan inisialisasi *Non-Negative Double Singular Value Decomposition* (NNDSVD). Hasil pengujian dilakukan dengan *5 – fold Cross Validation* mendapatkan hasil k terbaik pada saat $k = 89$ dengan nilai *Root Mean Square Error* (RMSE) 2.0105 dan running time selama 16.2983 menit pada inisialisasi NNDSVD. Dengan menggunakan nilai yang sama pada inisialisasi random diperoleh nilai RMSE 2.0199 dan running time selama 16.2983 menit.

Kata Kunci: *Collaborative filtering, k – fold Cross Validation, Non-Negative Double Singular Value Decomposition, Non-Negative Matrix Factorization, Root Mean Square Error, Sistem Rekomendasi.*

ABSTRACT

Rahilda Nurul Sakinah, Analysis Recommendation system of Airbnb's rating data using Initialization Non-Negative Double Singular Value Decomposition with Non-Negative Matrix Factorization method, under the guidance of **Yanne Irene, M.Si** dan **M. Irvan Septiar Musti, M.**

In 2019, 150 million people in Indonesia were internet users and have spend 9.3 billion US dollars for e-commerce specially on travel category including accommodation. It makes the rise online travel agent (OTA), one of them is Airbnb which has 800.995 rating data. This huge Airbnb dataset will be used to build the recommendation system. The recommendation system can help users to find what they need and they like. The researcher proposes Collaborative filtering technique based on other user preferences using Non-Negative Matrix Factorization (NMF) method with Non-Negative Double Singular Value Decomposition (NNDSVD) initialization for recommendation system. The result using 5 – fold Cross Validation get the best results when $k = 89$ with value Root Mean Square Error (RMSE) is 2.0105 and a running time of 16.2983 minutes on the NNDSVD initialization. Then for random initialization with the same k , get the value RMSE is 2.0199 and a running time of 16.2983 minutes.

Keywords: *Collaborative filtering, k – fold Cross Validation, Non-Negative Double Singular Value Decomposition, Non-Negative Matrix Factorization, Root Mean Square Error, Recommendation system.*

KATA PENGANTAR

Assalamu'alaikum Wr. Wb

Alhamdulillah, tiada hentinya penulis panjatkan puji dan syukur kepada Allah SWT karena berkat rahmat dan hidayah-Nya penulis dapat menyelesaikan penelitian ini. Shalawat serta salam peneliti curahkan kepada junjungan nabi besar Nabi Muhammad SAW beserta keluarganya, para sahabat dan para pengikutnya.

Penelitian ini penulis selesaikan untuk memperoleh gelar sarjana Matematika. Dalam penyusunan skripsi ini, peneliti tidak luput dari kesalahan dan hambatan. Namun, terbantu dengan adanya pihak-pihak yang memberikan doa, bantuan, motivasi dan selalu menyemangati sehingga penelitian ini dapat terselesaikan. Oleh karena itu peneliti mengucapkan terima kasih kepada:

1. Prof. Dr. Lily Surayya Eka Putri, M.Env.Stud selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta.
2. Ibu Dr. Suma'inna, M.Si, selaku Ketua Program Studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta dan Ibu Irma Fauziah M.Sc, selaku Sekretaris program studi Matematika Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta.
3. Ibu Yanne Irene, M.Si selaku pembimbing I dan Bapak M. Irvan Septiar Musti, M.Si selaku pembimbing II atas ilmu dan bimbingannya selama penyusunan skripsi ini sehingga dapat terselesaikan dengan baik.
4. Bapak Dr. Taufik Edy Sutanto, M.Sc.Tech selaku penguji I dan Ibu Dr. Suma'inna, M.Si. selaku penguji II, terima kasih atas kritik dan sarannya kepada penulis mulai dari seminar hasil hingga sidang skripsi, sehingga penelitian ini menjadi lebih baik.
5. Untuk kedua orang tua tersayang, Ayah Ishak dan Ibu Rusnani terima kasih sudah sabar dan tiada henti memberikan doa, kekuatan, dan materi hingga penulis mampu menyelesaikan skripsi ini dengan baik.

6. Untuk keempat kakak penulis, Bang hakim, Bang Rusli, Kak Nena, dan Kak Lia yang juga telah mendoakan dan memberikan dukungan hingga penulis mampu menyelesaikan skripsi ini.
7. Untuk Rizky Juliwardi, yang telah memberikan dukungan serta menjadi tempat berkeluh kesah dalam penulisan skripsi ini.
8. Ketiga sahabat penulis, Tanjung Kusumoningtyas, Maiyudi Mariska Windra Yahya, Alun Sagara Putra yang telah menjadi penghibur dikala suntuk mengerjakan skripsi.
9. Untuk Kak Nadya, Hamid, Sabrah, Shinta, Early, Wina dan Nci yang sering penulis repotkan dalam membantu mengerjakan skripsi.
10. Teman-teman angkatan 2015 matematika UIN Syarif Hidayatullah dan seluruh pihak yang secara langsung maupun tidak langsung telah membantu, mendukung, serta mendoakan penulis dalam penyelesaian skripsi ini.

Penulis menyadari bahwa masih ada kesalahan dalam penyusunan skripsi ini. Maka dari itu penulis mengharapkan kritik dan saran yang membangun supaya menjadi bahan perbaikan bagi peneliti selanjutnya. Penulis juga berharap penelitian ini bermanfaat bagi siapapun yang membacanya.

Wassalamu'alaikum Wr. Wb.

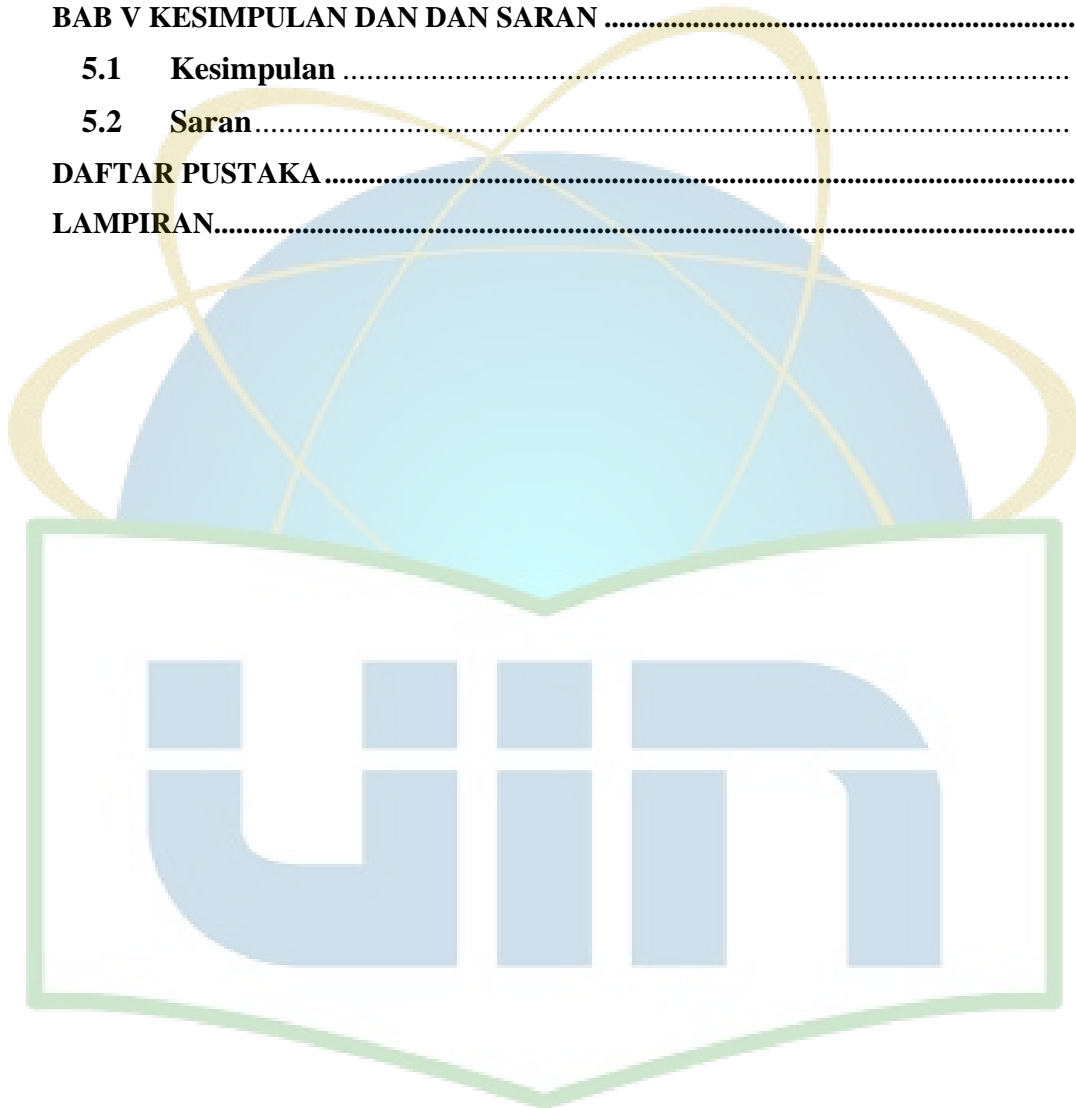
Jakarta, 2 Oktober 2020

Penulis

DAFTAR ISI

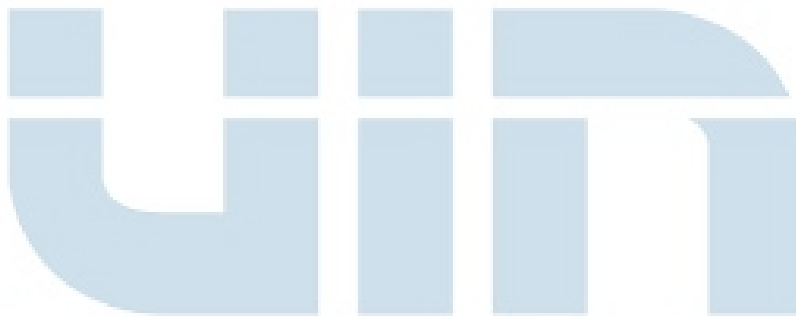
PERNYATAAN.....	ii
LEMBAR PENGESAHAN	iii
ABSTRAK	vi
ABSTRACT	vii
KATA PENGANTAR.....	viii
DAFTAR ISI.....	x
DAFTAR GAMBAR.....	xii
DAFTAR TABEL	i
BAB I PENDAHULUAN.....	1
1.1 Latar belakang.....	1
1.2 Perumusan masalah	4
1.3 Tujuan Penelitian	4
1.4 Batasan Masalah.....	5
1.5 Manfaat Penelitian	5
BAB II LANDASAN TEORI	6
2.1 Sistem Rekomendasi.....	6
2.2 <i>Collaborative Filtering</i>	7
2.3 Faktorisasi Matriks	8
2.4 Partisi Matriks	10
2.5 Nilai Eigen dan Vektor Eigen.....	11
2.6 Singular Value Decomposition	12
2.6.1 Truncated SVD.....	13
2.7 <i>Root Mean Square Error</i>	16
2.8 <i>K-fold Cross Validation</i>	17
BAB III METODOLOGI PENELITIAN	19
3.1 Data Penelitian.....	19
3.2 Non-Negative Matrix Factorization	20
3.3 Non negative Double Singular Value Decomposition.....	22

3.4	Alur Penelitian	29
BAB IV HASIL DAN PEMBAHASAN		32
4.1	Eksplorasi Data.....	32
4.2	Hasil Penelitian	33
4.3	Hasil Rekomendasi	39
BAB V KESIMPULAN DAN DAN SARAN		41
5.1	Kesimpulan	41
5.2	Saran.....	41
DAFTAR PUSTAKA.....		43
LAMPIRAN.....		45



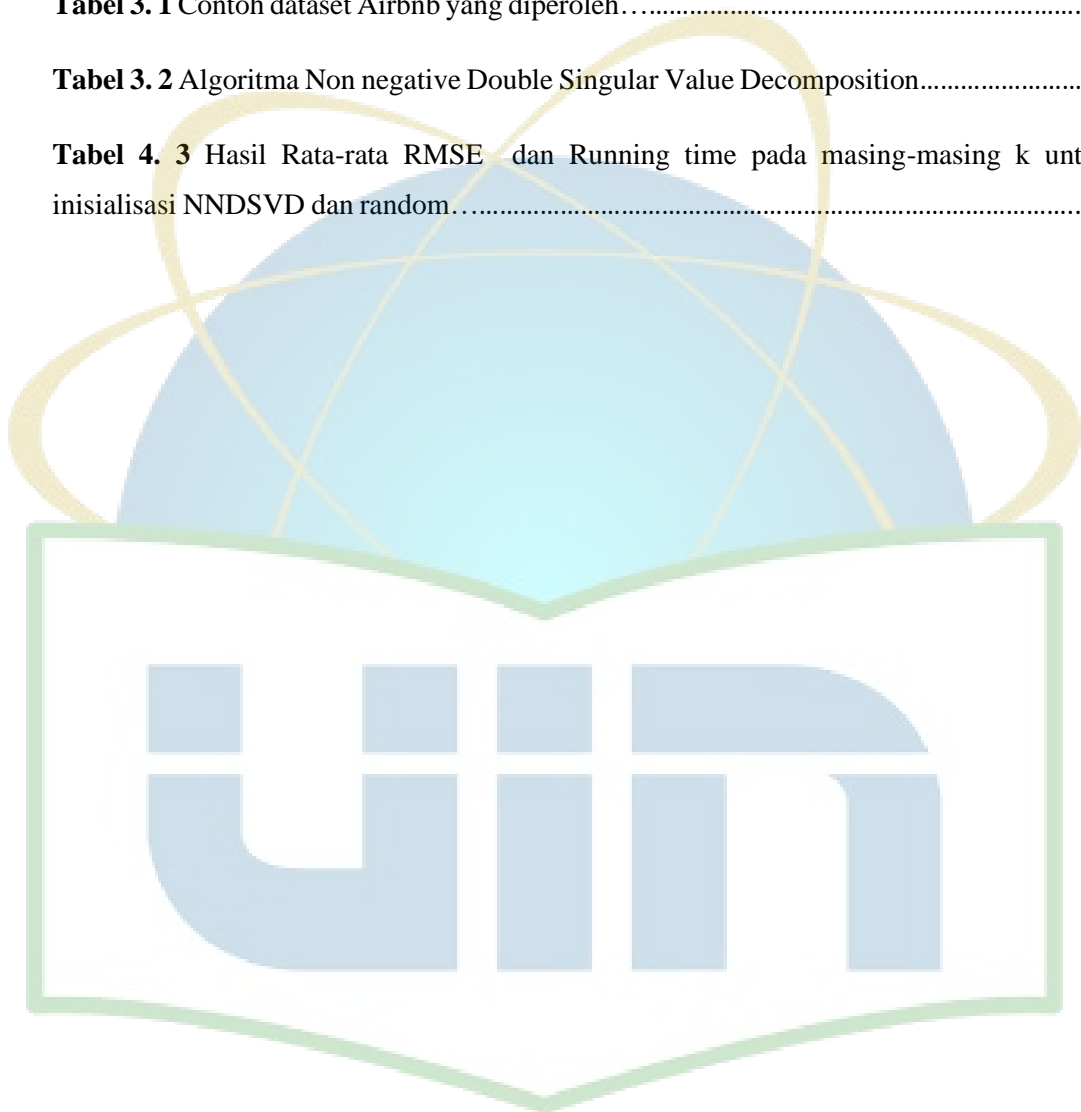
DAFTAR GAMBAR

Gambar 1. 1	Pengguna Internet di Indonesia pada bulan Januari 2018- Januari 2019	1
Gambar 1. 2	Kategori pengeluaran untuk e-commerce di Indonesia	2
Gambar 2. 1	Struktur matriks SVD.....	13
Gambar 2. 2	Struktur matriks Truncated SVD.....	14
Gambar 2. 3	Prinsip K-fold Cross Validation	18
Gambar 3. 1	Ilustrasi Non-Negative Matrix Factorization.....	20
Gambar 3. 2	Diagram Alur Penelitian.....	31
Gambar 4. 1	Jumlah masing-masing entri pada data rating.....	32
Gambar 4. 2	Jumlah masing-masing rating setelah entri 0 dihilangkan.....	33
Gambar 4. 3	Rata-Rata RMSE dan Kompleksitas waktu dengan inisialisasi NNDSVD.....	34
Gambar 4. 4	Rata-Rata RMSE dan Kompleksitas waktu dengan inisialisasi random	34
Gambar 4. 5	Perbandingan Rata-rata RMSE inisialisasi NNDSVD dan random	39
Gambar 4. 6	Hasil rekomendasi untuk user index ke-100.....	39



DAFTAR TABEL

Tabel 2. 1 Contoh matriks rating user terhadap suatu item	9
Tabel 3. 1 Contoh dataset Airbnb yang diperoleh	19
Tabel 3. 2 Algoritma Non negative Double Singular Value Decomposition.....	24
Tabel 4. 3 Hasil Rata-rata RMSE dan Running time pada masing-masing k untuk inialisasi NNDSVD dan random.....	35



BAB I

PENDAHULUAN

1.1 Latar belakang

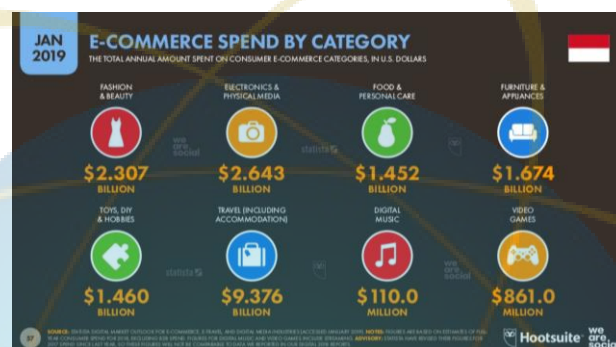
Perkembangan teknologi sekarang ini sedang mengalami kemajuan yang sangat pesat. Perkembangan teknologi tercipta karena adanya rasa ingin tahu manusia yang cukup tinggi terhadap kemudahan dalam menjalankan kehidupan. Dalam agama islam Allah SWT telah memerintahkan dalam Al-Quran surah Al-Mujadilah ayat 11 yang berbunyi: “Hai orang-orang beriman apabila dikatakan kepadamu: ‘Berlapang-lapanglah dalam majlis’, maka lapangkanlah niscaya Allah akan memberi kelapangan untukmu. Dan apabila dikatakan: ‘Berdirilah kamu’, maka berdirilah, niscaya Allah akan meninggikan orang-orang yang beriman di antaramu dan orang-orang yang diberi ilmu pengetahuan beberapa derajat. Dan Allah Maha Mengetahui apa yang kamu kerjakan”.

Dalam ayat tersebut Allah SWT telah menjanjikan akan meninggikan derajat orang-orang yang memiliki ilmu. Ilmu terus berkembang sesuai perkembangan zaman dan tidak hanya sebatas ilmu agama tetapi juga ilmu lain yang menunjang kehidupan manusia di setiap masa seperti. Internet merupakan ilmu dibidang teknologi dengan perkembangan paling cepat untuk masa sekarang.



Gambar 1.1 Pengguna Internet di Indonesia pada bulan Januari 2018- Januari 2019 [1].

Pada Gambar 1.1 dapat dilihat bahwa sebanyak 56% populasi penduduk di Indonesia atau sekitar 150 juta orang menggunakan internet. Terjadi peningkatan sebanyak 7% dari tahun 2018 yaitu sekitar 10 juta jiwa [1]. Peningkatan terjadi karena masyarakat yang mulai merasakan manfaat internet untuk mempermudah kehidupan diantaranya adalah untuk mencari informasi, bersosialisasi, serta melakukan jual-beli secara *online*.



Gambar 1.2 Kategori pengeluaran untuk e-commerce di Indonesia [1].

Survei yang dilakukan *Global Web Index* bulan Januari tahun 2019 yang pada Gambar 1.2 menunjukkan penduduk Indonesia banyak mengeluarkan uangnya dalam *e-commerce* salah satunya adalah untuk kepentingan *travel*. Melihat tingginya uang yang dikeluarkan penduduk Indonesia untuk *travel* menjadikan banyaknya muncul aplikasi dibidang *Online Travel Agent*. Salah satunya adalah Airbnb yang menawarkan jasa untuk pesan penginapan. Airbnb didesain sebagai sarana antara pemilik penginapan sebagai penjual jasa dan *traveler* sebagai pembeli jasa untuk bertransaksi secara online. Banyaknya informasi penginapan membuat user mengalami kebingungan untuk memilih penginapan dengan cepat, tepat dan sesuai dengan kebutuhannya. Hal tersebut dapat dibantu dengan adanya Sistem Rekomendasi.

Sistem rekomendasi dapat ditemukan pada hampir segala bidang seperti situs *electronic commerce* (e-commerce), situs sosial media, dan situs hiburan. Sistem rekomendasi pada situs e-commerce dibangun sebagai alat yang didesain para penyedia jasa untuk menentukan produk apa yang paling tepat direkomendasikan kepada user sesuai dengan kebutuhan dan selera mereka berdasarkan pada

informasi yang telah user berikan sebelumnya. Dengan adanya sistem rekomendasi user dimudahkan dari kebingungan yang sering kali timbul ketika ingin membeli atau menikmati suatu produk. Sistem rekomendasi dapat diklasifikasi berdasarkan sumber informasi user dan item menjadi tiga yaitu: *Content-Based*, *Collaborative Filtering*, dan *Hybrid* [2].

Collaborative filtering (CF) merupakan pendekatan yang paling umum digunakan dalam sistem rekomendasi. Dasar dari CF adalah memberikan rekomendasi berdasarkan penilaian dari user lain yang berfikir sama. Penilaian tersebut dapat diperoleh secara eksplisit (skalar) dan implisit (biner) [3]. Secara eksplisit penilaian dilihat berdasarkan rating yang tersedia dalam bentuk skalar sedangkan secara implisit penilaian dilihat dalam bentuk biner atau pernyataan suka/tidak suka. Dalam praktiknya CF memiliki beberapa kendala yaitu, masalah ukuran dari data matriks yang *sparse* [4]. Guna mengatasi masalah tersebut digunakanlah metode Faktorisasi Matriks (*Matrix Factorization*). Tujuan dari faktorisasi matriks adalah dekomposisi dimensi matriks yang besar menjadi dimensi matriks yang lebih kecil [5]. Metode faktorisasi matriks diantaranya adalah *Singular Value Decomposition* (SVD), *Principal Component Analysis* (PCA), *Probabilistic Matrix Factorization* (PMF), dan *Non-Negative Matrix Factorization* (NMF).

Metode NMF memiliki kelebihan dibandingkan dengan faktorisasi matriks lain seperti SVD dan PCA [6]. Pertama, model NMF menghasilkan data tak negatif sehingga lebih mudah untuk diinterpretasikan. Kedua, NMF secara umum cocok untuk mengolah data yang *sparse*. Hal ini memungkinkan NMF digunakan pada sistem rekomendasi yang memiliki data *sparse*.

Sebelumnya telah dilakukan penelitian untuk sistem rekomendasi diantaranya adalah Rekomendasi musik menggunakan *Collaborative filtering* dengan metode *K-Nearest neighbor* (KNN) dan didapatkan *Root Mean Square Error* (RMSE) sebesar 1,231 dan *Mean Absolute Error* (MAE) sebesar 0,940 [4]. Kemudian dilakukan juga penelitian untuk membandingkan algoritma sistem rekomendasi adaptasi dari *Non-negative matrix Factorization* yaitu: PSVD, SSVD, PRD,

RNMF, MIXD, MIXR pada data Movielens 1M [5]. Raden Sutrisman pada tahun 2018 melakukan penelitian dengan membandingkan inisialisasi NNDSVD dan *fuzzy c-means++* pada algoritma *Eigenspace based fuzzy c-means* untuk melakukan pendeteksian topik berita online di Indonesia, dari penelitian tersebut inisialisasi NNDSVD memperoleh tingkat akurasi sebesar 0,997 lebih baik dari inisialisasi *fuzzy c-means++* [7].

Keberhasilan dalam sistem rekomendasi menggunakan NMF selain bergantung pada model yang digunakan juga tergantung pada inisialisasi matriks awal W dan H . Inisialisasi matriks awal ini ditujukan untuk menambah performa dalam kompleksitas komputasi [8]. Inisialisasi umum yang digunakan pada algoritma NMF adalah *random*, sehingga menghasilkan matriks awal dengan angka numerik yang berbeda setiap dijalankan. Untuk itu, pada penelitian ini akan diimplementasikan penggunaan metode inisialisasi lain yang bersifat tidak random yaitu Non-Negative Double Value Decomposition (NNDSVD) pada algoritma NMF. Selanjutnya inisialisasi awal tersebut akan dianalisis nilai errornya dengan menggunakan database Airbnb.

1.2 Perumusan masalah

Berdasarkan latar belakang yang telah penulis sebutkan di atas, maka dapat diuraikan perumusan masalah dalam penilitan penulis adalah sebagai berikut :

1. Bagaimana implementasi sistem rekomendasi menggunakan metode NMF dengan inisialisasi NNDSVD?
2. Bagaimana menentukan parameter k terbaik dalam NMF untuk mendapatkan matriks W dan H paling optimal?
3. Bagaimana analisis nilai RMSE untuk inisialisasi NNDSVD pada metode NMF?

1.3 Tujuan Penelitian

Adapun tujuan yang ingin penulis capai dalam penilitian ini adalah:

1. Mendapatkan hasil implementasi dari sistem rekomendasi menggunakan metode NMF dengan inisialisasi NNDSVD.

2. Mendapatkan parameter k terbaik dalam NMF sehingga menghasilkan matriks W dan matriks H yang optimal.
3. Mendapatkan hasil analisis RMSE untuk inisialisasi NNDSVD pada metode NMF.

1.4 Batasan Masalah

Adapun Batasan masalah untuk penelitian penulis adalah:

1. Peneliti menggunakan data sekunder yang diambil dari Airbnb New York City.
2. Penelitian penulis hanya membahas dua buah inisialisasi awal yaitu random dan NNDSVD untuk metode NMF.
3. Penulis membatasi hanya mengambil data penginapan yang memiliki paling sedikit 10 review, dan user yang telah menilai paling sedikit 3 penginapan juga melakukan pengujian dengan k –*component* mulai dari 2 hingga 100.
4. Data diolah menggunakan bantuan software *Python 3.6.8* dengan *processor* Intel Core i5 serta RAM 6 GB.

1.5 Manfaat Penelitian

Manfaat yang dapat diambil dari penelitian ini diantaranya adalah:

1. Bagi penulis penelitian ini bisa menjadi ilmu baru yang bisa penulis implementasikan di kehidupan sehari-hari jika ingin merekomendasikan sebuah item kepada pengguna.
2. Bagi akademis, penelitian ini bisa dijadikan tambahan perkembangan ilmu dan bisa diimplementasikan terhadap dunia akademis.
3. Bagi pemerintah ataupun perusahaan kelak, penelitian ini bisa menjadi bahan pertimbangan dalam pemilihan metode untuk sistem rekomendasi

BAB II

LANDASAN TEORI

Bab ini akan menguraikan teori-teori yang akan penulis gunakan pada penelitian kali ini yang merupakan landasan dalam NMF. Penjelasan akan dibagi menjadi empat subbab. Subbab pertama akan dibahas mengenai sistem rekomendasi, kedua mengenai collaborative filtering, ketiga membahas faktorisasi matriks, dan keempat membahas tentang matriks evaluasi sebagai performa akurasi. Sementara penjelasan mengenai teori NMF dan inisialisasi dalam NMF akan dijelaskan pada bab selanjutnya.

2.1 Sistem Rekomendasi

Sistem rekomendasi adalah alat yang didesain untuk membantu para penyedia jasa guna menemukan produk (*item*) yang paling tepat sesuai dengan kebutuhan dan kesukaan pengguna (*user*) layanan itu sendiri [5]. Sistem rekomendasi akan menemukan pola dalam kumpulan data dengan mempelajari pilihan user dan item yang dihasilkan berkaitan dengan kebutuhan dan kesukaan mereka. Kesukaan pengguna terhadap suatu item dapat dilihat secara eksplisit dan implisit. Secara eksplisit kesukaan pengguna dilihat dari rating atau penilaian langsung yang user berikan terhadap item tersebut. Semakin tinggi rating yang diberikan berarti semakin tinggi pula user menyukai item itu. Secara implisit kesukaan user juga dapat dilihat dari seberapa lama atau seberapa sering user memilih item tersebut. Pada praktiknya seorang user tidak menilai semua item yang ada, oleh sebab itu beberapa item memiliki penilaian yang kosong. Kekosongan pada penilaian tersebut yang menjadi tantangan bagi penulis. Penulis akan memprediksi item apa yang dapat direkomendasikan kepada user walaupun user tersebut belum menggunakan item tersebut.

Sistem rekomendasi dapat diklasifikasikan berdasarkan sumber informasi user dan item menjadi *Content-Based*, *Collaborative Filtering*, dan *Hybrid* [9].

Pada *Content-Based filtering* user akan direkomendasikan sebuah item berdasarkan riwayat dari pilihan user sebelumnya. Item yang dinilai oleh user di masa lalu akan di observasi karakteristiknya, lalu informasi karakteristik tersebut digunakan untuk merekomendasikan item lain yang karakteristiknya mirip. *Content-Based* memiliki kendala untuk user baru. Kendala tersebut adalah user baru tidak memiliki riwayat terhadap item yang disukai, sehingga terdapat keterbatasan informasi terhadap karakteristik item. Keterbatasan karakteristik tersebut yang membuat rekomendasi item sulit dilakukan jika menggunakan pendekatan *Content-Based filtering*. Berbeda dengan *Content-Based* pada *Collaborative Filtering* rekomendasi didasarkan pada penilaian yang diberikan oleh user lain. *Collaborative Filtering* akan memberikan preferensi dari user lain. Preferensi tersebut digunakan untuk mencari kesamaan dari user. Kesamaan tersebut yang akan digunakan untuk merekomendasikan item untuk user. User yang memiliki kesamaan penilaian terhadap suatu item juga akan memiliki kesamaan pada item lain. Metode *Collaborative Filtering* memberikan performa dan akurasi yang lebih baik daripada *Content-Based Filtering* [2]. *Hybrid* adalah gabungan dari metode *Content-Based* dan *Collaborative Filtering*. Hybrid akan menggabungkan kedua metode tersebut sehingga akan memakan waktu yang lebih lama dalam segi komputasi.

2.2 Collaborative Filtering

CF merupakan salah satu metode yang paling sering digunakan dalam sistem rekomendasi. CF merekomendasikan item berdasarkan informasi dari user lain. CF akan menganalisis riwayat kebiasaan user untuk membangun hubungan antara user dengan suatu item untuk merekomendasikan produk berdasarkan pendapat user lain. Misalnya dalam *Content-Based*, item yang karakteristiknya sulit di dapat pada user baru masih bisa direkomendasikan ke user berdasarkan *rating* dari user lain yang menyukai item tersebut. CF dapat dikategorikan menjadi dua tipe : *memory-based* dan *model-based* [10].

Cara kerja *memory-based* pada intinya menghitung kemiripan yang dilihat dari rating yang telah pengguna berikan. Kemiripan ini dapat dilihat dari pengguna maupun produk. Berdasarkan kemiripan tersebut memory based dikategorikan

kembali menjadi *user-based* dan *item-based*. Kemiripan pada *User-based* dilihat berdasarkan kesukaan suatu pengguna dengan pengguna lain. Berbeda dengan *User-based*, *item-based* kemiripan dilihat dari suatu produk dengan produk lain. *Memory-based* memiliki tantangan yaitu *Sparsity* dan *Scalability* [10].

Pada dasarnya *Memory-based* menggunakan data penilaian untuk dihitung kemiripannya (*similarity*). Beberapa user akan memberikan penilaian terhadap produk yang mereka sukai, namun dalam beberapa kasus terdapat user yang tidak memberikan penilaian terhadap item yang mereka sukai atau terdapat item yang tidak dinilai oleh user. Hal ini menyebabkan tidak terdapat korelasi antara satu user dengan user lain atau suatu item dengan item lain. Tidak adanya korelasi juga menyebabkan banyak data yang kosong atau disebut *sparsity*. Data yang *sparse* dapat membuat *memory-based* tidak bisa digunakan untuk merekomendasikan secara akurat.

Scalability adalah bentuk ukuran dalam data. Dalam sistem rekomendasi jumlah data yang digunakan akan berbentuk sangat besar. Hal ini bisa mengakibatkan lambatnya rekomendasi dan akan menggunakan banyak memori. Selain itu besarnya data membuat perhitungan menjadi tidak efektif.

Berbeda dengan *memory-based*, *model-based* membangun model dari data asli yang tersedia untuk memprediksi rating yang belum diketahui. Dari hasil prediksi tersebut dapat dicari rekomendasi yang tepat berdasarkan prediksi rating yang paling besar. Hal ini dikarenakan rating yang semakin besar menandakan tingkat kesukaan yang semakin tinggi pula.

2.3 Faktorisasi Matriks

Data pada penelitian kali ini direpresentasikan berbentuk matriks dimana pada matriks tersebut setiap baris merupakan user dan tiap kolom merupakan item dan hubungan antara user dan item adalah rating yang diberikan. Matriks tersebut ditulis dalam bentuk $\mathbb{R}_{m \times n}$ di mana m merupakan banyaknya baris dan n merupakan banyaknya kolom. r_{ij} merupakan penilaian yang diberikan oleh user- i ke item- j .

Tabel 2.1 Contoh matriks rating user terhadap suatu item

	Item 1	Item 2	Item 3	Item 4	...	Item n
User 1	r_{11}	r_{12}	r_{13}	r_{14}	...	r_{1n}
User 2	r_{21}	r_{22}	r_{23}	r_{24}	...	r_{2n}
User 3	r_{31}	r_{32}	r_{33}	r_{34}	...	r_{3n}
\vdots	\vdots	\vdots	\vdots	\vdots	...	\vdots
User m	r_{m1}	r_{m2}	r_{m3}	r_{m4}	...	r_{mn}

Umumnya dalam matriks pada Tabel 2.1 akan terdapat penilaian yang memiliki entri kosong. Hal ini dikarenakan user tidak memberikan penilaian kepada semua item yang ada. Banyaknya user yang tidak memberikan penilaian terhadap suatu item akan membuat entri matriks menjadi sparse. Untuk mengatasi matriks yang sparse dapat dilakukan dengan faktorisasi matriks.

Teknik faktorisasi matriks merupakan metode yang paling umum digunakan dalam sistem rekomendasi. Penelitian kali ini menggunakan faktorisasi matriks dengan metode *aproksimasi*. Prinsip utama dari metode aproksimasi NMF adalah jika diberikan suatu matriks $A_{m \times n}$ maka kemudian kita menentukan terlebih dahulu dua buah matriks $W_{m \times k}$ dan $H_{k \times n}$ sedemikian sehingga perkalian matriks W dan matriks H akan menghasilkan suatu matriks $\hat{A}_{m \times n}$ yang dapat mengaproksimasi matriks A , dapat dinyatakan sebagai $\hat{A} \approx A$.

Misalkan $A_{m \times n}$ adalah sembarang matriks,

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Dalam hal ini matriks A merupakan matriks rating dari data asli. Maka dalam aproksimasi faktorisasi matriks maka akan ditentukan terlebih dahulu matriks sembarang misalnya $W_{m \times k}$ dan $H_{k \times n}$

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{bmatrix}, H = \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1n} \\ h_{21} & h_{22} & \dots & h_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{m1} & h_{m2} & \dots & h_{mn} \end{bmatrix}$$

Sedemikian sehingga $WH = \hat{A}$

$$\hat{A} = \begin{bmatrix} \hat{a}_{11} & \hat{a}_{12} & \dots & \hat{a}_{1n} \\ \hat{a}_{21} & \hat{a}_{22} & \dots & \hat{a}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{a}_{m1} & \hat{a}_{m2} & \dots & \hat{a}_{mn} \end{bmatrix}$$

Entri dari matriks \hat{A} merupakan prediksi rating. Matriks \hat{A} inilah yang akan digunakan untuk mengaproksimasi matriks A , kita tulis sebagai $\hat{A} \approx A$.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \approx \begin{bmatrix} \hat{a}_{11} & \hat{a}_{12} & \dots & \hat{a}_{1n} \\ \hat{a}_{21} & \hat{a}_{22} & \dots & \hat{a}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{a}_{m1} & \hat{a}_{m2} & \dots & \hat{a}_{mn} \end{bmatrix}$$

Dalam penelitian kali ini matriks $W_{m \times k}$ merupakan matriks *feature* user atau karakteristik yang menggambarkan komponen dari user dan matriks $H_{k \times n}$ adalah matriks *feature* produk atau karakteristik yang menggambarkan komponen komponen produk sedangkan k adalah banyaknya fitur yang dipilih dalam faktorisasi matriks. Faktorisasi matriks merupakan cara yang cukup berhasil dalam menemukan *latent variable*. Ada beberapa jenis model faktorisasi matriks, diantaranya adalah: *Singular Value Decomposition* (SVD), *Principal Component Analysis* (PCA), *Probabilistic Matrix Factorization* (PMF), dan *Non-Negative Matrix Factorization* (NMF).

2.4 Partisi Matriks

Sebuah matriks dapat dibagi atau dipartisi menjadi menjadi beberapa matriks yang lebih kecil dengan cara menyisipkan garis-garis horizontal dan vertikal diantara baris dan kolom yang diinginkan [11]. Mempartisi matriks dapat dilakukan dengan tiga kemungkinan. Matriks-matriks yang ukurannya kecil hasil partisi matriks disebut sub matriks. Partisi matriks digunakan untuk menyederhanakan matriks yang ukurannya besar menjadi matriks kecil sehingga lebih mudah

dioperasikan untuk tujuan tertentu. Setiap sub matriks hasil partisi selalu dapat dikembalikan ke dalam matriks asalnya. Contohnya untuk matriks $A_{4 \times 5}$ Pertama partisi matriks A menjadi 4 submatriks atau 4 blok $A_{11}, A_{12}, A_{21}, A_{22}$. Kedua partisi matriks A menjadi matriks-matriks baris r_1, r_2, r_3, r_4 . Ketiga adalah partisi matriks A menjadi matriks-matriks kolom c_1, c_2, c_3, c_4, c_5 .

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad (2.1)$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix} \quad (2.2)$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix} = [c_1 \ c_2 \ c_3 \ c_4 \ c_5] \quad (2.3)$$

2.5 Nilai Eigen dan Vektor Eigen

Sebelum membahas mengenai inisialisasi NNDSVD, penulis akan menjelaskan teori yang berhubungan dengan SVD, yaitu nilai eigen dan vektor eigen.

Misalkan A adalah sebuah matriks persegi $n \times n$, terdapat vektor tak nol x pada \mathbb{R}^n dan suatu skalar λ sehingga,

$$Ax = \lambda x, \quad x \neq 0 \quad (2.4)$$

Skalar λ disebut nilai eigen dari A dan vektor $x \neq 0$ disebut vektor eigen yang bersesuaian dengan λ [12]. Untuk memperoleh nilai eigen dari sebuah matriks A berukuran $n \times n$, persamaan (2. 4) dapat dituliskan kembali menjadi

$$\det(\lambda I - A) = 0 \quad (2. 5)$$

Dimana I merupakan suatu matriks identitas dari A . Kemudian untuk mencari vektor eigen dari A dapat menggunakan persamaan berikut:

$$(\lambda I - A)X = 0 \quad (2. 6)$$

Persamaan (2. 6) disebut persamaan karakteristik matriks A .

Definisi 2. 1 Nilai singular dari matriks A berukuran $m \times n$ adalah akar kuadrat positif dari nilai eigen tak nol matriks simetri $(A^t A)$ [13].

Dari definisi di atas, dapat diketahui hubungan antara nilai eigen dan nilai singular. Bahwa untuk mendapatkan nilai singular maka terlebih dahulu harus ditentukan nilai eigen.

2.6 Singular Value Decomposition

Singular Value Decomposition (SVD) merupakan salah satu teknik faktorisasi matriks. SVD pertama kali diperkenalkan oleh Beltrami dan Jordan pada tahun 1870 untuk dekomposisi matriks [13].

Definisi 2. 2 Misalkan matriks A berukuran $m \times n$. Sebuah hasil kali $A = U\Sigma V^t$ adalah sebuah singular value decomposition untuk A jika U merupakan matriks orthogonal berukuran $m \times m$, Σ merupakan matriks dengan entri tak nol yang hanya terletak di sepanjang diagonal utama berukuran $m \times n$ dengan seluruh entrinya adalah nonnegative, dan V merupakan matriks ortogonal berukuran $n \times n$. Entri dari diagonal matriks Σ disebut nilai singular dari A [14].

Teorema 2. 1 Jika A matriks riil berukuran $m \times n$, maka terdapat matriks ortogonal

$$U = [u_1, u_2, \dots, u_m] \in \mathbb{R}^{m \times m} \text{ dan } V = [v_1, v_2, \dots, v_n] \in \mathbb{R}^{n \times n}$$

Sedemikian sehingga,

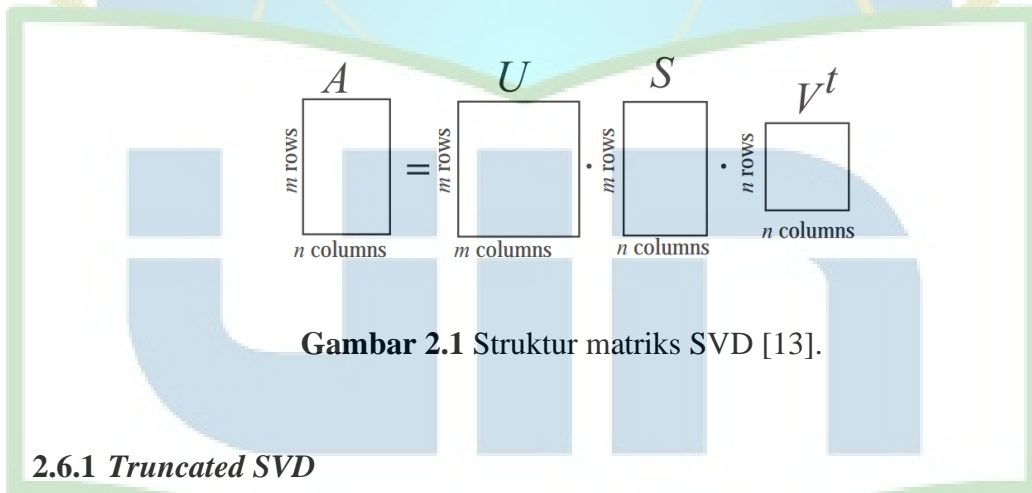
$$U^T A V = [\text{diag}(\sigma_1, \sigma_2, \dots, \sigma_m)] \in \mathbb{R}^{m \times n}, p = \min \{m, n\}$$

dengan $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p \geq 0$ [15].

Misalkan A sebuah matriks yang memiliki ordo $m \times n$, SVD matriks A berdasarkan Definisi 2. 2 dinyatakan sebagai berikut :

$$A = U S V^T \quad (2. 7)$$

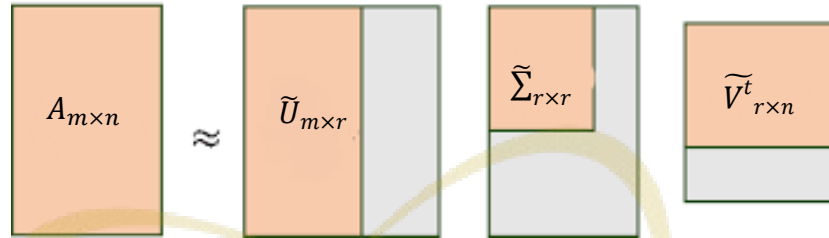
Dimana U merupakan matriks orthogonal berordo $m \times m$ yang dibentuk dari vektor eigen dari AA^T . Matriks V merupakan matriks orthogonal berordo $n \times n$ yang dibentuk dari nilai eigen vektor dari $A^T A$. Kemudian S adalah sebuah matriks $m \times n$ yang elemen pada diagonal utamanya hanya berisi nilai tak-nol dari vektor σ yang merupakan akar kuadrat positif dari nilai eigen U . σ itu sendiri disebut sebagai nilai *singular* dari matriks A .



2.6.1 Truncated SVD

Berdasarkan definisi Definisi 2. 2, SVD akan mendekomposisi matriks A yang berukuran $m \times n$ menjadi U, Σ , dan V^T dimana U merupakan matriks berukuran $m \times m$, matriks Σ berukuran $m \times n$, dan V^T berukuran $n \times n$. Truncated SVD akan menghasilkan matriks $\tilde{U}, \tilde{\Sigma}$ dan \tilde{V}^T dimana matriks \tilde{U} merupakan matriks berukuran $m \times r$ dengan $r < \text{rank}(A)$ yang diperoleh dengan mengambil r –kolom pertama dari matriks U , $\tilde{\Sigma}$ merupakan matriks yang berukuran $r \times r$ yang diperoleh dengan mengambil r –kolom pertama dan r –baris pertama dari Σ

dan matriks \widetilde{V}^t merupakan matriks yang berukuran $r \times n$ yang diperoleh dengan mengambil r –baris pertama dari matriks \widetilde{V}^t .



Gambar 2.2 Struktur matriks Truncated SVD.

Perkalian matriks \widetilde{U} , $\widetilde{\Sigma}$ dan \widetilde{V}^t akan menghasilkan matriks A_p yang berukuran $m \times n$, dimana A_p adalah matriks aproksimasi rank- r terbaik untuk matriks A .

$$A_p = \widetilde{U} \widetilde{\Sigma} \widetilde{V}^t \quad (2.8)$$

Berikut merupakan contoh sederhana dalam menentukan dekomposisi matriks menggunakan SVD dan Truncated SVD:

Misalkan terdapat matriks $A = \begin{bmatrix} 4 & 4 \\ -3 & 3 \end{bmatrix}$

$$B = AA^T = \begin{bmatrix} 4 & 4 \\ -3 & 3 \end{bmatrix} \begin{bmatrix} 4 & -3 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 32 & 0 \\ 0 & 18 \end{bmatrix} \quad (2.9)$$

Berdasarkan persamaan (2. 9) nilai eigen dari matriks B adalah:

$$\det(\lambda I - B) = 0 \quad (2.10)$$

$$\det \left(\begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} - \begin{bmatrix} 32 & 0 \\ 0 & 18 \end{bmatrix} \right) = 0 \quad (2.11)$$

$$\det \left(\begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} - \begin{bmatrix} 32 & 0 \\ 0 & 18 \end{bmatrix} \right) = 0 \quad (2.12)$$

$$\det \begin{bmatrix} \lambda - 32 & 0 \\ 0 & \lambda - 18 \end{bmatrix} = 0 \quad (2.13)$$

$$\det((\lambda - 32)(\lambda - 18)) = 0, \text{ diperoleh nilai } \lambda_1 = 32 \text{ dan } \lambda_2 = 18$$

Nilai singular A adalah $\sigma_1 = \sqrt{32}$, $\sigma_2 = \sqrt{18}$, sehingga diperoleh :

$$\Sigma = S = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \sqrt{32} & 0 \\ 0 & \sqrt{18} \\ 0 & 0 \end{bmatrix} \quad (2.14)$$

Untuk menentukan matriks **U** yang maka terlebih dahulu menentukan nilai vektor eigen berdasarkan persamaan (2. 6), sehingga diperoleh :

$$(\lambda I - B)(X) = 0 \quad (2.15)$$

$$\begin{bmatrix} \lambda - 32 & 0 \\ 0 & \lambda - 18 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \quad (2.16)$$

Perhatikan bahwa $\lambda_1 = 32$, kemudian substitusikan nilai λ_1 pada persamaan (2. 16) sehingga diperoleh :

$$\begin{bmatrix} 0 & 0 \\ 0 & 14 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \quad (2.17)$$

Persamaan (2. 17) dapat ditulis $0x_1 + 0x_2 = 0$ dan $0x_1 + 14x_2 = 0$, atau dapat ditulis $x_2 = 0$. Karena tidak terdapat keterangan mengenai x_1 , maka x_1 dapat dinyatakan sebagai suatu parameter, misalkan $x_1 = k$. Oleh karena itu, diperoleh :

$$u_1 = \begin{bmatrix} k \\ 0 \end{bmatrix} = k \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (2.18)$$

Kemudian lakukan hal yang sama untuk $\lambda_2 = 18$ disubstitusikan persamaan (2. 16) diperoleh:

$$\begin{bmatrix} -14 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \quad (2.19)$$

$-14x_1 + 0x_2 = 0$ dan $0x_1 + 0x_2 = 0$, atau dapat ditulis $x_1 = 0$. Karena tidak terdapat keterangan mengenai x_2 , maka x_2 dapat dinyatakan sebagai suatu parameter, misalkan $x_2 = s$. Oleh karena itu, diperoleh vektor eigen u_2 , sebagai berikut :

$$u_2 = \begin{bmatrix} 0 \\ s \end{bmatrix} = s \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2.20)$$

Selanjutnya, dengan menormalisasikan u_1 dan u_2 :

$u_1 = \frac{u_1}{\|u_1\|} = \frac{u_1}{\sqrt{(1)^2+(0)^2}} = u_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ dan $u_2 = \frac{u_2}{\|u_2\|} = \frac{u_2}{\sqrt{(0)^2+(1)^2}} = u_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, dengan menggabungkan u_1 dan u_2 diperoleh matriks \mathbf{U} :

$$\mathbf{U} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.21)$$

Tahap selanjutnya menentukan matriks \mathbf{V} . Sama seperti mencari matriks \mathbf{U} , perbedaannya hanya matriks \mathbf{V} merupakan matriks ortogonal dari $\mathbf{C} = \mathbf{A}^T \mathbf{A} = \begin{bmatrix} 4 & -3 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 4 & 4 \\ -3 & 3 \end{bmatrix} = \begin{bmatrix} 25 & 7 \\ 7 & 25 \end{bmatrix}$. Melalui tahapan yang sama seperti mencari matriks \mathbf{U} diperoleh :

$$\mathbf{V} = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}, \mathbf{V}^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

Dari proses ini didapat hasil dekomposisi matriks \mathbf{A} dengan SVD adalah sebagai berikut:

$$\mathbf{U} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{\Sigma} = \begin{bmatrix} \sqrt{32} & 0 \\ 0 & \sqrt{18} \\ 0 & 0 \end{bmatrix}, \mathbf{V}^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

Apabila dipilih $r=2$, Truncated SVD akan membentuk matriks aproksimasi untuk matriks \mathbf{A} sebagai berikut

$$\mathbf{A}_p = \tilde{\mathbf{U}} \tilde{\mathbf{\Sigma}} \tilde{\mathbf{V}}^t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{32} & 0 \\ 0 & \sqrt{18} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

2.7 Root Mean Square Error

Matriks prediksi yang diperoleh dengan metode NMF akan mendapatkan nilai keakuratan yang optimum. Root Mean Square Error (RMSE) merupakan persamaan yang digunakan untuk menghitung tingkat akurasi. RMSE merupakan satuan ukur yang digunakan untuk menilai keakuratan kinerja sistem

rekomendasi [16]. Secara umum rumus RMSE akan ditunjukkan pada persamaan (2. 22)

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{Y}_{ui} - Y_{ui})^2} \quad (2. 22)$$

Dengan,

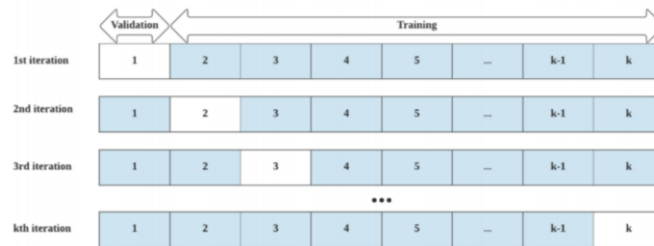
n = Banyaknya entri rating

Y_{ui} = Rating sebenarnya pada user- u terhadap item- i

\hat{Y}_{ui} = Rating prediksi pada user- u terhadap item- i

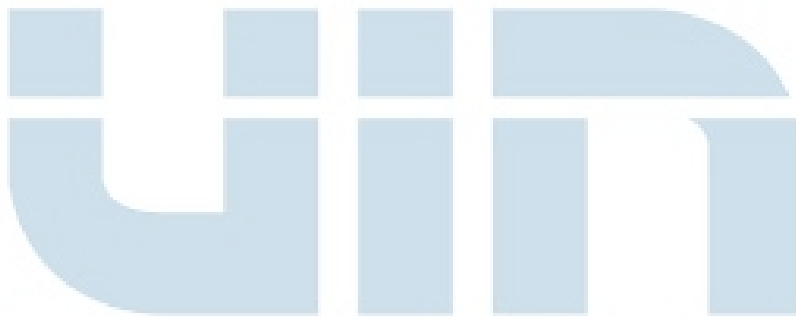
2.8 *K-fold Cross Validation*

Untuk meyakinkan bahwa model yang kita buat merupakan model terbaik, kita perlu melakukan validasi terhadap model sebelum digunakan. Validasi model adalah mengukur kinerja model dengan menghitung segala bentuk tingkat kesalahan prediksi pada model. Mengetahui seberapa baik kinerja model dapat membantu kita untuk mengoptimalkan parameter pada model itu sendiri sehingga model jauh lebih akurat. K-fold merupakan salah satu metode cross validation. Konsep k-fold cross validation tidak hanya membuat beberapa sampel data uji berulang kali, tetapi membagi dataset menjadi bagian terpisah dengan ukuran yang sama. Model dilatih oleh subset data latih dan divalidasi oleh subset validasi (data uji) sebanyak k [17]. Dengan k-fold cross validation dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi model.



Gambar 2.3 Prinsip K-fold Cross Validation [18].

Langkah K-fold cross validation seperti Gambar 2.3 adalah dengan membagi data sebanyak banyaknya k -fold yang dipilih. Kemudian setelah data terbagi sama rata sebanyak k kemudian data pada masing-masing fold dilatih dan diuji untuk melihat error dari masing-masing fold. Contohnya pada iterasi pertama atau pada fold = 1 Gambar 2.3 data dilatih pada gabungan fold 2 sampai k kemudian diuji pada fold 1. Hal ini dilakukan juga untuk iterasi kedua atau fold = 2, maka data dilatih pada gabungan fold 1,3 hingga k kemudian data diuji pada fold 2. Proses iterasi ini terus berulang hingga semua fold berkesempatan menjadi data uji.



BAB III

METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan mengenai data yang penulis gunakan serta metode Non-Negative Matrix Factorization (NMF) menggunakan dua buah inisialisasi yaitu *random* dan *Non-Negative Double Singular Value Decomposition* (NNDSVD).

3.1 Data Penelitian

Data yang digunakan dalam penelitian kali ini berupa data sekunder *Airbnb New York City* yang diambil dari situs Kaggle pada tahun 2019. Airbnb merupakan komunitas online *marketplace* yang menjadi penghubung antara pihak pengguna jasa (user) dan penyedia jasa. Airbnb sendiri telah tersedia dalam situs Airbnb <https://www.airbnb.com> maupun aplikasi ponsel. Jasa yang disediakan Airbnb adalah akomodasi khususnya penginapan. Penginapan yang tersedia pada situs maupun aplikasi Airbnb berupa penyewaan rumah, apartemen, maupun kamar pribadi.

Tabel 3.1 Contoh dataset Airbnb yang diperoleh.

Listing_id	Reviewer_id	Rating
9452127	6279455	5
6921831	125212928	5
6184827	14279077	4
16859979	118861692	3
16859979	36701279	3
16407022	123758421	5

Data yang penulis dapatkan seperti pada Tabel 3.1 merupakan data kategorik dengan format *Comma Separated Value* (CSV) yang berisi aktivitas perilaku user penyewa penginapan yang telah tersedia pada situs kaggle. Fitur yang diperoleh terdiri dari tiga kolom yaitu: *Reviewer_id*, *Listing_id*, dan *Rating*. *Reviewer_id*

merupakan identitas dari penilaian masing-masing user. Listing_id merupakan identitas dari masing-masing penginapan yang tersedia pada dataset. Rating merupakan penilaian yang user berikan terhadap sebuah penginapan. Rating yang tersedia berada dijangkauan 1 sampai 5. Dalam contoh pada Tabel 3.1 penginapan dengan id 16859979 diberikan rating 3 oleh dua orang user yang berbeda dengan id 118861692 dan id 36701279. Artinya adalah user dengan id 118861692 dan 36701279 memiliki preferensi yang sama karena telah menilai item dengan rating yang sama.

Dataset kemudian dibentuk menjadi matriks menjadi ukuran $\mathbb{R}_{m \times n}$ di mana m merupakan banyaknya user aktif yang pernah menggunakan layanan penginapan dan n merupakan banyaknya penginapan yang telah dinilai oleh user. r_{ij} merupakan rating yang diberikan user- i untuk penginapan- j . Setelah dataset berbentuk matriks original, selanjutnya akan ditentukan matriks awal W dan H menggunakan inisialisasi Random dan NNDSVD. Untuk inisialisasi random nilai awal yang diberikan berupa nilai acak yang ditentukan oleh sistem.

3.2 Non-Negative Matrix Factorization

Penelitian yang berhubungan dengan NMF telah banyak dilakukan dan telah penulis sebutkan pada bab 2. Bentuk persamaan NMF didefinisikan sebagai,

$$A \approx WH \quad (3.1)$$

Diberikan matriks $A \in \mathbb{R}^{m \times n}$ yang tiap elemennya non-negatif, $m_{ij} \geq 0$ dan sebuah integer $k < \min \{m, n\}$ [5]. Tujuan dari NMF adalah untuk menemukan dua faktor matriks tak negatif $W \in \mathbb{R}^{m \times k}$ dan $H \in \mathbb{R}^{k \times n}$ dimana jika WH akan mendekati matriks A . Berikut merupakan langkah-langkah sederhana mendapatkan matriks W dan H yang non-negatif.

$$\begin{array}{c} \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \approx \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & \dots & h_{1n} \\ h_{21} & h_{22} & \dots & h_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{m1} & h_{m2} & \dots & h_{mn} \end{bmatrix} \\ A \qquad \qquad \qquad W \qquad \qquad \qquad H \end{array}$$

Gambar 3.1 Ilustrasi Non-Negative Matrix Factorization.

Bentuk $A \approx WH$ pada persamaan (3. 1) dapat dijelaskan sebagai bentuk kesamaan antara matriks A dengan hasil perkalian W dan H. Untuk mencapai kondisi kesamaan antara matriks A dengan WH, diperlukan suatu kriteria yang disebut sebagai *Cost Function*. *Cost function* dimaksudkan untuk meminimalkan jarak antara dua matriks non negatif A dan B seperti berikut:

$$\|A - B\|^2 = \sum_{ij} (A_{ij} - B_{ij})^2 \quad (3. 2)$$

Persamaan (3. 2) menunjukkan aturan *Cost Function* untuk jarak antara A dan B yang memiliki batas bawah nol. Kondisi batas bawah nol terpenuhi jika dan hanya jika $A = B$.

Dalam penelitian ini, proses pengukuran menuju kondisi $A \approx WH$ menggunakan aturan *Norm Frobenius* yang dibuat berdasarkan aturan *cost function* seperti dijelaskan pada persamaan (3. 2). Dengan aturan ini, terdapat dua buah matriks yang akan dihitung jarak keduanya, yaitu matriks A dengan matriks hasil perkalian W dan H. Aturan *Norm Frobenius* yang digunakan dalam metode dekomposisi NMF dijelaskan pada persamaan seperti berikut.

$$f(W, H) = \|A - WH\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n \left(A_{ij} - \sum_{l=1}^k W_{il} H_{lj} \right)^2 \quad (3. 3)$$

Adapun bentuk umum dari norm frobenius adalah $\|A\|_F^2 = \sum_{i,j} |a_{ij}|^2$. Nilai variabel A menunjukkan suatu matriks, sehingga nilai Norm Frobenius dari A atau $\|A\|_F^2$ dinyatakan dalam bentuk jumlah kuadrat dari masing-masing elemen matriks penyusun matriks A.

Dengan demikian bentuk Norm Frobenius yang digunakan dalam metode NMF seperti persamaan (3. 3) menunjukkan hasil perhitungan norm frobenius untuk selisih masing-masing elemen dari matriks A dengan matriks hasil perkalian matriks W dan H. Matriks W dan H yang digunakan untuk meminimumkan fungsi diatas dilakukan dengan cara iterasi hingga fungsi tersebut mencapai suatu kondisi

yang konvergen, untuk menentukan nilai awal W dan H bisa dilakukan dengan inisialisasi random yang umum digunakan ataupun NNDSVD.

3.3 Non negative Double Singular Value Decomposition

Umumnya metode NMF menggunakan inisialisasi secara *random* dengan nilai *non-negative* pada elemen matriks W dan H . Inisialisasi digunakan untuk memberikan hasil output yang berbeda setiap kali eksekusi. Untuk permasalahan sistem rekomendasi kali ini digunakanlah inisialisasi yang tidak *random*. Metode yang digunakan adalah NNDSVD. Dasar dari algoritma NNDSVD adalah dua prosesnya dilakukan menggunakan SVD. Misalkan diberikan suatu matriks A yang merupakan matriks dengan entri rating yang diberikan user terhadap sebuah penginapan. Langkah pertama adalah menguraikan matriks A dengan SVD yang telah dijelaskan pada bab sebelumnya. Langkah kedua adalah menguraikan matriks U dan V^T . Berikut adalah langkah-langkah proses inisialisasi dengan NNDSVD [19].

Menguraikan matriks $A_{m \times n}$ dengan metode Truncated SVD menjadi,

$$A_{m \times n} \approx U_{m \times r} S_{r \times r} V_{r \times n}^T \quad (3.4)$$

Simbol \approx memiliki makna perkalian dari matriks U, S, V^T akan mendekati matriks asli A . Selanjutnya pendekatan (3.4) akan diubah menjadi

$$A_{m \times n} \approx (U_{m \times r} S_{r \times r}) (S_{r \times r} V_{r \times n}^T) \quad (3.5)$$

Pada NMF, bagian $U_{m \times r} S_{r \times r}$ pada matriks persamaan (3.5) digunakan untuk menginisialisasi matriks W dan bagian $S_{r \times r} V_{r \times n}^T$ digunakan untuk menginisialisasi matriks H . Matriks $S_{r \times r}$ adalah akar dari nilai eigen positif dari $A^t A$.

Dengan $r \leq \min(m, n)$ pendekatan (3.4) dapat dinyatakan dalam bentuk penjumlahan dari r faktor singular berikut:

$$A = \sum_{j=1}^r \sigma_j u_j v_j^T \quad (3.6)$$

Dimana $\sigma_j \geq \dots \geq \sigma_r \geq 0$ adalah nilai singular tak nol dari A . Nilai singular sebuah matriks A berukuran $m \times n$ adalah akar yang bernilai positif dari nilai eigen tak nol matriks simetri $A^t A$. Dan $\{u_j, v_j\}_{j=1}^r$ adalah vektor yang berkorespondensi dengan nilai singular, maka persamaan (3. 6) menjadi:

$$A = \sum_{j=1}^r \sigma_j C^{(j)} \quad (3. 7)$$

Dimana $C^{(j)} = u_j v_j^T$. Untuk mendapatkan matriks yang tak negatif yang akan menjadi inisialisasi matriks W dan H , persamaan (3. 7) akan dimodifikasi dengan mencari bagian positif dari $C^{(j)}$ yaitu $C_+^{(j)}$.

Lemma 3.1 Diberikan matriks $C \in \mathbb{R}^{p \times n}$ sedemikian sehingga $rank(C) = 1$, dan dapat dituliskan menjadi $C = C_+ - C_-$. Maka $rank(C_+), rank(C_-) \leq 2$ [14].

Berdasarkan Lemma 3.1, maka bagian positif $C^{(j)}$ dapat didekatkan dengan menggunakan bagian positifnya yaitu $C_+^{(j)}$, berdasarkan teorema 3.1

Teorema 3. 1 Misalkan $C \in \mathbb{R}^{m \times n}$ memiliki rank sama dengan 1, sedemikian sehingga $C = xy^T$ untuk $x \in \mathbb{R}^m, y \in \mathbb{R}^n$. Misalkan $\hat{x}_\pm := \frac{x_\pm}{\|x_\pm\|}$ $\hat{y}_\pm := \frac{y_\pm}{\|y_\pm\|}$ adalah normalisasi bagian positif dan negatif dari x dan y , dan $\mu_\pm = \|x_\pm\| \|y_\pm\|$ dan $\epsilon_\pm = \|x_\pm\| \|y_\mp\|$. Maka ekspansi tak terurut nilai singular C_+ dan C_- adalah

$$C_+ = \mu_+ \hat{x}_+ \hat{y}_+^T + \mu_- \hat{x}_- \hat{y}_-^T \quad (3. 8)$$

$$C_- = \epsilon_+ \hat{x}_+ \hat{y}_-^T + \epsilon_- \hat{x}_- \hat{y}_+^T \quad (3. 9)$$

Maksimum triplet dari C_+ adalah $(\mu_+, \hat{x}_+, \hat{y}_+)$ jika triplet dari C_+ adalah $(\mu_+, \hat{x}_+, \hat{y}_+)$ jika $\mu_+ = \max(\|x_+\| \|y_+\|, \|x_-\| \|y_-\|)$ selain itu adalah $(\mu_-, \hat{x}_-, \hat{y}_-)$. Dengan hal yang sama didapatkan maksimum triplet C_- adalah $(\epsilon_+, \hat{x}_+, \hat{y}_-)$ jika $\epsilon_+ = \max(\|x_+\| \|y_-\|, \|x_-\| \|y_+\|)$. Selain itu adalah $(\epsilon_-, \hat{x}_-, \hat{y}_+)$.

Dengan menggunakan ekspansi pada Teorema 3.1 maka bagian C_+ dapat didekomposisi menjadi matriks non negatif W dan H . Matriks tersebut digunakan untuk inisialisasi pada penyelesaian NMF. Algoritma NNDVD akan diberikan pada tabel berikut:

Tabel 3.2 Algoritma Non negative Double Singular Value Decomposition

Algoritma Non negative Double Singular Value Decomposition(NNDSVD)

Input : Matriks non negatif $A_{m \times n}$, bilangan bulat $k < \min(m, n)$.

1. Hitung triplet utama $A: [U, S, V]$
2. Inisialisasi $w_{p1} = \sqrt{S_{11}} \times u_{p1}$ dimana $p = 1, \dots, m$
3. Inisialisasi $h_{1q} = \sqrt{S_{11}} \times v_{q1}^T$ dimana $q = 1, \dots, n$
4. **for** $j = 2 : k$
5. $\mathbf{x} = u_{pj}, \mathbf{y} = v_{qj}$ untuk $p = 1, \dots, m$ dan $q = 1, \dots, n$
6. $x_+ = \text{positive}(x),$
 $y_+ = \text{positive}(y),$
 $x_- = \text{negative}(x),$
 $y_- = \text{negative}(y)$
7. $\|x_+\|, \|y_+\|, \|x_-\|, \|y_-\|$
8. $\mu_+ = \|x_+\| \|y_+\|$ dan $\mu_- = \|x_-\| \|y_-\|$
9. **if** $\mu_+ > \mu_-$
10. $\mathbf{u} = \frac{x_+}{\|x_+\|}, \mathbf{v} = \frac{y_+}{\|y_+\|}, \text{sigma} = \mu_+$
11. **else**
12. $\mathbf{u} = \frac{x_-}{\|x_-\|}, \mathbf{v} = \frac{y_-}{\|y_-\|}, \text{sigma} = \mu_-$
13. **end if**
14. $w_{pj} = \sqrt{S_{jj} \times \text{sigma}} \times u, p = 1, \dots, m$ dan
15. $h_{jq} = \sqrt{S_{jj} \times \text{sigma}} \times v, q = 1, \dots, n$
16. **end for**

Output: Matriks W ukuran $m \times k$ dan matriks H ukuran $k \times n$

Berikut adalah contoh dari ilustrasi inisialisasi NNDSVD:

Misalkan diberikan matriks A berukuran $m \times n$, dengan $m = 2$ dan $n = 3$

$$A = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & -1 \end{bmatrix} \quad (3.10)$$

Triplet utama dari A diperoleh dengan metode truncated SVD dengan langkah sebagai berikut:

$$B = AA^T = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & -1 \end{bmatrix} \begin{bmatrix} 3 & -1 \\ 1 & 3 \\ 1 & -1 \end{bmatrix} = \begin{bmatrix} 11 & 1 \\ 1 & 11 \end{bmatrix} \quad (3.11)$$

Berdasarkan persamaan (3.11) nilai eigen dari matriks B adalah:

$$\det(\lambda I - B) = 0 \quad (3.12)$$

$$\det\left(\begin{bmatrix} 11 & 1 \\ 1 & 11 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) = 0 \quad (3.13)$$

$$\det\left(\begin{bmatrix} 11 & 1 \\ 1 & 11 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}\right) = 0 \quad (3.14)$$

$$\det \begin{bmatrix} 11 - \lambda & 1 \\ 1 & 11 - \lambda \end{bmatrix} = 0 \quad (3.15)$$

$$\det((11 - \lambda)(11 - \lambda) - 1^2) = 0, \text{ diperoleh nilai } \lambda_1 = 12 \text{ dan } \lambda_2 = 10$$

Nilai singular A adalah $\sigma_1 = \sqrt{12}, \sigma_2 = \sqrt{10}$, sehingga diperoleh :

$$\Sigma = S = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \sqrt{12} & 0 \\ 0 & \sqrt{10} \\ 0 & 0 \end{bmatrix} \quad (3.16)$$

Untuk menentukan matriks U maka terlebih dahulu menentukan nilai vektor eigen berdasarkan persamaan (3.15), sehingga diperoleh :

$$(\lambda I - B)(X) = 0 \quad (3.17)$$

$$\begin{bmatrix} \lambda - 11 & 1 \\ 1 & \lambda - 11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \quad (3.18)$$

Perhatikan bahwa $\lambda_1 = 12$, kemudian substitusikan nilai λ_1 pada persamaan (3.18) sehingga diperoleh :

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \quad (3.19)$$

Persamaan (3. 19) dapat ditulis $x_1 + x_2 = 0$ dan $x_1 + x_2 = 0$, atau dapat ditulis $x_2 = 0$. Karena tidak terdapat keterangan mengenai x_1 , maka x_1 dapat dinyatakan sebagai suatu parameter, misalkan $x_1 = k$. Oleh karena itu, diperoleh :

$$u_1 = \begin{bmatrix} k \\ 1 \end{bmatrix} = k \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad (3. 20)$$

Kemudian lakukan hal yang sama untuk $\lambda_2 = 10$ disubstitusikan persamaan (3. 18) diperoleh:

$$\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0 \quad (3. 21)$$

$-x_1 + x_2 = 0$ dan $x_1 - x_2 = 0$, atau dapat ditulis $x_1 = 0$. Karena tidak terdapat keterangan mengenai x_2 , maka x_2 dapat dinyatakan sebagai suatu parameter, misalkan $x_2 = s$. Oleh karena itu, diperoleh vektor eigen u_2 , sebagai berikut :

$$u_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix} = s \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (3. 22)$$

Selanjutnya, dengan menormalisasikan u_1 dan u_2 :

$$u_1 = \frac{u_1}{\|u_1\|} = \frac{u_1}{\sqrt{(1)^2+(1)^2}} = u_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \text{ dan } u_2 = \frac{u_2}{\|u_2\|} = \frac{u_2}{\sqrt{(1)^2+(-1)^2}} = u_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix},$$

dengan menggabungkan u_1 dan u_2 diperoleh matriks U :

$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \quad (3. 23)$$

Tahap selanjutnya menentukan matriks V . Sama seperti mencari matriks U , perbedaannya hanya matriks V merupakan matriks ortogonal dari $C = A^T A = \begin{bmatrix} 3 & -1 \\ 1 & 3 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & -1 \end{bmatrix} = \begin{bmatrix} 11 & 5 \\ -1 & 9 \end{bmatrix}$. Melalui tahapan yang sama seperti mencari matriks U diperoleh :

$$V = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 2 \\ 1 & 0 & -5 \end{bmatrix}, V^t = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 0 \\ 1 & 2 & -5 \end{bmatrix}$$

Dari proses ini didapat hasil dekomposisi matriks A dengan SVD adalah sebagai berikut:

$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}, \Sigma = \begin{bmatrix} \sqrt{12} & 0 \\ 0 & \sqrt{10} \\ 0 & 0 \end{bmatrix}, V^t = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 0 \\ 1 & 2 & -5 \end{bmatrix}$$

Apabila dipilih $r=2$, Truncated SVD akan membentuk matriks aproksimasi untuk matriks A sebagai berikut

$$A_p = \tilde{U} \tilde{\Sigma} \tilde{V}^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{12} & 0 \\ 0 & \sqrt{10} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 0 \\ 1 & 2 & -5 \end{bmatrix}$$

Inisialisasi matriks W dengan $w_{p1} = \sqrt{s_{11}} \times u_{p1}$ dimana $p = 1, 2, \dots, m$ menjadi

Pertama kita pilih $p = 1$ maka $w_{11} = \sqrt{s_{11}} \times u_{11}$ kemudian dilanjutkan dengan dengan $p = 2$ maka $w_{21} = \sqrt{s_{11}} \times u_{21}$. Sehingga matriks awal W menjadi,

$$W = \begin{bmatrix} 2,42 & 0 \\ 2,42 & 0 \end{bmatrix} \quad (3.24)$$

Kemudian lakukan inisialisasi matriks H dengan $h_{1q} = \sqrt{s_{11}} \times v_{1q}^t$ dimana $q = 1, 2, \dots, n$. Dengan langkah pertama pilih $q = 1$ maka $h_{11} = \sqrt{s_{11}} \times v_{11}^t$. Lanjutkan langkah tersebut sampai $q = 3$ karena $n = 3$.

$$H = \begin{bmatrix} 3,46 & 6,92 & 3,46 \\ 0 & 0 & 0 \end{bmatrix} \quad (3.25)$$

Karena matriks A berukuran 2×3 , maka pilih $c \leq \min(m, n)$ misalkan 2, maka iterasi yang tersisa hingga j bernilai 2. Kemudian hitung $x = u_{p2}$ dan $y = v_{2q}^t$

$$x = \begin{bmatrix} 0,70 \\ -0,70 \end{bmatrix} \text{ dan } y = \begin{bmatrix} 2 & 1 & 0 \end{bmatrix}$$

Kemudian tentukan x_+, y_+, x_-, y_-

$$x_+ = \begin{bmatrix} 0,70 \\ 0 \end{bmatrix}, y_+ = [2 \quad 1 \quad 0], x_- = \begin{bmatrix} 0 \\ -0,70 \end{bmatrix}, y_- = [0 \quad 0 \quad 0]$$

Hitung *norm* dari $\|x_+\|, \|y_+\|, \|x_-\|, \|y_-\|$

$$\|x_+\| = 0,70, \|y_+\| = 2,23, \|x_-\| = 0,70, \|y_-\| = 0$$

Sehingga bisa dicari nilai $\mu_+ = \|x_+\| \|y_+\| = 1,56$ dan $\mu_- = \|x_-\| \|y_-\| = 0$

Selanjutnya cek apakah $\mu_+ > \mu_-$, karena $1,56 > 0$ maka akan dihitung $u = \frac{x_+}{\|x_+\|}, v = \frac{y_+}{\|y_+\|}$ dan $\sigma = \mu_+$

$$u = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, v = [0,89 \quad 0,44 \quad 0], \sigma = 1,56$$

Hitung $w_{p2} = \sqrt{s_{22} \times \sigma} \times u$ dan $h_{2q} = \sqrt{s_{22} \times \sigma} \times v$ kemudian masukkan nilainya ke dalam matriks W dan H sehingga didapatkan matriks W dan H sebagai berikut

$$W = \begin{bmatrix} 2,42 & 4,93 \\ 2,42 & 0 \end{bmatrix} \text{ dan } H = \begin{bmatrix} 3,46 & 6,92 & 3,46 \\ 4,39 & 2,17 & 0 \end{bmatrix}$$

Matriks W dan H yang dihasilkan dari metode NNDSVD akan digunakan sebagai inisialisasi awal algoritma NMF.

Penentuan inisialisasi awal ini akan digunakan dalam membentuk model NMF. Setelah ditentukan model maka selanjutnya dapat ditentukan RMSE dari model dengan cara membuat matriks yang berisi prediksi rating. Prediksi rating diperoleh dengan cara mengalikan matriks W dan matriks H yang dihasilkan model. Berikut merupakan contoh perhitungan RMSE:

Misalkan terdapat matriks awal $A = \begin{bmatrix} 7 & 1 & 1 \\ 2 & 3 & 4 \\ 5 & 0 & 6 \end{bmatrix}$ kemudian terdapat matriks

prediksi $WH = \begin{bmatrix} 6 & 1 & 1 \\ 3 & 3 & 3 \\ 5 & 3 & 6 \end{bmatrix}$. Dengan Rumus yang terdapat pada persamaan (2. 22)

maka,

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\widehat{WH}_{ui} - A_{ui})^2}$$

$$RMSE = \sqrt{\frac{(7-6)^2 + (1-1)^2 + (1-1)^2 + (2-3)^2 + (3-3)^2 + (4-3)^2 + (5-5)^2 + (0-3)^2 + (6-6)^2}{3 \times 3}}$$

$$RMSE = \frac{1 + 0 + 0 + 1 + 0 + 1 + 0 + 9 + 0}{9} = 1,33$$

3.4 Alur Penelitian

Alur penelitian dari penelitian sebagai berikut :

Langkah 1. Eksplorasi Data

Mengunduh data rating Airbnb New York City pada situs Kaggle <https://www.kaggle.com/c/airbnb>. Menghapus entri 0 pada data untuk melihat banyak jumlah dari masing-masing rating 1 sampai 5. Kemudian dataset ini diseleksi dengan hanya mengambil data penginapan yang memiliki paling sedikit 10 review, dan user yang telah menilai paling sedikit 3 penginapan.

Langkah 2. Membagi data sebanyak k-fold.

Membagi data ini dimaksudkan agar setiap bagian dalam data memiliki kesempatan yang sama untuk dilatih dan di uji sehingga menghasilkan akurasi yang terbaik. Setiap fold akan dibentuk model NMF untuk mendapatkan tingkat error dari masing-masing fold.

Langkah 3. Membentuk model NMF

Setelah penulis menentukan banyaknya fold kemudian dilakukan pembentukan model. Hasil dari pembentukan model ini menghasilkan matriks W dan H, dimana perkalian dari matriks W dan H akan disebut sebagai matriks prediksi. Pada langkah ini matriks W dan H akan dilakukan inisialisasi awal NNDSVD.

Langkah 4. Evaluasi Model

Menghitung hasil performa model dan merepresentasikannya dalam bentuk diagram batang. Performa model dilihat dengan melihat RMSE dari setiap k -component pada model NMF yang diuji. RMSE digunakan dengan menghitung akar dari pangkat kuadrat matriks prediksi dikurangi matriks sebenarnya dibagi banyaknya data.

Langkah 5. Pemilihan model terbaik

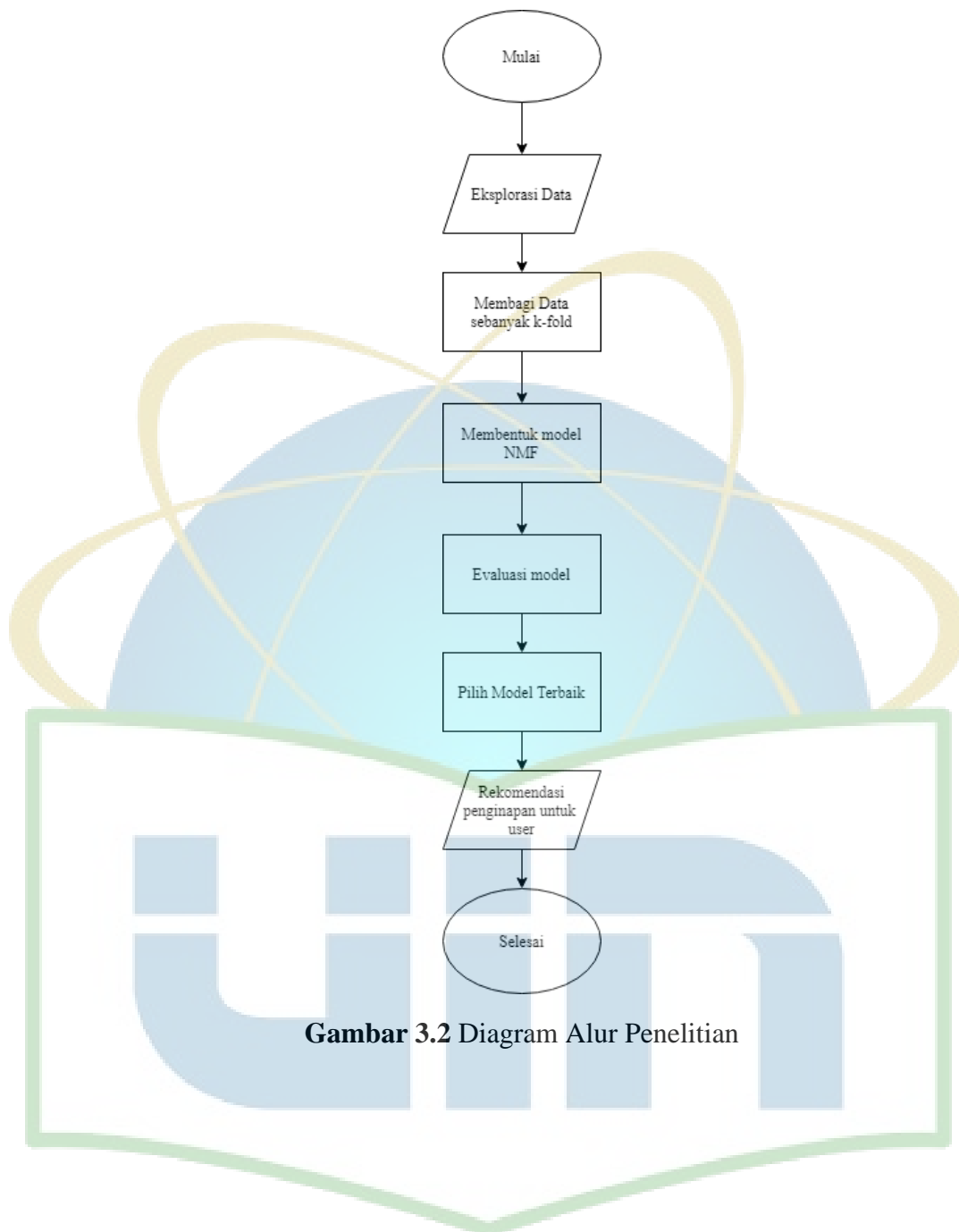
Memilih model terbaik dari metode NMF dengan melihat performa terbaik berdasarkan RMSE dan kompleksitas waktu.

Langkah 6. Rekomendasi

Setelah mendapatkan model terbaik maka model dapat digunakan untuk merekomendasikan penginapan kepada user. Rekomendasi dilihat berdasarkan entri rating pada matriks prediksi yang paling besar.

Berikut alur penelitian dari skripsi ini yang direpresentasikan berupa diagram.





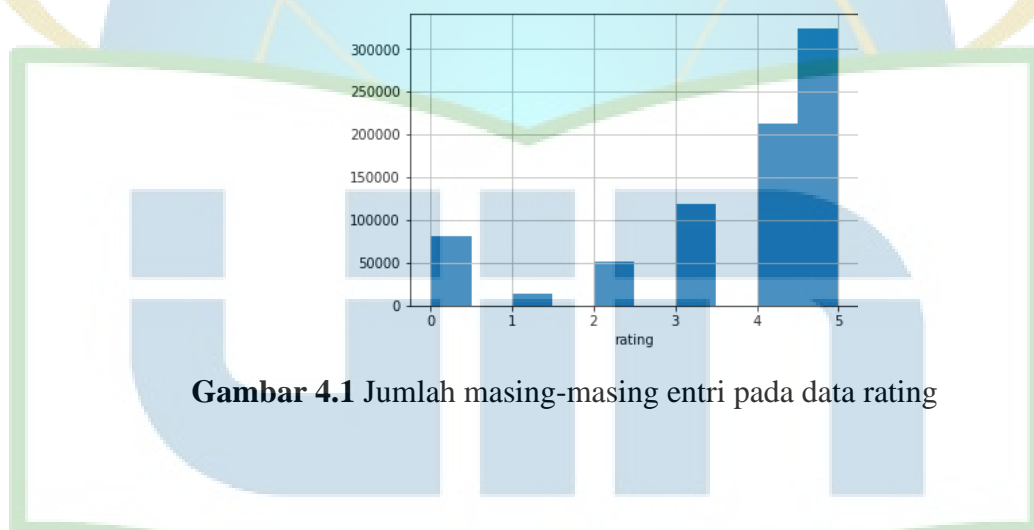
Gambar 3.2 Diagram Alur Penelitian

BAB IV

HASIL DAN PEMBAHASAN

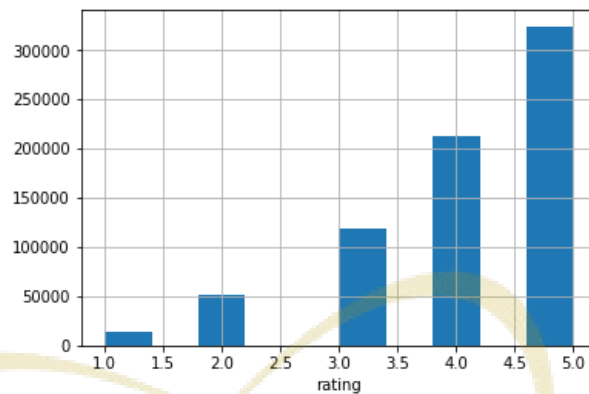
4.1 Eksplorasi Data

Bab ini akan menjelaskan hasil dari perhitungan NMF dengan inisialisasi random dan NNDSVD pada data sesungguhnya. Data yang digunakan dalam penelitian ini adalah database rating sebuah *marketplace* dalam bidang jasa penginapan Airbnb New York City. Data ini merupakan data yang bisa diakses untuk umum (*open source*) melalui situs (<https://www.kaggle.com/c/airbnb>). Secara umum data terdiri dari 800.995 rating dengan rentang 0-5 yang merupakan hasil penilaian dari 703051 user yang telah memberikan review terhadap 34839 daftar penginapan.



Gambar 4.1 Jumlah masing-masing entri pada data rating

Pada Gambar 4.1 entri terkecil adalah 0. Entri 0 dalam data artinya adalah user yang belum memberikan rating terhadap penginapan. Langkah pertama penulis menghilangkan entri 0 karena penulis ingin melihat rating yang sudah user berikan terhadap penginapan. Setelah menghilangkan entri 0 kini yang tersisa dalam data hanya rating 1-5. Kemudian masing-masing rating dilihat kembali jumlahnya seperti pada Gambar 4.2



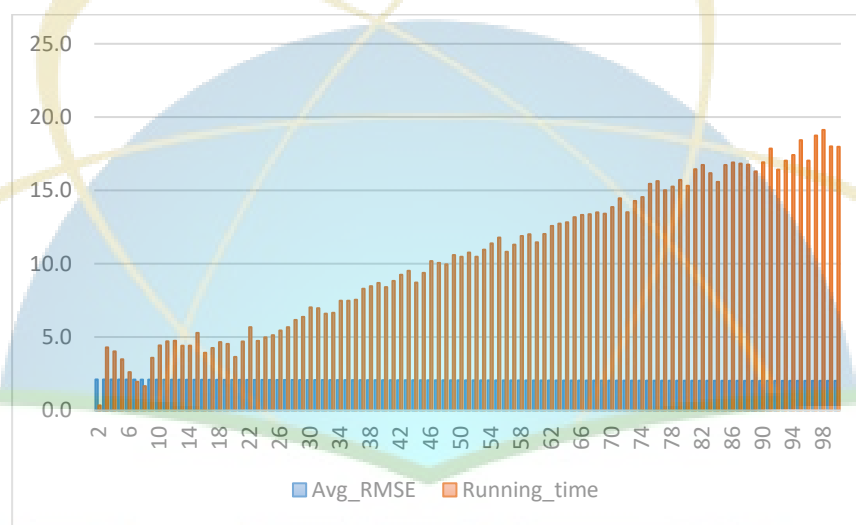
Gambar 4.2 Jumlah masing-masing rating setelah entri 0 dihilangkan.

Data rating setelah menghilangkan entri 0 sekarang berjumlah 719887. Karena penulis menggunakan pendekatan *collaborative filtering* untuk memprediksi rating maka penulis membatasi data penginapan yang memiliki paling sedikit 10 review, dan user yang telah menilai paling sedikit 3 penginapan. Pembatasan tersebut dimaksudkan agar penulis dapat melihat preferensi dari masing-masing user untuk digunakan sebagai rekomendasi kepada user lain. Sekarang penulis memiliki data `listing_id` berjumlah 26019, dan `reviewer_id` berjumlah 23079. Penulis mengambil nilai unik dari masing-masing `listing_id` dan `reviewer_id` sehingga didapatkan jumlah hasil akhir `listing_id` sebanyak 9503 dan `reviewer_id` sebanyak 5411. Jika ada sebanyak m user dan n daftar penginapan, maka matriks numpy akan berukuran 5411×9503 .

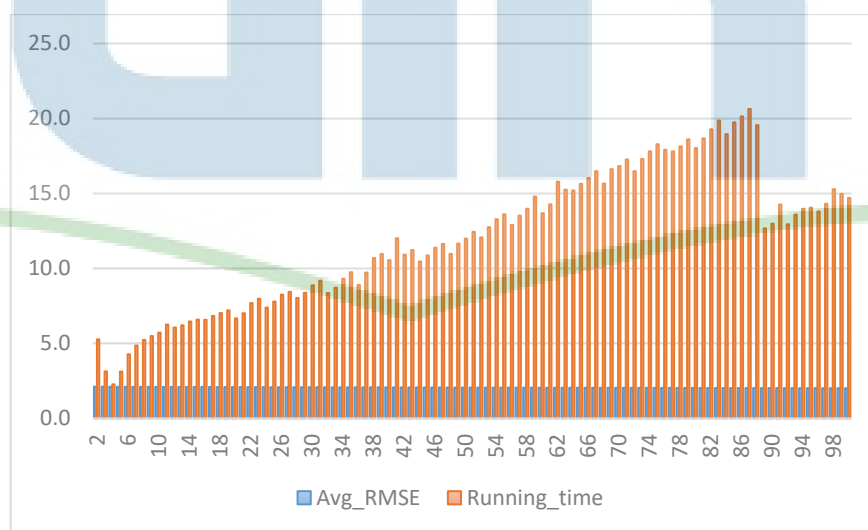
4.2 Hasil Penelitian

Langkah awal penelitian berdasarkan alur pada Gambar 3.2 adalah membaca data *Airbnb New York City* kemudian merepresentasikan data dalam bentuk matriks rating. Berdasarkan subab sebelumnya matriks sebenarnya pada penelitian ini berukuran $5411 \text{ user} \times 9503 \text{ penginapan}$ dengan entri rentang rating 1-5. Kemudian penulis membagi data sebanyak k -fold dengan memilih $k=5$. Dimana masing-masing fold akan dibentuk modelnya kemudian dihitung nilai RMSE. Setiap k -component pada NMF akan dihitung rata-rata RMSE dari kelima foldnya.

Untuk setiap parameter nilai n -component yang ditentukan pada model NMF akan dilakukan pengujian sehingga diketahui nilai formula dari masing-masing model. Dalam penelitian ini penulis melakukan simulasi dengan menggunakan algoritma NMF terhadap berbagai nilai k -component ($2 \leq k \leq 100$). Dimana setiap nilai k dilakukan 5 kali iterasi dengan proses K-fold cross validation. Berikut adalah perbandingan hasil rata-rata RMSE masing-masing nilai k -component untuk fold = 5 pada inisialisasi awal NNDSVD dan random.



Gambar 4.3 Rata-Rata RMSE dan Kompleksitas waktu dengan inisialisasi NNDSVD.



Gambar 4.4 Rata-Rata RMSE dan Kompleksitas waktu dengan inisialisasi random

Grafik batang berwarna oranye merupakan *running time* (kompleksitas waktu) dan batang berwarna biru merupakan rata-rata RMSE pada 5-fold. Pada Gambar 4.3 penulis mengasumsikan bahwa semakin besar nilai k maka semakin besar kompleksitas waktunya. Hal ini tidak terjadi untuk inisialisasi random Gambar 4.4 kompleksitas waktu cenderung naik ketika k berada di *range* 2-88 namun turun ketika $k = 89$ dan kembali naik hingga $k = 100$. Kemudian secara sekilas untuk nilai RMSE pada kedua inisialisasi NNDSVD dan random terlihat bahwa nilai tersebut konstan dan tidak ada perubahan yang signifikan. Untuk menganalisis lebih jelas masing-masing nilai dapat diperhatikan pada tabel berikut:

Tabel 4.1 Hasil Rata-rata RMSE dan Running time pada masing-masing k untuk inisialisasi NNDSVD dan random.

K	Inisialisasi NNDSVD		Inisialisasi Random	
	Avg_RMSE	Running_time	Avg_RMSE	Running_time
2	2.1129	0.34484	2.1129	5.2701
3	2.1105	4.3	2.1108	3.1335
4	2.1085	4.0273	2.1085	2.2715
5	2.1067	3.4894	2.1067	3.1197
6	2.105	2.612	2.1050	4.2723
7	2.1035	1.9551	2.1035	4.8586
8	2.102	1.6586	2.1020	5.2290
9	2.1005	3.5905	2.1005	5.4895
10	2.099	4.434	2.0990	5.7209
11	2.0976	4.7134	2.0976	6.2570
12	2.0963	4.75	2.0963	6.0596
13	2.0949	4.4149	2.0950	6.2084
14	2.0937	4.42	2.0937	6.4636
15	2.0923	5.2886	2.0924	6.5883
16	2.0911	3.9333	2.0912	6.5747
17	2.0899	4.2594	2.0900	6.8314
18	2.0886	4.6607	2.0887	7.0248
19	2.0874	4.5369	2.0875	7.2086

20	2.0863	3.6504	2.0863	6.6674
21	2.085	4.7031	2.0853	7.0111
22	2.0838	5.6785	2.0841	7.6927
23	2.0827	4.7442	2.0831	7.9757
24	2.0816	4.9856	2.0817	7.3905
25	2.0804	5.1321	2.0805	7.7923
26	2.0792	5.4514	2.0795	8.2540
27	2.0781	5.6782	2.0783	8.4401
28	2.0769	6.1716	2.0771	8.0302
29	2.0759	6.3805	2.0762	8.3674
30	2.0747	7.0282	2.0751	8.8766
31	2.0736	6.9715	2.0741	9.1853
32	2.0725	6.5998	2.0730	8.3607
33	2.0716	6.6586	2.0718	8.7171
34	2.0704	7.4831	2.0708	9.3154
35	2.0694	7.4826	2.0699	9.7497
36	2.0686	7.5502	2.0687	8.8972
37	2.0675	8.2987	2.0677	9.7268
38	2.0675	8.4728	2.0666	10.6954
39	2.0653	8.6944	2.0657	10.9619
40	2.0644	8.4111	2.0644	10.5536
41	2.0633	8.8324	2.0634	12.0128
42	2.0623	9.2517	2.0624	10.9210
43	2.0615	9.5226	2.0616	11.2281
44	2.0602	8.7202	2.0605	10.4685
45	2.0594	9.3813	2.0597	10.8713
46	2.0585	10.1846	2.0585	11.3839
47	2.0574	10.0621	2.0577	11.6346
48	2.0563	9.9639	2.0564	10.9710
49	2.0555	10.5967	2.0557	11.6577
50	2.0548	10.48	2.0547	11.9840
51	2.0535	10.7663	2.0538	12.4415
52	2.0526	10.4821	2.0529	12.0722
53	2.0515	10.9639	2.0518	12.7542
54	2.0508	11.391	2.0509	13.2848
55	2.0498	11.7886	2.0500	13.6127
56	2.0489	10.8194	2.0489	12.8986
57	2.0479	11.3044	2.0480	13.5222
58	2.047	11.8943	2.0473	13.9830

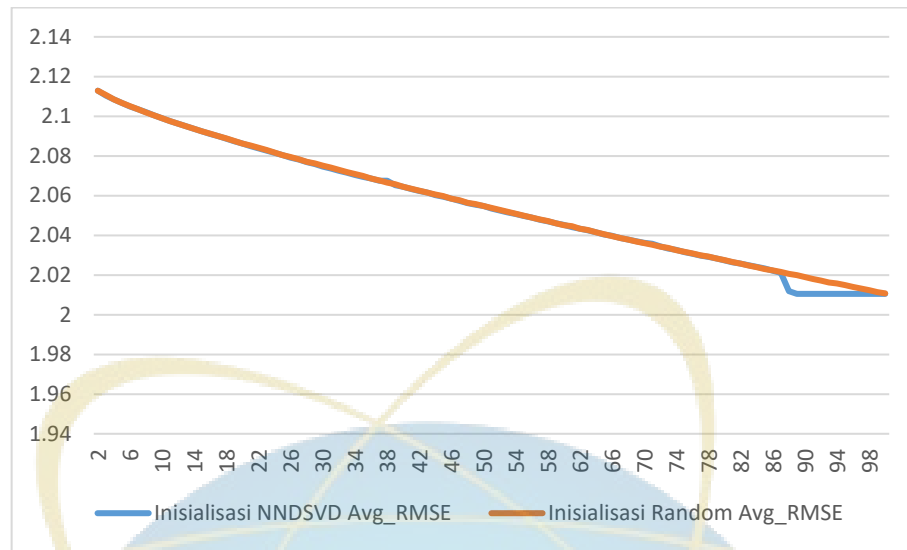
59	2.0461	12.0138	2.0461	14.7974
60	2.0453	11.4656	2.0450	13.6855
61	2.0442	12.0324	2.0446	14.2756
62	2.0434	12.5809	2.0433	15.7879
63	2.0426	12.7384	2.0426	15.2522
64	2.0415	12.8311	2.0415	15.2008
65	2.0405	13.1779	2.0406	15.6417
66	2.0398	13.3298	2.0397	16.0361
67	2.0387	13.3801	2.0387	16.4903
68	2.0379	13.5067	2.0378	15.6638
69	2.037	13.4174	2.0370	16.6209
70	2.0362	13.8684	2.0360	16.8286
71	2.0357	14.4655	2.0352	17.2727
72	2.0344	13.5171	2.0343	16.4855
73	2.0334	14.282	2.0336	17.2997
74	2.0327	14.5501	2.0325	17.8139
75	2.0316	15.4519	2.0317	18.2876
76	2.0308	15.6371	2.0309	17.9166
77	2.0298	15.0164	2.0300	17.8082
78	2.0292	15.2633	2.0293	18.1414
79	2.0284	15.718	2.0284	18.6084
80	2.0274	15.3187	2.0275	18.0298
81	2.0264	16.4493	2.0266	18.6798
82	2.0257	16.7384	2.0257	19.2769
83	2.025	16.1793	2.0247	19.8644
84	2.0242	15.5826	2.0240	18.9542
85	2.0234	16.7349	2.0229	19.7447
86	2.0222	16.909	2.0223	20.1400
87	2.021	16.8313	2.0216	20.6343
88	2.0119	16.7608	2.0206	19.5556
89	2.0105	16.2983	2.0199	12.6695
90	2.0105	16.9196	2.0190	12.9855
91	2.0105	17.8557	2.0180	14.2651
92	2.0105	16.4204	2.0172	12.9461
93	2.0105	17.0328	2.0162	13.5943
94	2.0105	17.4169	2.0157	13.9784
95	2.0105	18.43	2.0150	14.0534
96	2.0105	17.0347	2.0140	13.8012
97	2.0105	18.7444	2.0131	14.3158

98	2.0105	19.1272	2.0123	15.2854
99	2.0105	18.0046	2.0114	14.9744
100	2.0105	17.9785	2.0107	14.7024

Pada Tabel 4.1 penulis membulatkan nilai RMSE dan Running time hingga 4 angka setelah koma. Rata-rata RMSE paling kecil diperoleh pada saat $k = 100$ yaitu sebesar 2.010572361224466 untuk inisialisasi NNDSVD dan 2,010742685120310 pada inisialisasi random. Penentuan k terbaik dilakukan dengan memperhatikan nilai rata-rata RMSE terkecil dan juga kompleksitas waktu yang singkat, sementara dalam hasil yang diperoleh nilai RMSE terkecil juga membuat running time semakin besar.

Untuk inisialisasi NNDSVD penulis melihat saat $k = 89$ menuju $k = 100$ nilai RMSE mulai mengalami penurunan yang tidak signifikan bahkan cenderung sama saat dibulatkan yaitu sebesar 2.0105. Oleh karena itu penulis mengambil nilai terbaik dengan memperhatikan nilai RMSE dan running time pada saat $k = 89$ dengan running time selama 16.2983 menit.

Untuk inisialisasi random penulis melihat semakin besarnya nilai k maka nilai RMSE semakin kecil walaupun tidak mengalami perubahan yang signifikan. Oleh karena itu penulis menyimpulkan untuk mengambil nilai terbaik pada saat $k = 89$ dengan RMSE sebesar 2.0199 dan running time selama 12.6695 menit. Berikut merupakan perbandingan rata-rata RMSE untuk inisialisasi NNDSVD dan random dalam bentuk grafik.



Gambar 4.5 Perbandingan Rata-rata RMSE inisialisasi NNDSVD dan random

4.3 Hasil Rekomendasi

Setelah didapatkan model terbaik pada saat $k = 89$ maka penulis dapat melihat rekomendasi penginapan untuk user. Rekomendasi didapatkan dengan cara melihat matriks prediksi rating yang diperoleh pada saat $k = 89$. Entri dari matriks tersebut diurutkan berdasarkan nilai paling besar. Nilai paling besar ini merupakan tingkat kesukaan user terhadap penginapan yang direkomendasikan berdasarkan matriks prediksi. Sebagai contoh penulis ingin merekomendasikan user yang ada pada index ke-100 yaitu user 30031.

Recommendations listing for user 30031:

0.304	2135948
0.225	36435555
0.128	232419
0.12	7582057
0.1	32950436

Gambar 4.6 Hasil rekomendasi untuk user index ke-100

Pada Gambar 4.6 didapatkan 5 rekomendasi teratas untuk user 30031 adalah penginapan dengan identitas 2135948, 36435555, 232419, 7582057, 32950436.



BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil penelitian yang telah penulis lakukan pada bab sebelumnya terhadap data rating *Airbnb New York City* menggunakan dua buah inisialisasi yaitu NNDSVD dan Random, penulis menyimpulkan bahwa:

1. Rata-rata RMSE paling kecil diperoleh pada saat $k = 100$ yaitu sebesar 2.0105 untuk inisialisasi NNDSVD dan 2,0107 pada inisialisasi random dengan iterasi sebanyak 5-fold.
2. Penentuan k terbaik dilakukan dengan memperhatikan nilai rata-rata RMSE terkecil dan juga kompleksitas waktu yang singkat, sementara dalam hasil yang diperoleh nilai RMSE terkecil juga membuat running time semakin besar.
3. Untuk inisialisasi NNDSVD semakin besar nilai k maka semakin besar kompleksitas waktunya. Hal ini tidak terjadi untuk inisialisasi random dimana kompleksitas waktu cenderung naik ketika k berada di *range* 2-88 namun turun ketika $k = 89$ dan kembali naik hingga $k = 100$. Kemudian untuk nilai RMSE pada kedua inisialisasi NNDSVD dan random terlihat konstan dan tidak ada perubahan yang signifikan. Nilai terbaik didapatkan dengan memperhatikan nilai RMSE dan running time pada saat $k = 89$ dengan nilai RMSE 2.0105 pada inisialisasi NNDSVD dan 2.0199 pada inisialisasi random.

5.2 Saran

Setelah melakukan penelitian ini, penulis merasakan beberapa hal yang perlu dilakukan perbaikan dalam penelitian selanjutnya agar tercipta ilmu baru yang lebih berguna untuk kedepannya. Saran yang dapat penulis berikan untuk penelitian selanjutnya diantaranya adalah:

1. Eksperimen dilakukan dengan menggunakan data rating lain.

2. Pemilihan inisialisasi awal dirancang sendiri.
3. Melakukan analisis lanjutan terhadap metode NMF dengan memperhatikan parameter-parameter lain seperti *cost function*, dan metode pendekatan berbasis *content-based*.



DAFTAR PUSTAKA

- [1] Simon kemp & sarah moey, "DIGITAL 2019 SPOTLIGHT: ECOMMERCE IN INDONESIA," 2019. <https://datareportal.com/reports/digital-2019-ecommerce-in-indonesia> (accessed Mar. 22, 2020).
- [2] D. Bokde, S. Girase, and D. Mukhopadhyay, "Matrix Factorization model in Collaborative Filtering algorithms: A survey," *Procedia Comput. Sci.*, vol. 49, no. 1, pp. 136–146, 2015, doi: 10.1016/j.procs.2015.04.237.
- [3] B. Sarwar, G. Karypis, and J. Konstan, "Item-Based Collaborative Filtering Recommendation," *GroupLens Res. Group/Army HPC Res. Cent. Dep. Comput. Sci. Eng.*, pp. 286–295, 2001.
- [4] D. Sánchez-Moreno, A. B. Gil González, M. D. Muñoz Vicente, V. F. López Batista, and M. N. Moreno García, "A collaborative filtering method for music recommendation using playing coefficients for artists and users," *Expert Syst. Appl.*, vol. 66, pp. 1339–1351, 2016, doi: 10.1016/j.eswa.2016.09.019.
- [5] G. M. Del Corso and F. Romani, "Adaptive nonnegative matrix factorization and measure comparisons for recommender systems," *Appl. Math. Comput.*, vol. 354, pp. 164–179, 2019, doi: 10.1016/j.amc.2019.01.047.
- [6] R. Gaujoux and C. Seoighe, "A flexible R package for nonnegative matrix factorization," *BMC Bioinformatics*, vol. 11, 2010, doi: 10.1186/1471-2105-11-367.
- [7] R. T. Sutrisman and H. Murfi, "Analysis of non-negative double singular value decomposition initialization method on eigenspace-based fuzzy C-Means algorithm for Indonesian online news topic detection," in *2018 6th International Conference on Information and Communication Technology, ICoICT 2018*, 2018, doi: 10.1109/ICoICT.2018.8528791.
- [8] Z. Zheng, J. Yang, and Y. Zhu, "Initialization enhancer for non-negative matrix factorization," *Eng. Appl. Artif. Intell.*, vol. 20, no. 1, pp. 101–110, 2007, doi: 10.1016/j.engappai.2006.03.001.
- [9] O. Krasnoshchok and Y. Lamo, "Extended content-boosted matrix factorization algorithm for recommender systems," *Procedia Comput. Sci.*, vol. 35, no. C, pp. 417–426, 2014, doi: 10.1016/j.procs.2014.08.122.
- [10] G. Chen, F. Wang, and C. Zhang, "Collaborative filtering using orthogonal nonnegative matrix tri-factorization," *Inf. Process. Manag.*, vol. 45, no. 3, pp. 368–379, 2009, doi: 10.1016/j.ipm.2008.12.004.

- [11] H. Anton, "Anton - Elementary Linear Algebra with Applications 10e," *Methods Enzymol.*, 2009, doi: 10.1016/S0076-6879(09)05418-4.
- [12] Howard Anton and Chris Rorres, *Elementary Linear Algebra 11th Edition*. Canada Wiley, 2014.
- [13] R. L. Burden and J. D. Faires, *Numerical Analysis 9th Edition*. 2011.
- [14] A. G. Williamson and B. Jacob, "Linear Algebra," *Math. Gaz.*, 1990, doi: 10.2307/3618165.
- [15] G. W. S., G. H. Golub, and C. F. Van Loan, "Matrix Computations.," *Math. Comput.*, 1991, doi: 10.2307/2008552.
- [16] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," *Adv. Artif. Intell.*, 2009, doi: 10.1155/2009/421425.
- [17] T. T. Wong, "Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation," *Pattern Recognit.*, 2015, doi: 10.1016/j.patcog.2015.03.009.
- [18] Z. Xiong, Y. Cui, Z. Liu, Y. Zhao, M. Hu, and J. Hu, "Evaluating explorative prediction power of machine learning algorithms for materials discovery using k-fold forward cross-validation," *Comput. Mater. Sci.*, 2020, doi: 10.1016/j.commatsci.2019.109203.
- [19] C. Boutsidis and E. Gallopoulos, "SVD based initialization: A head start for nonnegative matrix factorization," *Pattern Recognit.*, 2008, doi: 10.1016/j.patcog.2007.09.010.

LAMPIRAN

Source Code Algoritma NMF

Memasukkan semua modul python

```
import numpy as np
import pandas as pd
import os
import time
import matplotlib.pyplot as plt
import gc
from scipy.sparse import csr_matrix
from scipy import sparse
from math import sqrt
from numpy import random as rd
# ML libraries
from sklearn.decomposition import NMF
from sklearn.utils.extmath import randomized_svd
from sklearn.metrics import mean_squared_error
```

Memanggil Data yang Digunakan

```
data =
pd.read_csv("""C:\\Users\\uin\\WinPython\\notebooks\\"""+'rating_prediction_by_
_sentiment_analysis.csv')
```

Membersihkan Data

```
data = data[data.rating !=0]
data.head()
counts_listing = data['listing_id'].value_counts()
counts_reviewer = data['reviewer_id'].value_counts()
```

```

selection_1 = data[data['listing_id'].isin(counts_listing[counts_listing >=
10].index)]

selection_final =
selection_1[selection_1['reviewer_id'].isin(counts_reviewer[counts_reviewer >
3].index)]

selection_final.shape

pivoted = pd.pivot_table(selection_final, values='rating', index='reviewer_id',
columns='listing_id').fillna(0)

Xround = np.ceil(pivoted)
X= pivoted.to_numpy()

```

Definisi Rumus RMSE

```

def calc_rmse(actual, pred):
    n = actual.sum() # this is we only want to sum entries that exist
    se = (actual - pred)**2

    return sqrt((1/n)*se.sum())

```

Code Prediksi Rating

```

def predict(X_hat, actual, ref, filter_actual, idx, n_rec):
    df_to_get_clauses = ref
    clauses = df_to_get_clauses.columns.values

    if filter_actual == True:
        diff = X_hat - actual
        diff = np.clip(diff,0,1)
    else:
        diff = X_hat

    print("For contract", idx, "the top", n_rec, "recommended clauses are:")

    return pd.DataFrame(data=diff[idx,:], index = clauses).sort_values(by=0, axis =
0, ascending = False).head(n_rec)

```

Membangun Model dan Membagi Data Sebanyak *k*-Fold

```
def train_val_NMF_model(data, components, alph, cross_val, method, export,
verbose):

    if type(verbose) != bool:

        raise ValueError("'verbose' variable is not boolean-type, use 'True' or 'False'
to control verbose")

    else:

        pass

    o_start = time.perf_counter()
    errors = []
    cv_errors = []
    config = []
    Ws = []
    Hs = []

    print("Initialisation:", method)
    print("Training and validating...")
    for comp in components:
        start = time.perf_counter()
        for alphas in alph:
            print("Config (components, alpha):",[comp,alphas])
            for cv in range(cross_val):
                np.random.shuffle(data) # shuffle data to perform cv and get rmse

                NMF_model = NMF(
                    verbose = verbose,
                    n_components = comp,
                    init = method,
                    solver = 'cd',
                    beta_loss = 'frobenius', # also called Euclidean Norm
                    tol = 1e-4,
```

```

        random_state = 0,
        alpha = alphas,
        max_iter = 1000
    )

```

```

    if verbose == True:
        print("Computing W...")
    else:
        pass

```

```

    W = NMF_model.fit_transform(data)

```

```

    if verbose == True:
        print("Computing H...")
    else:
        pass

```

```

        pass

```

```

    H = NMF_model.components_

```

```

    cv_error = calc_rmse(X, (W@H))

```

```

    #error = sqrt(error.mean())

```

```

    print(cv, "-fold" , "train_rmse:", cv_error)

```

```

    cv_errors.append(cv_error)

```

```

print("Avg rmse:", np.mean(cv_error))

```

```

print("Training time elapsed in minutes: ", (time.perf_counter() - start)/60)

```

```

print("")

```

```

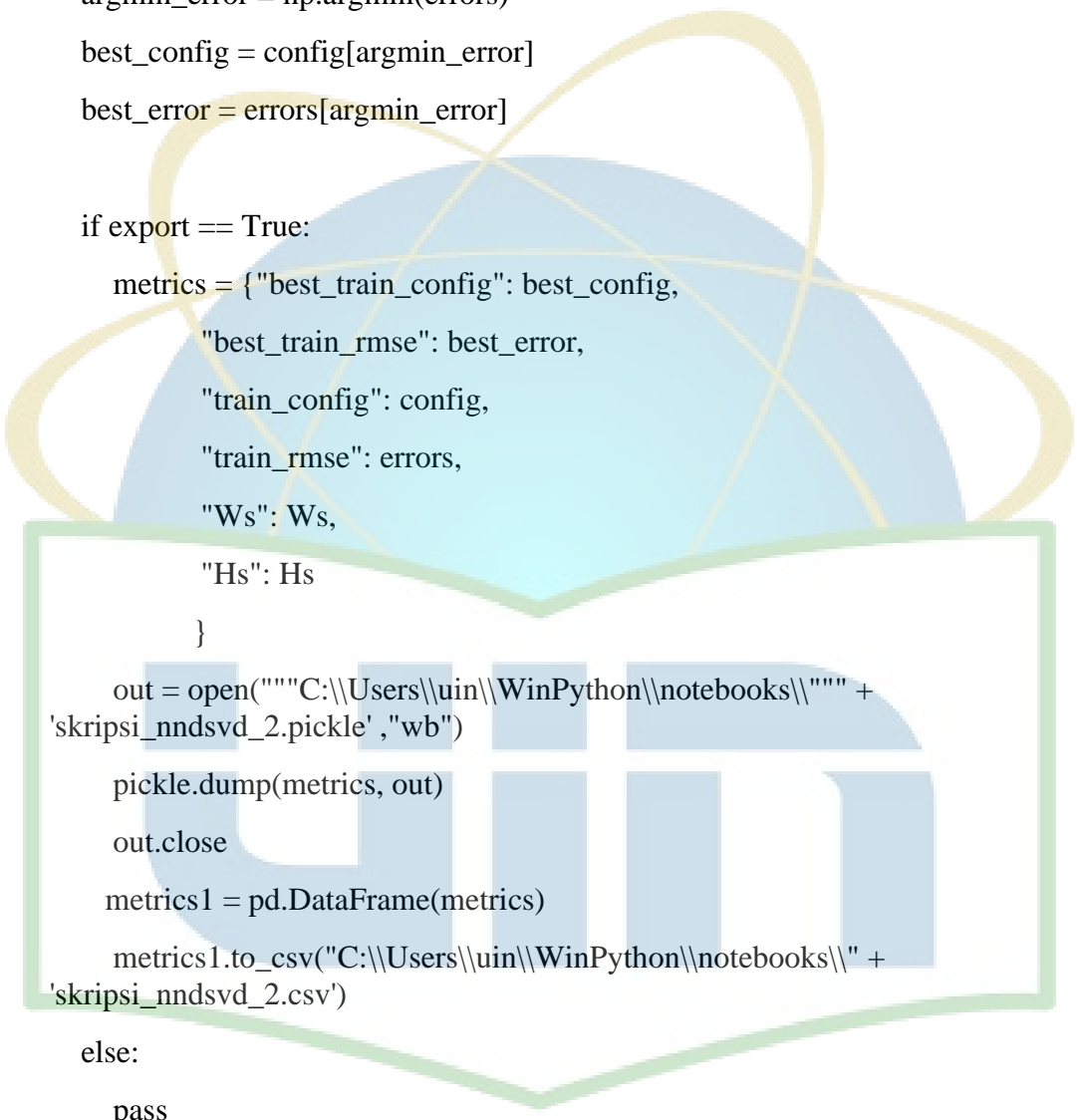
errors.append(np.mean(cv_error))

```

```

config.append([comp, alphas])

```



```

Ws.append(sparse.csr_matrix(W))
Hs.append(sparse.csr_matrix(H))
gc.collect()

argmin_error = np.argmin(errors)
best_config = config[argmin_error]
best_error = errors[argmin_error]

if export == True:
    metrics = {"best_train_config": best_config,
               "best_train_rmse": best_error,
               "train_config": config,
               "train_rmse": errors,
               "Ws": Ws,
               "Hs": Hs
              }

    out = open("C:\\Users\\uin\\WinPython\\notebooks\\" +
'skripsi_nndsvd_2.pickle', "wb")
    pickle.dump(metrics, out)
    out.close
    metrics1 = pd.DataFrame(metrics)
    metrics1.to_csv("C:\\Users\\uin\\WinPython\\notebooks\\" +
'skripsi_nndsvd_2.csv')
else:
    pass

print("Training and validating complete")
print("Total time elapsed in minutes: ", (time.perf_counter() - o_start)/60)
print("")
print("Best configuration:", best_config, "with error:", best_error)
print("Subset W, H at index:", argmin_error)

```

```
print("-----")
```

```
return errors, config, Ws, Hs
```

Uji model dengan k -component 2-100

```
comp = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67,
68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89,
90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```

```
r_error, r_configs, r_Ws, r_Hs = train_val_NMF_model(data = X,
                                                    components = comp,
                                                    alph = [0.01],
                                                    cross_val = 5,
                                                    method = 'nndsvd',
                                                    export = True,
                                                    verbose = False)
```

Hasil Prediksi Rating

```
list_User = list(X.columns)
list_penginapan = X.index.tolist()
def pred_recomendation_usr(NMF_Xhat,usr_idx, list_user, list_penginapan, k):
    usr = list_User[usr_idx]
    df = pd.DataFrame(NMF_Xhat, columns= list_User, index= list_penginapan)
    df.sort_values(by=[usr], inplace=True, ascending=False)
    recom = df[usr].index.tolist()[:k]
    value_nmf = list(df[usr])
    print("Recommendations listing for user { }:\n".format(usr))
    for idx, dt in enumerate(recom):
        print('{0:.3}\t{1}'.format(float(value_nmf[idx]), dt))
pred_recomendation_usr(NMF_Xhat, 50, list_User, list_penginapan, 5)
```