

**SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS  
COLLABORATIVE FILTERING**

**SKRIPSI**



oleh  
**DERRY SANTOSO**  
**72140011**

PROGRAM STUDI SISTEM INFORMASI  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS KRISTEN DUTA WACANA  
TAHUN 2018

**SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS  
COLLABORATIVE FILTERING**

**SKRIPSI**



Diajukan kepada Program Studi Sistem Informasi Fakultas Teknologi Informasi  
Universitas Kristen Duta Wacana  
Sebagai Salah Satu Syarat dalam Memperoleh Gelar  
Sarjana Komputer

Disusun oleh

**DERRY SANTOSO**  
**72140011**

PROGRAM STUDI SISTEM INFORMASI  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS KRISTEN DUTA WACANA  
TAHUN 2018

## PERNYATAAN KEASLIAN SKRIPSI

Saya menyatakan dengan sesungguhnya bahwa skripsi dengan judul:

### **Sistem Rekomendasi E-Commerce Amazon Berbasis Collaborative Filtering**

yang saya kerjakan untuk melengkapi sebagian persyaratan menjadi Sarjana Komputer pada pendidikan Sarjana Program Studi Sistem Informasi Fakultas Teknologi Informasi Universitas Kristen Duta Wacana, bukan merupakan tiruan atau duplikasi dari skripsi keserjanaan di lingkungan Universitas Kristen Duta Wacana maupun di Perguruan Tinggi atau instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Jika dikemudian hari didapati bahwa karya ilmiah ini adalah hasil plagiasi atau tiruan dari karya ilmiah lain, saya bersedia dikenai sanksi sesuai aturan yang berlaku di Universitas Kristen DutaWacana.

Yogyakarta, 26 Juni 2018



Derry Santoso

72140011

## **HALAMAN PERSETUJUAN**

**Judul Skripsi : Sistem Rekomendasi E-Commerce Amazon Berbasis Collaborative Filtering**

**Nama Mahasiswa : Derry Santoso**

**NIM : 72140011**

**Matakuliah : Skripsi**

**Kode : SI4426**

**Semester : Genap**

**Tahun Akademik : 2017/2018**

Telah diperiksa dan disetujui di Yogyakarta,  
Pada tanggal 26 Juni 2018

**Dosen Pembimbing I**



**Yetli Oslan, SKom., M.T.**

**Dosen Pembimbing II**



**Erick Kurniawan, S.Kom., M.Kom.**

## HALAMAN PENGESAHAN

### SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS COLLABORATIVE FILTERING

Oleh: DERRY SANTOSO / 72140011

Dipertahankan di depan Dewan Penguji Skripsi  
Program Studi Sistem Informasi Fakultas Teknologi Informasi  
Universitas Kristen DutaWacana - Yogyakarta  
Dan dinyatakan diterima untuk memenuhi salah satu syarat memperoleh gelar  
Sarjana Komputer  
pada tanggal  
26 Juni 2018

Yogyakarta, 26 Juni 2018

Mengesahkan,

Dewan Penguji:

- |                                    |       |
|------------------------------------|-------|
| 1. Yetli Oslan, SKom.,M.T.         | ..... |
| 2. Erick Kurniawan, S.Kom., M.Kom. | ..... |
| 3. Argo Wibowo, ST., MT.           | ..... |
| 4. Katon Wijana, S.Kom., M.T..     | ..... |

Dekan

Ketua Program Studi

(Budi Susanto, S.Kom, MT)

(Drs. Jong Jek Siang, M.Sc.)

## ABSTRAKSI

Amazon adalah E-Commerce yang menjual berbagai macam barang, diantaranya: buku, komputer, dvd, dll. Dalam penelitian ini, data yang digunakan adalah data rating buku Amazon dari tahun 1995-2014. Penelitian dilakukan agar data rating tersebut dapat berguna untuk perkembangan E-Commerce dalam hal penjualan barang dengan membuat sistem rekomendasi dengan metode *Collaborative Filtering*. Sistem Rekomendasi akan memberikan rekomendasi sesuai dengan barang yang sudah pernah diberi rating oleh *user*.

Penelitian akan dimulai dari data rating yang masih dalam bentuk JSON. Data JSON akan dilakukan *filtering* berdasarkan *k-cores*. Data yang sudah difilter akan diolah menggunakan metode *Collaborative Filtering* dan akan dilakukan pembuatan model agar dapat dipanggil oleh API. API akan menyimpan model kedalam memori, lalu API akan diakses oleh aplikasi web. Hasil akhir penelitian adalah daftar rekomendasi yang sudah dicetak dalam aplikasi web.

Daftar rekomendasi yang dicetak ada 2 macam, yaitu: *item related* dan *user recommendation*. *Item Related* adalah daftar *item* yang berdekatan dengan *detail item* yang sedang dilihat oleh *user*. *User Recommendation* adalah rekomendasi yang akan didapatkan oleh user berdasarkan dengan rating yang sudah pernah diberikan oleh *user*.

Kata kunci: Collaborative Filtering, Machine Learning, Sistem Rekomendasi.

## **KATA PENGANTAR**

Skripsi ini dicetak dengan tujuan memberikan informasi tentang bagaimana melakukan pengolahan data rating dan menggunakan data tersebut agar berguna untuk perkembangan sebuah E-Commerce. Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa, pembimbing pertama ibu Yetli Oslan, pembimbing kedua pak Erick Kurniawan, dan teman-teman yang mendukung dalam penyusunan skripsi. Skripsi ini akan menjelaskan tentang data rating yang masih dalam bentuk JSON, dilakukan pengolahan, dan dicetak pada aplikasi berbasis web sebagai sistem rekomendasi.

Jika ada kata-kata yang salah penulis mohon maaf sebesar-besarnya. Atas perhatian saudara penulis mengucapkan terima kasih.

Yogyakarta, Juni 2018

Penulis Skripsi,



Derry Santoso

## Daftar Isi

PERNYATAAN KEASLIAN SKRIPSI .....	2
HALAMAN PERSETUJUAN .....	3
HALAMAN PENGESAHAN .....	4
ABSTRAKSI .....	5
KATA PENGANTAR .....	6
BAB 1 .....	9
1.1. Latar Belakang .....	9
1.2. Rumusan Masalah .....	9
1.3. Batasan Masalah .....	10
1.4. Spesifikasi Sistem .....	10
1.4.1. Spesifikasi software .....	10
1.4.2. Spesifikasi hardware .....	10
1.4.3. Spesifikasi aplikasi .....	11
1.4.4. Spesifikasi kecerdasan pembangun .....	11
1.4.5. Spesifikasi kecerdasan pengguna aplikasi .....	11
1.5. Tujuan dan Manfaat Penelitian .....	11
1.6. Metodologi Penelitian .....	11
1.7. Sistematika Penulisan .....	12
BAB 2 .....	13
2.1. <i>Machine Learning</i> .....	13
2.2. <i>Collaborative Filtering</i> .....	13
2.2.1. <i>Collaborative Filtering Cost Function</i> .....	14
2.2.2. <i>Collaborative Filtering Stochastic Gradient Descent</i> .....	15
2.2.3. <i>Regularization</i> .....	16
2.3. <i>Cross Validation</i> .....	17
2.4. <i>Cosine Similarity</i> .....	18
BAB 3 .....	19
3.1. Data Penelitian .....	19
3.2. Gambaran Flowchart Penelitian .....	19
3.2. Gambaran Sistem .....	24
3.2.1. <i>DFD</i> .....	24
3.2.2. <i>Use Case</i> .....	25
3.2.3. <i>Activity Diagram</i> .....	28
BAB 4 .....	29



4.1 Pembuatan Sistem .....	29
4.1.1 Filter Amazon Rating JSON Menggunakan MongoDB .....	29
4.1.2 Pembuatan Model dengan Metode Collaborative Filtering .....	30
4.1.3 Menyiapkan Model untuk produksi dan Flask API .....	36
4.1.4 Aplikasi Web Laravel .....	39
4.2 Analisis .....	44
BAB 5 .....	47
5.1 Kesimpulan .....	47
5.2 Saran .....	47
Daftar Pustaka .....	48

# **BAB 1**

## **PENDAHULUAN**

### **1.1. Latar Belakang**

(O'Brien & Marakas, 2007) E-Commerce bukan hanya sekedar tempat untuk melakukan jual beli, tetapi juga tempat untuk berkembang, promosi, mengirim, dan servis produk. Kelebihan E-commerce adalah sebesar/sekecil apapun bisnis retailnya, jarak fisik tidak akan menjadi penghalang untuk melakukan bisnis. Semua pebisnis dan konsumen di bumi memiliki peluang untuk melakukan transaksi pada E-commerce.

Aplikasi E-Commerce yang akan dibuat adalah C2C (Customer to Customer). Aplikasi diharapkan dapat memberikan rekomendasi yang bersifat personal untuk setiap pembeli. Sistem Rekomendasi dengan Metode Collaborative Filtering akan memberikan rekomendasi berdasarkan : sejarah produk yang sudah diberi rating oleh pembeli terdahulu, sejarah *user* yang sudah memberi rating pada produk, fitur referensi tiap *user* terhadap produk, sejarah rating pembelian *user*.

Penelitian akan berfokus pada bagaimana pengukuran keakuratan rekomendasi di dalam aplikasi E-Commerce yang sudah siap produksi. Pembuatan model Sistem Rekomendasi akan dilakukan menggunakan bahasa pemrograman python. Model yang sudah jadi akan dikirimkan melalui Application Programming Interface (API) python ke Laravel. Framework Laravel akan digunakan sebagai dasar aplikasi E-Commerce berbasis Web.

### **1.2. Rumusan Masalah**

1. Bagaimana mengimplementasikan Sistem Rekomendasi pada E-Commerce dalam memberikan rekomendasi buku yang bersifat pribadi untuk setiap pelanggan?
2. Bagaimana pengukuran keakuratan sistem rekomendasi yang akan diimplementasikan ke dalam Aplikasi E-Commerce berbasis Web?

### **1.3. Batasan Masalah**

1. Data *User* Amazon tahun 1995 – 2014 yang sudah pernah memberi rating pada produk buku dan kode produk buku yang diberi rating
2. Metadata produk Amazon dalam kategori buku pada dataset tahun 1995 – 2014.
3. Semua data berasal dari repositori Julian McAuley institusi UCSD (University of California San Diego) (R. He, 2016).
4. Sample rating akan diambil dari setiap *user* sebanyak 10 rating sebagai test set, sisa rating yang tidak diambil akan digunakan sebagai training set.
5. Stakeholder yang terkait adalah pembeli (*user*) dalam aplikasi E-Commerce berbasis web.
6. Metode machine learning yang digunakan adalah Collaborative Filtering.
7. Bahasa yang digunakan untuk memproses data dan pembuatan model sistem rekomendasi adalah Python dan Flask untuk memproses API.
8. Framework yang digunakan sebagai dasar aplikasi E-Commerce adalah Laravel dengan bahasa pemrograman PHP.
9. Aplikasi E-Commerce web akan berfokus pada fitur yang dibutuhkan untuk melakukan penelitian sistem rekomendasi.

### **1.4. Spesifikasi Sistem**

#### **1.4.1. Spesifikasi software**

1. Sistem Operasi Windows 10 64 bit
2. Browser Mozilla Firefox / Chrome versi terbaru

#### **1.4.2. Spesifikasi hardware**

1. 3.5 Ghz i7 6700HQ
2. 16 GB RAM
3. 1 Terabyte HDD
4. DirectX 12 graphics device
5. Nvidia GTX 1060 6GB VRAM
6. Akses Internet

7. Monitor 15”, resolusi (1920x1080)
8. Keyboard dan Mouse

#### **1.4.3. Spesifikasi aplikasi**

1. Sistem Rekomendasi mampu memberikan rekomendasi yang bersifat personal kepada *user* dengan metode Collaborative Filtering
2. Model Sistem Rekomendasi yang digunakan dalam aplikasi berbasis web sudah teroptimisasi dan siap untuk tahap produksi

#### **1.4.4. Spesifikasi kecerdasan pembangun**

1. Kemampuan dalam pengolahan data menggunakan Python.
2. Kemampuan pembuatan aplikasi web menggunakan framework Laravel dengan bahasa pemrograman PHP.
3. Kemampuan melakukan tes keakuratan dan optimisasi model sistem rekomendasi.

#### **1.4.5. Spesifikasi kecerdasan pengguna aplikasi**

1. Mampu menggunakan komputer dan mengakses website melalui browser
2. Memahami istilah bahasa inggris yang digunakan dalam aplikasi

### **1.5. Tujuan dan Manfaat Penelitian**

1. Membuat suatu sistem rekomendasi yang dapat memberikan rekomendasi buku ke tiap pelanggan secara pribadi menggunakan metode Collaborative Filtering.
2. Membuat Sistem Rekomendasi yang sudah siap digunakan sampai ke tahap produksi untuk aplikasi berbasis web.

### **1.6. Metodologi Penelitian**

#### **1. Studi Pustaka**

Mempelajari bagaimana cara pembuatan sistem rekomendasi menggunakan metode Collaborative Filtering melalui website dan jurnal ilmiah. Lalu mempelajari cara pemrosesan data menggunakan python dan cara mengirim

model yang sudah jadi ke aplikasi web berbasis Laravel menggunakan API. Terakhir adalah mempelajari cara optimisasi sistem rekomendasi pada aplikasi web dan cara pengukuran keakuratan model yang dibuat.

## **2. Pengumpulan Data**

Mengumpulkan data review buku Amazon melalui repositori Julian McAuley institusi UCSD

## **3. Analisis Data**

Analisa menggunakan sample rating yang diambil dari setiap *user* sebanyak 10 rating sebagai test set, sisa rating yang tidak diambil akan digunakan sebagai training set.

## **4. Rancangan Sistem**

Mendesain sistem yang akan dibangun

## **5. Implementasi Sistem**

Pembuatan model sistem rekomendasi menggunakan metode Collaborative Filtering dengan Python, pembuatan aplikasi E-Commerce Web menggunakan Laravel, membuat API untuk mengirim data dari python ke Laravel, tampilkan data hasil rekomendasi melalui aplikasi web.

## **6. Pengujian dan Analisis Sistem**

Melakukan uji keakuratan sistem rekomendasi menggunakan metode cross validation pada 10 data rating dari setiap *user* sebagai test set.

## **7. Penyelesaian Laporan**

Membuat kesimpulan, saran, kelebihan dan kelemahan sistem yang dibangun.

### **1.7 Sistematika Penulisan**

Bab 1 akan menjelaskan tentang latar belakang, rumusan masalah, batasan masalah, spesifikasi sistem, tujuan dan manfaat penelitian, metodologi penelitian, dan sistematika penulisan. Bab 2 berisi penjelasan teori metode yang akan digunakan dalam penelitian. Bab 3 berisi rancangan sistem, sedangkan Bab 4 akan dilakukan pembuatan, penerapan, dan analisis sistem. Bab 5 berisi penutup yang berisi tentang kesimpulan dan saran penelitian.

## BAB 2

### LANDASAN TEORI

#### 2.1. *Machine Learning*

(Kohavi & Provost, 1998) *Machine Learning* adalah bidang studi ilmiah yang berfokus pada pembuatan algoritma yang dapat belajar sendiri. (Bluma & Langley, 1997) Machine Learning digunakan untuk sistem yang memiliki data yang banyak, rumit, dan memiliki masalah dalam memfokuskan pada informasi yang relevan. E-Commerce dapat fokus memberikan rekomendasi buku pada pelanggan menggunakan machine learning.

#### 2.2. *Collaborative Filtering*

Menurut (Sarwar, Karypis, Konstan, & Riedl, 2002) *CF (Collaborative Filtering)* bekerja sebagai berikut: dibuat sebuah database yang berisi sejarah preferensi *user*, jika ada pelanggan baru yang cocok dengan preferensi database tersebut, maka sistem akan memberikan rekomendasi pada pelanggan baru tersebut.

Fase suatu Sistem Rekomendasi menurut (Isinkayea, Folajimib, & Ojokohc, 2015) adalah: proses pengumpulan data, lalu proses pembelajaran (machine learning dengan metode collaborative filtering), setelah itu rekomendasi akan diberikan untuk *user*, setelah *user* memberikan feedback pada rekomendasi, maka siklus akan kembali lagi ke tahap pengumpulan data.



Gambar 1 Skema Collaborative Filtering (Bossenbroek & Gringhuis, 2014)

Bagian Collaborative Filtering terdiri dari:

### 2.2.1. Collaborative Fitering Cost Function

(Ng, 2017) Cost Function berfungsi untuk mengukur seberapa akurat hypothesis ( $h\theta(x)$ ) yang merupakan hasil dari  $(\theta^{(j)})^T x^{(i)}$ . Algoritma Collaborative Filtering Cost Function tanpa Regularization adalah sebagai berikut:

$$J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}) = \frac{1}{2} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2$$

Keterangan :

- $y$  : berisi Matrix dengan dimensi jumlah\_buku x jumlah\_user untuk menyimpan rating  $y^{(i,j)}$  yang diberikan oleh *user* kepada buku. Range rating adalah 1-5
- $r$  : berisi Matrix yang menunjukkan apakah buku sudah diberi rating atau belum. Buku yang sudah diberi rating akan memiliki isi  $r^{(i,j)} = 1$ , sedangkan yang belum berisi  $r^{(i,j)} = 0$ . Algoritma memiliki peraturan sum  $r^{(i,j)} = 1$  sehingga data yang akan diproses oleh algoritma adalah data buku yang sudah pernah diberi rating. Dimensi matrix sama dengan  $y$ , yaitu jumlah\_buku x jumlah\_user.
- $x$  : Matrix feature buku yang akan digenerate secara random. Baris ke  $i$ -th menunjuk ke 1 judul buku. Bentuk  $x$  yang sudah mengalami factorization akan menggunakan transpose ( $T$ ):

$$X = \begin{bmatrix} - (x^{(1)})^T - \\ - (x^{(2)})^T - \\ \vdots \\ - (x^{(n_m)})^T - \end{bmatrix}$$

- $\theta$  (*Theta*): Matrix preferensi *user* terhadap buku yang akan digenerate secara random. Baris ke  $i$ -th menunjuk ke 1 *user*. Bentuk  $\theta$  yang sudah mengalami factorization akan menggunakan transpose ( $T$ ):

$$\text{Theta} = \begin{bmatrix} - (\theta^{(1)})^T - \\ - (\theta^{(2)})^T - \\ \vdots \\ - (\theta^{(n_u)})^T - \end{bmatrix}$$

### 2.2.2. Collaborative Filtering Stochastic Gradient Descent

(Ng, 2017) Algoritma yang digunakan untuk melakukan *minimize cost function*. Algoritma akan melakukan loop sampai nilai gradient *converge ke local/global minimum*. Algoritma gradient berikut belum menggunakan *regularization* :

$$\frac{\partial J}{\partial x_k^{(i)}} = \sum_{j:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)}$$
$$\frac{\partial J}{\partial \theta_k^{(j)}} = \sum_{i:r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)}.$$

Keterangan:

- $k$  : nomor iterasi

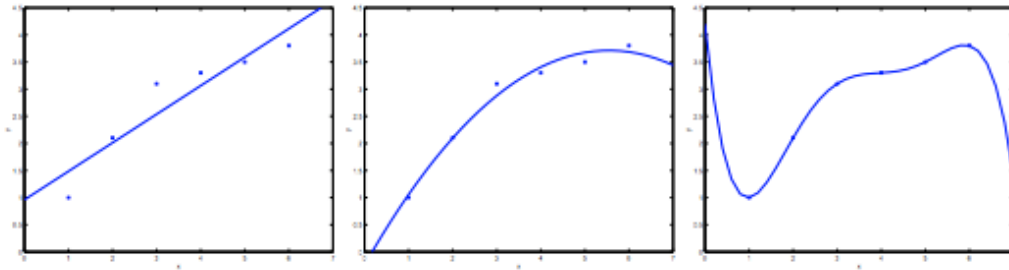
Theta dan  $x$  akan diupdate secara bersamaan pada setiap iterasi, sehingga prosesnya adalah:

1. Iterasi 1 : hitung  $\theta$  dengan algoritma gradient > hitung  $x$  dengan algoritma gradient > update  $\theta$  > update  $x$  > masuk ke iterasi 2
2. Iterasi 2 : hitung  $\theta$  dengan algoritma gradient > hitung  $x$  dengan algoritma gradient > update  $\theta$  > update  $x$  > masuk ke iterasi 3
3. dan seterusnya...



### 2.2.3. Regularization

(Ng, 2017) Regularization digunakan untuk mencegah overfitting. Overfitting terjadi jika terlalu banyak menggunakan feature, sehingga akan sangat cocok terhadap training set (cost function mendekati 0), tetapi garis yang dihasilkan akan kacau dan akan gagal dalam memprediksi nilai (rating) data baru.



Gambar 2 dari kiri ke kanan: grafik 1 (underfit dan High bias), grafik 2 (just right), grafik 3 (overfitting, High variance). a)

Berikut adalah algoritma Collaborative Filtering Cost Function yang sudah mengalami regularization:

$$J(x^{(1)}, \dots, x^{(n_m)}, \theta^{(1)}, \dots, \theta^{(n_u)}) = \frac{1}{2} \sum_{(i,j):r(i,j)=1} ((\theta^{(j)})^T x^{(i)} - y^{(i,j)})^2 + \left( \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2 \right) + \left( \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2 \right)$$

Keterangan :

- $\lambda$  (*lambda*) : regularization parameter. Jika terlalu besar maka akan terjadi underfitting, sehingga dibutuhkan percobaan berulang kali untuk menemukan jumlah lambda yang pas.
- Algoritma regularization ada 2 :  $\left( \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^n (\theta_k^{(j)})^2 \right)$  dan  $\left( \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^n (x_k^{(i)})^2 \right)$  karena didalam cost function collaborative filtering akan memproses *user* ( $\theta$ ) dan feature ( $x$ ) secara sekaligus.

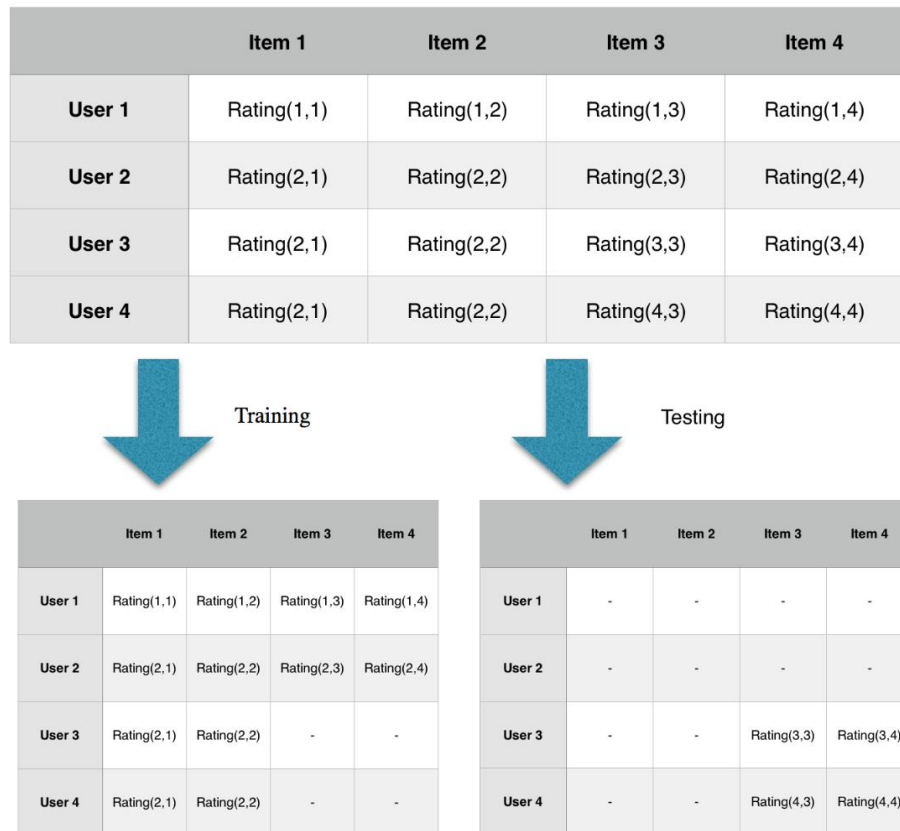
b) Berikut adalah algoritma Cost Function Gradient Descent yang sudah mengalami regularization :

$$\frac{\partial J}{\partial x_k^{(i)}} = \sum_{j:r(i,j)=1} ((\theta_k^{(j)})^T x^{(i)} - y^{(i,j)}) \theta_k^{(j)} + \lambda x_k^{(i)}$$

$$\frac{\partial J}{\partial \theta_k^{(j)}} = \sum_{i:r(i,j)=1} ((\theta_k^{(j)})^T x^{(i)} - y^{(i,j)}) x_k^{(i)} + \lambda \theta_k^{(j)}.$$

### 2.3. Cross Validation

(Barwicki, 2017) Cross Validation digunakan untuk melakukan benchmarking terhadap keakuratan suatu Sistem Rekomendasi yang menggunakan metode Collaborative Filtering.



Gambar 3 : User-Item utility matrix – Cross Validation

Sample rating akan diambil dari setiap *user* sebanyak 10 rating sebagai test set, sisa rating yang tidak diambil akan digunakan sebagai training set. Rating dari test set akan dibandingkan dengan rating prediksi yang dihasilkan oleh Collaborative Filtering (CF) dan diukur menggunakan Root Mean Squared Error (RMSE). Jika angka yang dihasilkan oleh RMSE makin kecil, maka akan menunjukkan bahwa sistem rekomendasi yang dibuat makin akurat.

## 2.4. Cosine Similarity

(Perone, 2013) *Cosine Similarity* adalah pengukuran dengan menghitung besar sudut cosine antara dua vector. Cosine similarity menghitung persamaan dari dot product untuk  $\cos \theta$ :

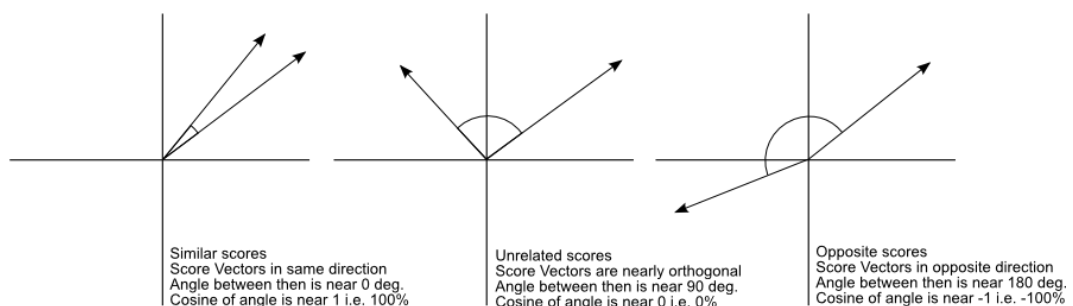
$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos \theta$$

$$\cos \theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

Keterangan :

- $\vec{a}$  : vector pertama
- $\vec{b}$  : vector kedua

Cosine similarity akan menghasilkan ukuran yang akan menjelaskan seberapa dekat vector pertama dengan kedua:



Gambar 4 : Hasil Cosine Similarity untuk berbagai macam vector,  
 1 (arah sama), 0 ( $90^\circ$ ), -1 (arah berlawanan)

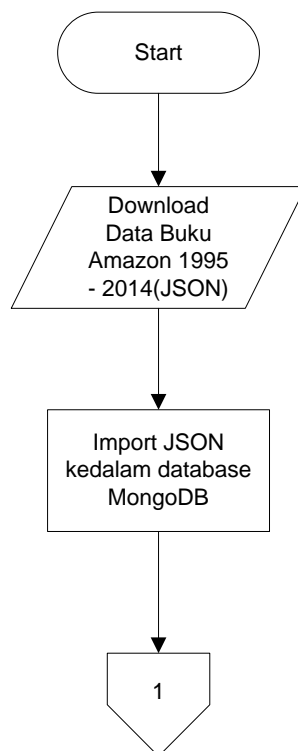
## BAB 3

### ANALISIS DAN RANCANGAN

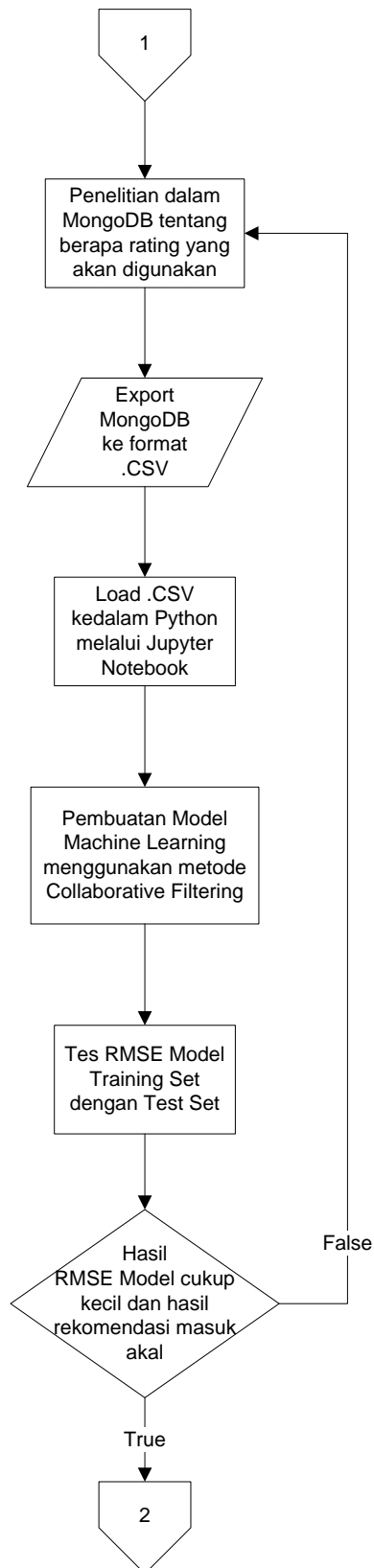
#### 3.1. Data Penelitian

1. Data *User* Amazon tahun 1995 – 2014 yang sudah pernah memberi rating pada produk buku dan kode produk buku yang diberi rating (8.898.041 review). Data rating *user* amazon yang diambil sudah difilter dengan menghilangkan *user* yang memberi rating kurang dari 5 buah rating (5-Core).
2. Metadata produk Amazon dalam kategori buku pada dataset tahun 1995 – 2014 (2.370.585 products).
3. Semua data berasal dari repositori Julian McAuley institusi UCSD (University of California San Diego) (<http://jmcauley.ucsd.edu/data/amazon/links.html>)

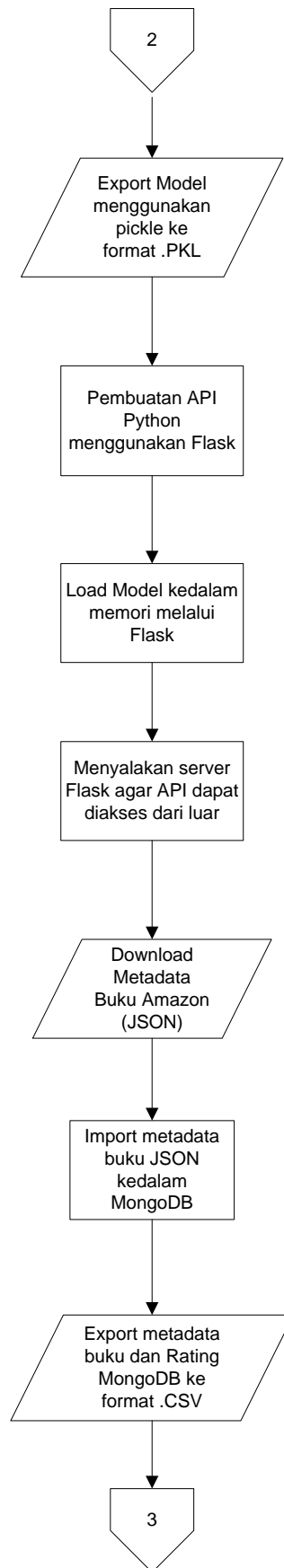
#### 3.2. Gambaran Flowchart Penelitian



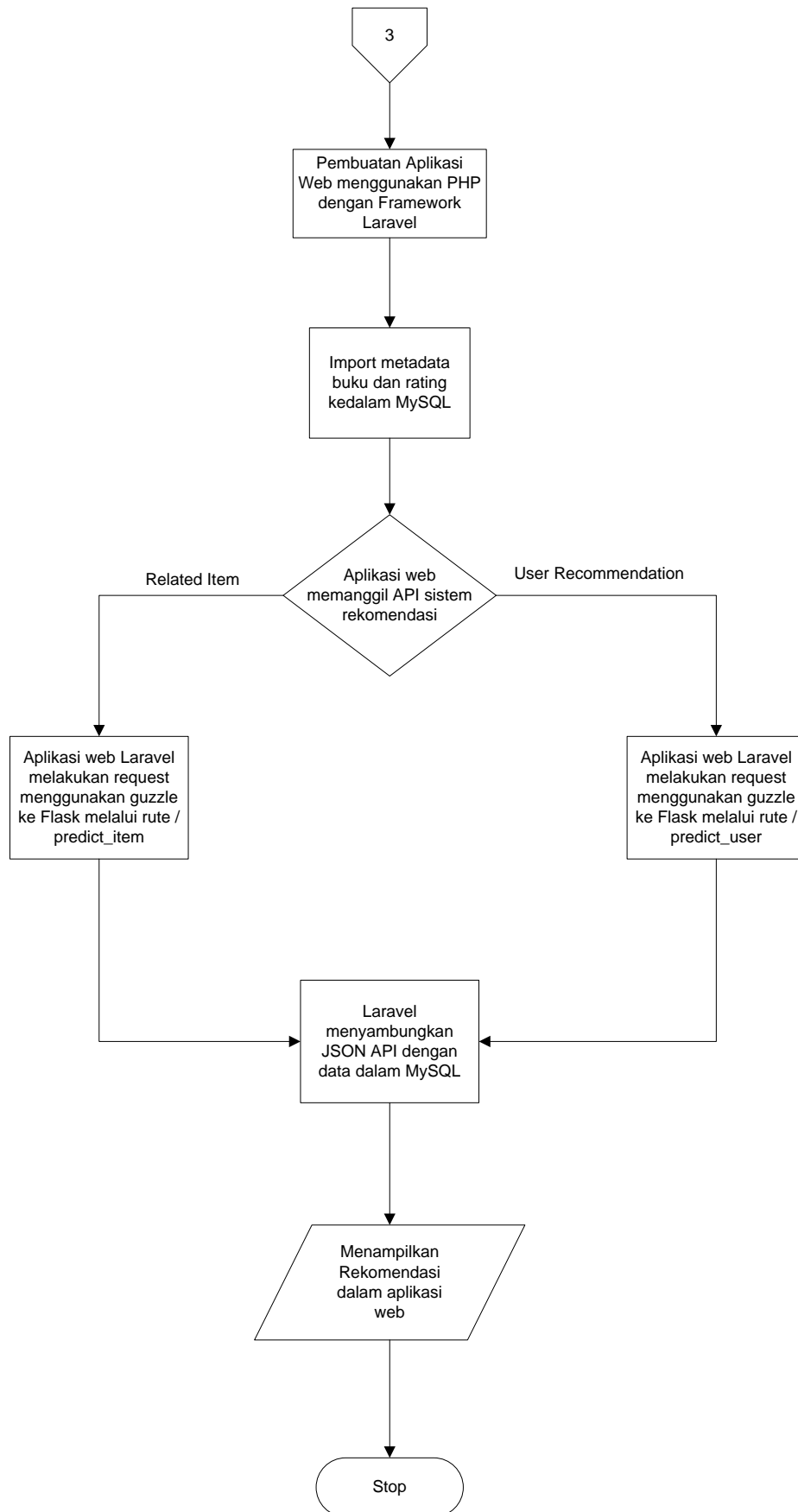
*Gambar 5 : Flowchart Penelitian bagian 1*



Gambar 5.2 : Flowchart Penelitian bagian 2



*Gambar 5.3 : Flowchart Penelitian bagian 3*



Gambar 5.4 : Flowchart Penelitian bagian 4

Data Rating Buku Amazon (JSON) akan di-load ke MongoDB. Didalam MongoDB akan dilakukan penelitian berapa rating yang dibutuhkan untuk penelitian. Setelah data rating difilter dalam MongoDB, akan *diexport* menjadi CSV. CSV akan di-load kedalam Python melalui Jupyter Notebook. Data akan dipisah menjadi training set dan test set, test set diambil dari 10 rating setiap *user* dan sisa ratingnya akan menjadi training set. Proses pembuatan model dilakukan menggunakan metode Collaborative Filtering. Setelah keluar hasil RMSE yang kecil dan keluaran rekomendasi masuk akal, model siap untuk *diexport*, jika hasil RMSE besar/keluaran rekomendasi tidak masuk akal, maka harus dilakukan penelitian kembali didalam MongoDB dan debugging kode python.

Model lalu akan *diexport* ke format PKL menggunakan pickle. API python dibuat menggunakan framework Flask. PKL akan di-load kedalam Flask. Setelah server Flask dinyalakan, maka API sistem rekomendasi siap dipanggil dari luar sebagai back end. Metadata buku (JSON) akan diimport kedalam MongoDB, lalu akan *diexport* sebagai CSV, data rating juga akan *diexport* menjadi CSV. Aplikasi Web mulai dibuat sebagai front end menggunakan Bahasa PHP dan framework Laravel. Metadata buku dan rating akan diimport kedalam MySQL agar dapat dipanggil oleh Laravel.

Input yang dilakukan *user* pada aplikasi web akan dikirimkan ke database, lalu akan dikirimkan ke model sistem rekomendasi kembali melalui API. Model sistem rekomendasi pada Flask yang siap dipakai akan menerima input *user* dan mengirim daftar rekomendasi ke aplikasi Laravel berbasis web melalui API. Input *user* akan digunakan sebagai preferensi buku *user* tersebut. Preferensi *user* yang berasal dari rating yang sudah diberikan *user* kepada *item*. Output yang diterima oleh *user* adalah Daftar rekomendasi *related item / user recommendation*.

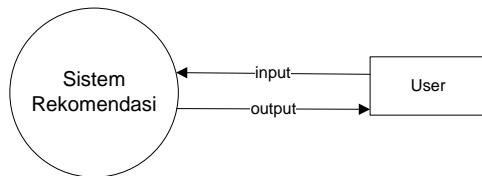


## 3.2. Gambaran Sistem

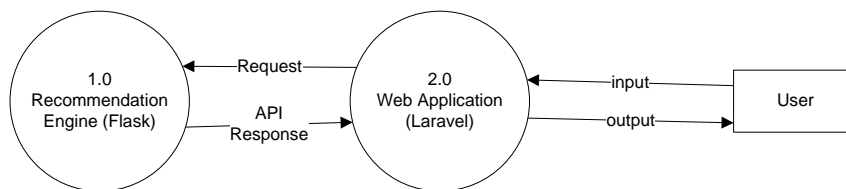
### 3.2.1 DFD

*Data Flow Diagram* menggambarkan proses aliran data yang terjadi dalam sistem sehingga jelas batasan kerja, interaksi, dan transformasi data dalam sistem tersebut. Simbol kotak menunjukkan entitas luar, arah panah menunjukkan aliran data, lingkaran menunjukkan proses, dan kotak tanpa garis sisi menunjukkan berkas atau penyimpanan data.

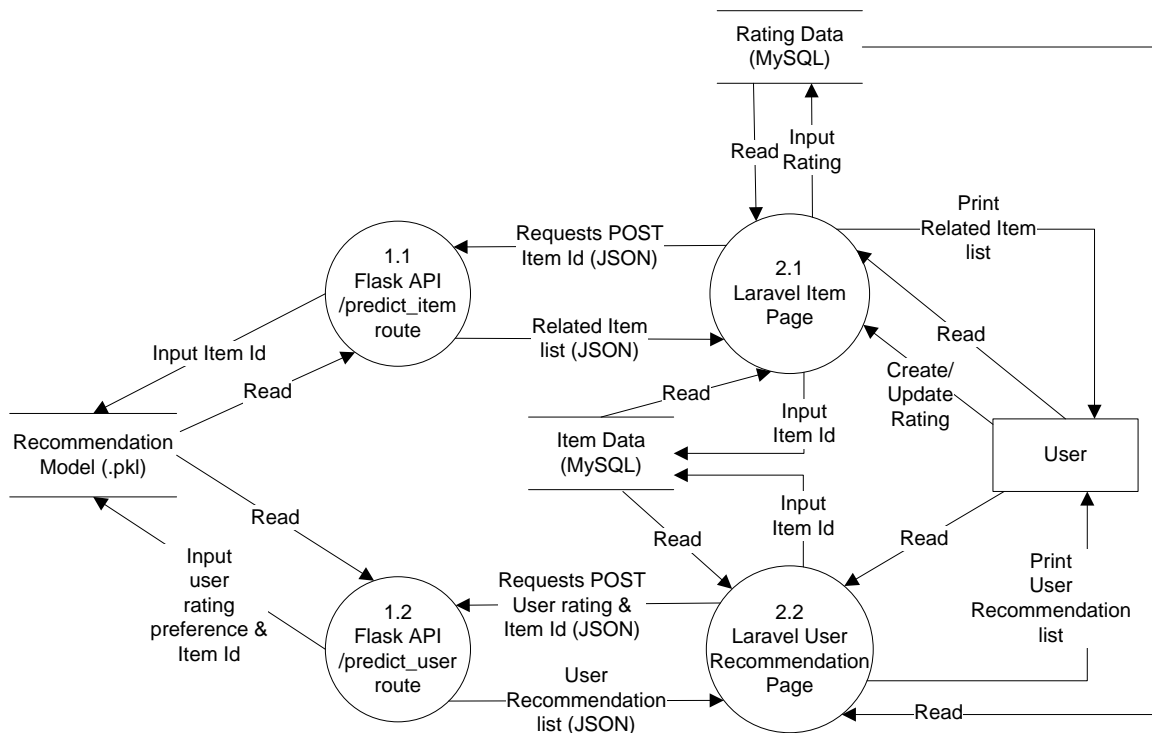
#### 3.2.1.2 Diagram Context (Level 0)



#### 3.2.1.2 DFD Level 1



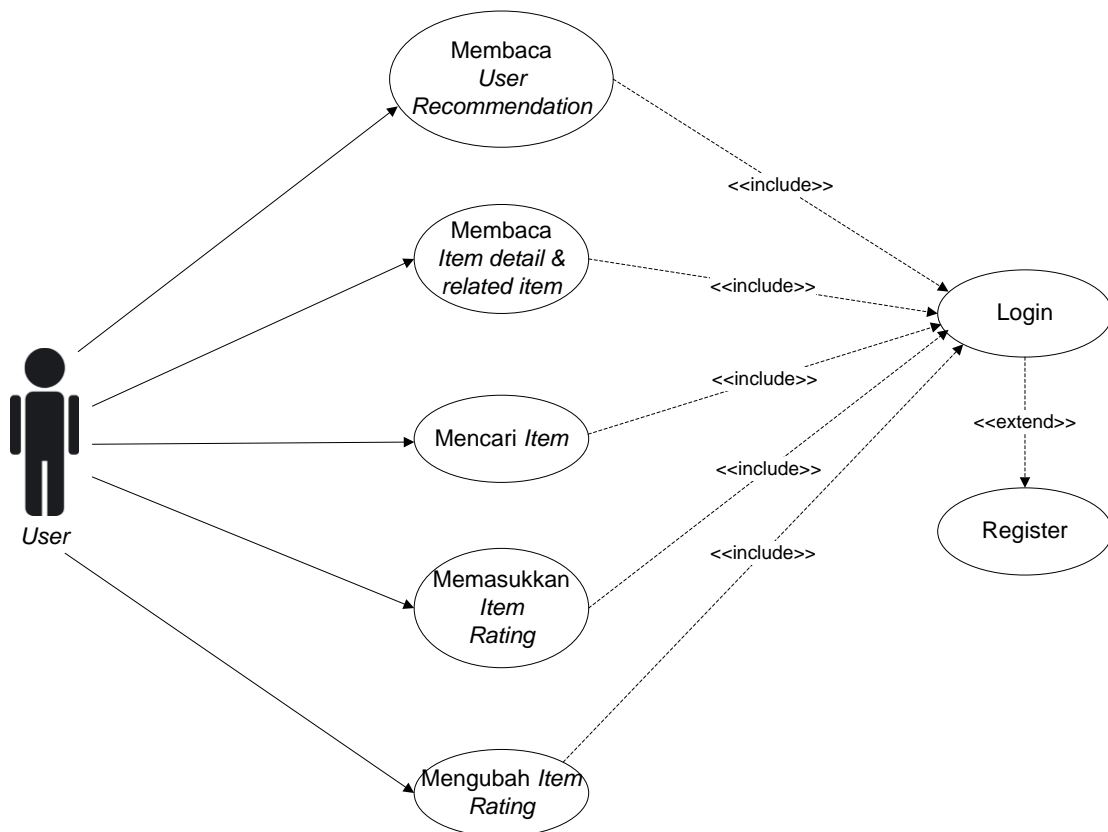
#### 3.2.1.3 DFD Level 2



Deskripsi :

- *User* yang membuka halaman detail *item*, akan diberi daftar *item* yang berdekatan (*related item*) dengan *item* yang sedang dibuka. *Item id* dari *item* yang sedang dibuka akan dikirim menggunakan POST requests Laravel (JSON) ke API Flask melalui */predict\_item route*. Flask menjalankan algoritma lalu mengakses model sistem rekomendasi (.pkl) dan mengirim daftar *related item* kembali ke Laravel (JSON). JSON dari Flask akan diparsing oleh Laravel dan daftar *related item* akan dicetak ke *user*.
- *User* harus sudah memberikan rating minimal pada satu *item* untuk mendapatkan rekomendasi menurut preferensi *user* (*user recommendation*). Daftar *item id* dan rating akan dikirim menggunakan POST requests Laravel (JSON) ke API Flask melalui */predict\_user route*. Flask menjalankan algoritma lalu mengakses model sistem rekomendasi (.pkl) dan mengirim daftar *related item* kembali ke Laravel (JSON). JSON dari Flask akan diparsing oleh Laravel dan daftar *user recommendation* akan dicetak ke *user*.

### 3.2.2 Use Case



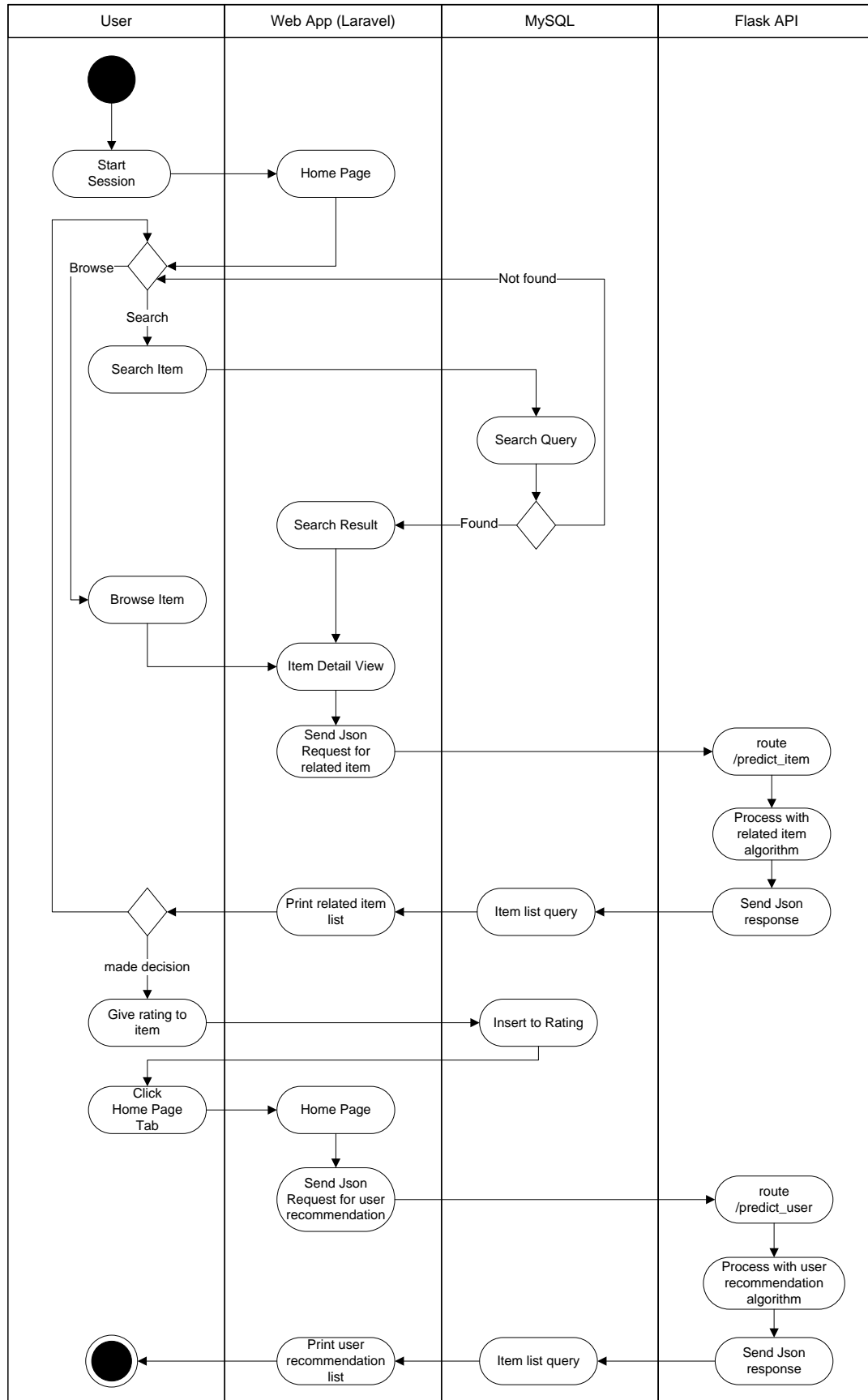
Nama Use Case		Memasukkan <i>Item Rating</i> .
Aktor		<i>User</i> .
Deskripsi		Use Case mendeskripsikan bagaimana <i>User</i> menambah rating baru.
Kondisi Awal		<i>User</i> sudah Log In terlebih dahulu.
Alur Kejadian	1	Use Case dimulai ketika <i>User</i> sudah Log In.
	2	<i>User</i> masuk ke detail <i>item</i> .
	3	Sistem menampilkan <i>detail item</i> dan daftar <i>related item</i> .
	4	<i>User</i> menekan salah satu tombol nomor rating (1-5).
	5	Use Case selesai.
Kondisi Akhir		Rating <i>item</i> sudah ditambahkan.

Nama Use Case		Mengubah <i>Item Rating</i> .
Aktor		<i>User</i> .
Deskripsi		Use Case mendeskripsikan bagaimana <i>User</i> mengganti rating.
Kondisi Awal		<i>User</i> sudah Log In terlebih dahulu dan <i>user</i> sudah pernah memberikan rating pada <i>item</i> yang bersangkutan.
Alur Kejadian	1	Use Case dimulai ketika <i>User</i> sudah Log In dan sudah pernah memberikan rating.
	2	<i>User</i> masuk ke detail <i>item</i> .
	3	Sistem menampilkan rating sebelumnya, <i>detail item</i> , dan daftar <i>related item</i> .
	4	<i>User</i> menekan salah satu tombol nomor rating (1-5) selain rating yang sudah pernah diberikan.
	5	Use Case selesai.
Kondisi Akhir		Rating <i>item</i> sudah diganti.

Nama Use Case		Membaca <i>User Recommendation</i> .
Aktor		<i>User</i> .
Deskripsi		Use Case mendeskripsikan bagaimana <i>User</i> mendapatkan rekomendasi berdasarkan rating yang diberikan.
Kondisi Awal		<i>User</i> sudah Log In terlebih dahulu dan <i>user</i> sudah pernah memberikan rating pada <i>item</i> minimal satu kali.
Alur Kejadian	1	Use Case dimulai ketika <i>User</i> sudah Log In dan sudah pernah memberikan rating.
	2	<i>User</i> masuk ke Tab Home.
	3	Sistem menampilkan daftar rekomendasi.
	4	Use Case selesai.
Kondisi Akhir		Rekomendasi sudah ditampilkan.

Nama Use Case		Mencari <i>Item</i>
Aktor		<i>User</i> .
Deskripsi		Use Case mendeskripsikan bagaimana <i>User</i> mencari <i>item</i> yang diinginkan
Kondisi Awal		<i>User</i> sudah Log In terlebih dahulu
Alur Kejadian	1	Use Case dimulai ketika <i>User</i> sudah Log In
	2	<i>User</i> masuk ke Tab Home.
	3	<i>User</i> mengetik nama <i>item</i> sebagai keyword pada search bar.
	4	Sistem menampilkan <i>item</i> hasil pencarian.
	5	Use Case Selesai.
Kondisi Akhir		Hasil pencarian sudah ditampilkan.

### 3.2.3 Activity Diagram



## BAB 4

### PENERAPAN DAN ANALISIS SISTEM

#### 4.1 Pembuatan Sistem

##### 4.1.1 Filter Amazon Rating JSON Menggunakan MongoDB

Langkah pertama dalam penelitian adalah mendownload Amazon Rating JSON dari repositori Julian McAuley UCLA. Data rating buku yang didownload adalah yang sudah difilter minimal *item* yang sudah diberi 5 rating (5-Core). JSON akan diimport kedalam MongoDB, data rating yang masuk dalam mongodb adalah 8.898.041 rating. Langkah selanjutnya adalah melakukan filter pada JSON berdasarkan minimal jumlah rating yang sudah diberi oleh *user* dan jumlah rating yang sudah diberikan pada *item* agar tidak menjadi *noise* saat pembuatan model.

Target jumlah rating setelah filter adalah dibawah 1.000.000 agar saat dilakukan cosine similarity dapat cukup disimpan dalam *memory*, agar *density dataset* makin tinggi, dan dapat dilakukan penomoran matrix dalam Excel. Filter *item* akan selalu dilakukan terlebih dahulu dari filter *user*. Mongodb dapat mengimport file rating json sebesar 8.81 GB dalam waktu 14 menit 28 detik. Hasil filter menurut k-core:

**Tabel 4.1 Hasil Filter K-Core**

Item (K-core)	User (K-core)	Jumlah Rating
5	5	8.898.041
5	11	6.461.096
5	21	4.640.065
11	21	3.652.890
21	21	2.851.745
36	51	918.053

*Item* dengan 36-Core dan *User* dengan 51-Core dapat memadatkan dataset sampai dibawah 1.000.000, sehingga dataset dengan rating sebanyak 918.053 inilah yang akan digunakan untuk pembuatan model Collaborative Filtering. Kode yang digunakan untuk filter adalah sebagai berikut:

**a. Kode MongoDB untuk filter *item* :**

```
var cursor = db.reviews_50_35.aggregate([
  { "$group": {
    "_id": "$asin",
    "count": {"$sum": 1}
  }},
  { "$match": { "count": { "$lte": 35 }}}
]);
cursor.forEach(function (doc) {
  db.reviews_50_35.remove({"asin" : doc._id});
});
```

**b. Kode MongoDB untuk filter *user* :**

```
var cursor = db.reviews_50_35.aggregate([
  { "$group": {
    "_id": "$reviewerID",
    "count": {"$sum": 1}
  }},
  { "$match": { "count": { "$lte": 50 }}}
]);
cursor.forEach(function (doc) {
  db.reviews_50_35.remove({"reviewerID " : doc._id});
});
```

Selanjutnya dataset akan *diexport* dari MongoDB kedalam format CSV dengan field *reviewerID* (*user id*), *asin* (*item id*), dan *rating*.

#### **4.1.2 Pembuatan Model dengan Metode Collaborative Filtering**

Bahasa yang digunakan untuk pembuatan model adalah Python, karena library machine learning yang dimiliki Python lengkap untuk penelitian sistem rekomendasi. Python yang digunakan adalah versi 3.6.5. Dasar kode yang digunakan untuk split dataset, gradient descent, dan cosine similarity berasal dari *Medium Insight Data Science* (Rosenthal, 2016). Langkah-langkah pembuatan modelnya adalah sebagai berikut:

#### a. Penomoran matrix menggunakan Excel

*User id, Item id, dan rating* yang sudah *diexport* dari MongoDB tidak dapat langsung *diimport* kedalam Python, karena program tidak tahu dimana harus meletakkan tiap rating kedalam matrix. Setiap *user id* dan *item id* harus diberi *matrix id*, dalam penelitian ini saya menggunakan Excel untuk memberikan penomoran, tetapi penomoran juga dapat dilakukan dengan membuat program untuk melakukan penomoran tersebut. Penomoran matrix menggunakan Excel:

reviewerID (user id)	user id matrix	asin (item id)	item id matrix	rating
A10MPQWV3IIE6EC	53	2007770	1	5
A1129LM24YWSZV	80	2007770	1	5
A11ARYQ7SS070U	93	2007770	1	5
A11L3YX5WIDKJ	117	2007770	1	5
A11NL2A0RDEGF	125	2007770	1	5
A11PKVBGCKRVEK	132	2007770	1	5
A1237ROTM7659	154	2007770	1	5
A12HI5D3C2EGWJ	177	2007770	1	5
A12OB10E5L01V0	193	2007770	1	5

#### b. Import dataset, pembuatan matrix, dan split dataset

##### 1) Import kedalam Python:

```
import numpy as np
import pandas as pd
np.random.seed(0)
names = ['user_id', 'item_id', 'rating']
df = pd.read_csv('reviews_50u_35i_ready.csv', sep=',',
names=names)
df.head()
```

##### 2) Pembuatan Matrix:

```
n_users = df.user_id.unique().shape[0]
n_items = df.item_id.unique().shape[0]
ratings = np.zeros((n_users, n_items))
for row in df.itertuples():
    ratings[row[1]-1, row[2]-1] = row[3]
```

##### 3) Split Training Set dan Test Set:

```
def train_test_split(ratings):
    test = np.zeros(ratings.shape)
    train = ratings.copy()
    for user in xrange(ratings.shape[0]):
        test_ratings = np.random.choice(ratings[user,
:]).nonzero()[0], size=10, replace=False)
        train[user, test_ratings] = 0.
```



```

        test[user, test_ratings] = ratings[user,
test_ratings]

        assert(np.all((train * test) == 0))
        return train, test
train, test = train_test_split(ratings)

```

#### 4) Perhitungan Dataset Sparsity (rating yang tidak 0 dalam matrix):

```

print str(n_users) + ' users'
print str(n_items) + ' items'
sparsity = float(len(ratings.nonzero()[0]))
sparsity /= (ratings.shape[0] * ratings.shape[1])
sparsity *= 100
print 'Sparsity: {:.4.2f}%'.format(sparsity)

```

*hasil print:*

```

8625 users
25877 items
Sparsity: 0.41%

```

### c. Pencarian nilai parameter untuk menghasilkan RMSE paling kecil menggunakan algoritma Stochastic Gradient Descent (SGD)

#### 1) Tes RMSE tanpa *regularization*:

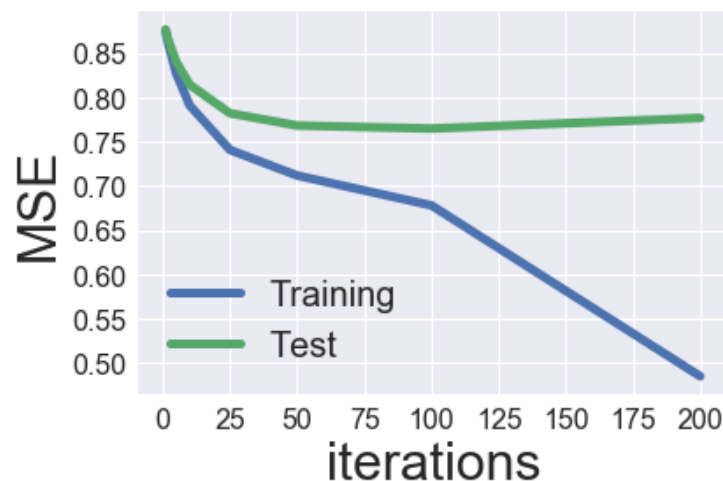
Tes menggunakan latent factor 40 dan learning rate 0.001.

```

MF_SGD = ExplicitMF(train, 40, learning='sgd',
verbose=True)
iter_array = [1, 2, 5, 10, 25, 50, 100, 200]
MF_SGD.calculate_learning_curve(iter_array, test,
learning_rate=0.001)

```

Learning Curve SGD:



Grafik menampilkan mulai iterasi 100 mulai terjadi *overfit* pada test set, Sehingga menunjukkan bahwa dibutuhkan *regularization*.

2) Tes untuk mencari *learning rate* dengan performa paling baik pada test set:

Semakin kecil RMSE, maka menunjukkan bahwa performa model makin baik. *Learning rate* yang di tes adalah: 0.01, 0.001, 0.0001, dan 0.00001. Test set RMSE dengan nilai yang lebih besar dari test set RMSE loop learning rate sebelumnya tidak akan ditampilkan:

```
iter_array = [1, 2, 5, 10, 25, 50, 100, 200]
learning_rates = [1e-5, 1e-4, 1e-3, 1e-2]

best_params = {}
best_params['learning_rate'] = None
best_params['n_iter'] = 0
best_params['train_mse'] = np.inf
best_params['test_mse'] = np.inf
best_params['model'] = None

for rate in learning_rates:
    print 'Rate: {}'.format(rate)
    MF_SGD = ExplicitMF(train, n_factors=40,
learning='sgd')
    MF_SGD.calculate_learning_curve(iter_array, test,
learning_rate=rate)
    min_idx = np.argmin(MF_SGD.test_mse)
    if MF_SGD.test_mse[min_idx] <
best_params['test_mse']:
        best_params['n_iter'] = iter_array[min_idx]
        best_params['learning_rate'] = rate
        best_params['train_mse'] =
MF_SGD.train_mse[min_idx]
        best_params['test_mse'] =
MF_SGD.test_mse[min_idx]
        best_params['model'] = MF_SGD
        print 'New optimal hyperparameters'
        print pd.Series(best_params)
```

Hasil RMSE:

**Tabel 4.2 Hasil RMSE *Learning Rate***

Learning Rate	Iteration	Test set MSE	Test set RMSE
0.00001	200	0.865158	0.930138699
0.0001	200	0.788713	0.888095153
0.001	100	0.764526	0.874371774

Hasil tes menunjukkan bahwa Learning Rate 0.001 menghasilkan test set RMSE yang paling kecil.

**3) Tes untuk mencari nilai *latent factor* dan *regularization* dengan performa paling baik pada test set:**

Semakin kecil RMSE, maka menunjukkan bahwa performa model makin baik. *Latent Factor* yang di tes adalah: 5, 10, 20, 40, 80. *Regularization* yang di tes adalah: 0.001, 0.01, 0.1, 1 . Learning Rate yang digunakan adalah 0.001 menurut hasil tes sebelumnya. Test set RMSE dengan nilai yang lebih besar dari test set RMSE loop learning rate sebelumnya tidak akan ditampilkan:

```
iter_array = [1, 2, 5, 10, 25, 50, 100, 200]
latent_factors = [5, 10, 20, 40, 80]
regularizations = [0.001, 0.01, 0.1, 1.]
regularizations.sort()

best_params = {}
best_params['n_factors'] = latent_factors[0]
best_params['reg'] = regularizations[0]
best_params['n_iter'] = 0
best_params['train_mse'] = np.inf
best_params['test_mse'] = np.inf
best_params['model'] = None

for fact in latent_factors:
    print 'Factors: {}'.format(fact)
    for reg in regularizations:
        print 'Regularizations: {}'.format(reg)
        MF_SGD = ExplicitMF(train, n_factors=fact,
learning='sgd', \
                                user_fact_reg=reg,
item_fact_reg=reg, \
                                user_bias_reg=reg,
item_bias_reg=reg)
        MF_SGD.calculate_learning_curve(iter_array,
test, \

learning_rate=0.001)
        min_idx = np.argmin(MF_SGD.test_mse)
        if MF_SGD.test_mse[min_idx] <
best_params['test_mse']:
            best_params['n_factors'] = fact
            best_params['reg'] = reg
            best_params['n_iter'] = iter_array[min_idx]
            best_params['train_mse'] =
MF_SGD.train_mse[min_idx]
            best_params['test_mse'] =
MF_SGD.test_mse[min_idx]
            best_params['model'] = MF_SGD
            print 'New optimal hyperparameters'
            print pd.Series(best_params)
```

**Tabel 4.3 Hasil RMSE *Latent Factors* dan *Regularization***

Latent Factors	Iteration	Regularization	Test set MSE	Test set RMSE
5	50	0.001	0.775969	0.880891026
5	50	0.01	0.774411	0.88000625
5	100	0.1	0.763977	0.874057778
10	200	0.1	0.762139	0.873005727
20	200	0.1	0.761719	0.872765146

Hasil tes menunjukkan bahwa *latent factors* 20, *iteration* 200, dan *regularization* 0.1 menghasilkan test set RMSE yang paling kecil.

**4) Melatih model menggunakan parameter terbaik:**

*Regularization* terbaik: 0.1

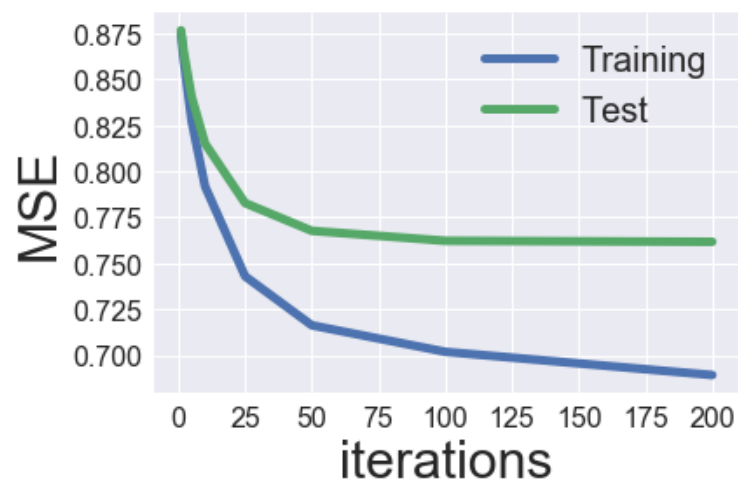
*Latent Factor* terbaik : 20

*Iteration* terbaik : 200

*Learning Rate* terbaik: 0.001

```
best_sgd_model = ExplicitMF(ratings, n_factors=20,
learning='sgd', \
                                item_fact_reg=0.1,
user_fact_reg=0.1, \
                                user_bias_reg=0.1,
item_bias_reg=0.1)
best_sgd_model.train(200, learning_rate=0.001)
```

Learning Rate SGD:



### 5) Melatih ulang model dengan normalisasi:

Normalisasi bertujuan agar skala rating dari *item* yang satu dengan *item* lainnya menjadi sama. Jika tidak dilakukan normalisasi, maka *item* dengan rating yang tinggi akan selalu direkomendasikan oleh sistem, hal ini menyebabkan rating yang diberikan oleh *user* baru tidak akan berdampak banyak dengan rekomendasi yang dikeluarkan oleh sistem.

#### Normalisasi menggunakan rata-rata rating setiap *item* :

```
users = ratings.shape[0]
books = ratings.shape[1]
books_mean = np.zeros((1, books))
books_norm = np.zeros((users, books))

R = np.zeros((users, books))
for us in range(users):
    for mo in range(books):
        if ratings[us, mo] >= 1.0:
            R[us, mo] = 1
        else:
            R[us, mo] = 0

for i in range(books):
    idx = np.where(R[:, i] == 1)[0]
    books_mean[0][i] = ratings[idx, i].mean()
    books_norm[idx, i] = ratings[idx, i] - books_mean[0][i]
```

#### Melatih ulang model dengan rating yang sudah dinormalisasi:

```
best_sgd_model = ExplicitMF(books_norm, n_factors=20,
                             learning='sgd', \
                             item_fact_reg=0.1,
                             user_fact_reg=0.1, \
                             user_bias_reg=0.1,
                             item_bias_reg=0.1)
best_sgd_model.train(200, learning_rate=0.001)
```

### 4.1.3 Menyiapkan Model untuk produksi dan Flask API

#### a. Export Model Menggunakan Pickle

Model yang sudah jadi dalam variabel `best_sgd_model` saat ini hanya berguna bagi developer. Untuk menggunakan model tersebut dalam produksi, model harus *diexport* dalam bentuk file atau dimasukkan kedalam database. Dalam penelitian ini, model akan *diexport* kedalam file dengan format `.pkl` menggunakan *library* Pickle dalam Python:

```
import pickle
with open("python_best_sgd_model_item.pkl", "wb") as
file_handler:
    pickle.dump(best_sgd_model, file_handler,
protocol=pickle.HIGHEST_PROTOCOL)
```

## b. Pembuatan Flask API

Flask API dibuat agar model sistem rekomendasi dapat diakses oleh aplikasi web. Ada 2 *route* yang akan dibuat dalam Flask API : *related item route* dan *user recommendation route*.

### 1) *Related Item Route*

*Related Item Route* dengan alamat “/predict\_item” dibuat untuk menangani *requests* dari aplikasi web pada saat *user* membuka halaman detail produk. *Related Item Route* akan menerima input berupa *matrix id* 1 *item* dari aplikasi web. Dalam halaman detail produk, pada bagian bawah akan ditampilkan daftar *item* yang berdekatan dengan *item* yang sedang dilihat oleh *user* menggunakan algoritma *cosine similarity*.

#### Algoritma *Cosine Similarity*:

```
def cosine_similarity(model):
    sim = model.item_vecs.dot(model.item_vecs.T)
    norms = np.array([np.sqrt(np.diagonal(sim))])
    return sim / norms / norms.T
```

#### *Related Item Route Flask API*:

```
@app.route("/predict_item", methods=['POST'])
def predict_item():
    if request.method == 'POST':
        try:
            data = request.get_json()
            book_mat = int(data["ItemMat"])

            idx_to_book = {}

            with
io.open('reviews_50u_35i_info_items.csv', mode='r',
encoding='utf-8-sig') as f:
                for line in f.readlines():
                    info = line.split(',')

                    idx_to_book[int(info[0])] =
'{"mat_id":"' + info[0] + '","item_id":"' + info[1] + '"}'

            book_indices =
np.argsort(sgd_model.sgd_sim[book_mat, :])[:-1]
            book_idx_list = []
            k_ctr = 0
            i = 1
            while k_ctr < 80:
```

```

        book = idx_to_book[book_indices[i]]
        book_idx_list.append(book)
        k_ctr += 1
        i += 1

    string_list = str(book_idx_list)
    string_list = string_list.replace("'",
    "").replace("\\n", "")

    except ValueError:
        return jsonify("Error Predict Item.")

    return string_list

```

## 2) *User Recommendation Route*

*User Recommendation Route* dengan alamat “/predict\_user” dibuat untuk menangani *requests* dari aplikasi web pada saat *user* membuka halaman *home*. *User Recommendation Route* akan menerima input berupa *matrix id* dan *rating* beberapa *item* dari aplikasi web. Rekomendasi yang ditampilkan dalam halaman *home* adalah berdasarkan dari rating yang sudah pernah diberikan *user* pada *item*, agar halaman *home* dapat menampilkan rekomendasi, *user* harus sudah memberikan rating minimal pada 1 *item*. Rekomendasi *User* didapat dengan:  $\text{my\_ratings.T (vector rating 1-5 new user transpose)} * \text{item\_vecs (all item latent factor)} * \text{item\_vecs.T (all item latent factor transpose)} = \text{new\_user\_rec (new user rating predictions)}$ .

### *User Recommendation Route Flask API:*

```

@app.route("/predict_user", methods=['POST'])
def predict_user():
    if request.method == 'POST':
        try:
            data = request.get_json()

            my_ratings =
            np.zeros((sgd_model.item_vecs.shape[0], 1))
            for i in data:
                my_ratings[i['ItemMat']] =
                i['ItemRating']

            new_user_rec =
            my_ratings.T.dot(sgd_model.item_vecs) \
                .dot(sgd_model.item_vecs.T)

            idx_to_book = {}

```

```

        with
        io.open('reviews_50u_35i_info_items.csv', mode='r',
        encoding='utf-8-sig') as f:
            for line in f.readlines():
                info = line.split(',')

                idx_to_book[int(info[0])] =
                '{"mat_id":"' + info[0] + ', "item_id":"' + info[1] + '"}'

        book_indices = np.argsort(new_user_rec[0, :],
        axis=0)[::-1]

        book_idx_list = []
        k_ctr = 0
        i = 1
        while k_ctr < 80:
            book = idx_to_book[book_indices[i]]
            book_idx_list.append(book)
            k_ctr += 1
            i += 1

        string_list = str(book_idx_list)

        string_list = string_list.replace("'",
        "").replace("\n", "")

        except ValueError as e:
            return jsonify("Error Predict User")

        return string_list

```

Sekarang model sistem rekomendasi sudah siap untuk menerima *input request* dari aplikasi web. Langkah selanjutnya adalah membuat aplikasi web sebagai front end menggunakan framework Laravel dengan bahasa PHP.

#### 4.1.4 Aplikasi Web Laravel

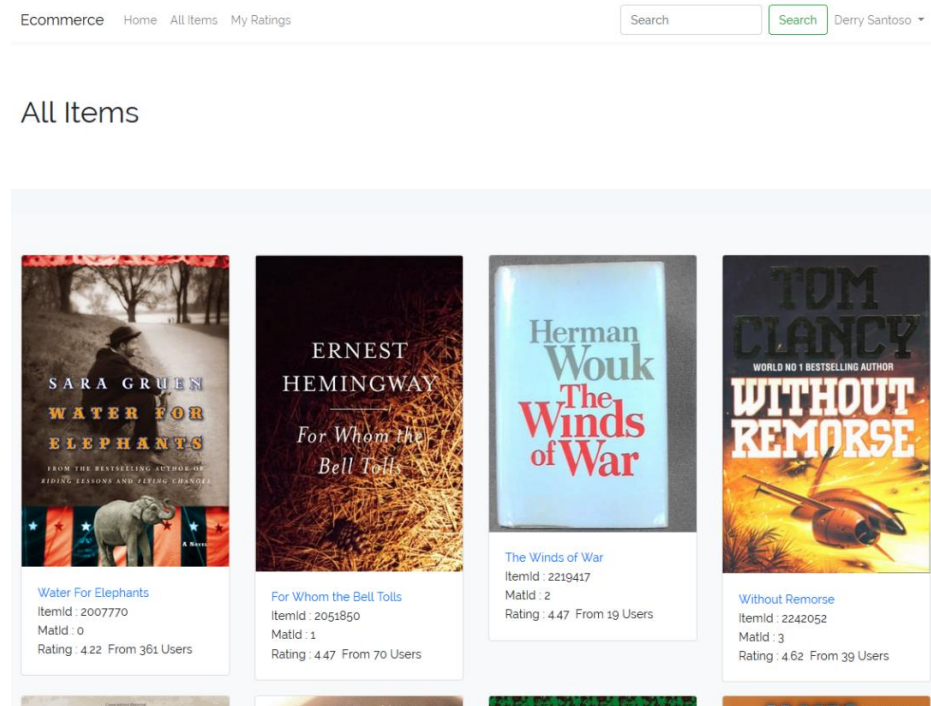
Aplikasi Web Laravel berfungsi sebagai *front end* yang dapat diakses oleh *user*. Fitur website yang dibuat berfokus pada fitur yang mendukung penelitian sistem rekomendasi menurut dataset yang tersedia secara publik (data rating). Aplikasi web akan menggunakan MySQL sebagai *back end* pertama untuk menyimpan data rating, item, dan *user*. Aplikasi web juga menggunakan Flask API sebagai *back end* kedua untuk memproses sistem rekomendasi.



**a. Halaman *All Items***

Halaman ini akan menampilkan semua buku yang ada didalam E-commerce.

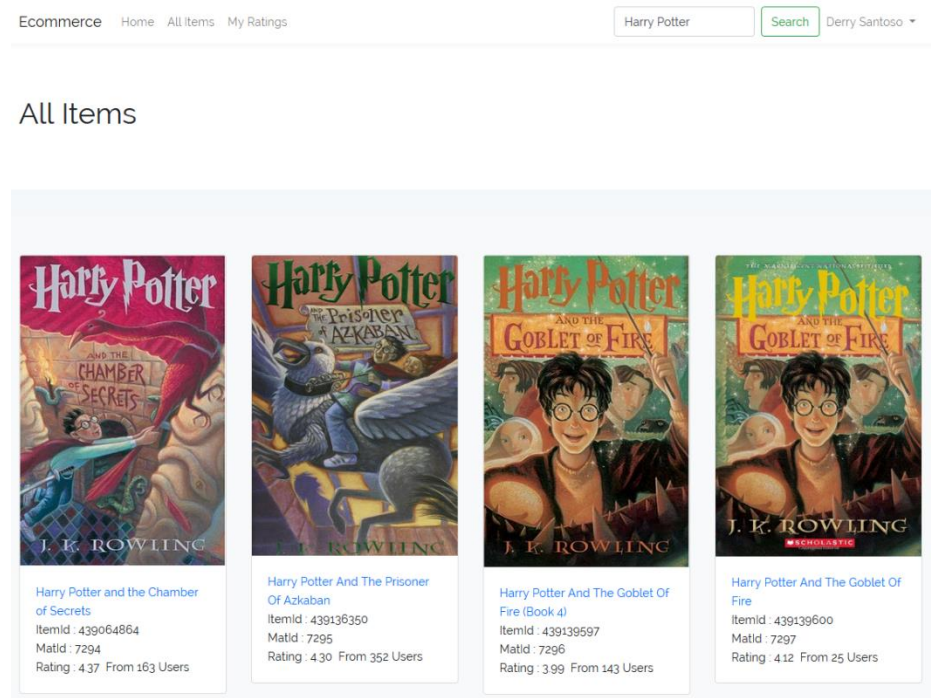
URL : <http://skripsi.test/items>



**b. Halaman *All Items* + Search Result**

Tampilan halaman *all items* setelah dilakukan pencarian “Harry Potter”

URL : <http://skripsi.test/items?result=Harry%20Potter>



### c. Halaman *Detail Item*

Halaman akan menampilkan detail 1 *item* dan pada bagian bawah akan ditampilkan daftar *related item*. Bagian *related item* akan mengirim *request* ke Flask API berbentuk JSON menggunakan Guzzle. Dalam Flask API, JSON akan masuk kedalam */predict\_item route* dan hasil perhitungan dalam Flask API akan dikirim kembali ke aplikasi web dalam bentuk JSON. JSON yang sudah sampai dalam aplikasi web akan diparsing dan mengakses data *item* dalam database MySQL, lalu daftar *item* dari MySQL akan di *print* dalam browser pada bagian *related item*.

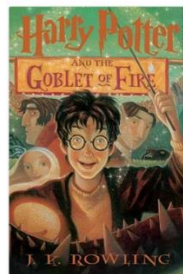
URL : <http://skripsi.test/items/detail/7297>

Harry Potter And The Goblet Of Fire (Book 4)

MatId:7296

ItemId:439139597

3.99 From 143 User Ratings



People who bought this item also bought

<p>The Barkeep ItemId:161218195 3 MatId:19219 Rating : 4.00 From 10 Users</p>	<p>The Good Lord Bird: A Novel ItemId:159448634 4 MatId:18502 Rating : 4.20 From 45 Users</p>	<p>Waiting for Columbus ItemId:385529139 MatId:6296 Rating : 4.39 From 33 Users</p>	<p>Head in the Clouds ItemId:764207563 MatId:10417 Rating : 4.07 From 45 Users</p>	<p>Cut ItemId:439324599 MatId:7303 Rating : 2.78 From 18 Users</p>	<p>On the Rocks (Last Call Series) (Volume 1) ItemId:194088312 1 MatId:20173 Rating : 3.80 From 83 Users</p>
<p>Unexpected Treasure - The MatId:19219</p>	<p>Hannibal Rising ItemId:375435417 MatId:5745</p>	<p>Separation of Power (Mitch Rapp Novels)</p>	<p>The Price of Glory: Verdun 1916</p>	<p>Resurrection ItemId:451458346</p>	<p>Catch Me: A MatId:19219</p>

### Kode *Related Item* :

```
$client = new Client(['base_uri' =>
'http://10.0.2.2:5000']);
$res = $client->request('POST', '/predict_item', ['json' =>
['ItemMat' => $item->item_mat_id]]);

$item_reccs = json_decode($res->getBody());

$reccs_array = [];
foreach($item_reccs as $recc){
    $reccs_array[] = $recc->item_id;
}
$reccs_array_imploded = implode(',',array_fill(0,
count($reccs_array), '?'));
$reccs_complete = Item::whereIn('item_id', $reccs_array)
->where('title','!=','')
-
>orderByRaw("field(item_id,{ $reccs_array_imploded})", $reccs_
array)
->paginate(12);
```

#### d. Halaman *My Ratings*

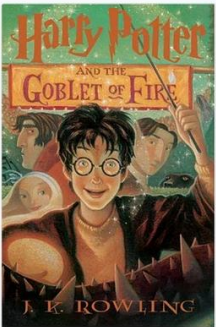
Halaman ini menampilkan *rating* pada *item* apa saja yang sudah pernah diberikan oleh *user*.

URL : [http://skripsi.test/items/my\\_ratings](http://skripsi.test/items/my_ratings)

[Ecommerce](#) [Home](#) [All Items](#) [My Ratings](#)


Derry Santoso ▾

### My Ratings



Harry Potter And The Goblet Of Fire (Book 4)  
Itemid : 439139597  
Matid : 7296  
Rating : 3.99 From 143 Users

1 2 3 4 5



Head in the Clouds  
Itemid : 764207563  
Matid : 10417  
Rating : 4.07 From 46 Users

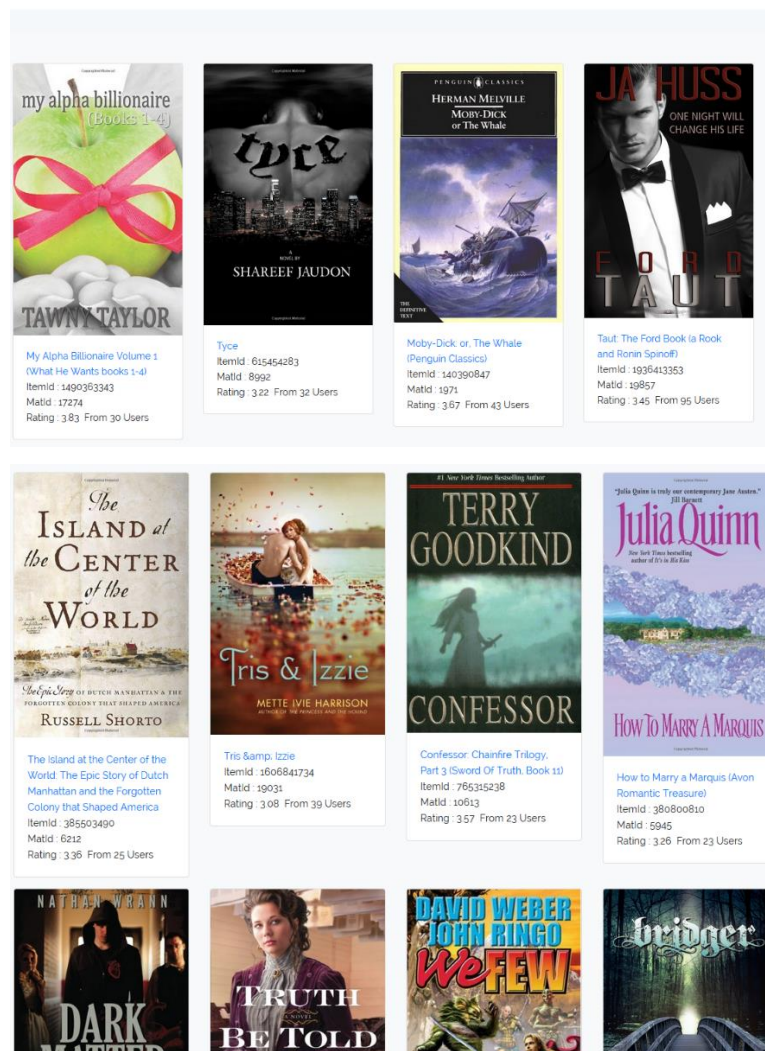
1 2 3 4 5

### e. Halaman *Home*

Halaman home akan menampilkan *user recommendation* berdasarkan rating yang sudah pernah diberikan *user* pada *item*. *Home* akan mengirim *request* ke Flask API berbentuk JSON menggunakan Guzzle. Dalam Flask API, JSON akan masuk kedalam */predict\_user route* dan hasil perhitungan dalam Flask API akan dikirim kembali ke aplikasi web dalam bentuk JSON. JSON yang sudah sampai dalam aplikasi web akan diparsing dan mengakses data *item* dalam database MySQL, lalu daftar *item* dari MySQL akan di *print* dalam browser pada halaman *home*. Jika *user* belum pernah memberikan rating pada *item*, maka pada halaman *home* akan ada pesan “*Rate Item to get Recommendation*”.

URL : <http://skripsi.test/home>

Home



### Kode User Recommendation :

```
if (\App\Rating::where('user_id', Auth::user()->user_id)-
>get()->count())
{
    $items_array = \App\Rating::where('user_id',
Auth::user()->user_id)->get();
    $client = new Client(['base_uri' =>
'http://10.0.2.2:5000']);

    $data = array(
        "json" => array()
    );
    foreach($items_array as $item)
    {
        $data['json'][] = array(
            'ItemMat' => (int)$item->item-
>item_mat_id,
            'ItemRating' => (int)$item->rating
        );
    }

    $res = $client->request('POST', '/predict_user',
    $data);

    $item_reccs = json_decode($res->getBody());

    $reccs_array = [];
    foreach ($item_reccs as $recc)
    {
        $reccs_array[] = $recc->item_id;
    }

    $reccs_array_imploded =
implode(',', array_fill(0, count($reccs_array), '?'));
    $items = Item::whereIn('item_id', $reccs_array)
        ->where('title', '!=', '')
        -
>orderByRaw("field(item_id,{$reccs_array_imploded})", $reccs_
array)
        ->paginate(12);
```

## 4.2 Analisis

Filter dataset menggunakan k-cores untuk meningkatkan *density* memiliki hasil *density* 0.41% yang berarti 99.58% data dalam matrix tidak memiliki rating (0). *Density* 0.41% kurang ideal jika dibandingkan dengan penelitian lain soal Collaborative Filtering dari (Joonseok Lee, 2012) dengan *density* 1-5% dan (Rosenthal, 2016) dengan *density* 6.30%. Dataset dengan *density* 0.41% tetap digunakan karena matrix sudah cukup dalam memory dan agar rating tidak terfilter terlalu banyak.



RMSE test set terkecil yang dihasilkan oleh Stochastic Gradient Descent (SGD) tanpa *regularization* adalah 0.874371774 dengan learning rate 0.001. RMSE bertambah tinggi setelah iterasi 100 karena dataset mengalami *overfit*. Salah satu solusi untuk mengangani *overfit* adalah dengan menambah *regularization* pada algoritma.

RMSE test set terkecil oleh SGD dengan *regularization* adalah 0.872765146. RMSE tidak memiliki pengukuran pasti tentang berapa RMSE yang baik/buruk karena perbedaan algoritma berpengaruh banyak dengan hasil RMSE. Salah satu cara adalah membandingkan RMSE dengan penelitian lain yang juga menggunakan metode Collaborative Filtering, RMSE 0.872765146 cukup baik jika dibandingkan dengan penelitian lain: hasil penelitian data movielens oleh (Rosenthal, 2016) menunjukkan bahwa hasil RMSE terkecil dengan algoritma ALS (Alternating Least Square) adalah 2.245 dan RMSE algoritma SGD adalah 0.8718, penelitian Collaborative Filtering yang lain adalah Netflix Prize (Netflix, 2009) dengan RMSE awal 0.9525 dan RMSE pemenang 0.8567.

Rekomendasi visual yang dikeluarkan pada aplikasi web cukup sulit untuk dinilai secara intuisi apakah berhubungan atau tidak, contoh:

- Halaman detail “Harry Potter Goblet of Fire” tidak menampilkan buku Harry Potter yang lainnya pada bagian *Related Item*
- Buku tentang Komputer seperti “My Macbook : John Ray” tidak menampilkan buku komputer lainnya pada bagian *Related Item*.

Hasil *Related Item* tersebut disebabkan karena customer asli dapat membeli barang dan memberi rating pada buku dengan kategori, author, genre, dan seri yang berbeda dari buku yang dibeli sebelumnya. Beberapa contoh sejarah rating *user* dalam dataset :

- *User* yang memberi rating buku “Harry Potter Goblet of Fire” belum tentu membeli buku Harry Potter yang lainnya. Walaupun *user* membeli seri buku Harry Potter yang lain, buku selain Harry Potter yang sudah diberi rating oleh *user* tersebut juga akan mempengaruhi sistem rekomendasi.
- *User* yang membeli buku komputer “My Macbook : John Ray”, juga membeli buku dalam kategori drama.

Agar hasil rekomendasi yang dikeluarkan oleh sistem lebih masuk akal dan tidak sepenuhnya bergantung pada sejarah rating *user*, maka diperlukan tambahan metode sistem rekomendasi yang disebut Content Filtering. Content Filtering dapat menjadi pelengkap Collaborative Filtering, gabungan keduanya disebut Hybrid Filtering. Content Filtering dapat melakukan beberapa *filtering item* contohnya: menurut keyword judul , pengarang, seri (Harry Potter, The Hunger Games), dan kategori (drama, komputer, fiksi).

## **BAB 5**

### **PENUTUP**

#### **5.1 Kesimpulan**

Hasil penelitian sistem rekomendasi dengan metode Collaborative Filtering dapat mengeluarkan hasil akhir RMSE yang cukup baik sebesar 0.872765146. Tetapi walaupun hasil RMSE cukup baik, rekomendasi visual yang keluar dalam aplikasi web cukup sulit untuk dinilai secara intuisi apakah hasil *related item* pada halaman detail *item* dan *user recommendation* pada halaman *home* benar-benar cocok untuk *user*.

#### **5.2 Saran**

- a. Menggunakan Metode Content Filtering sebagai pelengkap Collaborative Filtering agar hasil rekomendasi tidak sepenuhnya bergantung dengan dataset, daftar rekomendasi visual dalam web lebih masuk akal, dan rekomendasi dapat dinilai secara intuisi.
- b. Menggunakan Metode Content Filtering untuk memberikan rekomendasi *related item* pada *item* yang belum pernah diberi rating oleh *user*.
- c. Penelitian lebih lanjut tentang metode untuk memproses item dan rating baru yang tidak ada dalam dataset. Item dan rating baru dapat digunakan untuk melatih ulang model agar performa model meningkat (RMSE bertambah kecil).
- d. Penelitian lebih lanjut untuk memfilter *user* yang sejarah pembeliannya terlalu bervariasi agar tidak menjadi *noise* (contoh: *user* yang membeli buku komputer juga membeli buku drama dan fiksi).



## Daftar Pustaka

- Barwicki, A. (2017, June 16). *Recommendation System: Simple Theory of Collaborative Filtering*. Retrieved from Adrian's Space:  
<http://adrianbarwicki.com/2017/06/16/collaborative-filtering-recommendation-systems-simple-theory/>
- Bluma, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 245-271.
- Bossenbroek, H., & Gringhuis, H. (2014). Recommendation in E-Commerce. *Luminis Recommendation Services*.
- Isinkayea, F., Folajimib, Y., & Ojokohc, B. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, 261–273.
- Joonseok Lee, M. S. (2012). A Comparative Study of Collaborative Filtering.  
*arXiv:1205.3193v1 [cs.IR]*.
- Kohavi, R., & Provost, F. (1998). Glossary of Terms. *Machine Learning*, 271-274.
- Netflix. (2009). *Netflix Prize Leaderboard*. Retrieved from Netflix:  
<https://www.netflixprize.com/leaderboard.html>
- Ng, A. (2017). *Machine Learning by Stanford University*. Retrieved from Coursera:  
<https://www.coursera.org/learn/machine-learning/home/welcome>
- O'Brien, J., & Marakas, G. (2007). *Management Information System 10th ed*. New York: McGraw Hill.
- Perone, C. S. (2013, 9 12). *Machine Learning :: Cosine Similarity for Vector Space Models (Part III)*. Retrieved from Terra Incognita:  
<http://blog.christianperone.com/2013/09/machine-learning-cosine-similarity-for-vector-space-models-part-iii/>
- R. He, J. M. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. *WWW*.
- Rosenthal, E. (2016, March 16). *Explicit Matrix Factorization: ALS, SGD, and All That Jazz*. Retrieved from Medium Insight Data Science:  
<https://blog.insightdatascience.com/explicit-matrix-factorization-als-sgd-and-all-that-jazz-b00e4d9b21ea>
- Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2002). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. *Proceedings of the fifth international conference on computer and information technology* (pp. 1-6). Citeseer.



## RINGKASAN PROPOSAL

Dicetak tanggal: 27-11-2017 23:43:41

### Identitas

NIM : 72140011 IPK : 3.92  
Nama : DERRY SANTOSO IPro : 4.00  
Konsentrasi : SISTEM INFORMASI BISNIS HP : 089688366212  
Judul : SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS COLLABORATIVE FILTERING  
Dosen Pengarah (jika ada) : YETLI OSLAN, S.KOM., M.T.  
Apakah Skripsi ini kelanjutan dari KP? : Tidak  
Pembimbing KP :

### Gambaran Skripsi yang akan dibuat

Aplikasi E-Commerce yang akan dibuat adalah C2C (Customer to Customer). Aplikasi diharapkan dapat memberikan rekomendasi yang bersifat personal untuk setiap pembeli. Rekomendasi dengan Metode Collaborative Filtering akan memberikan rekomendasi berdasarkan : sejarah produk yang sudah diberi rating oleh pembeli terdahulu, sejarah user yang sudah memberi rating pada produk, fitur referensi tiap user terhadap produk, click stream, kategori dan sub kategori tiap produk.

### Spesifikasi Input-Output

#### Input :

- Data User Amazon tahun 1995 – 2014 yang sudah pernah memberi rating pada produk dan kode produk yang diberi rating
- Data Produk Amazon dalam semua kategori pada dataset tahun 1995 – 2014 yang sudah pernah diberi rating oleh user dan kode user yang memberi ratingnya

**Output :** Rekomendasi daftar produk untuk setiap pembeli

### Pengguna

Customer yang ingin membeli/menjual dalam website

### Metode

Collaborative Filtering



Universitas Kristen Duta Wacana  
Fakultas Teknologi Informasi Program Studi Sistem Informasi  
Jl. Dr. Wahidin Sudirahusada 5-25 Yogyakarta 55224  
Telp.: (0274)563929 Faks.: (0274)513235



#### Data

- Data User Amazon tahun 1995 – 2014 yang sudah pernah memberi rating pada produk dan kode produk yang diberi rating
- Data Produk Amazon dalam semua kategori pada dataset tahun 1995 – 2014 yang sudah pernah diberi rating oleh user dan kode user yang memberi ratingnya
- Semua data berasal dari repositori Julian McAuley institusi UCSD (University of California San Diego) (<http://jmcauley.ucsd.edu/data/amazon/links.html>)
- Data tahun 1995 – 2013 akan digunakan sebagai training set, sedangkan tahun 2014 sebagai test set.

#### Evaluasi

Siapkan proposal dan beri gambaran kerja sistem

#### Status Ringkasan Proposal: Diterima

Ringkasan Proposal ini disepakati dalam *desk evaluation* tanggal: **Jumat, 17 November 2017**

(Tim Desk Evaluation)



Universitas Kristen Duta Wacana  
Fakultas Teknologi Informasi Program Studi Sistem Informasi  
Jl. Dr. Wahidin Sudirahusada 5-25 Yogyakarta 55224  
Telp.: (0274)563929 Faks.: (0274)513235



## BERITA ACARA KOLOKIUUM

Dicetak tanggal: 14-12-2017 10:10:49

Dengan ini dinyatakan bahwa PROPOSAL SKRIPSI berjudul:

**SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS COLLABORATIVE FILTERING**

Yang diajukan oleh mahasiswa:

NIM/NAMA : 72140011 / DERRY SANTOSO

Dosen Pengarah : YETLI OSLAN, S.Kom., M.T.

Dinyatakan: ☒ LULUS ☐ LULUS BERSYARAT ☐ BELUM DAPAT DITERIMA

Adapun beberapa catatan terkait dengan proposal skripsi yang disepakati dalam kolokium ini:

Rekomendasi atas dasar: Click Stream, sejarah pembelian, favorit atas perilaku ybs kemudian diolah.

Tools: Python, Laravel

Data: tahun 1995-2014 (bisa disesuaikan jika terlalu berat)

Proposal final harus dikumpulkan selambat-lambatnya hari: **Sabtu, 5 Januari 2008**

Disepakati dalam KOLOKIUUM tanggal: Rabu, 13 Desember 2017

YETLI OSLAN, S.KOM., M.T.  
Dosen Pengarah

DERRY SANTOSO  
Mahasiswa ybs.

(  
TIM KOLOKIUUM

### Catatan:

Berita acara ini dan hasil Evaluasi Ringkasan Proposal harus dilampirkan di proposal final.



Universitas Kristen Duta Wacana  
Fakultas Teknologi Informasi Program Studi Sistem Informasi  
Jl. Dr. Wahidin Sudirahusada 5-25 Yogyakarta 55224  
Telp.: (0274)563929 Faks.: (0274)513235



F23.09.001

## BERITA ACARA UJIAN SKRIPSI

(Diisi oleh Ketua Tim Penguji)

Pada hari ini : **Selasa, 26 Juni 2018**

Telah dilakukan Ujian Skripsi untuk mahasiswa tersebut dibawah ini :

Nama Mahasiswa : **DERRY SANTOSO**  
No. Induk Mahasiswa : **72140011**  
Judul Skripsi : **SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS COLLABORATIVE FILTERING**

Dosen Pembimbing I : **YETLI OSLAN, S.Kom., M.T.**  
Dosen Pembimbing II : **ERICK KURNIAWAN, S.Kom., M.Kom.**




### NILAI

(Lingkari yang dipilih)

<b>A</b>	A-	B+	B	B-	C+	C	D
----------	----	----	---	----	----	---	---

Keterangan : **LULUS / TIDAK LULUS**




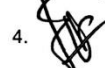
(Coret salah satu)

No.	CATATAN PERBAIKAN
1.	Konsisten bahasa pd USE CASE
2.	Flowchart simbol sambung :  
	

Perubahan di atas harus sudah diselesaikan paling lambat tanggal Kamis, 26 Juli 2018

Dewan Penguji Skripsi :

1. YETLI OSLAN, S.Kom., M.T.
2. ERICK KURNIAWAN, S.Kom., M.Kom.
3. ARGO WIBOWO, ST., MT.
4. KATON WIJANA, S.Kom., M.T.

1.   
2.   
3.   
4. 

Mahasiswa yang diuji,

  
(DERRY SANTOSO)

Yogyakarta, 26 Juni 2018  
Ketua Tim Penguji

  
(YETLI OSLAN, S.Kom., M.T.)

Catatan: 1 (satu) lembar untuk mahasiswa  
1 (satu) lembar untuk arsip



Universitas Kristen Duta Wacana  
Fakultas Teknologi Informasi Program Studi Sistem Informasi  
Jl. Dr. Wahidin Sudirahusada 5-25 Yogyakarta 55224  
Telp.: (0274)563929 Faks.: (0274)513235



## LEMBAR PENGANTAR SKRIPSI

Dicetak tanggal: 22-02-2018 14:21:58

Nama Mahasiswa	DERRY SANTOSO
No. Induk Mahasiswa	72140011
Pengambilan Tugas Akhir	Semester GASAL Tahun Ajaran 2018/2019
Judul Tugas Akhir	Sistem Rekomendasi E-Commerce Amazon Berbasis Collaborative Filtering
Dosen Pembimbing I	YETLI OSLAN, S.Kom., M.T.
Dosen Pembimbing II	ERICK KURNIAWAN, S.Kom., M.Kom.





## FORMULIR PERBAIKAN (REVISI) SKRIPSI

Dicetak tanggal: 27-06-2018 08:07:31

Yang bertanda tangan di bawah ini:

Nama : DERRY SANTOSO  
NIM : 72140011  
Judul Skripsi : SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS COLLABORATIVE  
FILTERING

Tanggal Pendadaran : Selasa, 26 Juni 2018 pukul 15:00 WIB

Telah melakukan perbaikan tugas akhir dengan lengkap.

Demikian pernyataan kami agar dapat dipergunakan sebagaimana mestinya.

Yogyakarta, Rabu, 27 Juni 2018

Dosen Pembimbing I

YETLI OSLAN, S.Kom., M.T.

Dosen Pembimbing II











ERICK KURNIAWAN, S.Kom., M.Kom.













## KARTU KONSULTASI SKRIPSI

Mulai Sem. Gsl/2018

NIM : 72140011 Nama Mahasiswa : DERRY SANTOSO  
Judul Skripsi : SISTEM REKOMENDASI E-COMMERCE AMAZON BERBASIS COLLABORATIVE FILTERING  
Pembimbing I : YETLI OSLAN, S.Kom., M.T. Pembimbing II : ERICK KURNIAWAN, S.Kom., M.Kom.

Pembimbing I	
1	Tanggal Konsultasi : 2-4-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>o Coba pertimbangan &amp; memasukkan hasil ke tabel / file</li> <li>o Lanjutkan ...</li> </ul>	 
2	Tanggal Konsultasi : 19-4-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>o Demo hasil reset (Rekomendasi) → Ok</li> <li>→ buat tampilan front end</li> </ul>	 
3	Tanggal Konsultasi : 2-5-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>o Demo program</li> <li>→ buat tampilan rekomendasi dari fix (5 per baris)</li> <li>→ Lanjutkan</li> </ul>	 
4	Tanggal Konsultasi : 14-5-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>o Revisi Bab-3</li> <li>→ format penulisan</li> <li>o Buat Bab 4 &amp; 5</li> </ul>	 
5	Tanggal Konsultasi : 22-5-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>o Bab 4 → tambahkan analisis</li> <li>o Buat Bab 5</li> </ul>	 

Pembimbing II	
1	Tanggal Konsultasi : 02-04-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>- konsul teknis consume API</li> </ul>	 
2	Tanggal Konsultasi : 16-04-2018
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>- konsul API, cara penganggilan</li> </ul>	 
3	Tanggal Konsultasi :
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>- konsul Guzzle dan API requests</li> </ul>	 
4	Tanggal Konsultasi :
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>- demo program</li> </ul>	 
5	Tanggal Konsultasi :
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
<ul style="list-style-type: none"> <li>- Bab 3</li> </ul>	 



6 Tanggal Konsultasi : 24-5-2018	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
o Acc s/d bab 5 → Cetak lengkap	YB
	Paraf Mahasiswa:
	hary
7 Tanggal Konsultasi : 31-5-2018	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
o Buat abstraksi 3 alinea + kata kunci	YB
	Paraf Mahasiswa:
	hary
8 Tanggal Konsultasi : 4-6-2018	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
o Acc Pddk P	YB
	Paraf Mahasiswa:
	hary
9 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
	Paraf Mahasiswa:
10 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
	Paraf Mahasiswa:
11 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
	Paraf Mahasiswa:

6 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
Bab 4	hary
	Paraf Mahasiswa:
	hary
7 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
Bab 5	hary
	Paraf Mahasiswa:
	hary
8 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
- Acc pendadaran	hary
	Paraf Mahasiswa:
	hary
9 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
	Paraf Mahasiswa:
10 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
	Paraf Mahasiswa:
11 Tanggal Konsultasi :	
Catatan Perkembangan/Revisi Skripsi:	Paraf Dosen:
	Paraf Mahasiswa:

Mengetahui,  
Koordinator Skripsi SI  
  
(Drs. Wimmie Handiwidjono, MIT.)

NIM: \_\_\_\_\_