e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

Sistem Rekomendasi Menggunakan *Item-Based Collaborative Filtering* Berbasis *Mobile* iOS

Bagas Ilham Rabbani, Herlina Napitupulu, Elis Hertini
Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan, Universitas Padjadjaran
Email: bagas19006@mail.unpad.ac.id, herlina@unpad.ac.id, elis.hertini@unpad.ac.id

Abstrak

Kecerdasan buatan menjadi topik yang sering diperbincangkan dalam beberapa tahun terakhir. Salah satu implementasi kecerdasan buatan adalah pembuatan sistem yang dapat memberikan rekomendasi. Adanya sistem rekomendasi yang bisa memberikan saran seperti pada rekomendasi restoran, kursus daring, mata kuliah, dan lain sebagainya. Pilihan-pilihan yang direkomendasikan tersebut akan membantu *user* terutama untuk cakupan pilihan yang sangat banyak. Penelitian ini bertujuan untuk menghasilkan perhitungan nilai kemiripan antar *item* dalam bahasa pemrograman Swift, perhitungan prediksi nilai *rating* dalam bahasa pemrograman Swift, dan implementasi sistem rekomendasi dalam aplikasi mobile iOS.

Kata Kunci: Sistem Rekomendasi, *Item-Based Collaborative Filtering*, Aplikasi *Mobile* iOS, *k-Nearest Neighbors*

Abstract

Artificial intelligence has become a topic that is often discussed in recent years. One of the implementations of artificial intelligence is the creation of a system that can provide recommendations. Recommender system can provide suggestions such as restaurant recommendations, online courses, and so on. The recommended options will help the user, especially for the wide range of choices. This study aims to discuss the creation of an item-based collaborative filtering recommendation system application that can calculate the similarity value between objects which is then used to provide recommendations to users based on rating values from other users. This paper also discusses the program code for making a recommendation system for iOS mobile applications.

Keywords: *Recommender System, Item-Based Collaborative Filtering, Mobile iOS Application, k-Nearest Neighbors*

1 PENDAHULUAN

Kecerdasan buatan (*artificial intelligence* atau AI) semakin banyak digunakan dalam kehidupan sehari-hari. Pesatnya perkembangan teknik komputasi dan pemrosesan informasi telah mempercepat kemajuan dan aplikasi AI, yang bertujuan untuk memungkinkan komputer melakukan berbagai tugas dengan mensimulasikan kecerdasan manusia (Duan *et al*., 2019). Saat ini, AI telah diterapkan di berbagai bidang, seperti pengenalan gambar dan ucapan, pengambilan keputusan, dan pemrosesan bahasa alami serta penerjemahan antar bahasa (Hwang *et al*., 2020). Salah satu cabang AI di

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

bidang pengambilan keputusan adalah sistem rekomendasi.

Sistem rekomendasi adalah perangkat lunak atau teknik yang memberikan saran atau rekomendasi item yang akan digunakan oleh pengguna (Ricci *et al*., 2015). Salah satu metode dari sistem rekomendasi adalah *collaborative filtering*, yaitu memberikan rekomendasi item kepada pengguna berdasarkan preferensi pengguna lain sebelumnya. *Item-based collaborative filtering* adalah sistem rekomendasi yang mengacu pada item yang mirip dengan item lain yang sebelumnya disukai oleh pengguna lain (Sarwar *et al*., 2001).

Item-based collaborative filtering telah banyak digunakan dalam penelitian pembuatan algoritma sistem rekomendasi, beberapa di antaranya adalah: Jepriana & Hanief (2020) dengan studi kasus konsentrasi jurusan di STMIK STIKOM Bali; Pangestu (2022) dengan studi kasus materi kursus online Coursera, dan Yusmar dkk. (2021) dengan studi kasus restoran.

Pembuatan aplikasi *mobile* iOS sistem rekomendasi *item-based collaborative filtering* menggunakan bahasa pemrograman Swift dapat menjadi dasar untuk pengembangan sistem rekomendasi yang dapat berjalan secara *native* pada perangkat iOS dan untuk digunakan pada berbagai kasus sesuai kebutuhan.

Pada penelitian ini dibahas bagaimana perhitungan nilai kemiripan antar item dalam bahasa pemrograman Swift, bagaimana perhitungan prediksi nilai rating dalam bahasa pemrograman Swift, dan implementasi sistem rekomendasi dalam aplikasi mobile iOS.

2 KAJIAN PUSTAKA

2.1 Sistem Rekomedasi

Sistem Rekomendasi adalah alat dan teknik perangkat lunak yang memberikan saran untuk *item* yang berguna bagi *user*. Saran yang diberikan kepada *user* berkaitan dengan berbagai macam proses pengambilan keputusan seperti mengenai barang apa yang harus dibeli, lagu apa yang harus didengar, atau berita daring apa yang harus dibaca. Sistem rekomendasi sangat bermanfaat ketika *user* dihadapkan dengan banyaknya pilihan yang ditawarkan oleh sebuah layanan (Ricci *et al*., 2015)

2.2 Item-Based Collaborative Filtering

Collaborative filtering adalah metode sistem rekomendasi yang membandingkan preferensi dari tiap *user* yang telah menilai produk dengan cara yang sama terhadap *user* aktif lainnya (Jannach *et al*., 2012). Algoritma *collaborative filtering* bertujuan memberikan saran item baru atau memprediksi kegunaan dari suatu item untuk *user* tertentu berdasarkan apa yang disukai sebelumnya oleh *user* atau pendapat pengguna lain yang berpikiran sama (Sarwar *et al*., 2001).

Item-based collaborative filtering berfokus pada kumpulan item-item yang sudah diberi rating oleh *user* aktif, dan menghitung seberapa mirip *item-item* tersebut dengan item i yang dituju. Kemudian diambil sebanyak *k item* yang paling mirip $\{i_1, i_2, \dots, i_k\}$. Ketika *item-item* yang paling mirip sudah ditemukan, prediksi rating dihitung berdasarkan nilai *rating* yang telah diberikan *user* terhadap item dengan *k item* yang paling mirip dengan item tersebut (Sarwar *et al.*, 2001).

2.3 K-Nearest Neighbor

Algoritma *k-nearest neighbor*, juga dikenal sebagai kNN, adalah sebuah *classifier* pembelajaran non-parametrik yang diawasi, yang menggunakan kedekatan (*proximity*) untuk membuat klasifikasi atau prediksi tentang pengelompokan titik data individu. Meskipun dapat digunakan untuk masalah regresi atau klasifikasi, ini biasanya digunakan sebagai algoritma klasifikasi, yang bekerja dengan mengasumsikan bahwa titik serupa dapat ditemukan di dekat satu sama lain (IBM, 2022). Dalam sistem rekomendasi, algoritma kNN digunakan untuk menentukan berapa banyak nilai kemiripan tertinggi yang digunakan untuk menghitung prediksi nilai rating untuk suatu item yang belum diberi

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

rating oleh *user*. Algoritma kNN dijelaskan sebagai berikut (Christopher, 2021):

a. Tentukan nilai k dari banyaknya tetangga.
b. Hitung nilai jarak antara titik data dengan semua tetangga.
c. Ambil sebanyak k tetangga terdekat berdasarkan jarak terdekat.
d. Dari k tetangga terdekat tersebut, hitung banyaknya titik data untuk tiap kategori.
e. Tentukan kategori titik data berdasarkan kategori dengan banyak anggota maksimum.

2.4 *Adjusted Cosine Similarity*

Cosine similarity digunakan untuk menghitung kemiripan antara *item-item*, yaitu dengan menganggap *rating-rating* untuk item adalah vektor-vektor dalam ruang berdimensi m (banyaknya baris pada matriks rating $m \times n$) dan mengukur kemiripan antar *item* dengan menghitung *cosinus* antara vektor-vektor tersebut (Sarwar *et al*., 2001).Perhitungan nilai *cosine similarity* antara item i dan item j $sim(i,j)$ ditunjukkan pada persamaan

$$sim(i,j) = \cos(\vec{\imath},\vec{\jmath}) = \frac{\vec{\imath} \cdot \vec{\jmath}}{\|\vec{\imath}\| * \|\vec{\jmath}\|} \qquad (1)$$

di mana:

$\vec{\imath}$: vektor *item i*
$\vec{\jmath}$: vektor *item j*
$\cos(\vec{\imath},\vec{\jmath})$: nilai *cosinus* antara $\vec{\imath}$ dan $\vec{\jmath}$.

Vektor $\vec{\imath}$ dan $\vec{\jmath}$ merupakan vektor kolom dengan anggota nilai *rating* pada *item i* dan *item j*. Nilai $\cos(\vec{\imath},\vec{\jmath})$ merupakan nilai *cosinus* sudut yang dibentuk antara $\vec{\imath}$ dan $\vec{\jmath}$. Nilai $sim(i,j)$ berkisar antara 0 sampai 1. Semakin besar nilainya, semakin mirip *item* i dan *item* j.

Adjusted cosine similarity digunakan untuk memperbaiki kekurangan dari perhitungan *cosine similarity*, yaitu perbedaan pada skala *rating* antar *user* yang tidak diperhitungkan sebelumnya. Perbaikan oleh *adjusted cosine similarity* dilakukan dengan melakukan pengurangan nilai *rating* dengan rata - rata *rating* yang diberikan oleh setiap *user* pada kasus *co–rated* (Jepriana dan Hanief, 2020). Perhitungan nilai kemiripan *item i* dan *item j* menggunakan *adjusted cosine similarity* $sim_{adj}(i,j)$ ditunjukkan pada persamaan

$$sim_{adj}(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_u)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_u)^2}} \qquad (2)$$

di mana

$\bar{R}_u$: rata-rata *rating* dari *user* ke-*u*
$R_{u,i}$: *rating user* ke-*u* untuk *item* ke-*i*
$R_{u,j}$: *rating user* ke-*u* untuk *item* ke- *j*.

Nilai $sim_{adj}(i,j)$ berkisar antara -1 sampai 1. Semakin besar nilainya, semakin mirip *item* i dan *item* j.

2.5 Adjusted Weighted Sum

Weighted sum (jumlah pembobotan) digunakan untuk menghitung prediksi dari *item j* terhadap *user u* dengan menghitung jumlah dari *rating* yang diberikan oleh *user* dengan *item* yang mirip dengan *item i*. Setiap *rating* dibobotkan sesuai dengan nilai kemiripannya pada $sim(i,j)$ (Sarwar *et al*., 2001). *Weighted sum* ditunjukkan pada persamaan

$$P_{u,j} = \frac{\sum_{i \in I(u)} sim(i,j) * R_{u,i}}{\sum_{i \in I(u)} sim(i,j)} \qquad (3)$$

di mana $P_{u,j}$ adalah prediksi untuk *user u* terhadap *item j*, $sim(i,j)$ menyatakan nilai kemiripan *cosine similarity* antara *item i* dan *item j*, dan $I(u)$ menyatakan himpunan *item* yang telah diberikan *rating* oleh *user u*.

Secara umum, pendekatan ini mencoba mempelajari bagaimana *user* memberikan *rating* terhadap *item* yang mirip. Perhitungan ini diskalakan dengan jumlah nilai kemiripan untuk memastikan prediksi berada dalam rentang yang telah ditentukan.

Perhitungan prediksi dengan menggunakan *weighted sum* tidak dapat digeneralisasi untuk kumpulan data yang memiliki nilai *sparse* yang tinggi. Hasil perhitungan nilai menjadi kurang tepat jika pada hasil dari *item–item similarity* terdapat nilai negatif, lalu diperkirakan membuat hasil prediksi *rating* yang nilainya negatif juga (Ghazanfar & Prugel-Bennett, 2010).

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

Perhitungan *adjusted weighted sum* ditunjukkan pada persamaan

$$P_{u,j} = \bar{R}_j + \frac{\sum_{i=1}^{n}(R_{u,i} - \bar{R}_i)\, sim_{adj}(i,j)}{\sum_{i=1}^{n}|sim_{adj}(i,j)|} \quad (4)$$

di mana

$P_{u,j}$: prediksi untuk *user u* terhadap *item j*
$sim_{adj}(i,j)$: nilai kemiripan *adjusted cosine* antara *item i* dan *item j*
$\bar{R}_j$: rata-rata *rating* untuk *item j*
$R_{u,i}$: *rating* dari *user u* untuk *item i*
$\bar{R}_i$: rata-rata *rating* untuk *item i.*

2.6 Swift

Swift adalah bahasa pemrograman yang dikembangkan oleh Apple Inc. dan komunitas *open source* pada tahun 2014 untuk menggantikan bahasa Objective-C yang digunakan Apple secara luas sebelumnya. Swift merupakan bahasa yang ringkas namun tetap ekspresif, mendukung banyak fitur modern, dan diperbarui secara berkala (Apple Inc, 2020).

2.7 Xcode

Xcode adalah aplikasi perangkat lunak yang dikembangkan oleh Apple Inc. yang menyediakan fasilitas komprehensif untuk pengembangan perangkat lunak. Xcode digunakan untuk mengembangkan perangkat lunak atau aplikasi untuk perangkat yang menjalankan sistem operasi macOS, iOS, iPadOS, watchOS, dan tvOS (Apple Inc., 2018). Pengembang dapat menggunakan perangkat asli atau menggunakan *simulator* yang sudah terintegrasi dengan Xcode untuk menjalankan aplikasi yang dibangun.

2.8 iOS

iOS adalah sistem operasi eksklusif untuk perangkat iPhone dan iPod yang dikembangkan oleh Apple Inc. pada tahun 2007, dan diperbarui secara *major* setiap tahun. Pengembangan aplikasi iOS dapat memanfaatkan kelebihan yang dimiliki oleh bahasa pemrograman Swift, dan lebih optimal jika dibangun dengan menggunakan Xcode, walaupun pengembangan dapat menggunakan bahasa lain seperti *framework* React Native dan perangkat lunak Visual Studio Code. Aplikasi yang dibangun menggunakan bahasa Swift (*native*) dan didesain untuk dijalankan pada perangkat iOS hanya bisa dibangun dengan menggunakan Xcode.

3 METODE PENELITIAN

3.1 Objek Penelitian

Objek penelitian dalam penelitian ini adalah sistem rekomendasi *item-based collaborative filtering* yang diimplementasikan dalam bahasa Swift untuk dijalankan pada perangkat iPhone.

3.2 Metode Penelitian

Pada penelitian ini, dibuat sistem rekomendasi menggunakan metode *item-based collaborative filtering* menggunakan bahasa pemrograman Swift. Sistem rekomendasi yang dibuat memiliki kemampuan menghitung nilai kemiripan antar *item* dan menghitung prediksi nilai *rating*.

Langkah awal dalam proses rekomendasi berdasarkan nilai *rating* yang diberikan oleh *user* adalah menghitung nilai kemiripan antara *item*. Perhitungan dilakukan menggunakan *adjusted cosine similarity* yang menghasilkan nilai antara -1 hingga 1, dengan nilai 1 menunjukkan kedua *item* memiliki nilai *rating* yang sama persis dan nilai -1 menunjukkan adanya perbedaan rating yang maksimum. Perhitungan nilai *adjusted cosine similarity* dihitung dengan menggunakan Persamaan (2). Setelah mendapatkan nilai kemiripan *adjusted cosine similarity*, nilai-nilai tersebut dapat diurutkan berdasarkan kedekatannya dan digunakan dengan jumlah data yang ditentukan berdasarkan nilai kNN yang dipilih.

Langkah selanjutnya adalah menghitung prediksi nilai *rating* untuk mata kuliah yang belum pernah diberi *rating* oleh *user*. Pada tahap ini, perhitungan prediksi dilakukan menggunakan *adjusted weighted sum* berdasarkan nilai kemiripan dari

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

perhitungan *adjusted cosine similarity*. Perhitungan *adjusted weighted sum* dihitung dengan menggunakan Persamaan (4).

4 HASIL DAN PEMBAHASAN

4.1 Pembentukan Objek Data

Sebelum membuat matriks *user-item* untuk perhitungan nilai kemiripan dan prediksi, langkah awal yang dilakukan adalah mempersiapkan objek-objek yang merepresentasikan data yang telah dikumpulkan. Hal ini bertujuan untuk memudahkan penggunaan data secara tepat dan efektif pada proses perhitungan berikutnya. Sebagai contoh, sistem rekomendasi yang dibuat adalah sistem yang memberikan rekomendasi suatu produk barang. Maka akan dibentuk objek produk, objek *user*, dan objek *rating*.

Objek pertama dan kedua adalah objek produk dan objek *user*, dan masing-masing objek memiliki atribut ID dan nama, dengan tipe data String. Objek ketiga adalah objek *rating*, dengan atribut ID *user* bertipe data String, ID *item* bertipe data String, dan nilai *rating* bertipe data Double.

```
struct Product {
var id: String?
var name: String?
}

struct User {
var id: String?
var name: String?
}

struct Rating {
var userId: String?
var itemId: String?
var value: Double?
}
```

Gambar 1. Pembentukan objek data produk, *user*, dan *rating*.

4.2 Perhitungan Nilai Kemiripan

Perhitungan nilai kemiripan antar *item* dilakukan untuk menentukan seberapa mirip suatu *item* dengan *item* lainnya berdasarkan data *rating* yang diberikan oleh *user*. Nilai kemiripan dihitung dengan menggunakan *adjusted cosine similarity* sesuai dengan persamaan 2.2. Berikut adalah kode fungsi menghitung nilai kemiripan, dengan parameter masukan kumpulan semua *rating* dan ID dari dua *item* yang akan dihitung nilai kemiripannya.

```
func cosineSimilarity(
    ratings: [Rating],
    item1: String?,
    item2: String?,
    isAdjusted: Bool
) -> Double {

let adjustedPrefix = isAdjusted ? "Adjusted " : ""
let ratingAverage = ratingAverage(by: .user)
let item1Ratings = ratings.filter({ $0.itemId ?? "0" ==
item1 })
let item2Ratings = ratings.filter({ $0.itemId ?? "0" ==
item2 })
var numerator = 0.0 // pembilang
var item1Denominator = 0.0 // penyebut (kiri)
var item2Denominator = 0.0 // penyebut (kanan)
for rating1 in item1Ratings {
for rating2 in item2Ratings {
if let firstRatingValue = rating1.value,
let secondRatingValue = rating2.value,
rating1.userId == rating2.userId {
let averageRatingUser1 = isAdjusted ?
ratingAverage[rating1.userId ?? "0"] ?? 0.0 : 0.0
let averageRatingUser2 = isAdjusted ?
ratingAverage[rating2.userId ?? "0"] ?? 0.0 : 0.0
let ratingValue1 = firstRatingValue -
averageRatingUser1
let ratingValue2 = secondRatingValue -
averageRatingUser2
numerator += ratingValue1 * ratingValue2
item1Denominator += pow(ratingValue1, 2)
item2Denominator += pow(ratingValue2, 2)}}}
if item1Denominator == 0.0 || item2Denominator ==
0.0 { return 0.0 }
let similarityValue = numerator /
(sqrt(item1Denominator) * sqrt(item2Denominator))
return similarityValue
}
```

Gambar 3. Kode fungsi perhitungan nilai kemiripan antar *item*.

4.3 Penentuan Nilai kNN

Tahap penentuan kNN dilakukan untuk menentukan berapa banyak nilai kemiripan yang akan digunakan untuk perhitungan prediksi nilai *rating* untuk setiap *item*.

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

4.4 Perhitungan Prediksi Nilai *Rating*

```
func weightedSum(
        ratings: [FBRating],
        userId: String?,
        itemId: String?,
        isAdjusted: Bool,
        k: Int,
        similarityValues: [SimilarityValue]? = nil,
        _ completion: @escaping ([SimilarityValue]) ->
Void
    ) -> Double {
        let allCourse = FBCourse.allCourses()
        let adjustedPrefix = isAdjusted ? "Adjusted " : ""
        let ratingAverage = ratingAverage(by: .item)
        var sum = 0.0
        var similaritySum = 0.0
        var averageUnratedItem = 0.0
        var neighborRatings: [(rating: FBRating, similarity:
Double)] = []
        var newSimilarityValues = similarityValues == nil ?
[SimilarityValue]() : similarityValues!
for rating in ratings {
if rating.userId == userId && rating.itemId != itemId {
let averageRatingItem = isAdjusted ?
(ratingAverage[rating.itemId ?? ""] ?? 0.0) : 0.0
averageUnratedItem = isAdjusted ?
(ratingAverage[itemId ?? ""] ?? 0.0) : 0.0
let ratingValue = (rating.value ?? 0) - averageRatingItem
let similarity = newSimilarityValues.first(where: {
($0.course.id == itemId && $0.similarToCourse.id ==
rating.itemId)
|| ($0.course.id == rating.itemId &&
$0.similarToCourse.id == itemId) })?.similarityValue ??
cosineSimilarity(ratings: ratings, item1: itemId, item2:
rating.itemId, isAdjusted: isAdjusted)
neighborRatings.append((rating: rating, similarity:
similarity))
if !newSimilarityValues.contains(where: { $0.course.id ==
itemId && $0.similarToCourse.id == rating.itemId }) {
newSimilarityValues.append(SimilarityValue(course:
allCourse.first(where: { $0.id == itemId })!,
similarToCourse: allCourse.first(where: { $0.id ==
rating.itemId })!,                similarityValue:
similarity))}
sum += similarity * ratingValue

similaritySum += abs(similarity)
        } else if rating.userId == userId && rating.itemId
== itemId {
            return rating.value ?? 0
        }
    }
```

Gambar 3. Kode fungsi perhitungan prediksi nilai *rating*.

Perhitungan prediksi rating dilakukan untuk memperkirakan nilai *rating* yang akan diberikan mahasiswa terhadap suatu mata kuliah yang belum diberikan *rating*. Nilai kemiripan antar mata kuliah dihitung dengan menggunakan *adjusted weighted sum* sesuai dengan Persamaan (4). Berikut adalah kode fungsi menghitung prediksi nilai *rating*, dengan parameter masukan kumpulan semua nilai *rating*, ID *user*, ID *item*, dan nilai kNN sebagai penentuan nilai kNN itu sendiri.

4.5 Implementasi Sistem Rekomendasi Pada Aplikasi iOS

Fungsi-fungsi sistem rekomendasi yang telah dibuat kemudian dikumpulkan dalam satu objek *class*. Berikut adalah kode deklarasi *class* sistem rekomendasi dengan fungsi-fungsinya.

```
final class IBCFRecommenderSystem {
func cosineSimilarity(
        ratings: [Rating],
        item1: String?,
        item2: String?,
        isAdjusted: Bool
    ) -> Double {
        let ratingAverage = ratingAverage(by: .user)
        let item1Ratings = ratings.filter({ $0.itemId ?? "0" ==
item1 })
...
```

Gambar 4. Deklarasi *class* objek sistem rekomendasi.

Sebagai contoh, dideklarasikan sebuah konstanta *class* sistem rekomendasi di dalam sebuah *view controller*. Untuk menghitung nilai kemiripan dan prediksi nilai *rating* bagi suatu *user*, cukup dengan memanggil fungsi-fungsi yang telah dibuat dengan memasukkan parameter masukan yang telah ditentukan sebelumnya.

Kode tersebut akan mencetak pada konsol nilai kemiripan antar *item* dan nilai prediksi *rating* untuk suatu *item* sesuai dengan parameter masukan yang disediakan. Implementasi sistem rekomendasi dapat disesuaikan dengan kebutuhan, cukup dengan mengganti parameter masukan dari tiap fungsi seperti: siapa *user* yang akan diberi nilai prediksi *rating*, *item* mana yang akan dihitung nilai kemiripannya dibanding *item* lain, dan lain sebagainya.

e-ISSN: 2655-867X
p-ISSN: 2655-8661
Volume 5 No. 2 Agustus 2023

```
final class HomeViewController {
let recommenderSystem = IBCFRecommenderSystem()
...
override func viewDidLoad() {
...
print(recommenderSystem.cosineSimilarity(
      ratings: [...],
      item1: "1",
      item2: "2",
      isAdjusted: true
  )) // prints similarity value to console
print(recommenderSystem.weightedSum(
      ratings: [...],
      userId: "1",
      itemId: "3",
      isAdjusted: true,
      k: 50
     ) { _ in } )
// prints weighted sum value to console
...
}
```

Gambar 5. Kode contoh implementasi sistem rekomendasi pada suatu halaman *home* aplikasi iOS.

5 SIMPULAN

Perhitungan nilai kemiripan antar *item* dapat dilakukan menggunakan metode *adjusted cosine similarity*, dan dapat dituliskan dalam bahasa pemrograman Swift dengan kode yang telah ditunjukkan pada pembahasan. Perhitungan nilai kemiripan antar *item* dapat dilakukan menggunakan metode *adjusted weighted sum*, dan dapat dituliskan dalam bahasa pemrograman Swift dengan kode yang telah ditunjukkan pada pembahasan.

Sistem rekomendasi dapat diimplementasikan dalam aplikasi mobile iOS sesuai dengan kebutuhan, seperti pada contoh yang telah ditunjukkan pada pembahasan. Sistem dibuat dalam bentuk objek *class.*

DAFTAR PUSTAKA

Apple Inc. (2018). *Xcode - Apple Developer*. Apple.

Apple Inc. (2020). Swift - Apple Developer. In *Swift: The powerful programming language that is also easy to learn.*

Duan, Y., Edwards, J. S., & Dwivedi, Y. K. (2019). Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda. *International Journal of Information Management*, *48*. https://doi.org/10.1016/j.ijinfomgt.2019.01.021

Ghazanfar, M., & Prugel-Bennett, A. (2010). Novel Significance Weighting Schemes for Collaborative Filtering: Generating Improved Recommendations in Sparse Environments. *DMIN'10, the 2010 International Conference on Data Mining!*

Hwang, G. J., Xie, H., Wah, B. W., & Gašević, D. (2020). Vision, challenges, roles and research issues of Artificial Intelligence in Education. In *Computers and Education: Artificial Intelligence* (Vol. 1). https://doi.org/10.1016/j.caeai.2020.100001

Jannach, D., Zanker, M., Ge, M., & Gröning, M. (2012). Recommender systems in computer science and information systems - A landscape of research. *Lecture Notes in Business Information Processing*, *123 LNBIP*. https://doi.org/10.1007/978-3-642-32273-0_7

Jepriana, I. W., & Hanief, S. (2020). Metode Item-Based Collaborative Filtering Untuk Model Sistem Rekomendasi Konsentrasi Penjurusan Di Stmik Stikom Bali. *Jurnal Teknologi Informasi Dan Komputer*, *6*(1).

Ricci, F., Shapira, B., & Rokach, L. (2015). Recommender systems handbook, Second edition. In *Recommender Systems Handbook, Second Edition*. Springer US. https://doi.org/10.1007/978-1-4899-7637-6

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web, WWW 2001*. https://doi.org/10.1145/371920.372071