UNIVERSIDAD DE LOS ANDES

FACULTAD DE CIENCIAS ECONÓMICAS Y SOCIALES

INSTITUTO DE ESTADÍSTICA APLICADA Y COMPUTACIÓN

PROGRAMA DE MAESTRÍA EN ESTADÍSTICA

METODOLOGÍA PARA EL DESARROLLO DE SISTEMAS DE RECOMENDACIÓN DE COMERCIO ELECTRÓNICO BASADA EN EL FILTRADO COLABORATIVO CON RETROALIMENTACIÓN IMPLÍCITA.

UN NUEVO CAMPO DE ACCION DE LA ESTADÍSTICA.

Autor: María Elena Naranjo Sánchez

Tutor: José Luciano Maldonado

TRABAJO DE GRADO

Presentado ante la Ilustre Universidad de Los Andes
como requisito final para optar al Grado Académico de
Magíster Scientiae en Estadística

MÉRIDA, VENEZUELA
Enero, 2017

RESUMEN

METODOLOGÍA PARA EL DESARROLLO DE SISTEMAS DE RECOMENDACIÓN DE COMERCIO ELECTRÓNICO BASADA EN EL FILTRADO COLABORATIVO CON RETROALIMENTACIÓN IMPLÍCITA. UN NUEVO CAMPO DE ACCION DE LA ESTADÍSTICA.

por

María Elena Naranjo Sánchez

Los sistemas de recomendación, basados específicamente en el filtrado colaborativo, son sistemas que generan recomendaciones a un usuario activo según la similitud que éste tiene con otros usuarios.

En este trabajo se propone una metodología para desarrollar sistemas de recomendación enfocados a los comercios electrónicos, recurriendo tan solo al historial de visitas, partiendo de que difícilmente un cliente califica un producto en todas las visitas. El tiempo relativo que ha estado cada cliente en un ítem es una información implícita que se aprovecha en esta investigación para hacer recomendaciones que puedan ser de interés del usuario activo.

La metodología fue establecida haciendo uso de la técnica de los k vecinos más cercanos con base a la información implícita asociada al tiempo de visita a los ítems. Se desarrollaron programas de computación en los que se implementa el algoritmo para encontrar los k vecinos más cercanos de forma eficiente, con el fin de reducir, sustancialmente, el tiempo de generación de las recomendaciones. Esta propuesta metodológica fue validada con los datos de un determinado comercio electrónico; los resultados obtenidos son alentadores como alternativa para la implementación de los sistemas de recomendación.

Palabras Claves: *Sistemas de recomendación, filtrado colaborativo, los k vecinos más cercanos, comercio electrónico, información implícita.*

INDICE

INDICE DE FIGURAS

INDICE DE TABLAS

CAPÍTULO I

Introducción

En este capítulo se presenta una breve descripción del tema que se aborda, específicamente se muestra la forma en que trabajan los sistemas de recomendación, se formula el problema tratado y se presentan los objetivos que fueron trazados para la investigación a partir de una diversidad de trabajos revisados como parte de los antecedentes, en los que se indican diferentes métodos para el desarrollo de los sistemas de recomendación.

1.1 Formulación del problema

Qué película ver, qué perfume comprar, qué música escuchar, a quién seguir en una red social, son recomendaciones que alguien podría darnos según nuestros gustos y necesidades. Pero, ¡qué tal si esas recomendaciones las hace un sistema de forma automática!

En este sentido, la diversidad de opciones que se le presentan a un usuario, de este tipo de sistemas, al realizar una búsqueda ha creado la dificultad de encontrar los ítems que más se adapten a sus intereses. Es así que surgen los Sistemas de Recomendación (SR), los cuales sugieren ítems que sean de interés para un usuario, siendo un “ítem” aquello que el sistema le recomienda al usuario.

La recomendación puede ser de varios tipos, entre ellas está la recomendación basada en el Filtrado Colaborativo (FC), que se fundamenta en que a un usuario se le recomiendan ítems que usuarios con gustos parecidos, tienen en su historial. Estos recomendadores se han convertido en una herramienta imprescindible en el

comercio electrónico (*e-commerce*) al ayudar a los consumidores a tomar una decisión en su compra [1].

Por otro lado, existen recomendaciones no personalizadas como serían: los productos más populares, los productos más vendidos, los productos mejor evaluados. Sin embargo, este tipo de recomendación no es objeto de estudio en este trabajo, puesto que esta investigación se centrará en crear un modelo para predecir los ítems que a un usuario le pueden interesar a partir de su similitud con otros usuarios. El filtrado colaborativo es más abierto, le puede recomendar al usuario un ítem que él ni se hubiera podido imaginar que existía o que necesitaría y todo en base a sus gustos. No se enfrasca en proporcionar ítems del mismo género, va más allá de lo que el usuario pudiera necesitar y le abre una ventana de posibilidades de ítems que por su cuenta no hubiera podido encontrar.

El sistema de recomendación basado en el filtrado colaborativo trabaja con una matriz de datos, como se muestra en la tabla 1, la cual está compuesta por un conjunto de m usuarios U y un conjunto de n ítems I. En la matriz se reflejan las opiniones de los clientes sobre ciertos productos; cada celda $r_{u,i}$ corresponde a la calificación que el usuario $u \in U$ tiene de un ítem $i \in I$, cada usuario tiene una única calificación para un ítem $i \in I$.

Tabla 1. Matriz de usuarios-ítems

	Ítems					
	I_1	I_2	...	I_j	...	I_n
U_1	5	3	?	?	1	?
U_2	?	?	?	?	5	?
Usuarios	?	5	2	2	?	?
U_x	?	?	?	3	?	5
:	4	2	?	?	2	?
U_m	3	?	?	?	1	4

Para que el SR pueda generar recomendaciones a los usuarios es indispensable tener información previa de los usuarios, referente al contenido que se va a recomendar. Esta información puede ser mediante retroalimentación implícita o retroalimentación explícita. La retroalimentación implícita es la observación que se

tiene sobre las acciones de un usuario, éstas podrían ser: número de veces que el usuario u compró un ítem i, tiempo que el usuario u escuchó la canción i, tiempo que el usuario u estuvo en la página i, entre otras acciones que no requieren de la intervención directa del usuario para calificar un ítem. La retroalimentación explicita se refiere a la calificación que un usuario u le asigna a un ítem i; en este último mecanismo se expresa de manera inequívoca el interés de un usuario hacia un ítem.

Luego, se trata de un problema de comercio electrónico en el que dada una secuencia de clics en la sesión de cierto usuario, en una página de comercio electrónico, se pretende generar recomendaciones a usuarios activos con el fin de alentar sus compras.

El proceso de retroalimentación de la información se efectúa mediante una de las acciones básicas que realizan los usuarios al visitar un comercio electrónico, estas acciones son los clics sobre los ítems. Partiendo de las acciones de aquellos usuarios que no han realizado una compra en la sesión, se pretende estudiar el registro de los clics de los usuarios no compradores y así asignarle una calificación a cada usuario sobre los ítems cliqueados [2]. En este caso, la calificación es el puntaje asignado a un usuario u sobre cierto ítem i, lo cual indicará el interés del usuario sobre el ítem.

En general, el primer problema que se plantea, en esta investigación, es crear una matriz de calificaciones a partir de los datos. Es decir, hay que determinar la calificación de un usuario sobre los ítems a partir de la interacción que haya tenido con esos ítems.

Por otro lado, resolver el problema del sistema de recomendación basado en el filtrado colaborativo implica realizar las siguientes tareas [3]:

- **Predicción:** Consiste en obtener el valor numérico $P_{a,j}$ que expresa la predicción de la calificación de un ítem i_j por un usuario activo u_a, e indica cuanto le gusta o disgusta el ítem. El valor predicho pertenece a la misma escala que las calificaciones provistas por u_a.

- **Recomendación:** Consiste en la selección de un conjunto de N ítems que se aconsejan a un usuario activo puesto que se estima que serán de su interés. Hay que tomar en cuenta que las recomendaciones se realizan sobre ítems que el usuario no ha cliqueado.

Existen muchas formas para generar las predicciones y recomendaciones, la mayoría de estos enfoques emplean técnicas estadísticas para abordar el problema. Las técnicas son usadas bien sea para la búsqueda de similitud entre usuarios según ítems, o para crear modelos que sirvan para obtener predicciones.

El problema, particularmente abordado en esta investigación, parte de las recomendaciones que se deben de hacer en un comercio electrónico tomando en cuenta el hecho que en el comercio electrónico difícilmente un cliente califica un producto, y que, en general, tan solo se cuenta con el historial de sus visitas. El tiempo relativo que estuvo cada cliente en un ítem es una información implícita que se puede aprovechar para hacer recomendaciones que puedan ser de interés del usuario.

A partir de la naturaleza del problema descrito se planteó la creación de una metodología en donde se busca alentar la compra, por parte de los usuarios, haciendo uso de una base de datos correspondiente a no compradores en un comercio electrónico. Estos datos pertenecen a la empresa alemana YOOCHOOSE, especializada en la recomendación de contenidos a partir de los comportamientos de los usuarios, y que fueron utilizados en el RecSys Challenge 2015 [4].

1.2 Antecedentes

Los sistemas de recomendación han alcanzado gran popularidad en años recientes debido a la cantidad de opciones que se tienen al tomar una decisión de compra electrónica. El enfoque que ha obtenido mayor interés en el desarrollo de los sistemas de recomendación es el del filtrado colaborativo, un método en el que se

recomiendan ítems a un usuario en base en las preferencias de otros usuarios; intuitivamente se asume que si un grupo de usuarios se asemejan en la calificación realizada a ciertos ítems, entonces, también se asemejarán en la calificación de cualquier otro ítem. Dos de las técnicas principales para este tipo de recomendación están basadas en la del vecino más cercano y los modelos de factores latentes [2].

A continuación se describen algunos trabajos y aplicaciones que se han realizado, en los últimos años, para generar recomendaciones tomando en cuenta estos métodos:

En [5] se divide el proceso de recomendación en tres tareas; *representación, formación de los vecinos* y *generación de las recomendaciones*. En la *representación* se plasma, en un modelo, los productos que ya han sido comprados por un cliente. En la *formación de los vecinos* se identifican aquellos usuarios más cercanos a otro en cuanto a preferencias (usuarios vecinos). En la *generación de las recomendaciones* se buscan los *N* ítems de mayor puntaje dentro de los vecinos del usuario. Para cada subproceso se proponen diferentes técnicas y se compara la calidad de predicción de cada una de ellas.

En el caso de Amazon.com, que utiliza recomendaciones como estrategia de mercadeo y personaliza la página según los intereses de cada cliente [1], el algoritmo usado para generar recomendaciones fue desarrollado por dicha Empresa y lo llamaron *item-to-item collaborative filtering* (filtrado colaborativo ítem a ítem). Este algoritmo les permite producir recomendaciones en tiempo real escalando en grandes conjuntos de datos para generar calidad en sus recomendaciones. El algoritmo se enfoca en encontrar ítems similares, no usuarios similares, para ello se construye una tabla de ítems similares agrupando los ítems que los usuarios tienden a comprar en conjunto; este procedimiento se realiza enfocándose en el método del vecino más cercano.

En [6] se propone un modelo de filtrado colaborativo usando Redes Bayesianas, en donde se incluyen simultáneamente ítems y usuarios en el modelo; el modelo usa variables latentes para describir usuarios e ítems abstractamente como vectores reales. Los investigadores centraron más la atención en los algoritmos de recomendación cuando Netflix lanzó un premio por 1 millón de dólares para mejorar las recomendaciones de películas. El objetivo de la competencia era construir un algoritmo de recomendación que pudiera mejorar el algoritmo existente en un 10%, lo cual generó un interés rotundo tanto en el mundo académico como en los aficionados [7]. Los ganadores del concurso de Netflix mezclaron una gran cantidad de técnicas para su sistema de recomendación, entre ellas, análisis de componentes principales, vecino más cercano y redes neuronales [8]. En los algoritmos propuestos se tomó en cuenta el efecto temporal [9].

En [10] se presenta un análisis para el conjunto de datos de Netflix. Usando el método del vecino más cercano, se desarrolló un modelo simple para predecir la calificación que un usuario le asigna a una película basada en la calificación que el mismo usuario le ha dado a películas similares. En una segunda parte del trabajo se implementaron dos métodos basados en el vecino más cercano que fueron propuestos por el equipo ganador de la competencia, antes mencionada, y se prueba el rendimiento de ambas pruebas con la base de datos de Netflix.

En los antecedentes descritos hasta ahora, a excepción del ganador del premio de Netflix son estáticos, se supone que las preferencias de los usuarios no varían con el tiempo.

En [11] se propone romper el supuesto de que los gustos de los usuarios tienen un comportamiento estático. Se demuestra que el comportamiento de los usuarios varía con el tiempo. Se modelan las preferencias de los usuarios usando Modelos Ocultos de Markov (MOM) y se realizan comparaciones con algoritmos estáticos propuestos recientemente. Se obtuvieron resultados prometedores en el uso de los MOM para predecir las preferencias de los usuarios. El algoritmo propuesto toma únicamente en cuenta la secuencia de las observaciones ignorando la valoración que el usuario le ha dado a los ítems.

Uno de los problemas que se presentan en los sistemas de recomendación es la sobre-especialización, el sistema le muestra al usuario elementos similares a los que ya ha visto anteriormente. En [12] tratan este problema tomando regiones, de los ítems, que no se encuentran sobreexpuestas al usuario; este método fue denominado *Outside-The-Box recommendation* (recomendaciones fuera de la caja). Se pudo demostrar que con este método las recomendaciones eran de alta calidad y diferían significativamente si se comparaba con las realizadas mediante los métodos tradicionales del filtrado colaborativo.

Como problema general, en muchos casos es difícil o hasta imposible obtener calificaciones de los usuarios, o las mismas pudieran no ser confiables. Para generar recomendaciones de calidad se requiere que la calificación de los usuarios refleje el verdadero valor de sus preferencias partiendo del uso de información implícita. En [13] se presenta un método de filtrado colaborativo de gran precisión para tratar información implícita, este método está basado en la técnica del vecino más cercano. Se trabaja con una matriz en la que se califica con 1 cuando el usuario *u* ha comprado el ítem *i* y con otra matriz donde se indica el tiempo en que un usuario compró, cierto producto, después que este fue lanzado al mercado. En los resultados se observó precisión al incorporar información temporal, confirmando que las compras más recientes reflejan mejor la preferencia actual de un usuario y los productos que han sido lanzados recientemente resultan más atractivos para los usuarios.

1.3 Justificación

Este trabajo aborda un campo de investigación, relativamente nuevo, prácticamente desconocido en nuestro país. Además de su uso en el área comercial, esta disciplina cuenta con una gran cantidad de aplicaciones y la Estadística es fundamental para validar los métodos y resultados.

Al trabajar con un conjunto de datos de millones de registros se debe buscar la forma de que su manejo sea eficiente en sentido computacional y, a su vez, se obtengan resultados consistentes, a pesar de los diversos métodos encontrados para la resolución del problema no se ha conseguido alguno donde aborden estos dos puntos, considerándose ambos importantes en los algoritmos de los sistemas de recomendación.

1.4 Objetivos

Objetivo general

Generar una metodología para el desarrollo de sistemas de recomendación de comercio electrónico basada en el filtrado colaborativo con retroalimentación implícita.

Objetivos específicos

- Comprender diferentes metodologías relacionadas con el desarrollo de los sistemas de recomendación.
- Establecer un algoritmo que convierta la información implícita en un valor numérico.
- Generar recomendaciones de calidad y consistentes con base al comportamiento de los usuarios.
- Alcanzar la escalabilidad del sistema de recomendación en sentido computacional.
- Validar experimentalmente el método.
- Generar conclusiones comparativas con distintos parámetros del método propuesto.

CAPÍTULO II

Marco teórico

En este capítulo se describen los fundamentos teóricos necesarios para el desarrollo de la investigación, específicamente se muestra, con suficiente detalle, la teoría asociada a los sistemas de recomendación, haciendo particular énfasis en que existen diferentes técnicas para su construcción; y se resalta que un sistema de recomendación puede variar según los intereses de su aplicación, incluso cada sistema de este tipo puede basarse en un algoritmo distinto. Igualmente, se hace énfasis en diferentes métodos estadísticos para realizar el filtrado colaborativo en el cual se centró este estudio.

2.1 Sistemas de recomendación

Los sistemas de recomendación constituyen un conjunto de técnicas y herramientas que proporcionan a un usuario sugerencias de ítems que se estima sean de su interés, como lo describe la fig. 1 [14]. Debido a su amplia generalidad de aplicación esta tecnología ha ido creciendo, siendo de interés tanto para los usuarios como para los proveedores de servicios electrónicos. Entre las motivaciones de los sistemas de recomendación para los usuarios se encuentran [15]: la búsqueda de ítems que le sean atractivos, la ubicación de los mejores ítems, encontrar ítems que complementen sus preferencias. Para los proveedores de servicios les resulta útil para incrementar el número de ítems vendidos, vender mayor variedad de ítems, incrementar la satisfacción del usuario, entre otras ventajas.

Figura 1. Esquema general de los sistemas de recomendación

Todo sistema de recomendación procesa datos, asociados con los ítems, para generar recomendaciones que reciben los usuarios.

En sentido general, un sistema de recomendación controla tres tipos de objetos: ítems, usuarios y transacciones [2].

- **Ítems:** Son los objetos a ser recomendados. Los ítems son caracterizados por su complejidad y utilidad. Pueden tomar diferentes calificaciones, entre ellas, positiva si resulta útil para el usuario o negativa si no es apropiada para el usuario.
- **Usuarios:** Son quienes utilizan los sistemas de recomendación; pueden tener diferentes tareas. Muchos sistemas de recomendación exploran la información del usuario para construir sus modelos de recomendación.
- **Transacciones:** Son los datos que se generan durante la interacción humano-computadora, constituyen la materia prima que utilizan los algoritmos de producción de las recomendaciones. En cada transacción se almacena la información de las preferencias de un usuario acerca de un ítem, esta información puede ser por retroalimentación explícita o implícita [2]. En la retroalimentación explícita se le pregunta al usuario su opinión acerca de un ítem en una escala de rangos. Las calificaciones pueden ser: numéricas (p. ej. 1 a 5 estrellas), binarias, ordinales, entre otras. En la retroalimentación implícita el sistema infiere la opinión de los usuarios con base a sus acciones, estas acciones pudieran ser el historial de compras o de búsqueda del

usuario, patrones de búsqueda, tiempo porcentual de visitas, número de veces que un usuario ha observado un ítem o hasta el movimiento del ratón de su computadora.

2.2 Técnicas de generación de recomendaciones

Existen diferentes formas para realizar recomendaciones. En [2] se plantean seis tipos, como se muestra en la fig. 2.

Figura 2. Técnicas de recomendación

- **Recomendación basada en contenido**

El sistema aprende a recomendar ítems que son similares a aquellos por los cuales el usuario ha demostrado interés en el pasado. La similitud en los ítems se calcula con las características asociadas a los ítems, p. ej. si un usuario ha escuchado una canción del género pop entonces el sistema le recomendará otras canciones de este género.

- **Filtrado colaborativo**

Este sistema es uno de los más populares actualmente, se basa en recomendar ítems a un usuario según los ítems que son de interés a usuarios similares. Este sistema tiene la ventaja de que las personas son las que valoran la calidad de los productos a recomendar. La investigación que se describe se enfocó en este tipo de técnica, por lo tanto, en la siguiente sección se muestra detalladamente en que consiste el filtrado colaborativo.

- **Recomendación demográfica**

Este tipo de sistema recomienda ítems con base al perfil demográfico del usuario, p. ej. usuarios que son direccionados a ciertas páginas basándose en su ubicación actual.

- **Recomendación basada en conocimiento**

Se realizan recomendaciones basándose en el conocimiento que da el usuario sobre sus necesidades o preferencias. Del conocimiento que tenga el sistema sobre los productos se realizan las recomendaciones que mejor se adapten al usuario. Las recomendaciones son estáticas, a los usuarios con iguales requerimientos se les proporciona la misma recomendación, p. ej. el usuario da un ejemplo del tipo de producto que está buscando y el sistema le recomendará productos similares al ejemplo que se le proporciona.

- **Recomendación basada en la comunidad**

Este tipo de sistema recomienda ítems basándose en las preferencias de los amigos de los usuarios. Se basa en la frase “Dime quiénes son tus amigos y te diré quién eres”.

- **Recomendación híbrida**

Se basa en la combinación de diferentes técnicas para realizar las recomendaciones, aprovechando las ventajas de una técnica para solventar las desventajas de otra. Por ejemplo, en el sistema de recomendación de filtrado colaborativo existe el problema del arranque en frio, en el cual no se pueden recomendar ítems debido a la escasez de datos por lo que se usan otras técnicas de sistemas de recomendación para resolver el problema.

2.3 El Filtrado Colaborativo (FC)

El supuesto fundamental del filtrado colaborativo es que si los usuario *X* y *Y* califican *n* ítems de forma similar, ellos comparten los mismos gustos y por consiguiente en el futuro calificarán otros ítems similares [9]. En la fig. 3 se observa un ejemplo de este tipo de sistema: el usuario A compra dos productos similares al usuario B, por lo que el tercer producto que compra A, que no tiene en común con B, sirve de recomendación para el usuario B.

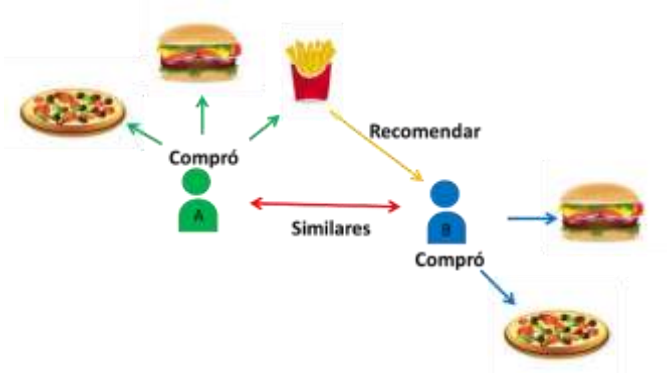

Figura 3. Supuesto del Filtrado Colaborativo

Los sistemas basados en el filtrado colaborativo surgieron de los sistemas basados en contenido, los cuales se enfocan en buscar ítems que al usuario le han gustado en el pasado, con ese tipo de sistema no se puede medir la calidad de las recomendaciones, algo que con el filtrado colaborativo sí, es decir, en el filtrado colaborativo se utiliza la información que varios usuarios han aportado sobre los ítems para realizar las recomendaciones. Con la opinión de otros usuarios se trata de predecir la valoración de un ítem para un usuario en particular. A partir de los gustos de un usuario se establece a qué grupo pudiera pertenecer según su similitud; de esta forma si en el grupo fue valorado un ítem de manera positiva se considera probable que el usuario activo valore de igual forma dicho ítem, y si no lo ha cliqueado el sistema se lo pudiera recomendar.

Los datos utilizados en el FC se controlan con una matriz de las preferencias de *m* usuarios por *n* ítems, como se observa en la tabla 1. Cada celda $r_{u,i}$ corresponde a

la calificación del usuario *u* sobre un ítem *i*. La calificación puede ser obtenida mediante retroalimentación explícita o implícita.

Tabla 1. Matriz de usuarios-ítems

			Ítems			
	I_1	I_2	...	I_j	...	I_n
U_1	5	3	?	?	1	?
U_2	?	?	?	?	5	?
Usuarios	?	5	2	2	?	?
U_x	?	?	?	3	?	5
:	4	2	?	?	2	?
U_m	3	?	?	?	1	4

Clasificación del filtrado colaborativo

El FC se puede clasificar de dos formas [16]: basados en memoria y basados en modelos.

a) Algoritmos basados en memoria

Son algoritmos que realizan la predicción tomando en cuenta toda la base de datos de los ítems calificados usando métricas de similitud. Entre estos algoritmos, el algoritmo basado en los vecinos más cercanos es el que sobresale por su popularidad. En este caso, cada usuario forma parte de un grupo con intereses similares, por lo tanto, se identifican los llamados “vecinos más cercanos” de un usuario y a partir de éstos se predicen las recomendaciones. Este tipo de algoritmo procesa la matriz cada vez que calcula una predicción o una recomendación.

La mayoría de los enfoques, con el algoritmo de los vecinos más cercanos, se generaliza a los siguientes pasos [17]:

I. Asignar un peso a todos los usuarios respecto a la similitud que tienen con el usuario activo (el usuario activo es a quien se le quiere generar la recomendación).
II. Seleccionar *k* usuarios que tienen la mayor similitud con el usuario activo (serán los llamados *vecinos más cercanos*).

III. Generar la predicción del usuario *a* sobre un ítem *i* a partir de las calificaciones de los *vecinos más cercanos* del usuario activo.

En la literatura se encuentran dos tipos de enfoques para buscar los vecinos más cercanos: basados en usuarios o basados en ítems; ambos esquemas siguen los mismos pasos explicados previamente. A continuación se describe cada uno de ellos.

El enfoque basado en usuarios

Como se muestra en la fig. 4, en este enfoque los ítems que previamente fueron calificados por un usuario activo juegan un papel importante en la búsqueda de sus vecinos más cercanos, siendo los vecinos más cercanos aquellos usuarios que tienen similitudes con el usuario activo.

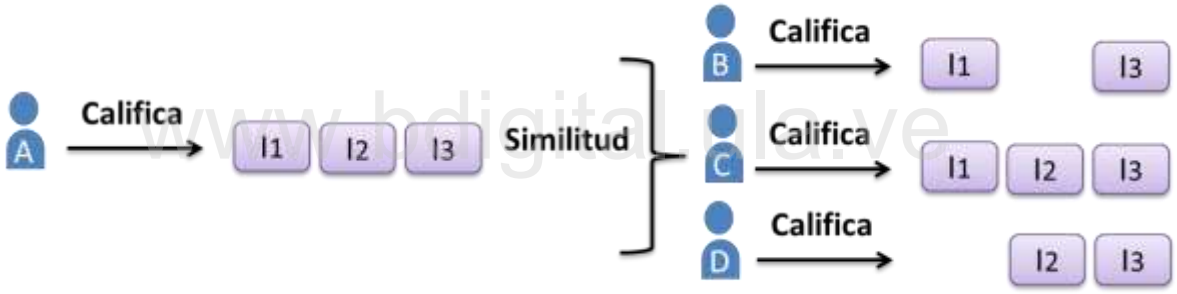

Figura 4. Sistema de Recomendación de filtrado colaborativo basado en usuarios

En el paso uno, para hallar los vecinos más cercanos se plantea buscar la similitud entre los usuarios, esta medida será el peso $w_{a,u}$ entre el usuario *u* y el usuario activo *a*. Existen ciertas medidas para la similitud, entre ellas, la más usual es la correlación de Pearson.

La correlación de Pearson se puede usar para calcular la similitud entre dos usuarios, *a* y *u* (ec. 1), según la correlación que presentan sus puntuaciones. Esta medida toma valores entre 1 y -1, donde el valor 0 indica ausencia de correlación entre los usuarios.

$$w_{a,u} = \frac{\sum_{i \in I}(r_{a,i} - \overline{r_a})(r_{u,i} - \overline{r_u})}{\sqrt{\sum_{i \in I}(r_{a,i} - \overline{r_a})^2 \sum_{i \in I}(r_{u,i} - \overline{r_u})^2}} \quad (1)$$

Dónde:

$w_{a,u}$, es el peso entre el usuario activo a y el usuario u. .

I, es el conjunto de ítems ya calificados por los dos usuarios.

$r_{a,i}$, es la calificación que el usuario a le ha dado al ítem i.

$r_{u,i}$, es la calificación que el usuario u le ha dado al ítem i.

$\overline{r_u}$, es el promedio de las calificaciones del usuario u.

$\overline{r_a}$, es el promedio de las calificaciones del usuario a.

Alternativamente, como medida de similitud a la correlación de Pearson se puede usar la similitud del coseno (ec. 2), la cual mide la similitud que hay entre dos usuarios en función del ángulo que se forma entre ellos. En esta medida no se pueden tener calificaciones negativas; los ítems sin calificar tomarán el valor cero. Un valor cercano a 1 indica similitud mientras uno cercano a 0 indica lo contrario.

$$w_{a,u} = \cos(\overrightarrow{r_a} \cdot \overrightarrow{r_u}) = \frac{\overrightarrow{r_a} \cdot \overrightarrow{r_u}}{\|\overrightarrow{r_a}\|_2 \times \|\overrightarrow{r_u}\|_2} = \frac{\sum_{i=1}^{m} r_{a,i} r_{u,i}}{\sqrt{\sum_{i=1}^{m} r_{a,i}^2} \sqrt{\sum_{i=1}^{m} r_{u,i}^2}} \quad (2)$$

Entre otras medidas de similitud se encuentran: la métrica de las singularidades [18], la diferencia cuadrática media, el coeficiente de correlación de Spearman y la similitud basada en el coseno ajustado [19].

En el segundo paso, la selección de los usuarios que tienen mayor similitud con el usuario activo se puede hacer de dos formas [20]:

- Establecer un umbral de correlación y seleccionar *K* los usuarios que superen el umbral establecido. Esta forma puede no tener buenos resultados si el

umbral que se establece es alto, ya que los usuarios que no tengan vecinos con correlación alta tendrán poca variedad de ítems para recomendarle.

- Seleccionar *k* vecinos con mayor similitud.

Una vez elegidos los *k* vecinos más cercanos, se procede a predecir la calificación que el usuario activo realizaría sobre los ítems que no ha calificado (paso 3). Esta predicción se puede hacer con la siguiente medida:

Suma ponderada de otras calificaciones: Para hacer la predicción de la preferencia del usuario *a* sobre un ítem *i*, se puede tomar el promedio ponderado de los pesos de todas las calificaciones de dicho ítem de los vecinos del usuario activo [21], este promedio ponderado se observa en la ec. 3.

$$P_{a,i} = \overline{r_a} + \frac{\sum_{u \in U}(r_{u,i} - \overline{r_u}) \cdot w_{a,u}}{\sum_{u \in U} |w_{a,u}|} \qquad (3)$$

Donde:

$\overline{r_a}$ y $\overline{r_u}$, es el promedio de las calificaciones Del usuario a y el usuario u de todos los ítems calificados.

La suma se realiza sobre los usuarios $u \in U$ quienes han calificado el ítem i.

La predicción se basa en los vecinos más cercanos del usuario activo; mientras más cercano se encuentre un vecino al usuario *a* mayor será el peso de su calificación.

Finalmente, teniendo los *K* vecinos más cercanos y habiendo predicho los ítems pendientes por calificar, se generan las recomendaciones para el usuario activo.

Ejemplo del enfoque basado en usuario

Dado un usuario activo, se pretende encontrar el conjunto de usuarios que tienen mayor similitud con el usuario activo para poder predecir el valor de los ítems que el usuario activo no ha calificado. En el caso de la tabla 2, serían el ítem 5 y el ítem 6.

Tabla 2. Ejemplo de matriz de calificaciones 6x4

	Ítems					
	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
Usuario activo	5	3	4	4	?	?
Usuario 1	4	3	4	3	5	4
Usuario 2	3	1	2	3	3	3
Usuario 3	1	5	5	2	1	1
Usuario 4	3	3	1	5	4	3

Para encontrar la similitud entre el usuario activo, llamémoslo A, y los demás usuarios, se usa la correlación de Pearson, obteniéndose las similitudes mostradas en la tabla 3.

Tabla 3. Ejemplo de similitud entre el usuario activo y un conjunto de usuarios

	Similitud
Sim(A,1)	0,59
Sim(A,2)	0,82
Sim(A,3)	-0,73
Sim(A,4)	0

Se seleccionan los vecinos más cercanos al usuario activo, usando como criterio aquellos usuarios cuya similitud con el usuario activo fue mayor a cero, en este caso correspondería al usuario 1 y al usuario 2. De esta forma, usando la ec. 3, correspondiente a la suma ponderada de otras calificaciones, se calcula la calificación del usuario activo hacia los ítems 5 y 6, obteniéndose los valores mostrados en las ec. 4 y 5.

$$P_{a,5} = 4 + \frac{(5-3,8)*0,59+(3-2,5)*0,82}{0,59+0,82} = 4,79 \quad (4)$$

$$P_{a,6} = 4 + \frac{(4-3,8)*0,59+(3-2,5)*0,82}{0,59+0,82} = 4,37 \quad (5)$$

A partir de los ítems predichos se procedería a recomendarle como primera opción el ítem 5, ya que obtuvo mayor puntaje, y como segunda opción el ítem 6.

El enfoque basado en ítems

Como se aprecia en la fig. 5, cuando se tiene un conjunto de datos de millones de usuarios el enfoque basado en usuarios pudiera no funcionar bien debido a la complejidad computacional, frente a esta situación se propone calcular la similitud entre los ítems [1]. Este algoritmo es similar al basado en usuarios, sólo que en vez de buscar similitud entre usuarios se busca similitud entre ítems, por lo tanto, se generan recomendaciones de ítems que tienen vecinos más cercanos a los ítems que un usuario activo previamente calificó.

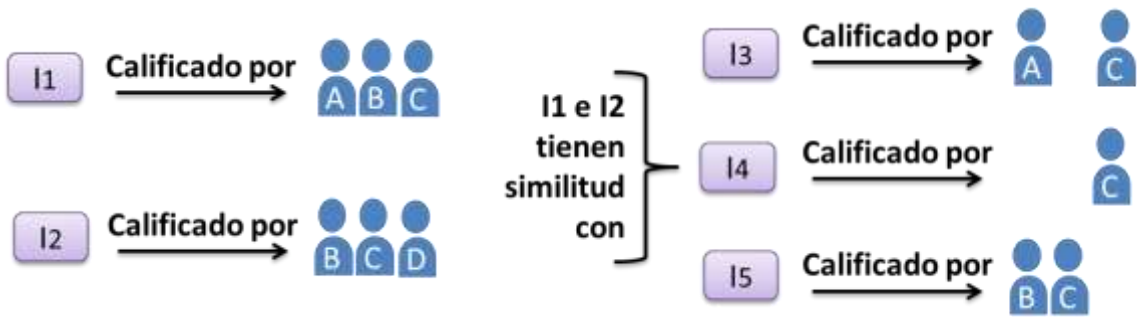

Figura 5. Sistema de Recomendación de filtrado colaborativo basado en ítems

Para hallar los vecinos más cercanos se calcula la similitud entre dos ítems *i* y *j*. Entre las medidas para hallar la similitud entre ítems, también, se encuentra la correlación de Pearson (ec. 6).

$$w_{i,j} = \frac{\sum_{u \in U}(r_{u,i} - \overline{r_i})(r_{u,j} - \overline{r_j})}{\sqrt{\sum_{u \in U}(r_{u,i} - \overline{r_i})^2}\sqrt{\sum_{u \in U}(r_{u,j} - \overline{r_j})^2}} \quad (6)$$

Dónde:

$w_{i,j}$, es el peso entre el usuario i y el usuario j.

U, es el conjunto de todos los usuarios que han calificado los ítems i y j.

$r_{u,i}$, es la calificación que el usuario u le ha dado al ítem i.

$r_{u,j}$, es la calificación que el usuario u le ha dado al ítem j.

$\overline{r_i}$, es el promedio de las calificaciones del ítem i por los usuarios.

$\overline{r_j}$, es el promedio de las calificaciones del ítem j por los usuarios.

La tarea de predicción es similar al enfoque basado en usuarios. Se seleccionan los vecinos más cercanos, en este caso los *k* ítems con mayor similitud a aquel ítem al que se desea predecir la calificación. Se realiza la predicción con la media ponderada de la calificación otorgada por el usuario actual a dichos vecinos, la ponderación se realiza con la similitud.

Para predecir la calificación de un ítem se propone la medida Promedio Simple del Peso [21], expresada en la ec. 7.

Promedio simple del peso:

Cuando se trabaja con la predicción basándose en los ítems se puede predecir la calificación del ítem *i* por el usuario *a,* usando un promedio del peso [17].

$$P_{a,i} = \frac{\sum_{j \in K} r_{a,j} w_{i,j}}{\sum_{j \in K} |w_{i,j}|} \quad (7)$$

Dónde:

K, es el conjunto de vecinos más cercanos al ítem i

$w_{i,j}$, es la similitud entre el ítem i y el ítem j

$r_{a,j}$, es la calificación del usuario a al ítem j

Finalmente, se realizan las recomendaciones de N ítems que, se estima, serían interesantes para un usuario particular. Serían aquellos ítems que el usuario no ha calificado y tienen mayor calificación predicha.

Para finalizar la sección de los algoritmos basados en memoria se debe tener en cuenta las siguientes consideraciones:

Normalización de las calificaciones

Cuando se califica un ítem, cada usuario lo hace de acuerdo a su propia escala. Para transformar las calificaciones a una escala universal hay varios métodos, dos de los más populares son [2]:

- **Normalización centrada en la media (*mean-centering*):** Con esta normalización se puede determinar cuándo una determinada calificación es positiva o negativa, comparándola con el promedio de todas las calificaciones. De igual forma, se aprecia a partir de qué nivel a un usuario le gusta o disgusta un ítem.

  Esta normalización puede ser aplicada tanto al enfoque basado en ítems como al enfoque basado en usuarios. En el enfoque basado en usuarios, toda una fila de las tablas de calificaciones realizadas por un usuario es

transformada a una normalización, $h(r_{u,i})$, respecto a la media al restarle a $r_{u,i}$ el promedio $\overline{r_u}$ de las calificaciones realizadas por el usuario *u*.

$$h(r_{u,i}) = r_{u,i} - \overline{r_u} \quad (8)$$

- **Normalización z-score:** Normaliza los datos de un usuario o un ítem ajustándose a la media y a la varianza.
  Para la normalización utilizando el enfoque basado en usuarios, se divide la calificación normalizada con respecto a la media para el algoritmo basado en usuarios, entre la desviación estándar σ_u de las calificaciones hechas por el usuario *u.* De esta forma la calificación normalizada queda así:

$$h(r_{u,i}) = \frac{r_{u,i} - \overline{r_u}}{\sigma_u} \quad (9)$$

Estas transformaciones, por lo general, se realizan en el paso tres, antes de generar las predicciones.

El enfoque basado en usuarios vs el enfoque basado en ítems

Al elegir entre un sistema de recomendación basado en usuarios o un sistema basado en ítems, se debe considerar [2]:

Precisión: La precisión depende de la razón entre el número de usuarios y número de ítems en el sistema. En los sistemas donde el número de usuarios es mucho más grande que el número de ítems se recomiendan los sistemas basados en ítems, mientras que para los sistemas que tienen miles de usuarios y cientos de miles de ítems pueden ser más beneficiosos los sistemas basados en usuarios.

Eficiencia: La eficiencia de un sistema depende de la razón entre el número de usuarios y el número de ítems. Cuando el número de usuarios excede el número de ítems se propone trabajar con el enfoque basado en ítems ya que se requiere menos tiempo y memoria para encontrar las similitudes.

Estabilidad: El escoger qué enfoque usar depende de la cantidad de cambios que hay en los usuarios o en los ítems. Si la lista de usuarios es más estática comparada con la lista de ítems, se recomendará usar el enfoque basado en usuarios, ya que la similitud entre los usuarios podría ser calculada con poca frecuencia para realizar recomendaciones.

Justificabilidad: El enfoque basado en ítems tiene la ventaja que la recomendación puede ser justificada, esto es debido a que, es a los ítems a los cuales se les hace un puntaje. Los ítems vecinos usados para la predicción pueden ser usados como una explicación de la recomendación. El enfoque basado en usuarios puede ser menos sensible ya que el usuario activo no conoce a los usuarios que sirven de vecinos para generar las recomendaciones.

Serendipia: El enfoque basado en ítems es menos aconsejable si se desean obtener recomendaciones inesperadas y que sean de interés para el usuario. Un sistema de recomendación basado en ítems buscará recomendaciones que estén relacionadas a los ítems que previamente le han gustado al usuario, por lo que no ayudaría al usuario a descubrir diferentes tipos de ítems que posiblemente sean de su interés.

Para la serendipia se recomienda el enfoque basado en usuarios, por ejemplo, un usuario A que ha escuchado solo canciones del género pop puede tener como vecino cercano al usuario B que a su vez ha escuchado canciones del género pop; si B ha escuchado canciones de otro género, estas canciones podrían ser recomendadas a A ofreciéndole diversidad en sus canciones y que seguramente serán de su agrado.

Problemas que se presentan con el uso de algoritmos basados en memoria

Escasez de datos: La escasez de datos puede ser de varios tipos: 1.- El problema de la corrida en frio es uno de los problemas más comunes en los sistemas de recomendación. En estos casos se recomienda pedir a los usuarios puntuar ciertos ítems para poder generar recomendaciones, otra solución, es emplear otro método

de sistema de recomendación donde se hagan recomendaciones basándose en información demográfica, basada en contenido o simplemente generar recomendaciones no personalizadas. 2.- El problema del nuevo producto. Cuando un ítem es nuevo en el sistema resultará difícil realizar recomendaciones de ese nuevo producto.

Escalabilidad: Un sistema de recomendación basado en memoria, al tratar millones de usuarios e ítems sufrirá serios problemas de escalabilidad al tener que realizar recomendaciones en tiempo real.

Sinonimia: Puede haber productos iguales o muy similares con diferentes nombres y el sistema de recomendación los tratará como si fueran diferentes.

Oveja gris: El filtrado colaborativo funciona mejor con usuarios que encajan en un grupo de usuarios con gustos similares. Cuando el perfil de un usuario no encaja en el perfil de otros usuarios se hace difícil determinar una recomendación adecuada, a este tipo de usuarios se les llama ovejas grises. También ocurre para los usuarios nuevos que no tienen vecinos más cercanos, por lo que el usuario no podrá recibir recomendaciones adecuadas.

Ataques y manipulaciones de preferencia: Habrán usuarios que quieran favorecer o perjudicar algunos productos frente a otros, al introducir información inadecuada al sistema. En este caso resulta necesario que el sistema de recomendación tome sus precauciones para evitar estas manipulaciones.

Diversidad: Cuando el número de productos es muy elevado, con respecto al número de usuarios, puede surgir que los usuarios solo hayan calificado los mismos productos y de esta forma no haya diversidad en las recomendaciones.

Ventajas de los algoritmos basados en memoria

Las mayores ventajas de los algoritmos basados en memoria son [2]:

- **Simplicidad:** Estos métodos son intuitivos y simples de implementar.
- **Justificabilidad:** Se provee una justificación para las predicciones propuestas.
- **Eficiencia:** No requiere un entrenamiento previo para generar las recomendaciones y el almacenamiento de los vecinos más cercanos requiere poca memoria.
- **Estabilidad:** Son afectados muy poco con el ingreso de nuevos usuarios, ítems y calificaciones**.** Una vez que se encuentra la similitud entre ítems, el sistema basado en ítems puede hacer recomendaciones a nuevos usuarios sin tener que entrenar nuevamente el sistema.

b) Algoritmos basados en modelos

Son algoritmos que usan un conjunto de calificaciones existentes para crear un modelo que predice calificaciones. Estos modelos son entrenados usando los datos disponibles y no requieren procesar continuamente la matriz de datos de entrenamiento. La ventaja de este enfoque es la rapidez y la escalabilidad. Entre los algoritmos que se pueden usar para construir este tipo de sistemas están: las redes bayesianas, el análisis clúster, las redes neuronales artificiales, los métodos de regresión, entre otros.

Ventajas y desventajas del FC

Explorando los dos métodos propuestos para el filtrado colaborativo, se presenta la tabla 4 donde se comparan las ventajas y desventajas de cada enfoque:

Tabla 4. Comparación de los algoritmos de filtrado colaborativo

Categoría del FC	Ventajas	Desventajas
Algoritmos basados en memoria	• Métodos simples e intuitivos de implementar • Mejores recomendaciones • Se pueden agregar nuevos datos fácilmente • No requiere de un entrenamiento previo para generar las recomendaciones	• Limitado en cuanto a escalabilidad con grandes volúmenes de datos • Corrida en frio. No se pueden realizar recomendaciones para nuevos usuarios o ítems. • Usa todos los datos para realizar la predicción teniendo que usar mucha memoria
Algoritmos basados en modelos	• Buen manejo de la escalabilidad y diversidad de datos • Rendimiento de la predicción, son más rápidos que los basados en memoria	• Pérdida de información con técnicas de reducción de la dimensionalidad de los datos • Modelo más complejo de construir computacionalmente • Inflexible, más difícil de agregar nuevos datos

2.4 Evaluación de la métrica

La calidad de los sistemas de recomendación puede ser evaluada comparando la recomendación con una base de datos conocida de calificaciones de ciertos usuarios. Sin embargo, estas evaluaciones no se consideran triviales ya que el

rendimiento de un algoritmo puede ser mejor o peor dependiendo del conjunto de datos que se esté usando.

Según la naturaleza del problema, el objetivo de los sistemas de recomendación puede variar, mientras que un sistema puede buscar generar recomendaciones no erróneas otro sistema puede buscar generar recomendaciones no triviales para el usuario, siempre buscando satisfacer al usuario.

Entre las métricas estadísticas que se proponen para evaluar la precisión se encuentran:

Métricas de precisión en la predicción: Estas calculan qué tan cerca es la calificación predicha por el sistema con la calificación proporcionada por el usuario [15]:

- **Error Medio Absoluto (MAE):** Mide la desviación de las observaciones predichas con su valor real. A menor MAE mejor es la predicción realizada por el sistema sobre la calificación de un usuario.

$$MAE = \frac{\sum_{\{i,j\}} |p_{i,j} - r_{i,j}|}{n} \qquad (10)$$

- **Raíz cuadrada del error cuadrático medio (RMSE):**

$$RMSE = \sqrt{\frac{1}{n}\sum_{\{i,j\}} (p_{i,j} - r_{i,j})^2} \qquad (11)$$

La principal diferencia con la anterior es que RMSE otorga una mayor importancia a los errores más grandes, lo que es bastante razonable pues estos errores son los que probablemente tengan un mayor impacto en la percepción del usuario.

Métricas de rendimiento: Se centran en la relevancia y en la capacidad del sistema de detectar qué producto es relevante y cual no para un determinado usuario.

Estos métodos son usados dividiendo el conjunto de ítems I en dos subconjuntos; $I_{entrenamiento}$ e I_{prueba}. El $I_{entrenamiento}$ será usado para calcular $L(u)$, el cual contiene N ítems de interés para u. Supongamos que $T(u) \subset I_u \cap I_{prueba}$, es el conjunto de ítems que el usuario u considera relevante. Si solo se tiene la lista de ítems comprados por un usuario u, $T(u)$ será el conjunto de estos ítems.

Estas medidas son definidas de la siguiente manera [22]:

- **Precisión (P):** es la probabilidad de que un elemento seleccionado sea relevante. Se puede ver como la proporción de recomendaciones que son buenas recomendaciones.

$$P(L) = \frac{1}{|U|}\sum_{u\in U}|L(u) \cap T(u)|/|L(u)| \qquad (12)$$

- ***Recall*** **(R):** es la probabilidad de que sea seleccionado un elemento relevante, aunque en los sistemas de recomendación la "relevancia" es algo totalmente subjetivo. Se puede ver como la proporción de buenas recomendaciones que aparecen en el top de recomendaciones.

$$R(L) = \frac{1}{|U|}\sum_{u\in U}|L(u) \cap T(u)|/|T(u)| \qquad (13)$$

CAPÍTULO III

Descripción de la metodología propuesta

En este capítulo se describe la metodología propuesta, que está basada en la técnica del vecino más cercano, para generar recomendaciones a un usuario que visita un comercio electrónico.

3.1 Sistema de recomendación propuesto

El sistema busca generar recomendaciones que atraiga la atención del usuario activo. Las recomendaciones están dirigidas a usuarios exploradores que no han definido su compra. Para construir el sistema que se describe hay que disponer de una base de datos suficientemente grande del comercio electrónico con el que se vaya a trabajar, de la cual se usan los registros de aquellos usuarios que no han realizado compras.

En el problema planteado no hay una retroalimentación explícita que ayude a generar las recomendaciones. Así, se plantea un esquema en el que se convierte el comportamiento del usuario en un valor numérico, es decir, se hace uso de información implícita.

Para generar las recomendaciones se toman en cuenta los tiempos que han permanecido usuarios no compradores en los ítems visitados, pues son buenos indicadores del grado de interés que tiene el usuario hacia un ítem.

Se supone que los usuarios no compradores están menos enfocados en un ítem particular y pueden visitar diversos ítems de su interés, mientras que un usuario comprador tiene definida la compra y los ítems que visita pudieran ser poco diversos.

Definiciones

- **Usuario activo:** Usuario que se encuentra actualmente en el comercio electrónico a quien se le dan recomendaciones.
- **Sesión:** Visita de un usuario al comercio electrónico. En una sesión puede realizarse uno o más clics. La sesión hace referencia a los registros de una visita del usuario, es decir, no toma en cuenta todo el historial de ítems visitados por un usuario en otras sesiones.

3.2 Generar recomendaciones con información implícita

Para generar las recomendaciones basadas en el filtrado colaborativo de usuarios no compradores, a partir de información implícita, se cumplen las tres etapas siguientes:

Pre procesamiento → Búsqueda de los vecinos más cercanos → Predicción

I. Preprocesado de las matrices - Transformación de los datos

Como ya se ha mencionado, el comportamiento del usuario está plasmado en el tiempo que estuvo un usuario visitando cada ítem, esto indica el grado de interés que tiene un usuario hacia un ítem. Considerando a *M* como el máximo número de diferentes ítems visitados por un usuario en la base de datos, la metodología se inicia de la siguiente manera:

Se crea una matriz de 3+2M columnas, con los ítems de cada usuario presente en la base de datos**.** Las tres primeras columnas corresponden al identificador del usuario, al número de n ítems almacenados del usuario y al tiempo total que el usuario estuvo en los ítems visitados. Las columnas

siguientes corresponden a cada uno de los n ítems visitados por el usuario, junto a los n valores numéricos que representan el grado de interés del usuario hacia el ítem.

En este paso se toman en cuenta las siguientes consideraciones:

- Se seleccionan aquellos usuarios que hayan visitado como mínimo 3 ítems, con esto se asegura que hayan ítems de un usuario que sirvan de recomendación.
- El mejor tiempo que se obtuvo de un usuario no se toma en cuenta para la búsqueda de similitud con un usuario activo, ya que dicho ítem será el candidato a ser una recomendación.

El valor numérico de un usuario u hacia un ítem i, $r_{u,i}$, se genera usando el supuesto de que mientras más tiempo haya estado un usuario en un ítem, mayor es su interés por ese ítem. Para cada ítem cliqueado por un usuario, sin tomar en cuenta el mejor, se le asigna un valor numérico que indica la proporción del tiempo que ha estado un usuario en ese ítem.

El tiempo que estuvo un usuario en un ítem se puede obtener según la naturaleza de la base de datos. Una de las formas es considerando la diferencia del tiempo en que un usuario realizó clic en un ítem i_l con respecto al tiempo en que realizó clic en su ítem sucesor, con esto se deduce el tiempo que el usuario pasó en un ítem i_l. De esta forma, se infiere en el presente trabajo, el tiempo que un usuario u pasó en un ítem i.

$$r_{u,i} = \frac{t_{u,i}}{\sum_{i=1}^{n} t_{u,i}} \qquad (14)$$

$r_{u,i}$, es el tiempo relativo que el usuario u estuvo en el ítem i, razón del tiempo que estuvo en el ítem i entre el tiempo total.

$t_{u,i}$, es el tiempo que el usuario u estuvo en el ítem i.

$\sum_{i=1}^{n-1} t_{u,i}$, es el tiempo total que el usuario u estuvo en los $n-1$ ítems cliqueados.

n, es el número de ítems vistos por el usuario.

$r_{u,i}$ mide el grado de interés del usuario u hacia el ítem i . Su valor está entre 0 y 1.

El algoritmo general para crear la matriz mencionada, funciona de la siguiente manera:

Para cada usuario:

1. Leer el registro de un usuario.
2. Descartar el usuario si ha visitado menos de 3 ítems.
3. Seleccionar el ítem con mejor tiempo y almacenarlo junto al tiempo que estuvo el usuario en ese ítem. Es importante resaltar que este ítem será la recomendación que aportaría ese usuario, por lo tanto, este ítem se ignora para el cálculo de los tiempos relativos y el cálculo de la distancia entre un usuario activo y los usuarios del historial.
4. Se almacena el identificador del ítem y los tiempos del resto de ítems. Los n ítems no se encuentran ordenados en ningún sentido.
5. A partir del segundo ítem al n se calcula $r_{u,i}$.

Una vez que se construye la matriz de los ítems por usuario, se crean los archivos de usuarios por ítems en donde se almacenan los usuarios que cliquearon un ítem determinado junto a los n ítems del usuario. Al crear un archivo por cada ítem que hay en el comercio electrónico se ahorra significativamente el tiempo de procesamiento, ya que no se tendrá que recorrer una matriz de todos los ítems junto a todos los usuarios. Esto se observará más adelante

La estructura de los archivos es tal y como se muestra en la fig. 6. Cada registro del archivo contiene la información de un usuario que cliqueó un ítem i_k. Esta información es; el identificador del usuario que cliqueó el ítem i_k, el número de ítems visitados por el usuario, tiempo total que el usuario estuvo en los ítems, seguido de los ítems cliqueados con el tiempo que pasó en dichos ítems.

ID usuario, U_j	Nº de ítems visitados por U_j	Tiempo total que U_j estuvo en los ítems	ID del ítem en que U_j pasó más tiempo	Tiempo que U_j pasó en I_1		ID del ítem en que U_j pasó menos tiempo	Valor numérico de un usuario U_j hacia el ítem I_m
U_j	n	$\sum_{i=1}^{n} t_{u,i}$	I_1	t_{u_j,i_1}	...	I_m	r_{u_j,i_m}

Figura 6. Estructura de los archivos de usuarios por ítems

II. Algoritmo de búsqueda de los k vecinos más cercanos

El algoritmo presentado se basa en la idea de podar caminos, por lo que se utiliza el cuadrado de la distancia euclídea la cual permite conseguir condiciones de podas bastante eficientes. Es decir, se usa la distancia euclídea como base de la medida de similitud para la búsqueda de los k vecinos más cercanos.

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + \cdots (x_n - y_n)^2} \qquad (15)$$

En esta etapa es fundamental el uso de los archivos de usuarios por ítems.

La filosofía del método se basa en realizar el análisis de acuerdo al historial. Es de esperar, que si el usuario activo es similar a un usuario u_j, entonces, el ítem que recomienda u_j será de interés para el usuario activo.

La búsqueda de los k vecinos se realiza tomando en cuenta aquellos usuarios que visitaron los ítems cliqueados por el usuario activo, iniciando con el ítem en el que pasó más tiempo el usuario activo.

Para explicar el algoritmo se hacen las siguientes definiciones:

- $d_{a,i}$ distancia euclídea entre el usuario a y el usuario i.
- n número de ítems cliqueados por el usuario a.
- $i_{a,1}$ ítem cliqueado por el usuario a con mayor valor numérico, $i_{a,2}$ ítem cliqueado por el usuario a con el segundo mayor valor numérico … $i_{a,n}$ es el ítem cliqueado por el usuario a con el menor valor numérico.

- $r_{a,j}$ es el valor numérico (tiempo relativo) del usuario a hacia el ítem j.
- D_{max} distancia al cuadrado entre a y el peor *k buen vecino.*
- I_{ua} conjunto de ítems visitados por el usuario activo.
- u_a es el usuario activo.
- u_j es un usuario de la base de datos candidatos a ser vecino cercano del usuario activo.

Se diferencian los *k buenos vecinos* de los *k vecinos más cercanos,* en el sentido que los *k vecinos más cercanos* son aquellos que, dentro de toda la base de datos, tienen la menor distancia euclídea con u_a mientras que *k buenos vecinos* son un conjunto de k usuarios que, en un instante de la búsqueda, tienen la menor distancia euclídea con u_a y son candidatos a *vecinos más cercanos*.

De los anteriores enunciados, la distancia euclídea entre un usuario activo u_a y un usuario u_j, queda expresada de la siguiente forma:

$$d(u_a,u_j)^2 = \underbrace{(r_{ua,1}-r_{uj,1})^2+(r_{ua,2}-r_{uj,2})^2+\cdots+(r_{ua,n}-r_{uj,n})^2}_{\text{Coincidentes}}$$

$$\underbrace{+(\hat{r}_{ua,1})^2+\cdots+(\hat{r}_{ua,m})^2}_{\text{No coincidentes}} \qquad (16)$$

$$+(\hat{r}_{uj,1})^2+\cdots+(\hat{r}_{uj,r})^2$$

Los términos coincidentes corresponden a aquellos que hacen referencia a los ítems cliqueados en conjunto por u_a y u_j y los no coincidentes son aquellos ítems que no han sido cliqueados en conjunto. En la ecuación 16, los términos que están en color verde son aquellos que hacen referencia a m ítems del usuario activo que no son coincidentes con u_j y los términos que están en color rojo son aquellos que hacen referencia a r ítems de u_j que no son coincidentes con el usuario activo.

¿Cómo escoger un buen k?

El k, el cual es seleccionado a través de experimentación por el diseñador del sistema de recomendación, dependerá de la base de datos que se tenga; las pruebas se realizan para determinar un k con el cual se obtienen los mejores tiempos computacionales. También, es importante analizar el error de predicción de los distintos k, puesto que se busca un k para que este error sea bajo. Otra consideración es que los tiempos relativos estimados de los ítems a recomendar sean altos.

El algoritmo de búsqueda de los k vecinos comprende los siguientes pasos:

1. Ordenar los ítems del usuario activo de mayor a menor valor numérico. No se toma en cuenta el ítem que acaba de cliquear.

$I_{ua} \rightarrow i_{a,1}, i_{a,2}, i_{a,3}, .. , i_{a,n}$ $i_{a,1}$ es el ítem con mayor grado de interés de u_a

$i_{a,n}$ es el ítem con menor grado de interés de u_a

2. Se busca el archivo correspondiente a $i_{a,1}$ (el ítem cliqueado por u_a con mejor valor numérico).
3. Se seleccionan los usuarios que cliquearon el ítem $i_{a,1}$ y se almacenan en una tabla, cuya primera columna contiene el valor que un usuario le asignó al ítem $i_{a,1}$, y la segunda columna contiene una lista de la información almacenada del usuario que también cliqueó $i_{a,1}$, esta información será el identificador del usuario respectivo junto a los ítems cliqueados por el usuario y el tiempo relativo correspondiente.
4. Se ordena la tabla en orden decreciente, en función de los valores numéricos que cada usuario le ha dado a $i_{a,1}$. Esto es, la tabla ordenada tendrá en la primera fila al usuario que mejor valor tiene sobre el ítem $i_{a,1}$, la fila k tendrá al usuario que tiene el k-esimo valor sobre el ítem $i_{a,1}$.
5. Se calcula la distancia euclídea entre el usuario activo y los *k* primeros usuarios, en cuyas recomendaciones no estén los ítems cliqueados por el

usuario activo. Si el mejor ítem cliqueado por un usuario se encuentra entre los ítems cliqueados por el usuario activo, es decir, se encuentra en el conjunto I_{ua}, se descarta el usuario como posible *k* vecino más cercano, pues él no aportará recomendaciones novedosas al usuario activo. Estos serán en principio k *buenos vecinos.*

6. La mayor distancia, al cuadrado, entre los k buenos vecinos actuales, con u_a, será almacenada en D_{max}.
7. Para los usuarios que cliquearon $i_{a,1}$, antes de calcularle la distancia con u_a, se verifica que:

$$(r_{ua,1} - r_{uj,1})^2 > D_{max} \qquad \text{(a.1)}$$

$$r_{ua,1} > r_{uj,1} \qquad \text{(a.2)}$$

Estas expresiones se justifican más adelante, la primera desigualdad (a.1) indica que ya no habrá usuarios que cliquearon el ítem $i_{a,1}$ con distancias menores a D_{max} por lo que los usuarios que se comparan a partir de ese momento no podrán mejorar la distancia de alguno de los k vecinos más cercanos. La segunda condición (a.2) va de la mano con la condición (a.1), esta condición es añadida debido a que los usuarios son ordenados de mayor a menor tiempo relativo y se considera la condición de salida a partir de que el tiempo relativo del usuario activo es mayor que el usuario considerado

Si la desigualdad (a.1) es falsa, calcular $d_{a,j}$.

Si ${d_{a,j}}^2 < D_{max}$ se elimina el último k buen vecino de u_a y se ubica al nuevo usuario u_j con su información en la posición que le corresponda.

En este punto hay que tomar en cuenta que aquellos usuarios en donde su mejor ítem cliqueado se encuentre entre los ítems cliqueados por el usuario activo serán descartados, se descartan ya que la recomendación que estos podrían aportar no será de interés para el usuario activo ya que el usuario activo ya habrá visto este ítem cliqueado.

Se calcula el nuevo D_{max}.

Si las desigualdades (a.1) y (a.2) son verdaderas para algún usuario (lo que indicaría que las desigualdades serían verdaderas para el resto de los usuarios), o ya se recorrieron todos los usuarios que cliquearon $i_{a,1}$ se continúa el proceso con los usuarios del siguiente ítem cliqueado por u_a, en este caso el siguiente ítem será $i_{a,2}$.

Al pasar a los usuarios que cliquearon el siguiente ítem, ya no habrá usuarios por comparar que hayan cliqueado $i_{a,1}$.

8. Para el resto de los ítems cliqueados por u_a, es decir, desde $i = 2\ hasta\ i = n$, o hasta que se consigan los k vecinos más cercanos.
    a. Buscar en los archivos de usuarios por ítems, los usuarios que cliquearon $i_{a,i}$.
    b. Antes de empezar a recorrer los usuarios que cliquearon $i_{a,i}$ verificar la siguiente desigualdad:

$$\sum_{l=1}^{i-1}(r_{a,l})^2 > D_{max} \quad \text{(b)}$$

$$donde\ l, será\ la\ referencia\ de\ los\ items\ cliqueados$$

$$por\ u_a\ que\ ya\ fueron\ comparados.$$

    Si se cumple, parar, ya se tienen los k vecinos más cercanos. Esto es, si la sumatoria de los valores numéricos de los ítems que ya se han comparado es mayor a D_{max}, significa que las distancias que se calcularán con los siguientes $i_{a,i}$ van a ser mayores a ese valor por lo que ya se habrán conseguido los mejores k vecinos. Si la desigualdad (b) no se cumple, hay que seguir con el paso c.

    c. Para los usuarios que cliquearon $i_{a,i}$:

$$(r_{ua,i} - r_{uj,i})^2 > D_{max} \qquad \text{(c.1)}$$

$$r_{ua,i} > r_{uj,i} \qquad \text{(c.2)}$$

Si la desigualdad (c.1) y (c.2) son verdaderas ya no se comparan más usuarios que hayan cliqueado $i_{a,i}$. Se aumenta i una unidad e ir al paso a.

Si la desigualdad (c) es falsa se calcula $d_{a,j}$ y se analiza la siguiente desigualdad:

$$d_{a,j}{}^2 < D_{max} \qquad \text{(d)}$$

Si la desigualdad (d) se cumple, se elimina el último k buen vecino de u_a y se ubica el nuevo usuario en la posición que le corresponda.

Se calcula el nuevo D_{max}.

Volver al paso (c) con el siguiente usuario que cliqueó $i_{a,i}$.

Se finaliza cuando se hayan recorrido todos los $i_{a,i}$ o se hayan conseguido los k vecinos más cercanos.

Justificación de la condición de poda (a) y (c)

En esta sección se prueba por qué las condiciones (a) y (c) permiten reducir el espacio de búsqueda de los k vecinos.

Se considera que el usuario activo ha visto k ítems, sin contar el ítem en el que está en un cierto instante, algunos de estos ítems coindicen con los del usuario u_j, siendo u_j un usuario que ha cliqueado por lo menos uno de los ítems cliqueados por el usuario activo.

Sean:

- $i_{a,1}, i_{a,2}, \ldots, i_{a,n}$ los ítems del usuario activo que coinciden con los del usuario u_j, ordenados de forma descendiente según el tiempo relativo.
- $r_{ua,i}$ y $r_{uj,i}$ son los tiempos relativos del usuario activo y del usuario u_j hacia el ítem $i_{a,i}$.
- $\hat{\imath}_{a,1}, \hat{\imath}_{a,2}, \ldots, \hat{\imath}_{a,m}$ los ítems del usuario activo que no coinciden con el usuario u_j .
- $\hat{r}_{ua,i}$ el tiempo relativo del usuario activo hacia el ítem $\hat{\imath}_{a,i}$.
- $\hat{\imath}_{uj,1}, \hat{\imath}_{uj,2}, \ldots, \hat{\imath}_{uj,r}$ los ítems del usuario u_j que no coinciden con el usuario activo.
- $\hat{r}_{uj,i}$ el tiempo relativo del usuario u_j hacia el ítem $\hat{\imath}_{uj,i}$.

Considerando los anteriores enunciados se descompone la distancia euclídea tal y como se presentó en la ecuación 16:

$$d(u_a, u_j)^2 = \underbrace{(r_{ua,1} - r_{uj,1})^2 + (r_{ua,2} - r_{uj,2})^2 + \cdots + (r_{ua,n} - r_{uj,n})^2}_{\text{Coincidentes}}$$

$$\underbrace{+(\hat{r}_{ua,1})^2 + \cdots + (\hat{r}_{ua,m})^2}_{\text{No coincidentes}} \qquad (16)$$

$$+(\hat{r}_{uj,1})^2 + \cdots + (\hat{r}_{uj,r})^2$$

Como:

$$(r_{ua,1} - r_{uj,1})^2 + (r_{ua,2} - r_{uj,2})^2 + \cdots + (r_{ua,n} - r_{uj,n})^2$$

$$+(\hat{r}_{ua,1})^2 + \cdots + (\hat{r}_{ua,m})^2$$

$$+(\hat{r}_{uj,1})^2 + \cdots + (\hat{r}_{uj,r})^2 \geq (r_{ua,i} - r_{uj,i})^2$$

Se tiene que si:

$$\left(r_{ua,n} - r_{uj,n}\right)^2 \geq D_{max} \qquad \text{y}$$

$$r_{ua,n} > r_{uj,n}$$

Entonces, por transitividad:

$$d\left(u_a, u_j\right)^2 \geq D_{max}$$

Justificación de la condición de poda (b)

En esta sección se demuestra por qué la condición (b), también, permite podar el espacio de búsqueda de los k vecinos más cercanos.

$$\sum_{l=1}^{i-1}(r_{a,l})^2 > D_{max} \quad \text{(b)}$$

$$donde\ l, será\ la\ referencia\ de\ los\ items\ cliqueados$$

$$por\ u_a\ que\ ya\ fueron\ comparados.$$

Cuando se visita el ítem i, de los ítems cliqueados por u_a, ya se habrán considerado todos los usuarios de la base de datos que cliquearon los ítems que corresponden a $\hat{r}_{ua,1}, \ldots, \hat{r}_{ua,\mathrm{i}-1}$. Estos ítems aparecen en la distancia euclídea como no coincidentes, tal y como se muestra en la ec. 17:

$$d\left(u_a, u_j\right)^2 = \left(\hat{r}_{ua,1}\right)^2 + \cdots + \left(\hat{r}_{ua,i-1}\right)^2 + \sum_{Resto\ de\ los\ términos}(\ldots)^2 \qquad (17)$$

La condición de poda (b) se justifica de la siguiente manera:

Si:

$$\left(\hat{r}_{ua,1}\right)^2 + \cdots + \left(\hat{r}_{ua,i-1}\right)^2 > D_{max}$$

Entonces, por transitividad:

$$d(u_a, u_j)^2 \geq D_{max}$$

Con estas dos condiciones de reducción del espacio de búsqueda, queda comprobado que al cumplirse las desigualdades ya se habrán conseguido los k vecinos más cercanos para el usuario activo.

En la fig. 7 se presenta un diagrama de las condiciones de poda o reducción del espacio de búsqueda para seleccionar los k vecinos de un usuario activo.

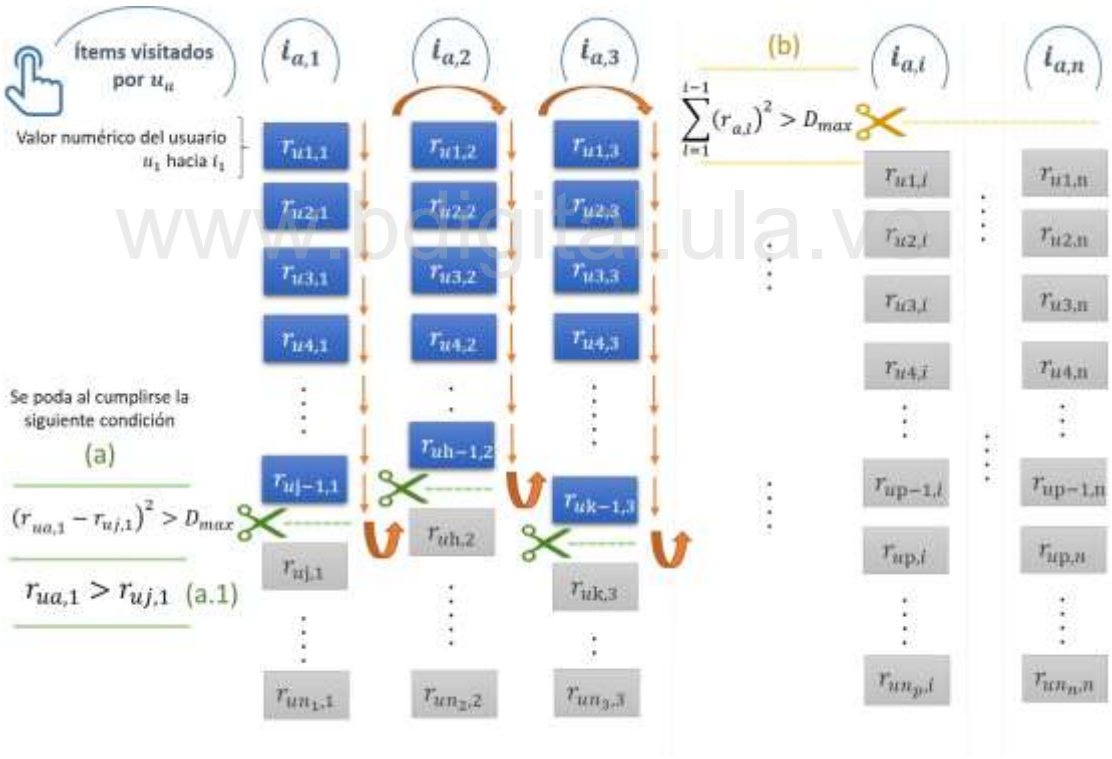

Figura 7. Diagrama de las condiciones de poda

De forma general, el método se inicia identificando los usuarios que cliquearon el mejor ítem para un usuario activo, $i_{a,1}$. El mejor ítem es aquel en el que el usuario activo pasó mayor tiempo. Los usuarios que cliquearon $i_{a,1}$ se ordenan en forma

descendente, según el tiempo relativo de cada usuario hacia el ítem $i_{a,1}$ y se seleccionan los primeros k usuarios como los k vecinos más cercanos; la mayor distancia entre estos k vecinos cercanos se almacenará en D_{max}. Se terminan de recorrer los usuarios que cliquearon $i_{a,1}$ revisando previamente las condiciones (a.1) y (a.2), si éstas se cumplen se dejan de recorrer los usuarios que cliquearon $i_{a,1}$, y se pasa a recorrer los usuarios que cliquearon $i_{a,2}$, el segundo mejor ítem cliqueado por el usuario activo. Los usuarios que cliquearon $i_{a,2}$ se ordenan de forma descendente según el tiempo relativo de cada usuario hacia el ítem $i_{a,2}$, antes de recorrer los usuarios se verifica la condición (b), si ésta se cumple ya se habrán obtenido todos los k vecinos para el usuario activo, si no se cumple se recorren los usuarios que cliquearon $i_{a,2}$, verificando previamente las condiciones (a.1) y (a.2), si las condiciones se cumplen ya no habrán usuarios que visitaron $i_{a,2}$ por recorrer y se pasa a recorrer los usuarios que cliquearon $i_{a,3}$. Así sucesivamente, continúa el algoritmo hasta recorrer todos los usuarios que visitaron los ítems cliqueados por el usuario activo o conseguir los k vecinos más cercanos. Este método permite considerar toda la base de datos, sin recorrerla toda, reduciendo el tiempo computacional de búsqueda de los k vecinos más cercanos.

¿Qué usuarios se considerarán para vecinos?

Se consideran vecinos aquellos cuyo mejor ítem no ha sido visitado por el usuario activo. Se calcula la distancia euclídea entre el usuario activo y los usuarios en la base de datos, sin considerar el mejor ítem del usuario activo. Es de esperar que, si el usuario activo es similar a un usuario u_j, entonces el ítem a recomendar por el usuario u_j le atraerá con una intensidad similar, al usuario activo, cuando éste lo visite.

Cada vecino aporta una recomendación, pudiendo ocurrir que dos vecinos coincidan en su recomendación, por lo tanto a lo más hay *k* ítems recomendados. También, puede llegar a ocurrir que se genere una sola recomendación.

III. Predicción para los ítems recomendados por los k mejores vecinos

Una vez que se tienen los ítems recomendados por cada vecino de los *k* más cercanos, se predice el valor numérico que el usuario activo le daría a cada ítem recomendado. Este valor es el estimado del tiempo relativo que el usuario activo permanece en el ítem recomendado.

Para predecir el tiempo relativo del usuario hacia un ítem recomendado se proponen pesos $w_{a,u}^{(i)}$, de tal forma que cuanto más cercano esté el vecino más peso tendrá el ítem recomendado dado por ese vecino. Los pesos $w_{a,u}^{(i)}$ se calculan para cada uno de los ítems i, que son los ítems recomendados por los *k* vecinos.

Se tiene que:

- $k^{(i)}$ es el número de vecinos que han visitado i
- $P_{a,i}$ es la predicción del tiempo relativo que pasaría el usuario a en el ítem i.

La predicción se realiza de la siguiente manera:

- Si $k^{(i)} = 1$, se define $P_{a,i} = r_{u,i}$ donde u es el vecino que visitó i.
- Si $k^{(i)} > 1$, se define:

$$P_{a,i} = \sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} \tilde{r}_{u,i} w_{a,u}^{(i)} \qquad (18)$$

Donde:

- $w_{a,u}^{(i)} = \frac{1}{k^{(i)}-1}\left(\frac{T^{(i)}-d_{u,a}}{T^{(i)}}\right)$
- $T^{(i)} = \sum_{\substack{u\ \text{vecinos} \\ que\ visitaron\ i}} d_{u,a}$
- $d_{u,a}$ distancia entre el usuario activo y el usuario u, sin tomar en cuenta el mejor ítem de u.

Los pesos $w_{a,u}^{(i)}$, son tales que cuanto más cercano esté el vecino, más peso tendrá el ítem recomendado dado por ese vecino.

De lo anterior es claro que $\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = 1$, tal y como se demuestra a continuación:

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = \sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} \left[\frac{1}{k^{(i)}-1}\left(\frac{T^{(i)}-d_{u,a}}{T^{(i)}}\right)\right]$$

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = \frac{1}{k^{(i)}-1} \sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} \left(\frac{T^{(i)}-d_{u,a}}{T^{(i)}}\right)$$

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = \frac{1}{k^{(i)}-1} \sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} \left(1-\frac{d_{u,a}}{T^{(i)}}\right)$$

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = \frac{1}{k^{(i)}-1}\left(k^{(i)} - \sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} \frac{d_{u,a}}{T^{(i)}}\right)$$

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = \frac{1}{k^{(i)}-1}\left(k^{(i)} - \frac{\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} d_{u,a}}{T^{(i)}}\right)$$

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = \frac{1}{k^{(i)}-1}\left(k^{(i)}-1\right)$$

$$\sum_{\substack{u\ vecinos \\ que\ visitaron \\ i}} w_{a,u}^{(i)} = 1$$

Consideraciones del predictor:

1) El ítem i es el mejor para, por lo menos, un buen vecino, pero se consideran en la suma, vecinos que hayan visitado i así no sea el mejor para esos otros vecinos.
2) El defecto de este predictor está en que para cada ítem se calculan los pesos.

CAPÍTULO IV

Validación experimental

En este capítulo se muestra la forma en que se miden los resultados alcanzados. La metodología se valida en la práctica mediante el uso de la métrica estadística error medio absoluto. Esta métrica se aplica sobre una muestra de una base de datos de usuarios que visitan un comercio electrónico; usuarios que para la validación son llamados usuarios activos.

Se calcula el error medio absoluto para cada usuario, de la muestra, considerando diferentes k pre-establecidos. Una vez calculados estos errores, se obtiene la media de los errores absolutos para determinar qué k es el que permite alcanzar los mejores resultados para los datos utilizados.

4.1 Descripción de la base de datos

Para realizar la validación de la metodología se trabajó con la base de datos de un comercio electrónico de la empresa alemana YOOCHOOSE, especializada en la recomendación de contenidos a partir de los comportamientos de los usuarios. Es de destacar que estos datos fueron utilizados en el RecSys Challenge 2015 [4]. Concretamente, en el trabajo que se describe, se utilizaron los registros de los usuarios no compradores que aparecen en esta base de datos.

La base de datos es real y está conformada por 20.922.069 de registros de usuarios no compradores, de los cuales, 1.994.313 registros corresponden a usuarios diferentes y a 40.619 ítems diferentes. Los datos contenidos en los registros de esta base de datos permiten calcular el tiempo que estuvo cada usuario en cada ítem.

4.2 Simulación y medida de evaluación

Para evaluar la calidad de la metodología, la cual se refleja en la calidad de las predicciones realizadas por los usuarios, se usa la métrica estadística **Error Medio Absoluto (MAE),** en donde se compara el tiempo relativo predicho, que un usuario activo pasaría en un ítem, con el tiempo relativo que el usuario pasó en el ítem:

$$MAE = \frac{\sum_{\{i,j\}} |p_{i,j} - r_{i,j}|}{n} \qquad (19)$$

Sobre cada cliente o usuario activo se realizan las siguientes actividades:

1. Se localiza el ítem en que el usuario activo pasó el mayor tiempo. Este se considera el mejor ítem del usuario activo, en la fig. 8 se ve representado como i_{p+1}.

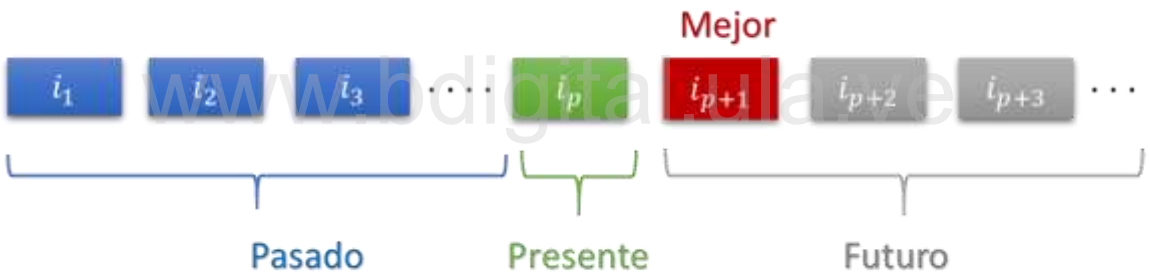

Figura 8. Simulación del usuario activo

En la fig. 8 se observan los clics realizados por un usuario activo. Esto corresponde a la simulación que se realiza en el momento de generarle las recomendaciones al usuario. En el proceso de simulación se define a i_p como el último ítem que acaba de cliquear el usuario, es decir, el ítem en que el usuario actual se encuentra en el presente. El ítem i_p se encontrará justo antes del ítem cliqueado por el usuario activo en el cual pasó mayor tiempo, i_{p+1}.

Por cuestiones prácticas, si el mejor ítem, i_{p+1}, fue cliqueado antes de que tuviera el mejor tiempo no se tomará este usuario activo en el proceso de simulación. Cabe

resaltar que el mejor ítem cliqueado por un usuario pudo haber sido cliqueado anteriormente pero con un tiempo menor, por lo que no se tomaría dicho clic como el mejor.

2. Se aplica el método de los k vecinos para generar R recomendaciones al usuario activo. En la fase de validación R será igual a *k*.

Para la búsqueda de los k vecinos se toman en cuenta aquellos ítems que se encuentran antes del mejor. Una vez se tienen los k vecinos se genera una lista de R ítems a recomendar los cuales estarán ordenados de mayor a menor preferencia.

En este punto, si el mejor ítem del usuario activo no se encuentra entre los ítems a recomendar se descarta a este usuario activo, ya que no se podrá calcular el MAE. Si el usuario activo no fue descartado se realiza lo siguiente:

a. Se calcula el estimador del tiempo relativo que pasaría el usuario en el mejor ítem, $p_{i,j}$.
b. Se calcula el tiempo relativo real que pasó el usuario en el mejor ítem, $r_{i,j}$.
c. Se calcula el error medio absoluto.

Una vez calculada una muestra suficientemente grande para diferentes k, se procede a calcular la media de los errores absolutos para cada uno de los k:

$$MAE_k = \frac{\sum_{\{i,j\}} |{p_{i,j}}^k - {r_{i,j}}^k|}{m} \qquad (20)$$

El tamaño de la muestra se establece en la fase experimental en donde se propone usar un tamaño de muestra para el cálculo de medias, tal y como se presenta en la ecuación 21.

$$n = \frac{z^2 * \sigma^2}{e^2} \qquad (21)$$

4.3 ¿Por qué realizar la simulación tomando en cuenta el historial del usuario activo antes del mejor?

En el método propuesto para realizar la simulación, se plantea que el usuario activo se encuentra en un ítem i_p, el cuál es el predecesor del mejor ítem cliqueado, i_{p+1} (ver fig. 8). Con esto, el historial que se toma del usuario activo sería de aquellos ítems cliqueados antes de cliquear el ítem en el que el usuario activo pasó el mayor tiempo, el cual se ha denominado “El mejor”.

Se decidió realizar la validación de esta forma ya que en la base de datos, que se usa para la validación, no fue implementado el método propuesto de los k vecinos más cercanos, como método de recomendación, por lo que no es de esperarse que los ítems recomendados sean los cliqueados por el usuario. Usando la simulación propuesta es más probable conseguir el mejor ítem de un usuario activo en la lista de ítems recomendados.

CAPÍTULO V

Resultados experimentales

En esta sección se presentan los resultados obtenidos al aplicar la metodología propuesta, como resultado de esta investigación, haciendo uso de la base de datos descrita en el capítulo IV, correspondiente a un comercio electrónico del cual se cuenta con 20.922.069 de registros de usuarios no compradores. Se analiza la técnica para diferentes valores de k con el fin de determinar aquel que presente un equilibrio en los resultados, buscando buenos resultados tanto en el sentido computacional como en la calidad de las recomendaciones en el sentido de mejor tiempo relativo de predicción y precisión.

5.1 Resultados obtenidos del error medio absoluto para diferentes k

Es de resaltar que la metodología propuesta fue implementada en el software estadístico R, por medio del cual se programó el algoritmo para la generación de las recomendaciones. Se realizaron corridas consecutivas del programa para evaluar diferentes valores de k; para cada k se hacen distintas corridas, simulando en cada corrida un solo usuario activo. En cada corrida se calcula el error absoluto. Esto permite calcular el error medio absoluto, tomando en cuenta todas las corridas para un mismo k.

Del tamaño de la muestra planteado en la ecuación 21 se determina cuál es el tamaño de muestra que se debe usar para el cálculo del error medio absoluto con los diferentes k. Inicialmente se tomó una muestra de 100 usuarios activos con el que se determinó la desviación estándar y así, se calculó el tamaño de la muestra

apropiado. Con un error del 2% y una desviación estándar del 95% se consigue que, para los diferentes k, el mayor tamaño de muestra que se debe tomar en cuenta es de 287 usuarios activos.

De lo anterior se decide trabajar con una muestra de 300 usuarios activos para cada k planteado.

Se utilizaron valores para k de 5, 10, 15, 25, 40, 50, 100, 200, 300 vecinos más cercanos, y se obtuvo los resultados que se muestran en la fig. 9.

Figura 9. Media del error medio absoluto con los diferentes k

En la fig. 9 se puede observar que los errores parecieran aumentar a partir de 40 vecinos más cercanos. El error óptimo pareciera alcanzarse alrededor de k=40.

Una interpretación posible de la fig. 9 se basa en que es de esperar que exista un valor óptimo de k en las pruebas realizadas: Si k es pequeño, hay pocos valores a promediar para estimar el tiempo que pasaría el usuario activo en los ítems a recomendar y por tanto su variancia es mayor. Si k es muy grande entonces se

estarían considerando vecinos que no son muy similares al usuario activo, es decir, de comportamiento distinto, así que ellos aportarían variabilidad. Esta interpretación puede verse como se presenta en la fig. 10.

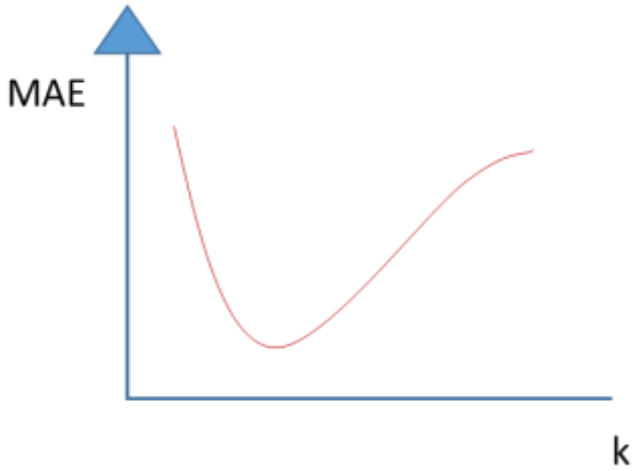

Figura 10. Interpretación de la media del error medio absoluto con los diferentes k

Desde el punto de vista de la precisión de las recomendaciones, k=40 parece ser el más confiable entre los k planteados en la simulación, pues k=40 resultó tener el menor error medio absoluto en las simulaciones.

5.2 Resultados obtenidos del tiempo relativo predicho para diferentes k

Otro de los aspectos más importantes de la escogencia del k es que el k escogido tiene que dar recomendaciones en que el tiempo relativo esperado sea alto.

En esta sección se muestran los resultados del experimento para examinar este aspecto. Para cada k se hacen distintas corridas, simulando en cada corrida un solo usuario activo. En cada corrida se encuentran los 10 mejores ítems a recomendar con diferentes k. Al final, para cada k y nivel de la recomendación se promedian los tiempos relativos.

Para calcular el tamaño de muestra de las simulaciones se determinó la desviación estándar para n=100, en el nivel 1 de recomendación, es decir, la primera recomendación que se aportaría. Con esto se obtiene el tamaño de la muestra con un error del 5% y un intervalo de confianza del 95%. El tamaño de muestra, n, fue de 36 usuarios activos, siendo menor al tamaño piloto por lo que se decidió trabajar con las corridas de la muestra piloto.

En la fig. 11 se presentan los resultados de los tiempos relativos promediados que se obtuvieron de simulaciones.

Como era de esperarse, para cada nivel de recomendación, el tiempo relativo promedio que resultó mayor fue el del mayor k. En general, se observa que a medida que aumenta el k, el tiempo relativo predicho de los ítems a recomendar también aumenta.

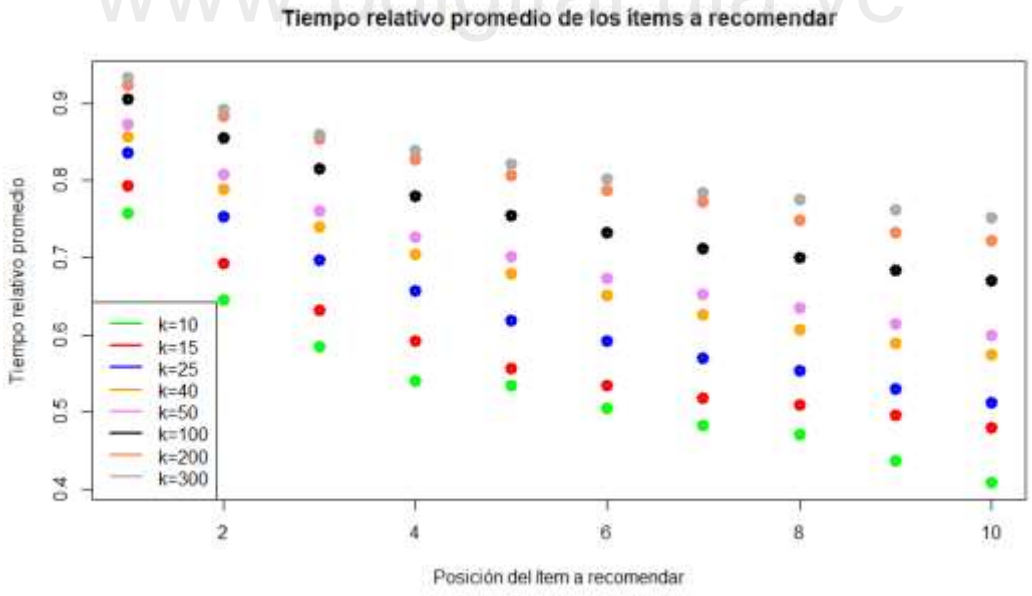

Figura 11. Tiempo relativo promedio de los ítems a recomendar

5.3 Resultados obtenidos del tiempo computacional para diferentes k

Otra consideración importante para la escogencia del k es el tiempo computacional que le tomaría al servidor buscar los k vecinos de un usuario activo y generar las recomendaciones. Para discutir este punto se procedió, para cada k, tomar de manera simulada 100 usuarios activos. Para cada usuario activo se calcularon los k vecinos, la lista de recomendaciones y se guardó el tiempo de procesamiento del CPU, este tiempo se obtiene con el software R y permite orientar en los tiempos de procesamiento para diferentes k. Luego de las 100 simulaciones se procedió a calcular el promedio de los 100 tiempos. Se aprovecharon las simulaciones de la sección 5.2.

En la fig. 12 se muestra el tiempo promedio de procesamiento para las simulaciones realizadas con los diferentes k utilizados en las simulaciones. El mayor tiempo de simulación se obtuvo con k=10, el tiempo de corrida disminuye significativamente a partir de k=25. Entre k igual a 40 y 50 pareciera haber un mínimo relativo. Luego sube para bajar. Ya el decrecimiento es poco a partir de 200.

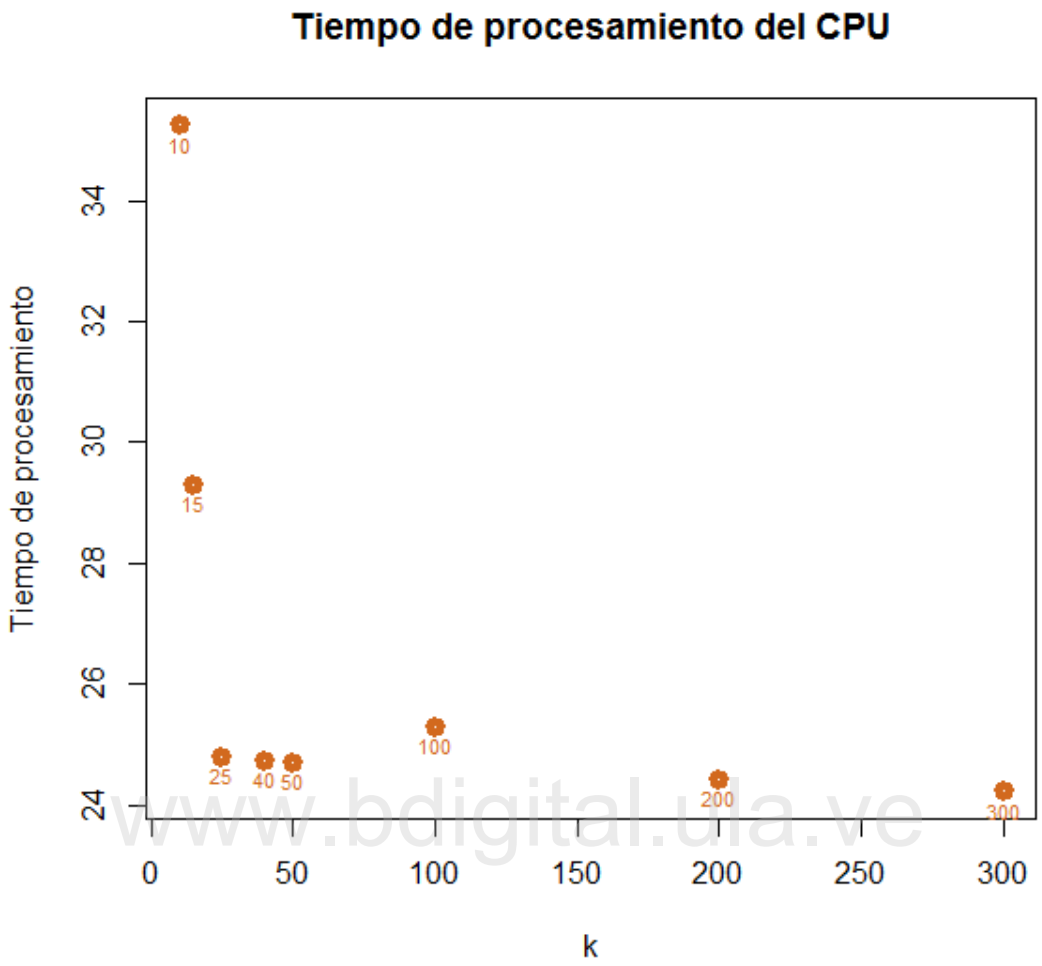

Figura 12. Tiempo de procesamiento del CPU para calcular los k vecinos más cercanos y generar las recomendaciones

CAPÍTULO VI

Conclusiones y recomendaciones

A continuación se describen diversas conclusiones y recomendaciones que se pueden considerar como consecuencias del desarrollo de esta investigación.

6.1. Conclusiones

- Muchos de los sistemas de recomendación desarrollados, en la actualidad, usan la técnica del vecino más cercano y muestran como su desventaja el tiempo computacional que requiere.
- Al hacer uso de una gran base de datos se presenta el inconveniente de la cantidad de memoria y tiempo de procesamiento que se requiere para generar las recomendaciones.
- La investigación realizada en este trabajo, que propone una metodología para reducir el tiempo computacional en el momento de generar las recomendaciones, produjo resultados alentadores para la implementación de la metodología para el desarrollo de sistemas de recomendación.
- La forma en que se utilizó el algoritmo general de los vecinos más cercanos, para realizar el filtrado colaborativo, logra generar recomendaciones adecuadas con base al comportamiento de los usuarios.
- El método propuesto tiene la ventaja, respecto a muchos desarrollos de sistemas de recomendación, en el sentido que permite considerar toda la base de datos sin recorrerla toda, realizando mecanismos de poda, reduciendo el tiempo computacional para la generación de las recomendaciones.
- En la investigación se esperaba que a medida que se aumentara el k, el error medio absoluto fuera disminuyendo, sin embargo, los gráficos hacen pensar que esta conjetura es falsa.

- A través del filtrado colaborativo implícito se puede construir sistemas de recomendación que complementarían las recomendaciones generadas por el filtrado colaborativo explícito.
- El filtrado colaborativo implícito, posiblemente, permite generar recomendaciones para un número más amplio de usuarios que el filtrado colaborativo explícito.
- Las recomendaciones que proporciona escogiendo k=40 tienen buenos tiempos relativos estimados, pero no tan buenos como k=200. Sin embargo, el error de la predicción es menor que el de k=200. Incluso con k=200 se tendría un intervalo de confianza para el tiempo relativo que sobrepasa los límites plausibles de los tiempos relativos.
- Con k=200 se obtienen buenos tiempos computacionales. Con este k, grande con respecto a k=40, se puede estar encontrando vecinos con gustos ya no tan similares al activo como con k=40. Sin embargo, puede estar proponiendo recomendaciones más novedosas y atractivas que con k=40.
- Con k=300 se tienen mayor error absoluto y tiempos relativos estimados de las recomendaciones similares a k=200, así que es preferible considerar k=200.

6.2. Recomendaciones

- Una forma de evaluar la verdadera contribución de este trabajo, requiere de su aplicación para generar recomendaciones a comercios electrónicos distintos al utilizado en esta investigación.
- Realizar validaciones con otros métodos de sistemas de recomendación, con el fin de comparar los tiempos de respuesta.
- La propuesta metodológica estuvo enfocada en buscar similitudes entre usuarios, esto se hizo ya que el planteamiento de crear archivos de ítems ayuda a disminuir el tiempo de búsqueda computacionalmente, sin embargo, se recomienda aplicar la metodología buscando similitudes entre ítems y realizar comparaciones con los resultados obtenidos en este trabajo.

- Se sugiere realizar estudios comparativos que midan la eficiencia entre esta metodología y una equivalente basada en filtrado colaborativo explícito.
- Al fin y al cabo el interés de un comercio electrónico es vender. Por esto, para evaluar la calidad de las recomendaciones se sugiere implementar un sistema de recomendación con k=40 y otro con k=200, de esta forma se podrán comparar no solo la efectividad en los tiempos relativos de las recomendaciones, sino también el índice de compras de ambos k.

BIBLIOGRAFIA

[1] Linden, G., Smith, B., y York, J. (2003). Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, 76-80.

[2] Ricci, F., Rokach, L., Shapira, B., y Kantor, P.B. (2011). *Recommender Systems Handbook*. Nueva York: Springer.

[3] Sarwar, B., Karypis G., Konstan, J. y Riedl, J. (2001). Item-based Collaborative Filtering Recommendation Algorithms.

[4] http://2015.recsyschallenge.com/ recuperado el 15 de Mayo de 2015.

[5] Sarwar, B., Karypis, G., Konstan, J., y Riedl, J. (2000). *Analysis of Recommendation Algorithms for E-commerce*. (Trabajo de investigación). Departamento de Ciencia Computacional e Ingeniería. Universidad de Minnesota.

[6] Langseth, H. (2009). Bayesian Networks for Collaborative Filtering. *NAIS*, 67-78.

[7] Ekstrand, M. D., Riedl, J. T., y Konstan, J. A. (2010). Collaborative Filtering Recommender Systems. *Now Publishers, 4*(2), 81-173.

[8] Töscher, A., y Jahrer, M. (2008). The BigChaos Solution to the Netflix Prize. Austria.

[9] Goldberg, K., Roeder, T., Gupta, D., y Perkins C. (2001). Eigentaste: a constant time collaborative filtering algorithm. *Information Retrieval*, 4(2), 133–151.

[10] Gjoka, M., y Soldo, F. (2008). Exploring collaborative filters: Neighborhood-based approach.

[11] Sahoo, N., Vir Singh, P., y Mukhopadhyay, T. (2012). A Hidden Markov Model for Collaborative Filtering. *EBSCOhost, 36*.

[12] Abbassi, Z., Sihem, A-Y., Lakshmanan, L., Vassilvitskii, S., y Yu, C. (2009). Getting Recommender Systems to Think Outside the Box. En *RecSys '09* (pp. 285-288). ACM.

[13] Lee T. Q., y Park, Y. A Time-based Recommender System using Implicit Feedback.

[14] Resnick, P. y Varian, H.R. (1997). Recommender systems. *Communications of the ACM,* 40(3), 56-58.

[15] Herlocker, J., Konstan, J., Riedl, J. y Terveen, L. (2004). Evaluating collaborative filtering recommender systems. *ACM Transaction on Information Systems*, 22(1), 5–53.

[16] Deshpande, M. y Karypis, G. (2004). Item-based top-*N* recommendation algorithms. *ACM Transactions on Information Systems*, 22(1), 143-177.

[17] Melville, P., y Sindhwani, V. Recommender Systems. *Watson Research Center, Yorktown Heights, NY*.

[18] Bobadilla, J., Hernando, A. y Ortega, F. (2012). A collaborative filtering similarity measure based on singularities. *Information Processing & Management,* 48(2), 204-2017.

[19] Breese, J., Heckerman, D. y Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. En *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*. (pp 43-52). San Francisco: Morgan Kaufmann Publishers Inc.

[20] Galán, N. (2007). Filtrado Colaborativo y Sistemas de Recomendación. Universidad Carlos III de Madrid.

[21] Su, X. y Khoshgoftaar, T. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* (2009), 1–20.

[22] Desrosiers, C. y Karypis G. (2011). A comprehensive survey of neighborhood-based recommendation methods. Springer. pp. 107-144.

[23] Koren, Y., Bell, R., y Volinsky, C. (2009). MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS. *IEEE Computer Society*, 42-49.

[24] Seguido, M. (2009). *Sistemas de recomendación para webs de información sobre la salud*. Tesis de Maestría. Departamento de Lenguaje y Sistemas Informáticos de la Universidad Politécnica de Cataluña