Implementación y prueba de algoritmo de recomendación consciente del contexto basado en factorización de matrices

Diego Armando González Delgado

Universidad del Bío-Bío

Facultad de Ciencias Empresariales, Departamento de Sistemas de Información.

Concepción, Chile

2015

Implementación y prueba de algoritmo de recomendación consciente del contexto basado en factorización de matrices

Diego Armando González Delgado

Proyecto de Título presentado en conformidad a los requisitos para obtener el título de Ingeniero Civil en Informática

Profesor Guía:
Pedro Gerónimo Campos Soto

Universidad del Bío-Bío
Facultad de Ciencias Empresariales, Departamento de Sistemas de Información.
Concepción, Chile
2015

No solo la Biblia, sino que todo el universo es un criptograma organizado por el todopoderoso, un rompecabezas ideado por Dios, nuestro deber es resolverlo.

Isaac Newton

"non nobis Domine non nobis. sed nomini tuo da gloriam"

"…et dimitte nobis debita nostra sicut et nos dimittimus debitoribus nostris…"

Agradecimientos

A Dios, porque sin su ayuda nada de esto sería posible, a mi familia, mis Padres por su apoyo incondicional, a Don Pedro Campos mi profesor guía por darme la oportunidad de trabajar en este tema y por su apoyo durante todo el proceso, a los angelitos que me cuidan desde el cielo y a esa personita especial, quien es el principal motivo para luchar y sacar adelante mis sueños.

Resumen

En este trabajo se analiza, modela e implementa un algoritmo de Recomendación Consciente del Contexto basado en la técnica de Factorización de Matrices. Se comienza presentando los Sistemas de Recomendación, para continuar con la Factorización de Matrices, y posteriormente introducir el concepto de “contexto”.

En base a dichos conceptos, se estudia el diseño e implementación del algoritmo de Factorización de Matrices Consciente del Contexto propuesto por Baltrunas et al. Implementación que luego es sometida a exhaustivas pruebas con distintas métricas de evaluación, para de este modo comprobar si efectivamente cumple con el propósito de entregar mejores recomendaciones que un algoritmo no consciente del contexto.

Los resultados obtenidos muestran que, utilizando esta nueva implementación, es posible obtener recomendaciones hasta un 40% más precisas en algunos casos, con respecto a la factorización no consciente del contexto. Sin embargo, la obtención de tales resultados requiere de un cuidadoso ajuste del valor de algunos parámetros del algoritmo.

Palabras clave: Sistemas de Recomendación, Factorización de Matrices, Contexto.

Contenido

Lista de figuras

Pág.

Lista de tablas

Pág.

1.Introducción

En los últimos años y debido principalmente a la sobre carga de información que tenemos en internet, han proliferado considerablemente los denominados sistemas de recomendación, los cuales proporcionan a los usuarios, información acerca de productos y/o servicios, que puedan ser del interés del usuario, esto tras realizar un “estudio” de su perfil, sus gustos e incluso de la forma en la que el usuario navega por internet.

Sin entrar mucho en detalle, podemos poner algunos ejemplos de webs que trabajan con sistemas de recomendación, donde la web de “Amazon” es la que tiene el sistema de recomendación más conocido por todo el mundo ya que una vez que el usuario se da de alta en su web, este registra la navegación que hace el usuario por sus productos y sus compras, para luego con estos datos ser capaz de ofrecer al usuario solo productos que le puedan interesar y que no sea el usuario el que deba revisar los miles de productos disponibles en Amazon. Otro sitio web que trabaja en base a sistemas de recomendación es YouTube. Esta web registra los últimos vídeos vistos por un usuario y recomienda videos similares, basándose seguramente en el contenido de los mismos o bien en los “tags” característicos de los vídeos.

Son precisamente estas técnicas las que se pretende estudiar e implementar, sumando una nueva variable al sistema de recomendación, esta nueva variable es el contexto. El contexto es un concepto bastante multifacético, estudiado a través de diferentes áreas como la inteligencia artificial, psicología, ciencia cognitiva y la lingüística entre otros.

En la bibliografía relacionada al tema, el contexto se definió inicialmente como la ubicación geográfica del usuario, la identidad de las personas cercanas a él, los objetos a su alrededor, o los cambios en estos elementos. Otros incluyen la fecha, la temporada, la temperatura, etc.

1.1 Origen del tema

Ciencia cognitiva, teoría de aproximación, recuperación de información, la teoría de predicción y los modelos de elección de los clientes en la comercialización, forman parte de los inicios de los sistemas de recomendación, que sin embargo surgió de forma independiente a mediados de los años 90, cuando las investigaciones comenzaron a basarse específicamente en los problemas de recomendación, los cuales en su formulación más común, se reducen al hecho de estimar puntuaciones para los elementos que aún no han sido vistos por un usuario [1].

Es dentro de esta perspectiva donde surgen diferentes formas de estimar dichas puntuaciones, siendo la más intuitiva, la de realizar las estimaciones en base a calificaciones entregadas por el mismo usuario a otros artículos similares, hecho esto, se recomienda al usuario el o los artículos con las más altas calificaciones estimadas.

A esta idea principal, paulatinamente se ha ido incorporando diferente tipo de información, esto para obtener estimaciones cada vez más precisas. En este marco, las últimas investigaciones sugieren que agregar información del contexto en el cual fueron realizadas las calificaciones por parte del usuario, incrementan notoriamente la precisión en las estimaciones [2].

Basándose en esta idea, se analizará e implementará un algoritmo de recomendación que incorpora dicha información contextual [2], para posteriormente, comparar los resultados con otro tipo de sistemas de recomendación, para de este modo comprobar y/o corroborar si efectivamente las recomendaciones otorgadas por dicho algoritmo son o no más precisas que las de otros algoritmos que no incorporen información contextual en sus estimaciones.

1.2 Objetivos

1.2.1 Objetivo General

El objetivo general del proyecto consiste en la implementación y prueba de un algoritmo de recomendación consciente del contexto basado en la extensión de un algoritmo básico de factorización de matrices para sistemas de recomendación.

1.2.2 Objetivos Específicos

• Estudiar y comprender algoritmos de recomendación, técnicas de factorización de matrices y algoritmos de recomendación conscientes del contexto, así como las librerías de software involucradas en su implementación.

• Extender una librería de algoritmos de recomendación existente, por medio de la implementación de un algoritmo de recomendación consciente del contexto basado en factorización de matrices.

• Realizar pruebas sobre el algoritmo para verificar y validar su correcto funcionamiento.

• Comparar el rendimiento del algoritmo implementado con respecto a otros algoritmos existentes en la librería.

1.3 Presentación de Capítulos

Este informe se encuentra organizado en 9 capítulos, cuyo contenido se describe a continuación:

El capítulo 1 (**Introducción**) describe el tema central de este trabajo, partiendo por el origen del tema, seguido de los objetivos generales y específicos.

El capítulo 2 (**Marco Teórico)** nos presenta lo que son los Sistemas de Recomendación, su historia y evolución, para terminar con una pequeña clasificación de los mismos.

El capítulo 3 presenta la denominada **Factorización de Matrices**, y como se utiliza dicha técnica en sistemas de recomendación, presentando un modelo básico, para luego agregar más información relevante con la finalidad de precisar aún más las estimaciones. Al final de este capítulo se presentan 3 implementaciones de dicha factorización existentes en la web.

El capítulo 4 (**Sistema de Recomendación Consciente del Contexto**) introduce de lleno el tema del contexto, nos explica un poco el concepto de “contexto”, para luego utilizar la técnica de Factorización de Matrices vista en el capítulo 3, pero insertando esta vez información contextual. Se presenta también una idea de cómo obtener dicha información de contexto.

El capítulo 5 (**Implementación de un Sistema de Recomendación Consciente del Contexto**) implementa toda la teoría vista en los capítulos anteriores, presentando un algoritmo de recomendación consciente del contexto junto a su respectivo diagrama de clases.

El capítulo 6 presenta las **Pruebas de la Implementación** realizadas al algoritmo consciente del contexto, definiendo con precisión los datos de prueba y la forma en que deben ser ejecutadas las mismas, junto con las métricas utilizadas para la comparación de la implementación en contraste con implementaciones de algoritmos de recomendación basados en factorización de matrices que no utilizan información contextual para realizar sus estimaciones.

El capítulo 7 (**Resultados de las Pruebas**) entrega un completo resumen de las pruebas realizadas en el capítulo anterior, presentando tablas y gráficos de los datos obtenidos. Luego de esto se presenta una pequeña evaluación de los resultados obtenidos junto con las conclusiones inferidas a través de las pruebas.

Finalmente, el capítulo 8 presenta las **Conclusiones** generales del proyecto, aportes y posibles trabajos futuros en la misma línea temática propuesta.

2. Marco Teórico

2.1 Sistemas de Recomendación

Hoy en día disponemos de tal cantidad de información, que la habilidad para identificar cual sería la más útil, para cada una de nuestras necesidades, se vuelve sumamente compleja, convirtiéndose en un caos de información, sin que el usuario pueda encontrar a veces lo que realmente desea.

Los Sistemas de Recomendación surgen para solucionar este problema, pasando a ser una herramienta indispensable en muchas tareas de la vida cotidiana, ante el problema de filtrar contenido interminable.

Básicamente, los sistemas de recomendación reciben información del usuario acerca de productos o servicios en los que el usuario se encuentra interesado y le recomienda aquéllos que estén más cercanos a sus necesidades.

R. Burke [3] define los Sistemas de Recomendación como “sistemas que producen recomendaciones personalizadas como salida o tienen el efecto de guiar al usuario de una forma personalizada a productos interesantes o útiles entre una gran cantidad de productos disponibles”.

El objetivo de un buen sistema de recomendación es hacer llegar a cada usuario la información que necesita, evitando la que no le interesa. “Sí le gusta este libro, tal vez le interesen también los siguientes”. Las personas simplemente valoran lo que han visto y el sistema le recomienda otros ítems afines a sus gustos.

Tal ha sido la expansión de los Sistemas de Recomendación, que desde el año 2007, se realiza en diferentes ciudades del mundo la denominada “ACM RecSys”, una conferencia dedicada en tu totalidad a los Sistemas de Recomendación, organizada por la *Association for Computing Machinery* [4], es en dicha conferencia donde se presentan los mayores avances en la materia y a su vez que se presentan los nuevos desafíos y trabajo futuro.

En la actualidad, existe una gran cantidad de sitios especializados en Internet, ofreciendo millones de productos y servicios para su consumo, que utilizan sistemas de recomendación, a continuación se citan algunos:

- Amazon (Página de compra por internet, incluye recomendaciones de productos),
- Netflix (Servicio de alquiler de Series y Películas con recomendación según gustos),
- Reddit (Sistema de recomendación de noticias),
- StumbleUpon (Recomendación páginas web)
- Youtube o Last.fm (Sistemas de música).

Ahora bien, formalmente el problema de la recomendación puede ser descrito de la siguiente manera [1]:

Sea U el conjunto de todos los usuarios del sistema e I el conjunto de todos los ítems del mismo, se define entonces la función de utilidad r, función que mide la utilidad de articulo i para un usuario u.

$$r = U\ x\ I\ \rightarrow R$$

Donde R es un conjunto ordenado (enteros no negativos o números reales entre un cierto rango)

Entonces, para cada usuario $u\ \in U$, queremos elegir un ítem $i\ \in I$ que maximice la utilidad del usuario, esto es:

$$\forall\ u\ \in U, i'_u\ = \arg max_{i \in I}\ r(u,i)$$

En este sentido, el argumento (ítem) con la máxima utilidad puede ser ninguno, solo uno o muchos.

Generalmente r no está definida para todo el conjunto, si no sólo para una parte de éste, por lo que necesita extrapolarse, este es el principal problema de los sistemas de recomendación.

La utilidad se representa a través de un rating o puntuación, la puntuación generalmente se da solo para ítems puntuados por usuarios, la idea de los sistemas de recomendación es estimar o predecir la puntuación de los ítems no puntuados para realizar de esta forma recomendaciones apropiadas.

Existen dos grandes formas de estimar o predecir estas puntuaciones, la primera está basada en heurísticas específicas para definir la función de utilidad o a través de validaciones empíricas, la segunda es estimar funciones de utilidad optimizando ciertos criterios de rendimiento tales como por ejemplo “Raíz del Error Medio Cuadrático” o el “Error absoluto medio” [5].

Una vez puntuados todos los ítems, por cualquiera de las dos formas mencionadas anteriormente, se selecciona el mejor o los N mejores para recomendar según sea el caso.

2.2. Historia y evolución de Sistemas de Recomendación

Como se mencionó anteriormente la rama de investigación de Sistemas de Recomendaciones surgió como tal a mediados de los años 90, previo a esto, las recomendaciones se realizaban mediante algoritmos procedentes de la Recuperación de Información [6], realizando recomendaciones sin personalización, es decir, a todos los usuarios del sistema se recomienda el mismo conjunto de productos [7]. Para ello, se utilizaban distintas herramientas:

- **Filtrado basado en características:** Esta técnica pretende buscar un producto o conjunto de productos que satisfaga una serie de propiedades que el usuario define en una consulta. El mayor problema de esta técnica es que el usuario debe conocer las características de los productos que se encuentran en el sistema y puede especificar consultas que no se correspondan con ningún producto.
- **Filtrado de productos sin personalización:** Con esta técnica se calcula una lista de productos que se corresponde con los productos mejor valorizados (mayor valoración media) o con los productos más populares del sistema (el más comprado).
- **Filtrado con datos generales de los consumidores:** Utilizan datos generales que el sistema conoce, sin realizar ninguna operación con los datos del usuario al que se desea recomendar.

Un tipo de sistema que también se utilizaba en los inicios de los Sistemas de Recomendación son los Sistemas de Recomendación Basados en Contenido [8]. En este tipo de sistemas se tiene información sobre el contenido de los productos, como sus atributos o descripciones de los mismos. El perfil de usuario se calcula utilizando la información de contenido de los productos que le gustan. Uno de los primeros Sistemas de Recomendación Basado en Contenido es un sistema de noticias en el que se toma cada palabra como una característica, para describir el contenido de la misma mediante representaciones en espacios vectoriales. También se han propuesto otros métodos para este tipo de Sistemas de Recomendación, tales como redes neuronales [9] y redes bayesianas [10].

Luego de esto, a mediados de los años 90 surgen los primeros trabajos sobre filtrado colaborativo [11], en este tipo de sistemas se realizan las estimaciones en base a otros usuarios con gustos similares, y uno de los primeros sistemas desarrollados en este campo es el llamado Tapestry [12] concebido como una forma eficiente de manejar grandes volúmenes de correo, proporcionando listas de correo, permitiendo a los usuarios suscribirse sólo a sus listas de interés, convirtiéndola en una importante área de investigación.

Existe en la actualidad un grupo de investigadores reunidos bajo el alero de GroupLens, una organización dedicada básicamente a la recomendación de noticias y películas

basándose en filtrado colaborativo, trabajando constantemente para mejorar dicha técnica y lograr importantes avances [13].

2.2 Clasificación de los Sistemas de Recomendación

Los Sistemas de Recomendación se clasifican según la forma de estimar los ítems no puntuados. La literatura generalmente propone la siguiente clasificación:

- **Recomendaciones Basadas en Contenido:**

Se recomienda al usuario ítems similares a los preferidos por el mismo usuario en el pasado.

- **Recomendaciones Basadas en Filtrado Colaborativo:**

Se recomiendan ítems que las personas con gustos similares al usuario, probaron y prefirieron en el pasado.

- **Aproximaciones Hibridas:**

Este método combina los métodos de recomendación colaborativos y basadas en contenido.

Los sistemas de recomendación basados en Contenido, fueron durante décadas la mejor forma de recomendación, hasta la llegada del método de Recomendaciones mediante Filtrado Colaborativo. Este tipo de sistemas Basados en contenido, utilizan técnicas de recuperación de información, es decir, en un ejemplo sencillo, un documento de texto es recomendado en base a una comparación entre el contenido del mismo, y el perfil del usuario de interés, que no es más que una lista de palabras claves con su respectiva ponderación. Por otra parte se realiza un análisis de frecuencia al documento para determinar sus palabras clave y sus respectivos pesos.

Uno de los principales problemas de este tipo de sistemas, es el hecho de que el sistema necesita información en texto que describa de forma adecuada a cada ítem, con el fin de tener un conjunto de características suficientes con las cuales trabajar, además dicha información debe ser obtenida de forma automática. En general, los métodos

automáticos de extracción de características, son mucho más difíciles de aplicar a los datos multimedia como imágenes, audio y/o secuencias de vídeo [1].

La integración de nuevos usuarios al sistema [14] es otro de los problemas más evidentes de los sistemas Basados en contenido, ya que, como la recomendación se basa en contenido valorado anteriormente por el usuario, en el caso de usuarios dados de alta recientemente, significa que no hay valoraciones previas con las cuales comparar.

Otro de los problemas que presentan los sistemas de recomendación basados en contenido es la llamada sobre-especialización. La sobre-especialización [1] se produce al recomendar sólo contenidos muy similares a los valorados anteriormente, sin tener conciencia de la arbitrariedad de gustos e intereses de los usuarios, por lo que se recomienda siempre un solo tipo de ítems. Este y otros problemas, suelen ser solucionados parcialmente con la incorporación de algoritmos de aleatoriedad en la recomendación del contenido [15].

Los sistemas de Recomendación basados en técnicas de Filtrado Colaborativo en cambio, basan sus recomendaciones en términos de la similitud entre los gustos de un grupo de usuarios (o ítems), es decir, para cada usuario se analiza una “vecindad” de usuarios con evaluaciones similares, y de los cuales se obtienen las puntuaciones para los ítems no calificados por el usuario en cuestión.

Otro enfoque diferente para el Filtrado Colaborativo, es el Modelo de Factores Latentes [16], donde el método más utilizada para encontrar dichos factores, es la llamada Factorización de Matrices [17], que utiliza la técnica de reducción de dimensiones de la Matriz de Datos. La Matriz de Datos, es una forma confiable de representar las puntuaciones usuario-ítem en un formato reducido. La Factorización de Matrices utiliza a su vez, la denominada Descomposición de Valores Singulares [18] (SVD por sus siglas en inglés) que es un método bastante conocido, y que se utiliza en la factorización de matrices, proporcionando las mejores aproximaciones de la matriz original. Este tema será abordado con más en detalle en el siguiente capítulo.

Si bien las recomendaciones suelen ser más precisas que en sistemas basados en contenido, los sistemas de Filtrado Colaborativo, traen consigo algunos problemas

heredados, como por ejemplo la incorporación de un nuevo usuario, que al igual que en el caso anterior, al no tener valoraciones de ítems, el sistema no podrá realizar recomendaciones hasta que su perfil este lo suficientemente completo para encontrar su propia vecindad. Un problema similar ocurre con el ingreso de un nuevo ítem, que al no haber sido puntuado por ningún usuario, no será parte de ninguna nueva recomendación.

Otro problema asociado al Filtrado Colaborativo, es el uso diferenciado de la escala de valoración por parte de los usuarios del sistema, esto es básicamente que una misma nota de puntuación, puede suponer diferentes apreciaciones por parte de los distintos usuarios, es decir, por ejemplo la nota 3 puede no significar lo mismo para un usuario u otro.

Este y otros problemas asociados al Filtrado Colaborativo pueden ser resueltos a través de varias técnicas.

La primera de ellas es el llamado “Filtrado Demográfico” que considera usuarios similares para conformar una vecindad, no solo en base a sus calificaciones y preferencias, si no también cuando dichos usuarios pertenecen al mismo segmento demográfico. Por ejemplo podemos utilizar información del usuario respecto a su género, edad, código de área, ocupación, etc [19].

En los últimos años, han comenzado a proliferar los sistemas de recomendación que intentan fusionar tanto los Sistemas Basados en Contenido como los de Filtrado Colaborativo, a este nuevo tipo de sistemas se les denomina Sistemas de Recomendación Híbridos.

Existen varias formas de clasificar estos sistemas según el grado en que una de las técnicas sea la base y cual el complemento, es decir, existen sistemas Basados en Contenido con características colaborativas o bien Sistemas de Filtrado Colaborativo con características basadas en contenido.

En la actualidad, se trabaja arduamente para conseguir un sistema hibrido, que logre unificar las más reconocidas técnicas de recomendación en un solo sistema, capaz se entregar recomendaciones cada vez más exactas.

3. Factorización de Matrices

Como se mencionó anteriormente, los Sistemas de Recomendación basados en Filtrado Colaborativo se abordan generalmente a través de dos áreas principales: El método de los “vecinos cercanos” y el modelo de factores latentes.

El método de los “vecinos cercanos” como se dijo en el capítulo anterior, se basa en las relaciones existentes entre usuarios y sus preferencias o bien entre ítems, en dicho enfoque orientado a los ítems, se considera como “vecinos” a los ítems que obtienen una puntuación similar a otro cuando son calificados por un usuario en particular.

El modelo de factores latentes en cambio, intenta explicar las puntuaciones de los usuarios caracterizando ítems y usuarios a través de un número determinado de factores latentes, dichos factores son inferidos a partir de patrones encontrados en las puntuaciones por parte de los usuarios. Estos factores son, en general no muy obvios, es decir, podríamos ser capaces de pensar de algunos de ellos, pero es difícil estimar su impacto sobre las calificaciones, esto implica que estos factores no necesariamente deben ser conocidos con anterioridad [20].

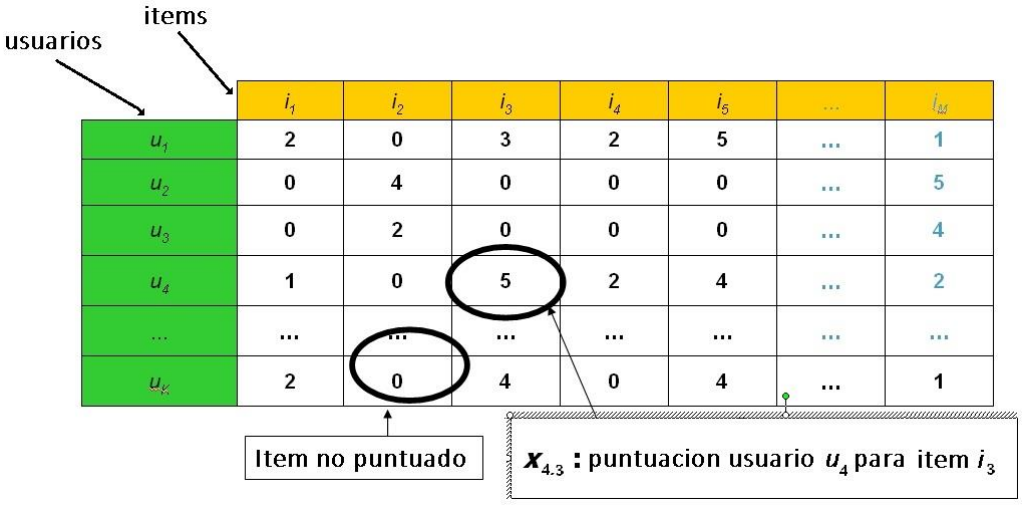
La principal idea del modelo de factores latentes, se basa en el hecho de que una calificación está profundamente influenciada por un conjunto de factores que son muy específicos para el dominio (por ejemplo, cantidad de acción en las películas, la complejidad de los personajes, etc.). El objetivo es inferir los llamados factores latentes de los datos de calificación mediante el uso de diferentes técnicas matemáticas, en este caso, Factorización de Matrices.

3.1 Sistemas de Recomendación Basados en Factorización de Matrices

El uso de la Factorización de Matrices para detectar factores latentes, se basa principalmente en el hecho de que dicha factorización permite encontrar factores latentes que no sólo hacen posible realizar predicciones, sino que además permite obtener información acerca de la relación entre usuarios e ítems, más específicamente la factorización de matrices, nos permite escalabilidad, flexibilidad y predicciones más certeras que otros métodos similares [21].

Factorización de Matrices caracteriza a ítems y usuarios a través de factores, que son inferidos mediante la búsqueda de patrones en la puntuación de los ítems. Existen diferentes maneras de obtener los datos de entrada con los cuales generar las distintas recomendaciones, la más efectiva y conveniente, es la llamada "Retroalimentación explicita de alta calidad", donde los usuarios explícitamente entregan su puntuación respecto a los ítems o productos evaluados.

Estos datos de puntuación se ubican generalmente en una matriz de dos dimensiones denominada Matriz de Datos, donde la primera dimensión (filas) hace referencia a los usuarios, la segunda (columnas) a los ítems a evaluar y la puntuación está inserta como dato dentro de la matriz (Figura 1).



	i_1	i_2	i_3	i_4	i_5	...	i_M
u_1	2	0	3	2	5	...	1
u_2	0	4	0	0	0	...	5
u_3	0	2	0	0	0	...	4
u_4	1	0	5	2	4	...	2
...	...	...	...	...	...	...	...
u_K	2	0	4	0	4	...	1

Figura 1: Matriz de Datos

Existen a su vez varias medidas de puntuación, por ejemplo la empresa de entretenimientos Netflix, utiliza una escala de estrellas de 1 a 5, mientras que en otros, la calificación es simplemente en términos de “buena o mala”. Esta forma de obtener los datos de entrada, tiene el siguiente problema, y es que probablemente cada usuario en particular, solo ha evaluado un pequeño porcentaje de ítems dentro de todos los ítems a puntuar, dicho problema conlleva a una matriz de datos bastante dispersa.

Una forma de solucionar este problema de la Matriz de Datos dispersa, es a través de lo que se denomina “Retroalimentación Implícita”, esto es, la obtención indirecta de la opinión de los usuarios, a través de observar por el ejemplo, el historial de compras, historial de navegación, patrones de búsqueda, o incluso movimiento del mouse.

3.2 Modelo Básico de Factorización de Matrices

El modelo más sencillo de Factorización de Matrices [20], modela las puntuaciones conocidas como una matriz M de dimensión m×n, donde m y n es la cantidad de usuarios e ítems respectivamente, y se denota a $r_{ui} \in$ M, como la puntuación del usuario u, al ítem i. Este modelo apunta a representar la matriz M en 2 matrices más pequeñas p y q de dimensiones m×f y f×n respectivamente, donde f hace referencia al número de factores latentes, con f << m y f << n.

Un usuario u se representa a continuación, como un vector columna $p_u \in p$ y un ítem i como un vector fila $q_i \in q$. El producto punto entre ambos vectores q_i y p_u, representado por ${q_i}^T p_u$ se estima como la calificación que el usuario u, daría al ítem i, esto lo denotamos como $\hat{r}_{ui}$. De esta forma, se puede obtener una predicción de valoración (recomendación) mediante:

$$\hat{r}_{ui} = {q_i}^T p_u \qquad (1)$$

De esta forma, podemos estimar fácilmente la calificación que un usuario le dará a cualquier elemento, mediante el uso de esta ecuación.

El siguiente desafío, es la obtención o identificación de los factores latentes. Es aquí donde adquiere una especial importancia la llamada "Descomposición de Valores Singulares" o "*singular value decomposition*" (SVD), técnica bastante conocida y utilizada generalmente para la identificación de factores semánticos latentes en recuperación de información.

Para poder aplicar descomposición de valores singulares en sistemas de recomendación basados en filtrado colaborativo, es necesario factorizar la matriz de puntuaciones de usuarios-elementos. Generalmente, esto es un gran problema, debido a la dispersión de datos de la matriz, pues SVD no está definido para matrices incompletas, sin embargo, podemos realizar algunas pequeñas modificaciones a esta técnica para poder utilizarla en aquellos casos de matrices dispersas, sin perjuicio de un gran riesgo de "sobreajuste" [22].

Para conocer los vectores de factores (q_i y p_u), el sistema minimiza el error cuadrático y lo normaliza en el conjunto de calificaciones conocidas [20]:

$$\min_{q*,p*} \sum_{(u,i)\in R} [(r_{ui} - {q_i}^T p_u)^2 + \lambda (\| q_i \|^2 + \| p_u \|^2)] \quad (2)$$

Donde R es el conjunto de los pares (u, i) para los cuales se conoce r_{ui} (Conjunto de entrenamiento).

Esto proceso se conoce como "aprendizaje" o "entrenamiento", ya que el sistema "aprende" el modelo mediante el ajuste de las calificaciones observadas anteriormente.

Es aquí donde debemos evitar el antes mencionado "sobreajuste" o "sobre-entrenamiento", que se produce cuando el sistema se ajusta demasiado a los valores de entrenamiento, quedando de esta forma ajustado o entrenado a características muy específicas de dichos datos, lo que produce un efecto negativo en los datos a predecir, que mayoritariamente no cuentan con dichas características sobreajustadas, es por esta

razón, que debemos evitar que el sistema sobreajuste los datos, y para ello se introduce en la ecuación la constante λ, que si bien no existe una forma exacta de calcular su valor, suele ser determinada a través de validación cruzada.

Para minimizar esta ecuación existen dos enfoques, el llamado mínimos cuadrados alternados (*alternating least squares, ALS*) y el descenso de gradiente estocástico (*stochastic gradient descent*), este último será el método a utilizar, por ser una técnica de fácil implementación y con un tiempo de ejecución relativamente rápido.

Este enfoque utiliza las derivadas parciales de cada uno de los parámetros (p y q) de la función de minimización (2) para realizar las actualizaciones correspondientes. La diferencia entre la puntuación real del usuario u, hacia el ítem i y la puntuación estimada para el mismo usuario e ítem, se puede definir como el error asociado a esa puntuación:

$$e_{ui} = r_{ui} - {q_i}^T p_u$$

Con esto, se actualizan los parámetros mediante sus derivadas parciales por una magnitud proporcional a γ en la dirección opuesta a la pendiente de la siguiente manera:

$$q_i \leftarrow q_i + \gamma * (e_{ui} * p_u - \lambda * q_i)$$

$$p_u \leftarrow p_u + \gamma * (e_{ui} * q_i - \lambda * p_u)$$

Donde γ también es conocido como paso, salto o tasa de aprendizaje.

Esta actualización se ejecuta en reiteradas ocasiones hasta lograr encontrar los factores q_i y p_u que caractericen de formas “precisa” (evitando el sobreajuste) a usuarios e ítems, lo que permite determinar de esta forma, las puntuaciones desconocidas en la matriz de datos.

3.3 Sesgos

Esta aproximación básica de Factorización de Matrices, no toma en consideración varios aspectos fundamentales, tales como la diferencia que existen en la forma en que los usuarios utilizan la escala de valoración de los ítems. Por ejemplo el caso en que un usuario sistemáticamente acostumbre a realizar valoraciones más altas a los demás usuarios, o ítems que suelen recibir una calificación más alta que los demás, después de todo existen ítems o productos considerados ampliamente como mejores o peores que otros.

Este tipo de efecto se conoce como sesgo [20] o la diferencia entre el valor esperado por un estimador y el verdadero valor del parámetro y puede llegar a generar grandes variaciones en las recomendaciones, por lo tanto no sería prudente utilizar el sistema sin tomar en consideración estos sesgos. Una primera aproximación para incluir el sesgo en la obtención de $\hat{r}_{ui}$ seria.

$$b_{ui} = \mu + b_i + b_u$$

Donde b_{ui} es el sesgo involucrado, μ es la calificación promedio general del ítem y b_i y b_u son las desviaciones observadas de ítems y usuarios respectivamente.

Aplicando este sesgo a la ecuación (1), el resultado es…

$$\hat{r}_{ui} = q_i{}^T p_u + \mu + b_i + b_u \quad (3)$$

Y respectivamente en la ecuación (2) de minimización del error:

$$\min_{q*,p*,b*} \sum_{(u,i)\in R} [(r_{ui} - q_i{}^T p_u - \mu - b_i - b_u)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2 + b_u^2 + b_i^2)]$$

3.4 Implementaciones Existentes

Existen a nivel mundial, variadas organizaciones dedicadas a la implementación de Sistemas de Recomendación, dentro de estas, no son pocas las que dedican todos sus esfuerzos a la implementación de sistemas basados en filtrado colaborativo y más aún, en el diseño e implementación de Sistemas de Recomendación basados en Factorización de Matrices.

Entre los sistemas de recomendación más destacadas, que contienen implementaciones basadas en Factorización de Matrices podemos encontrar:

- MyMediaLite (C#) - http://www.mymedialite.net/
- Apache Mahout (Java) - http://mahout.apache.org/
- LensKit (Java) - http://lenskit.org/
- GraphLab collaborative filtering library (C++) – http://graphlab.org/
- Crab (Python) http://muricoca.github.io/crab/

3.4.1 MyMediaLite

MyMediaLite [23] es una biblioteca de algoritmos de Sistemas de Recomendación para múltiples usos y se dirige especialmente a los dos escenarios más comunes en el filtrado colaborativo:

- Predicción de calificación (por ejemplo, en una escala de 1 a 5 estrellas), y
- Predicción de ítems según feedback (por ejemplo, de los clics, gustos, o compras).

MyMediaLite es un software libre, de código abierto, escrito en C#, Python, entre otros lenguajes de programación. La última versión disponible de MyMediaLite es la 3.10.

Esta librería fue desarrollada por Zenó Gantner, Steffen Rendle, Lucas Drumond, y Christoph Freudenthaler en la Universidad de Hildesheim.

MyMediaLite pretende ser un tipo de Framework [24] y se dirige tanto a los usuarios académicos e industriales, que pueden utilizar los algoritmos existentes en la biblioteca, o utilizar el marco para la implementación y evaluación de nuevos algoritmos.

3.4.2 LensKit

LensKit [25] es un completo set de herramientas de recomendación, desarrollado por Múltiples investigadores del grupo GroupLens - http://grouplens.org/ -, en particular MovieLens y BookLens, quienes usan LensKit en sus proyectos, proporcionando experiencia, informes de errores, y otra información valiosa (y códigos).

Está escrita en lenguaje Java y actualmente se desarrolla la versión 2.2 la cual ya se encuentra en su versión Beta.

Esta librería proporciona una implementación de FunkSVD, un algoritmo de filtrado colaborativo basado en SVD que utiliza descenso por gradiente estocástico para aprender los factores y desarrollar la factorización de la matriz [26].

MovieLens es un Sistema personalizado de recomendación de películas, que ayuda a sus usuarios a encontrar películas de su agrado, en base un perfil construido a través de la puntuación de diferentes películas ya vistas por el usuario.

GroupLens pone a libre disposición de investigadores y desarrolladores un completo set de datos de MovieLens [27] con más de 10 millones de puntuaciones realizadas por 72.000 usuarios a 10.000 películas, convirtiéndose de este modo en el más grande proveedor de este tipo de datos, los que se utilizan mayoritariamente para probar la efectividad de los sistemas de recomendación desarrollados.

3.4.3 Apache Mahout

Apache Mahout [27] es una completa biblioteca de Machine Learning, y se enfoca principalmente en 3 áreas, el filtrado colaborativo, almacenamiento en clúster y clasificación.

Mahout proporciona bibliotecas de Java para las operaciones matemáticas comunes centrado en el álgebra lineal y estadística. Mahout es un trabajo en progreso, y el número de algoritmos implementados crece constantemente.

Esta librería implementa un gran número de algoritmos de filtrado colaborativo, siendo la denominada Descomposición de Valores Singulares una de las más reconocidas [28].

Dicha implementación basada en Factorización de Matrices será utilizada como base para la implementación de una librería propia que incluirá factores de contexto en cada recomendación.

Mahout actualmente se encuentra en su versión experimental 1.0, sin embargo la versión a utilizar será la 0.9 por ser esta una versión completa y probada en su totalidad.

4. Sistema de Recomendación Consciente del Contexto

Los Sistemas de Recomendación clásicos, como se observó anteriormente, presentan ciertos problemas relacionados con el hecho de basarse únicamente en la interacción de usuarios e ítems. Uno de los nuevos problemas detectados es la diferencia que puede suponer para un usuario, necesitar o gustar de cierto ítem, dependiendo del contexto en el que éste se encuentre inmerso.

Por ejemplo, no es lo mismo recomendar paquetes vacacionales para una pareja de recién casados, para una familia con hijos o para un grupo de adolescentes; o bien, en el caso mismo de las vacaciones, también se debe considerar la época del año, ya que las vacaciones invernales difieren mucho de las vacaciones en época estival.

Una forma de intentar corregir dichos problemas, consiste en utilizar además de la información de usuarios e ítems, información relativa al contexto que rodea al usuario al momento de realizar la puntuación de un ítem.

Por contexto, podemos considerar variadas acepciones, como la localización de usuario, objetos alrededor del mismo, la fecha, estaciones del año, el nivel de experiencia de un usuario, la relación del usuario con los ítems a recomendar, etc... Que afecten de manera significativa o no la situación contextual [29].

Por ejemplo, en el caso de un sistema de recomendación de películas, no es lo mismo recomendar una película para una pareja, una familia o una persona sola, en este caso un contexto valido y útil para utilizar en las recomendaciones sería la compañía social con la que cuenta el usuario.

Cabe mencionar que el contexto es un componente formado de otros componentes, como pueden ser la hora o la compañía. En base a esto, cada aplicación define el contexto que más le conviene para una correcta recomendación.

El primer problema que se presenta en los Sistemas de Recomendación Conscientes del Contexto, es la forma de obtener dicha información de contexto. Entre las formas más utilizadas están [29]:

- **De forma Explicita:** Esto es, preguntando directamente a los usuarios u obteniendo dicha información de otro tipo de fuentes.
- **De forma Implícita:** Es decir, obteniendo la información a partir de otros datos, como lo pueden ser la localización o la marca de tiempo de alguna transacción.
- **Por Inferencia:** Obtener la información de contexto a través de minería de datos o algún otro tipo de procesamiento estadístico.

Una vez obtenida dicha información, por cualquiera de los métodos anteriores, llega el momento de aplicarla al Sistema de Recomendación para ayudar a este a generar mejores recomendaciones, buscando la adaptación de las recomendaciones al contexto actual (o deseado) del usuario (contexto objetivo). La manera en que aplicamos esta información puede realizarse de 3 diferentes maneras [29]:

- Explotar la información contextual en una primera etapa, es decir, se procesan los datos con la información contextual, para obtener un set de datos acorde al contexto objetivo. Luego, se puede utilizar cualquier técnica de recomendación sobre los datos ya filtrados. A esto se le conoce como **Pre-Filtrado Contextual**.
- Otra forma es ignorar la información contextual en la etapa de obtención de los ratings, para luego incluirla y contextualizar la situación de cada usuario. Este método se denomina **Post-Filtrado Contextual**.
- Por último, la información contextual puede ser usada directamente en la estimación de los ratings, esto se denomina **Modelado Contextual**. Esta última forma de incluir la información contextual en las recomendaciones, es la que se utiliza en este trabajo.

4.1 Factorización de Matrices Consciente del Contexto

En el capítulo anterior se describió cómo la Factorización de Matrices se ha convertido en uno de los métodos más eficientes y eficaces a la hora de calcular estimaciones para la recomendación de ítems. En este sentido la Factorización de Matrices Consciente del Contexto [2] (desde ahora CAMF, por sus siglas en inglés) no es más que una extensión de modelo clásico de Factorización de matrices pero que es capaz de utilizar información contextual en la predicción de las puntuaciones.

Lo primero que debemos mencionar, es que dicho modelo presentado en [2], permite representar diferentes tipos de "granularidad" en lo que respecta a la relación de la información contextual con las calificaciones. En particular, revisaremos 3 tipos de granularidad.

El primer nivel de granularidad, llamado CAMF-C es el caso más general, donde podemos asumir que cada condición de contexto influencia de manera global todas las puntuaciones, independiente del ítem. Para entender de mejor manera este concepto, veamos el siguiente ejemplo: Imaginemos que tenemos un factor contextual "clima", y uno de sus posibles valores de contexto es "soleado"; entonces podemos inferir que todos los ítems (lugares para visitar) independiente de su naturaleza, recibirán calificaciones más altas.

El segundo nivel, denominado CAMF-CC implica una granularidad media, e introducimos el caso donde una condición de contexto influencia solo a un grupo de ítems organizados por categoría y no a la totalidad de ellos, asumiendo que los ítems se pudiesen agrupar por categorías. Por ejemplo en un sistema recomendador de música, el factor contextual "compañía" y su valor "pareja", puede afectar positiva o negativamente a algunos géneros musicales, como la música romántica, pero a otros como el Rock no.

En un tercer nivel de granularidad llamado CAMF-CI utiliza la premisa de que el contexto influye de manera diferente para cada artículo. Por ejemplo, el tiempo "soleado" podría aumentar la calificación del "Parque de diversiones", pero no la calificación para el "museo de ciencias naturales". Este tipo de granularidad introduce un parámetro por cada

par de condición contextual e ítem. Resulta intuitivo que este nivel representa una granularidad considerablemente más fina, e introduce un número mucho mayor de parámetros al modelo, lo que quedará en evidencia en la siguiente sección.

4.2 Modelo de Factorización de Matrices Consciente del Contexto

El modelo de Factorización de Matrices presentaba a r_{ui}, como la puntuación del usuario u, hacia el ítem i , esto sin hacer ninguna referencia al contexto en el cual se estableció esa calificación por parte del usuario. Pues bien, en esta nueva representación se introduce en su lugar r_{uic} , como la puntuación del usuario u, hacia el item i , en la situación contextual c.

Esta nomenclatura puede denotar más de un factor contextual, más específicamente $r_{uic1\ldots ck}$ denota la puntuación del usuario u, al ítem i, realizada en los contextos $c1, \ldots, ck$, donde el factor contextual cj es igual a 0, 1, 2,…zj y cj = 0 significa que dicho factor contextual es desconocido, mientras que otros valores indican los posibles valores que puede tomar dicho factor contextual (ejemplo 1 = soleado, 2 = nublado, etc…). Si extendemos la ecuación (3) de Factorización de Matrices y sesgos al modelo Consciente del Contexto obtenemos:

$$\hat{r}_{uic1\ldots ck} = q_i{}^T p_u + \mu + b_u + \sum_{j=1}^{k} B_{ijc_j} \quad (4)$$

Donde B_{ijc_j} modela los parámetros de información contextual.

Podemos denotar a **K** $= z_1 + \cdots + z_k$, y donde k representa el número de factores contextuales y z_i el número de posibles valores para el i-ésimo factor contextual.

Dependiendo del tipo de granularidad que queramos representar, la ecuación (4) se torna más o menos compleja y el valor del parámetro K varía.

Si queremos representar la granularidad más fina, es decir el modelo CAMF-CI, se necesitara un parámetro B_{ijc_j} por cada par de condición contextual e ítem. Si el número de ítems es igual a n, tendremos Kn parámetros B_{ijc_j}

En el caso de CAMF-CC (granularidad media) en cambio, se necesitara un parámetro B_{ijc_j} por cada par de condición contextual y categoría de ítem, es decir, si t es el número de categorías (por ejemplo 5 o 10), la cantidad de parámetros B_{ijc_j} es Kt.

Por ultimo para el caso más simple de CAMF-C, se necesita un solo parámetro B_{ijc_j} por cada factor contextual.

Este modelo propuesto [2] podría ampliarse para tener en cuenta las posibles dependencias entre los factores contextuales. Dicho modelo más complejo podría encajar mejor en cierto grupo de datos, pero, si no hay suficientes datos de entrenamiento, la mayor complejidad puede tener un efecto negativo sobre la exactitud de las recomendaciones.

Una vez dicho esto, al igual que en el caso de la Factorización de Matrices básica, se define el procedimiento de aprendizaje como un problema de optimización [2].

$$\min_{q*,p*,b*,B*} \sum_{(u,i)\in R} \left[\left(\mathrm{r}_{\mathrm{uic1...ck}} - {q_i}^T p_u - \mu - b_u - \sum_{j=1}^{k} B_{ijc_j} \right)^2 + \lambda \left(\| q_i \|^2 + \| p_u \|^2 + b_u^2 + \sum_{j=1}^{k} \sum_{c_j=1}^{z_j} B_{ijc_j}^2 \right) \right]$$

(5)

Luego de esto, y al igual que en la Factorización de Matrices básica, se aplica el enfoque del “Descenso por gradiente estocástico” para determinar los factores de ítems y usuarios, por lo que se necesita calcular las derivadas parciales de cada parámetro y obtener de esta forma las actualizaciones correspondientes. Este procedimiento se explica con más detalle en la siguiente sección.

4.3 Cálculo de Factores Contextuales

Para entrenar adecuadamente el modelo de Factorización de Matrices Consciente del Contexto presentado en la sección anterior, se debe utilizar la ecuación (5) para determinar mediante la derivada parcial de cada uno de sus factores, las actualizaciones correspondientes. Fue necesario calcular manualmente todas las derivadas parciales de la ecuación (5).

Es importante recordar que para simplificar el cálculo de dichas derivadas parciales, llamaremos "error" a la diferencia entre la calificación real entregada por el usuario u, al ítem i, realizada en los contextos $c1, \dots, ck$ y la calificación estimada por el sistema para el mismo usuario, ítem y condiciones contextuales:

$$e_{uic1,\dots,ck} = \mathrm{r}_{\mathrm{uic1\dots ck}} - {q_i}^T p_u - \mu - b_u - \sum_{j=1}^{k} B_{ijc_j}$$

Una vez dicho esto, las actualizaciones toman la siguiente forma:

$$q_i \leftarrow q_i + \gamma * (e_{uic1,\dots,ck} * p_u - \lambda * q_i)$$

$$p_u \leftarrow p_u + \gamma * (e_{uic1,\dots,ck} * q_i - \lambda * p_u)$$

$$\mu \leftarrow \mu + \gamma * (e_{uic1,\dots,ck} - \lambda * \mu)$$

$$b_u \leftarrow b_u + \gamma * (e_{uic1,\dots,ck} - \lambda * b_u)$$

$$B_{ijc_j} \leftarrow B_{ijc_j} + \gamma * (e_{uic1,\dots,ck} - \lambda * B_{ijc_j})$$

Para realizar la implementación del Sistema de Recomendación Consciente del Contexto se utilizará el modelo de granularidad más simple.

5. Implementación de un Sistema de Recomendación Consciente del Contexto

Como se mencionó anteriormente se utilizará como base para la implementación del Sistema de Recomendación Consciente del Contexto, la Factorización implementada en la librería Apache Mahout.

Dicha implementación, en lenguaje Java, cuenta con todas las características necesarias para extender la Factorización básica a una factorización Consciente del Contexto. Utiliza además el enfoque de Descenso por Gradiente Estocástico, y cuenta a su vez con una extensa implementación que permite leer los datos de entrada de forma rápida y eficiente. Esta librería está totalmente optimizada para ahorrar espacio de memoria durante la ejecución.

5.1 Implementación

A continuación se presenta el algoritmo del Sistema de Recomendación Consciente del Contexto implementado.

Antes de comenzar la factorización como tal, se obtienen los datos de entrenamiento, los que son cargados en la Matriz de Datos a través de lo que en Mahout se conoce como DataModel. Dicho DataModel es entregado como parámetro al Sistema.

Los datos de entrenamiento provenientes de MovieLens, contienen insertos además de la puntuación del usuario al ítem respectivo, la marca de tiempo en que estas puntuaciones fueron realizadas. Esta marca de tiempo permite determinar variados tipos de contextos (obtención implícita de la información) relacionados justamente con el tiempo, como puede ser estación del año, mes, “semana – fin de semana”, “día – noche”, etc… En este trabajo se estudia el contexto “días de la semana”, es decir, se parte de la premisa que los usuarios puntúan de diferente manera según el día de la semana. Una vez que se cuenta con la información de contexto respectiva podemos iniciar la factorización.

El número de veces que se actualizarán los factores hasta encontrar los más representativos, se conoce como iteraciones. El número de iteraciones debe ser entregado como parámetro al momento de ejecutar el programa, al igual que el número de factores que se utilizarán para realizar la factorización.

Para poder procesar y utilizar información contextual en el proceso de predicción, fue necesario crear una nueva interface llamada “Context”, la cual define los métodos comunes a todo tipo de contexto. Luego se creó una clase abstracta denominada “TimeContext” que extiende la interfaz “Context”, añadiendo métodos propios de contexto de tipo temporal. Para finalmente crear la clase “DaysOfWeekContext” que extiende “TimeContext”, y es la encargada de procesar las marcas de tiempo en los datos de entrada, para convertirlas en el contexto respectivo. Esta implementación permite la posterior adición de otros tipos de contexto, que permitan precisar aún más las estimaciones.

5.1.1 Algoritmo Factorización de Matrices Consciente del Contexto

A continuación se describe el algoritmo implementado:

1. Inicialización de Parámetros:
    a. Regularización (evitar sobreajuste)
    b. Taza de aprendizaje
    c. Factores de usuarios e ítems se inicializan de forma aleatoria
    d. Factores de bias (sesgo) de usuarios e ítems y de Contexto se inicializan en cero
2. Para cada iteración:
    a. Para cada puntuación conocida:
        i. Realizar la predicción utilizando el factor de contexto correspondiente
        ii. Calcular el error entre la predicción y la puntuación real
        iii. Actualizar todos los parámetros (Factores ítems y usuarios, bias ítem y usuarios) utilizando error, regularización y taza de aprendizaje
        iv. Ajustar el factor de contexto correspondiente

En este algoritmo, los factores de usuarios e ítems son actualizados y ajustados dependiendo del factor contextual correspondiente, es decir, si una puntuación fue realizada un día lunes, el ajuste de los factores de usuarios e ítems, se estima utilizando el valor del factor contextual del día correspondiente. En el caso de “días de la semana”, el vector que contiene los valores contextuales está conformado por 7 factores, cada uno de ellos representa un día de la semana y son utilizados y ajustados cada uno de ellos en forma independiente dependiendo del día en que se realizó la puntuación del ítem por parte del usuario.

Los parámetros γ y λ, deben ser ajustados de forma manual solo en base al desarrollo de pruebas al algoritmo, ya que no existe una forma exacta de determinar dichos valores.

Una vez que se complete el total de iteraciones, los factores de usuarios e ítems estarán adecuadamente ajustados y deberían permitir una mejor estimación de las puntuaciones

desconocidas de la matriz de datos, en caso que dichas puntuaciones efectivamente dependan del contexto temporal en estudio (día de la semana).

5.2 Diagrama de Clases

Se presentan 2 diagramas de clases, el primero representa las nuevas clases creadas para añadir información contextual al sistema. El segundo diagrama hace referencia a las clases que realizan el proceso de factorización, su clase abstracta y respectiva interfaz.

5.2.1 Diagrama de clases Contexto

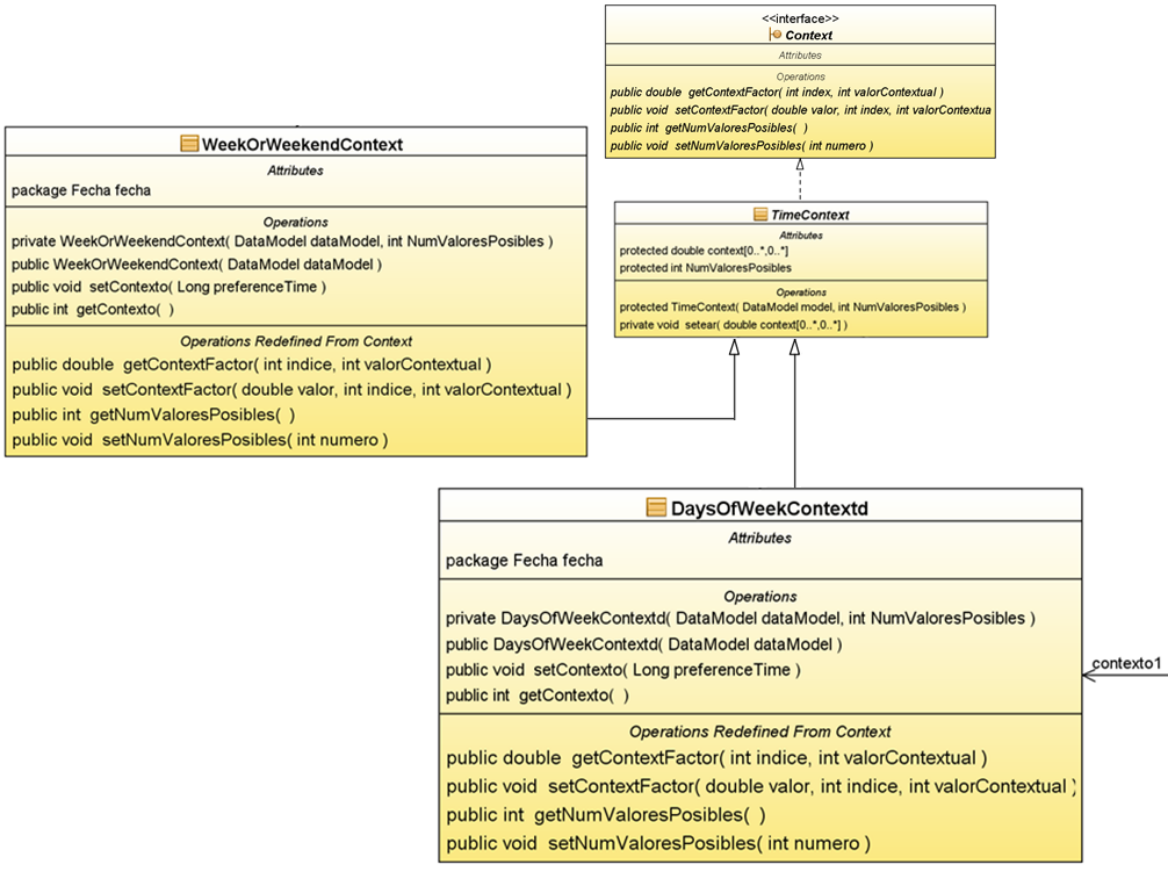


Figura 2: Diagrama de Clases de Contexto

Esta implementación, permite idear y agregar nuevos tipos de contexto a la librería.

5.2.2 Diagrama de clases Factorización

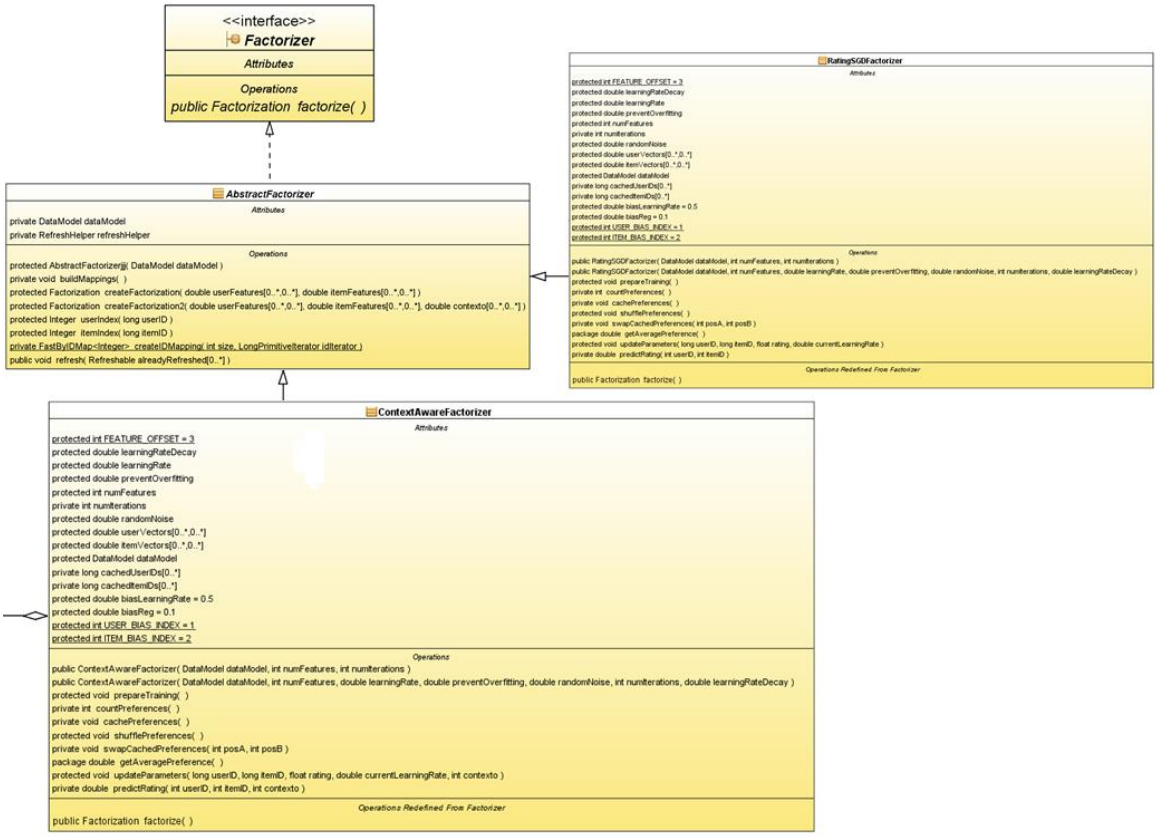


Figura 3: Diagrama de Clases de Factorización

La Interface Factorizer, la clase abstracta Factorizer y la clase RatingSGDFactorizer (Factorización de Matrices pura) son propias de la librería Mahout, la clase ContextAwareFactorizer es la implementación del algoritmo presentado en la sección anterior.

6. Pruebas de la Implementación

Para determinar si efectivamente el algoritmo de Recomendación Consciente del Contexto implementado realiza mejores predicciones que un sistema basado en Factorización de Matrices donde no se tiene en cuenta el contexto en el cual se realizan las recomendaciones, se realizará una serie de pruebas con distintos set de datos de entrenamiento, distinto número de iteraciones y distinto número de factores.

Raíz del Error Medio Cuadrático y Error Absoluto Medio o RMSE y MAE respectivamente por sus siglas en inglés (“Root Mean Square Error”; “Mean Absolute Error”) [5] serán los criterios utilizados para determinar la efectividad de la implementación realizada.

La librería Mahout provee un completo set de herramientas de evaluación para sus sistemas de recomendación, los que se utilizarán para obtener los valores de las métricas antes mencionadas. Para ejecutar dichas herramientas y obtener los valores de RMSE y MAE solo se debe indicar el nombre del algoritmo a evaluar y el porcentaje de datos que se usará para entrenamiento, dejando el porcentaje restante de datos como datos de prueba, para de esta forma comparar si las recomendaciones estimadas por el sistema coinciden con las puntuaciones reales del set de datos.

6.1 Datos de Prueba

Como se mencionó en el capítulo 3, GroupLens provee un completo set de datos de puntuaciones reales realizas a través de su sistema MovieLens, un sistema recomendador de películas.

Los set de datos de MovieLens se recogieron durante diversos períodos de tiempo, dependiendo del tamaño del conjunto. Es posible descargar diferentes conjuntos de datos, estos son:

- MovieLens 100k (Recogidos:19 Septiembre de1997 al 22 Abril de 1998)
  - ✓ 100.000 ratings de 1000 usuarios a 1700 películas.

- MovieLens 1M (Recogidos durante todo el año 2000)
  - ✓ 1 millón de ratings de 6000 usuarios a 4000 películas.

- MovieLens 10M
  - ✓ 10 millones de ratings de 72.000 usuarios a 10.000 películas.
  - ✓ Estos datos fueron recogidos desde el inicio de MovieLens.

Los datos provistos por MovieLens se presentan en el siguiente formato (separado por tabulación):

Usuario Ítem Rating Marca de Tiempo

El primer paso a realizar, es transformar dicho formato, ya que Mahout solo acepta datos de entrada en formato .csv (separado por comas):

Usuario, Ítem, Ratings, Marca de tiempo

Para ello, se ha creado un pequeño programa que se encarga de dicha transformación, y nos devuelve los datos en el formato adecuado para utilizar con Mahout.

6.2 Ejecución de las Pruebas

Por el gran volumen de datos a utilizar en las pruebas (MovieLens 10M), éstas se realizan en un servidor especial de mucha mayor potencia que un ordenador normal de escritorio o notebook, dicho servidor cuenta con 8 núcleos virtuales (4 reales) y 32 GB de memoria RAM.

Se crea entonces un programa en lenguaje Java, que recibe como parámetros el nombre del archivo de prueba, la cantidad de iteraciones y el número de factores. Este programa utiliza la librería Mahout y en particular la Factorización Consciente del Contexto implementada y entrega como salida el RMSE y MAE obtenidos durante la ejecución de la factorización.

A partir de dicho programa, obtenemos a través de su compilación, el archivo .jar respectivo, el cual podremos ejecutar desde cualquier consola de comandos utilizando la siguiente sintaxis.

java –jar “nombre archivo .jar” “nombre archivo de prueba” “n° factores” “n° iteraciones”

Ejemplo: java –jar FMCC.jar MovieLens10M.csv 30 60

Las pruebas se realizarán de la siguiente forma:

- 30 o 60 Factores, 10 a 60 iteraciones.

Esto es un total de 12 combinaciones. Cada una de estas combinaciones se prueba con 100 mil, 1 millón y 10 millones de datos. Esto genera un total de 36 pruebas.

Cada una de esas 36 pruebas se hará 5 veces, esto básicamente por la forma aleatoria de inicializar los factores de usuarios e ítems, lo que provoca pequeñas variaciones entre una ejecución y otra. En total se realizan 180 ejecuciones del programa.

Esto es válido tanto para Factorización de Matrices básica (prueba base) como para Factorización de Matrices Consciente del Contexto.

7. Resultados de las Pruebas

A continuación se presenta un completo resumen de las pruebas realizadas tanto al algoritmo de Recomendación basado en Factorización de Matrices Consciente del Contexto, como al algoritmo basado en Factorización de Matrices pura, el cual servirá como línea de base (baseline) para determinar la mejora obtenida por la explotación de información contexutal.

Posteriormente se presentan y discuten los resultados obtenidos por ambos algoritmos..

Se presenta en primera instancia los resultados según el criterio de la Raíz del Error Medio Cuadrático o RMSE para posteriormente presentar los resultados según el segundo criterio que es el de Error Absoluto Medio o MAE.

7.1 Resultados Obtenidos

Se presentan los resultados divididos por set de datos. Baseline hace referencia a la Factorización de Matrices pura y FMCC a la Factorización de Matrices Consciente del Contexto.

7.1.1 Criterio RMSE

MovieLens 100K / RMSE			
Factores	Iteraciones	Baseline	FMCC
30	10	0,915728242	0,763478897
30	20	0,908422878	0,57314225
30	30	0,884106133	0,542232489
30	40	0,863290435	0,527978625
30	50	0,831593941	0,527855185
30	60	0,8029156	0,526955193
60	10	0,925658927	0,713051818
60	20	0,899551816	0,453658512
60	30	0,880077247	0,410003677
60	40	0,853684576	0,398428386
60	50	0,820577275	0,393996169
60	60	0,786176106	0,392515143

Tabla 1: MovieLens 100K / RMSE

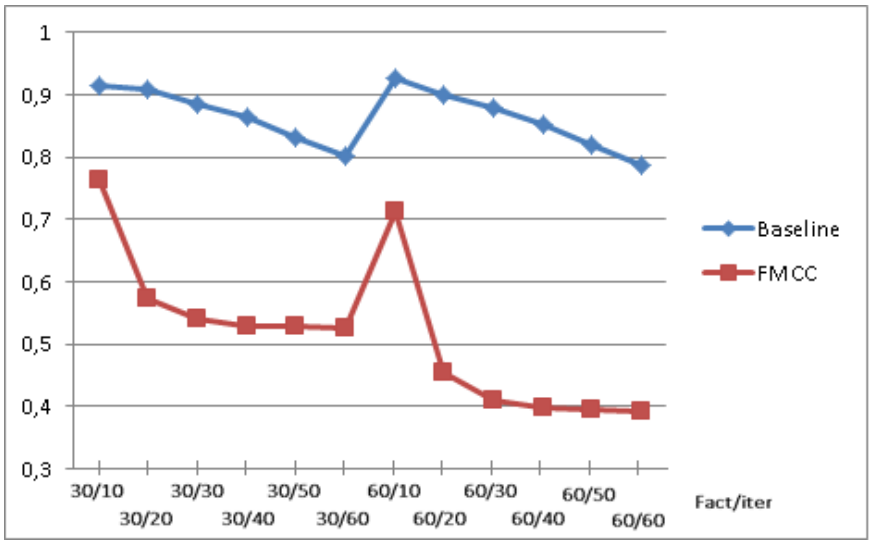


Figura 4: MovieLens 100k / RMSE

MovieLens 1M / RMSE			
Factores	Iteraciones	Baseline	FMCC
30	10	0,89860652	0,743727887
30	20	0,875668406	0,700472332
30	30	0,860600502	0,695744269
30	40	0,850952342	0,696158879
30	50	0,845149244	0,696347603
30	60	0,84106797	0,696110204
60	10	0,896275106	0,699746017
60	20	0,873340798	0,617321116
60	30	0,859168187	0,606803619
60	40	0,849932759	0,605013151
60	50	0,842404642	0,605128795
60	60	0,839912065	0,604709641

Tabla 2: MovieLens 1M / RMSE

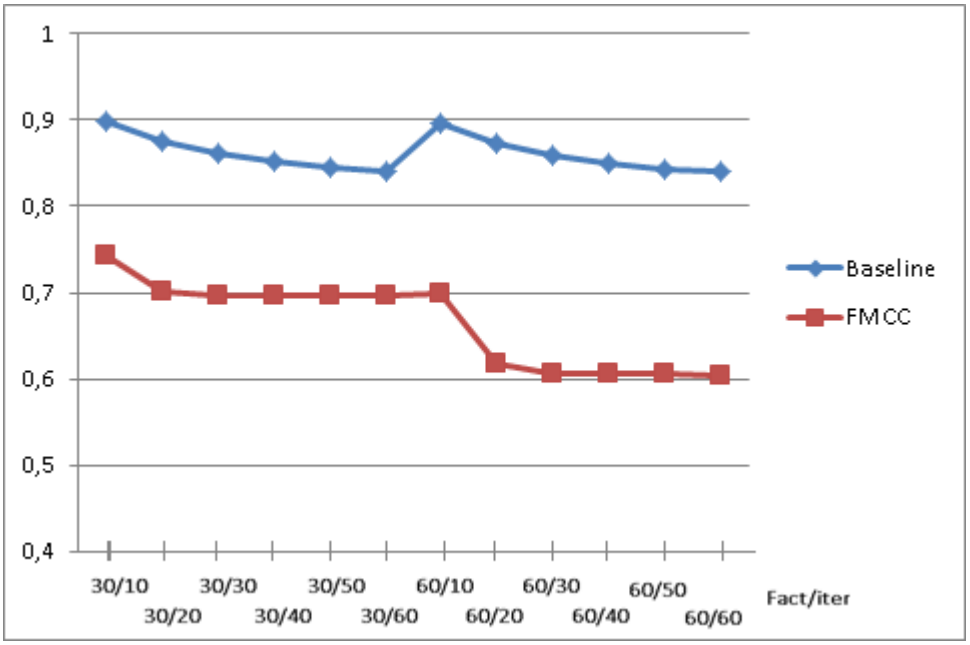


Figura 5: MovieLens 1M / RMSE

MovieLens 10M / RMSE			
Factores	Iteraciones	Baseline	FMCC
30	10	0,84199828	0,732314477
30	20	0,826939813	0,714099488
30	30	0,818235071	0,712361728
30	40	0,811712906	0,713788994
30	50	0,806925753	0,715548235
30	60	0,805012039	0,71645594
60	10	0,841270234	0,71387569
60	20	0,825888986	0,719921475
60	30	0,81708101	0,669718235
60	40	0,810986953	0,669305402
60	50	0,806902703	0,669817465
60	60	0,804103784	0,670567481

Tabla 3: MovieLens 10M / RMSE

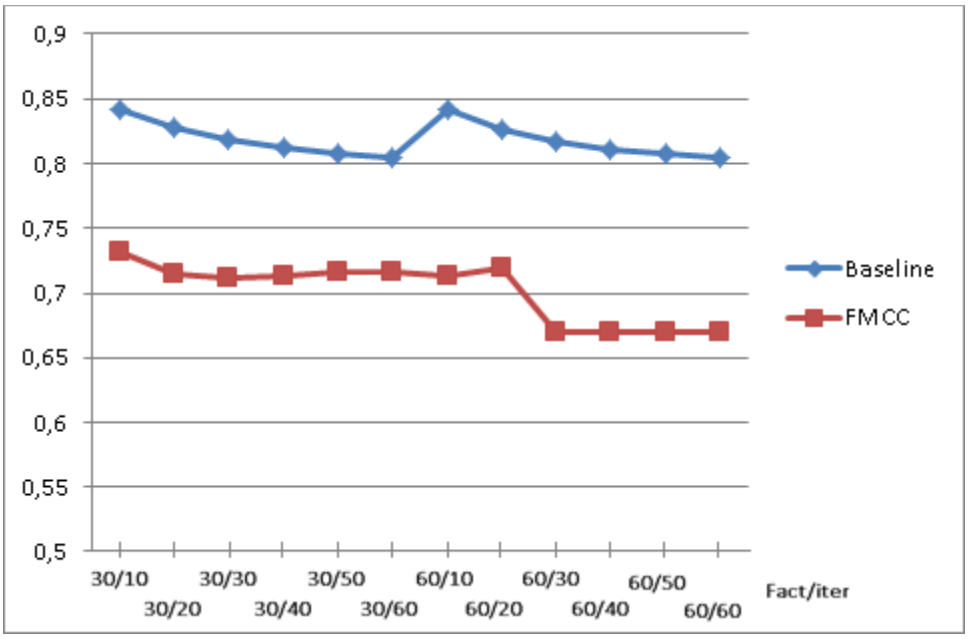


Figura 6: MovieLens 10M / RMSE

7.1.2 Criterio MAE

MovieLens 100K / MAE			
Factores	Iteraciones	Baseline	FMCC
30	10	0,729370299	0,603516424
30	20	0,717532434	0,45421492
30	30	0,701441267	0,424947775
30	40	0,682627541	0,418022515
30	50	0,660127678	0,414471287
30	60	0,638865705	0,412808975
60	10	0,729265245	0,566374455
60	20	0,714933857	0,35781829
60	30	0,69984346	0,321720235
60	40	0,675320126	0,311294264
60	50	0,650935906	0,310392878
60	60	0,625313898	0,3098759

Tabla 4: MovieLens 100K / MAE

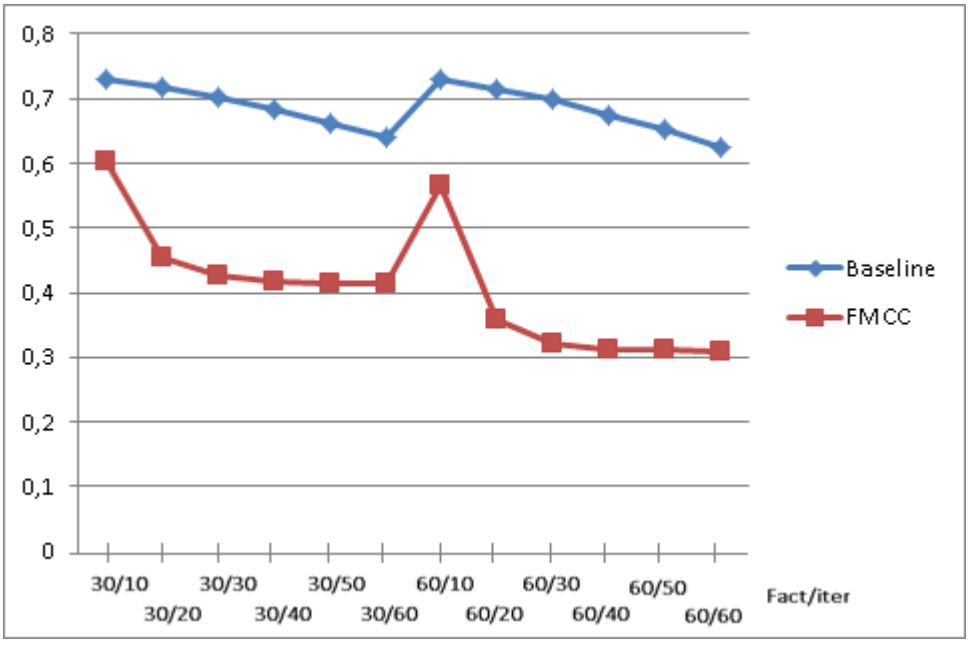


Figura 7: MovieLens 100K / MAE

MovieLens 1M / MAE			
Factores	Iteraciones	Baseline	FMCC
30	10	0,708939046	0,589413013
30	20	0,689659602	0,555028
30	30	0,678754032	0,55070414
30	40	0,673541369	0,550294134
30	50	0,669022057	0,550758796
30	60	0,665300988	0,550694915
60	10	0,708444719	0,55638824
60	20	0,689563577	0,486314564
60	30	0,678597014	0,478717056
60	40	0,67111206	0,475735318
60	50	0,666667522	0,476333436
60	60	0,663892549	0,475309098

Tabla 5: MovieLens 1M / MAE

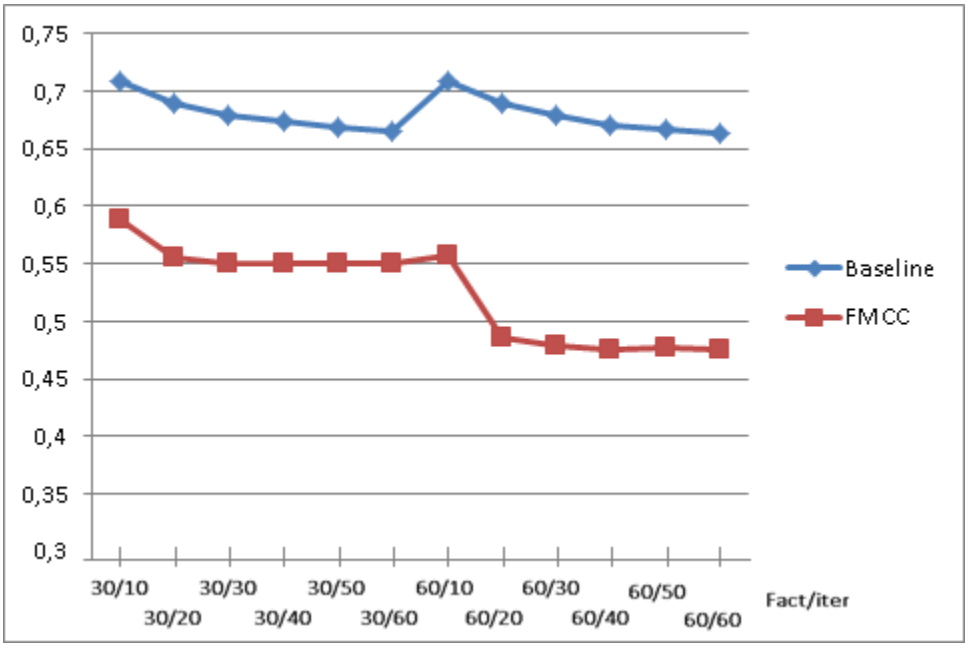


Figura 8: MovieLens 1M / MAE

MovieLens 10M / MAE			
Factores	Iteraciones	Baseline	FMCC
30	10	0,650964208	0,569626808
30	20	0,63860864	0,555546635
30	30	0,632222219	0,555483063
30	40	0,627133091	0,556319631
30	50	0,624033328	0,55622724
30	60	0,62234622	0,556805561
60	10	0,649648507	0,556749676
60	20	0,637515843	0,56107081
60	30	0,63148328	0,520675265
60	40	0,626135021	0,520048155
60	50	0,623576557	0,520652199
60	60	0,621401244	0,520390183

Tabla 6: MovieLens 10M / MAE

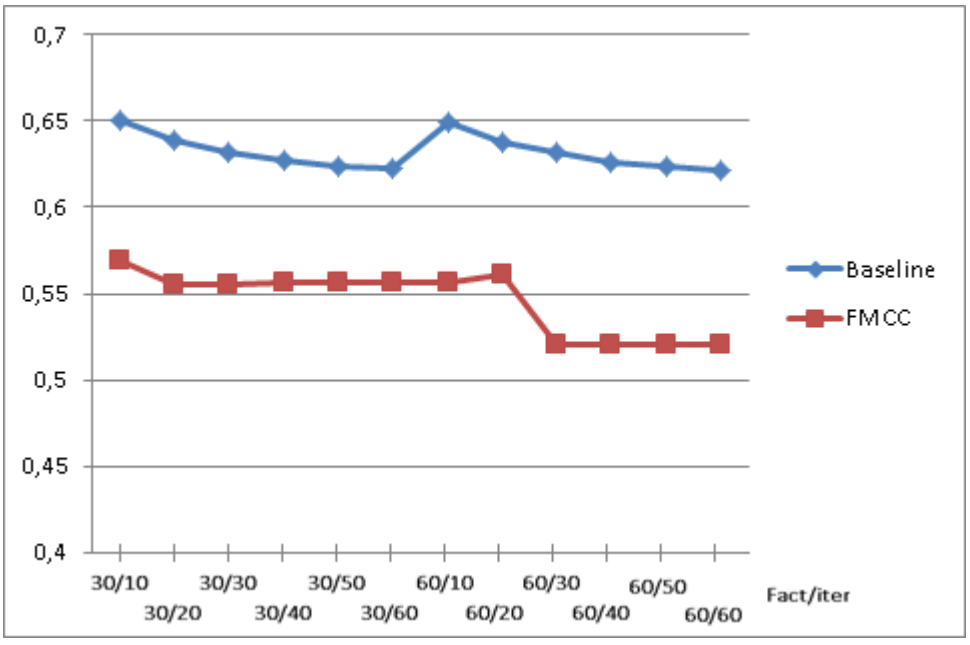


Figura 9: MovieLens 10M / MAE

7.2 Conclusiones de los resultados obtenidos

Como se observa en tablas y gráficos de la sección 7.1, en todas y cada una de las pruebas, la implementación de Factorización de Matrices Consciente del Contexto, resultó ser más precisa en la estimación de calificaciones, llegando en algunos casos a ser hasta un 40% más efectiva que la Factorización pura.

Se observa que las recomendaciones Conscientes del Contexto, son en promedio un 25% más precisas que las recomendaciones realizadas sin utilizar información contextual.

En base a estos resultados, se puede concluir que la inclusión de información contextual repercute positivamente en la obtención de recomendaciones cada vez más precisas. Tomando en cuenta que la implementación realizada en este trabajo, solo representa la granularidad más gruesa, es decir, donde un contexto afecta de manera igualitaria a todos los ítems, lleva a pensar que con una granularidad más fina, los resultados podrían ser potencialmente mucho más favorables que los obtenidos durante esta investigación.

Sin embargo, es importante recalcar que en la obtención de estos resultados, es muy importante el adecuado ajuste de los parámetros γ y λ, regularización y taza de aprendizaje respectivamente, ya que un cuidadoso ajuste de dichos parámetros es un elemento fundamental en la obtención de mejores recomendaciones. No existe una forma exacta de determinar los valores de dichos parámetros, y si bien se recomiendan valores cercanos al 0,1 para ambos, estos deben ser ajustados manualmente en base a prueba y error, hasta obtener las mejores estimaciones.

Vale destacar el hecho de que estas conclusiones son válidas para el dominio de “Recomendación de Películas”, que intuitivamente puede ser bastante parecido a un sistema recomendador de música u otro tipo de contenido multimedia, pero no necesariamente para otros dominios como lo pueden ser la “Recomendación de paquetes vacacionales” o sistemas de similares características.

8. Conclusiones del Proyecto

Los Sistemas de Recomendación han sufrido un significativo progreso en la última década, y los Sistemas de Recomendación Conscientes del Contexto son la prueba de aquello.

En este proyecto, se estudió, modeló e implementó un algoritmo de Recomendación Consciente del Contexto basado en la técnica de Factorización de Matrices propuesto recientemente en la literatura. Dicho algoritmo fue arduamente probado, obteniendo notables resultados en comparación a algoritmos basados en una Factorización de Matrices pura.

En primer término, se revisó en profundidad diferentes algoritmos de recomendación, ya sean los denominados Basados en contenido o Filtrado Colaborativo, además de un estudio acabado de la técnica de Factorización de Matrices, así como las librerías de software que implementan dicha técnica, para luego proseguir con el estudio de algoritmos de recomendación conscientes del contexto.

Una vez terminado el trabajo investigativo, se procedió a realizar la extensión de la librería de algoritmos de recomendación Mahout, por medio de la implementación del algoritmo de recomendación consciente del contexto basado en factorización de matrices propuesto por Baltrunas et al. [2].

El potencial de dichos sistemas es altísimo, y las estimaciones de los ratings o puntuaciones son cada vez más precisas. La información del contexto específico en el cual se realizan las puntuaciones de contenido por parte de los usuarios, resultó ser de

gran importancia en el proceso de predicción de puntuaciones, logrando en algunos casos hasta un 40% más de precisión para la posterior recomendación de dicho contenido, en este caso películas.

Cabe señalar, eso sí, que se requiere de un cuidadoso ajuste de los parámetros de regularización y taza de aprendizaje (γ y λ), ya que dichos parámetros influyen de forma directa en la obtención de los factores latentes, y sus valores varían dependiendo de la cantidad de factores tanto de ítems y usuarios, como de factores contextuales.

Dichas conclusiones fueron obtenidas a través de un exhaustivo proceso de prueba del algoritmo implementado, verificando y validando su correcto funcionamiento.

Una vez validado el correcto funcionamiento del algoritmo implementado, se realizó un proceso de comparación con respecto a un algoritmo de Factorización No consciente del contexto, dicho algoritmo fue extraído de la misma librería Mahout. Dicha comparación permitió obtener los resultados antes mencionados, junto con la convicción de la importancia del contexto en el proceso de recomendación.

La implementación realizada en este trabajo, puede ser aplicada a cualquier sistema que necesite recomendaciones precisas de ítems para usuarios, con la única condición, de que los datos de entrada, deben proveer al menos un tipo de información de contexto temporal para ser procesada.

Referencias

[1] G. Adomavicius and a. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[2] L. Baltrunas, L. Bernd, and F. Ricci, “Matrix factorization techniques for context aware recommendation,” *RecSys ’11 Proc. fifth ACM Conf. Recomm. Syst.*, pp. 301–304, 2011.

[3] R. Burke, “Hybrid Recommender Systems : Survey and Experiments †,” pp. 1–29.

[4] Association for Computing Machinery, “ACM RecSys.” [Online]. Available: http://recsys.acm.org/. [Accessed: 20-Dec-2014].

[5] C. Res, C. J. Willmott, and K. Matsuura, “Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance,” vol. 30, pp. 79–82, 2005.

[6] C. D. Manning, P. Raghavan, and H. Schutze, “Boolean Retrieval,” *Introd. to Inf. Retr.*, pp. 1–18, 2009.

[7] P. W. Foltz and S. T. Dumais, “Personalized information delivery: an analysis of information filtering methods,” *Commun. ACM*, vol. 35, no. 12, pp. 51–60, 1992.

[8] M. J. Pazzani and D. Billsus, “Content-Based Recommendation Systems,” pp. 325–341, 2007.

[9] J. S. C. L and R. F. E, “Modelación de sistemas de recomendación aplicando redes neuronales artificiales Recommendation systems modeling applied artificial neural networks Resumen : Este artículo desarrolla y describe un modelo para un sistema reco-,” no. 2, pp. 45–56, 2013.

[10] L. M. de Campos, J. M. Férnandez-Luna, and J. F. Huete, “A Bayesian Network approach to Hybryd Recommending Systems,” *Dpto. Ciencias la Comput. e I.A.*

[11] J. Ben Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative Filtering Recommender Systems,” pp. 291–324.

[12] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information Tapestry .,” vol. 61, no. 10, pp. 1–10, 1992.

[13] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens : An Open Architecture for Collaborative Filtering of Netnews,” 1994.

[14] Y. Seroussi, F. Bohnert, and I. Zukerman, “Personalised rating prediction for new users using latent factor models,” *Proc. 22nd ACM Conf. Hypertext hypermedia - HT ’11*, p. 47, 2011.

[15] M. Gori and A. Pucci, “ItemRank : A Random-Walk Based Scoring Algorithm for Recommender Engines,” pp. 2766–2771, 2002.

[16] B. Chen, “Latent Factor Models for Web Recommender Systems.”

[17] P. O. Hoyer, “Non-negative Matrix Factorization with Sparseness Constraints,” vol. 5, pp. 1457–1469, 2004.

[18] R. L. Burden and J. D. Faires, “Descomposición en Valores singulares (SVD),” pp. 1–12, 2010.

[19] A. Pazzani and M. J, “A Framework for Collaborative, Content-Based and Demographic Filtering,” *Artif. Intell. Rev.*, vol. 13, pp. 5–6, 1999.

[20] Y. Koren, R. Bell, and C. Volinsky, “Matrix Factorization Techniques For Recommender Systems,” pp. 42–49, 2009.

[21] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl, “Application of Dimensionality Reduction in Recommender System -- A Case Study,” 1998.

[22] D. M. Hawkins, “The problem of overfitting.,” *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 1, pp. 1–12, 2004.

[23] “MyMediaLite Recommender System Library.” [Online]. Available: http://www.mymedialite.net/index.html. [Accessed: 03-Nov-2014].

[24] S. Rendle, C. Freudenthaler, and L. Schmidt-thieme, “MyMediaLite : A Free Recommender System Library,” 2011.

[25] “LensKit Recommender Toolkit.” [Online]. Available: http://lenskit.org/. [Accessed: 28-Oct-2014].

[26] S. Funk, “Simon Funk on the Stochastic Gradient Descent algorithm,” 2006. [Online]. Available: http://sifter.org/~simon/journal/20061211.html.

[27] Apache, “Apache Mahout.” [Online]. Available: http://mahout.apache.org/. [Accessed: 20-Dec-2014].

[28] S. Schelter and S. Owen, “Collaborative Filtering with Apache Mahout,” vol. i, 2012.

[29] G. Adomavicius and A. Tuzhilin, “Context-aware recommender systems,” *Proc. 2008 ACM Conf. Recomm. Syst. - RecSys ’08*, p. 335, 2008.