Universidad de Granada

Depto. de Ciencias de la Computación e Inteligencia Artificial

Modelos de Recomendación Basados en Redes Bayesianas

MEMORIA QUE PRESENTA

Miguel Ángel Rueda Morales

PARA OPTAR AL GRADO DE DOCTOR EN INFORMÁTICA

DIRECTOR

Prof. Dr. Juan Francisco Huete Guadix

Granada, Septiembre 2011

Editor: Editorial de la Universidad de Granada
Autor: Miguel Ángel Rueda Morales
D.L.: GR 1144-2012
ISBN: 978-84-695-1065-0

La memoria titulada “Modelos de Recomendación Basados en Redes Bayesianas“, que presenta D. Miguel Ángel Rueda Morales para optar al grado de doctor, ha sido realizada dentro del programa de doctorado ”Modelos Probabilísticos para la Inteligencia Artificial y la Minería de Datos“ del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección del Profesor Doctor D. Juan Francisco Huete Guadix.

Granada, Septiembre 2011.

Miguel Ángel Rueda Morales	Juan F. Huete Guadix
El alumno	V.B. Director

Agradecimientos

Gracias a mi esposa, María Isabel, por todo su amor y paciencia.
Gracias a mis padres, Manuel y Ángeles, y a mis hermanos, Manuel, Antonio y María Teresa, por estar ahí, por su confianza y su apoyo.
Gracias al resto de mi familia, directa y política, por su interés y cariño.
Gracias a mi director de tesis y amigo, Prof. Dr. Juan Fco. Huete, por toda su ayuda, sabiduría, insistencia y, sobre todo, paciencia.
Gracias al Prof. Dr. Luis Miguel De Campos y al Prof. Dr. Juan Manuel Fernández, por el apoyo dado durante estos años.
Gracias a mis compañeros del grupo, ya doctores, Carlos Martín y Alfonso E. Romero por toda la ayuda y amistad que me han dado.
Gracias a mis compañeros de fatigas del "Frikitorio" y alrededores, por todos esos buenos ratos que hemos pasado estos años: Carlos, Javi, Borja, Edu, Antonio, Alberto, Pedro, Fernando García, Fernando Bobillo, Jesús, Sergio, Mariló, Marta, Julián, Jose Manuel, Victor, Juanra, Andrés, Cora y Nacho.
Gracias a mis amigos por servirme de "escape" cientos de veces, por quererme y hacerme sentir tan afortunado por tenerlos a mi lado: Pedro, Yago, Rub, Alex, Inma, Migue, Pepa, Tobas, Sole, Chus, Alberto, Tinx, Oscar, Gonzalo, Felipe, Juanlu y muchos más que se merecen estar en esta lista que, obviamente, está incompleta.
Gracias a mis compañeros y amigos de la OTRI, por su confianza y ánimo.
Sé que falta gente en estos agradecimientos, por eso quiero dar las gracias a todos los que, de alguna forma, han contribuido al feliz desarrollo de esta memoria. GRACIAS.

Índice general

Capítulo 1

Introducción

Los *Sistemas de Recomendación* (en adelante SR) son técnicas y herramientas software que ofrecen sugerencias sobre productos (o acciones) en un dominio concreto que el usuario considera interesante [1]. Las sugerencias están relacionadas con procesos de toma de decisión, tal como qué canción escuchar, qué película ver o qué viaje realizar. A lo largo de la literatura podemos encontrar distintos métodos para predecir las preferencias que tienen los usuarios respecto a los productos, normalmete expresadas en forma de votos: KNN, Naive Bayes, árboles de decisión, redes neuronales y modelos probabilísticos entre otros. El trabajo que presentamos en esta memoria está basado en los modelos probabilísticos, concretamente en Redes Bayesianas [2, 3, 4], mediante las que hemos creado modelos que nos permiten obtener buenas recomendaciones.

Este trabajo tiene su origen dentro del proyecto TIN2005-02516 bajo el título de "Aplicación de Modelos Gráficos Probabilísticos a la Recuperación de Información Estructurada". El objetivo principal del proyecto era el desarrollo de un Sistema de Recuperación de Información Estructurada basado en el formalismo de los Modelos Gráficos Probabilísticos. Nuestro trabajo se enmarca en el objetivo número 5 de la memoria del proyecto: "Desarrollo de Sistemas de Recomendación", que trata sobre la adaptación de las soluciones encontradas en Recuperación de Información estructurada a los dominios de aplicación concretos de los Sistemas de Recomendación (librerías, viajes, inmobiliarias, etc.). La tarea clave será la de modelar las relaciones entre usuarios y productos (mediante modelos basados en contenido) y los usuarios entre sí (modelos colaborativos). Para ello, usaremos las Redes Bayesianas, que nos permitirán combinar una representación cualitativa de esas relaciones (representando explícitamente las relaciones de dependencia e independencia en una estructura gráfica) con una representación cuantitativa, que mide la fuerza de las relaciones mediante un conjunto de distribuciones de probabilidad.

Las distintas problemáticas de los SR que se abordarán en esta memoria se pueden clasificar en:

- Sistemas Basados en Contenido: El objetivo de estos sistemas es el de recomendar al usuario productos que son similares a aquellos que le gustó en el pasado. Enmarcado en esta problemática se propondrá un modelo capaz de tener en cuenta las descripciones de un producto.

- Sistemas Colaborativos: Estos sistemas recomiendan aquellos productos que han sido votados positivamente por usuarios similares. En este marco, se diseñará un modelo que permita representar el conjunto de usuarios similares a un usuario mediante una Red Bayesiana, diseñando mecanismos eficientes de almacenamiento y propagación de las probabilidades implicadas en el proceso. Además, se abordará uno de los problemas que presentan los sistemas colaborativos como es la dispersión de la matriz de votos, que hace que sea difícil recomendar aquellos productos que no han sido votados previamente. Para ello, se diseñarán mecanismos que alivien esta problemática mediante el uso de información de segunda mano. Por otra parte, se diseñará un nuevo modelo cuyo objetivo será encontrar el vecindario que con mayor verosimilitud pueda generar los votos dados por el usuario.

- Sistemas Híbridos: Se desarrollará un modelo que sea capaz de integrar la información de contenido y la colaborativa. Esta integración se conseguirá permitiendo las conexiones entre la Red Bayesiana que representa el contenido, con la red que representa la componente colaborativa.

- Recomendación para grupos: Se estudiarán los problemas de la recomendación a grupos junto con los problemas relacionados con los mecanismos de agregación de la incertidumbre. Para ello desarrollaremos modelos que permitan manejar criterios como el mayoritario, el máximo o mínimo.

Los resultados obtenidos, que presentamos en este trabajo mediante la fórmula de compendio de artículos, se concretan en las siguientes cuatro publicaciones:

- Managing Uncertainty in Group Recommending Processes - *Manejando la incertidumbre en el proceso de la Recomendación en Grupo.* User Modeling and User-Adapted Interaction. 2008. JCR: 3.074

- Combining Content-based and Collaborative Recommendations: a Hybrid approach based on Bayesian networks - *Combinando Recomendaciones Colaborativas y Basadas en Contenido: una aproximación Híbrida basada en redes Bayesianas.* International Journal of Approximate Reasoning. 2010. JCR: 1.679

- Using second-hand information in Collaborative Recommender Systems - *Usando Información de Segunda Mano en Sistemas de Recomendación Colaborativos.* Soft Computing. 2009. JCR: 1.512

- Using past-predictions accuracy in Recommender Systems - *Usando las predicciones precisas del pasado en Sistemas de Recomendación.* Information Sciences. 2011 (sometido). JCR: 2.833

El contenido de esta memoria está estructurado de la forma: en el Capítulo 2 realizaremos una introducción sobre los SR, mostrando sus características y diferenciando los tipos de sistemas que podemos encontrarnos. En el Capítulo 3 expondremos unas nociones básicas sobre Redes Bayesianas, haciendo hincapié en los modelos canónicos [5], básicos para nuestro trabajo. En el Capítulo 4 presentaremos las contribuciones de los trabajos que componen esta memoria, haciendo un breve resumen del planteamiento del problema, la solución propuesta y los resultados obtenidos. A continuación, en el Capítulo 5, mostraremos los artículos completos y, para terminar, en el Capítulo 6 presentaremos un listado de aquellas publicaciones realizadas a lo largo del periodo de trabajo y que han servido de complemento para los trabajos que aquí presentamos.

Capítulo 2

Sistemas de Recomendación

Internet se ha convertido en una herramienta indispensable para nuestro día a día. Hace no muchos años, a la hora de buscar información sobre algún tema, había que recurrir a enciclopedias, revistas especializadas impresas, bibliotecas públicas, etc. Desde el auge de Internet, el acceso a la información es mucho más rápido y sencillo. No obstante, lo que parecía ser una ventaja, se ha convertido en un gran problema ya que existe una gran cantidad de información sobre tantos y tantos productos. Para intentar paliar este problema se han popularizado herramientas automáticas que ayudan a los usuarios a encontrar productos que coincidan con sus gustos. Los SR surgen por este motivo. En términos generales, un SR ofrece sugerencias sobre productos (o acciones) en un dominio concreto que el usuario considera interesante [1]. Normalmente denominamos producto o ítem al término general que se usa para indicar qué es lo que el sistema recomienda a los usuarios (canciones, películas, noticias, etc.).

Existen una serie de problemas bien identificados que un SR puede ayudar a resolver [6]:

- Encontrar buenos items: Recomendar por medio de una lista ordenada aquellos items que se consideran buenos indicando cuánto les gustaría (por ejemplo mediante una escala de 1 a 5 estrellas).

- Encontrar todos los items buenos: Recomendar todos los items que pueden satisfacer las necesidades de algún usuario. En estos casos es insuficiente encontrar sólo algunos items buenos, han de ser todos. Podemos encontrar ejemplos de este tipo cuando el número total de items es relativamente pequeño o bien el SR tiene aplicaciones críticas (medicina, finanzas).

- Anotación en contexto: Dado un contexto existente, como por ejemplo una lista de items, enfatizar algunos de ellos dependiendo de las preferencias del usuario, superponiendo la información de la recomendación a la información listada.

- Recomendar una secuencia: En vez de centrarse en la generación de una recomendación individual, la idea es recomendar una secuencia de items que agrade en su conjunto. Ejemplos clásicos: compilaciones de música o series de TV.

- Recomendar un paquete: Sugerir un conjunto de items que encajan bien juntos. Un ejemplo muy claro es la planificación de unas vacaciones, que puede estar compuesto por varios destinos, varias atracciones y varios hoteles para pernoctar. Desde el punto de vista del usuario, todas estas alternativas pueden considerarse como un único destino vacacional.

- Sólo navegar: En esta tarea, el usuario navega por un catálogo sin la intención inminente de comprar un producto. La tarea del recomendador es ayudar al usuario a encontrar los items en los que probablemente esté interesado en esa sesión de navegación.

- Encontrar recomendaciones creíbles: Algunos usuarios no se creen los resultados propuestos por los SR de primeras. Lo que hacen es jugar con ellos para ver si son buenos realizando recomendaciones (probando si les recomiendan sus películas favoritas, por ejemplo).

- Mejorar el perfil: El usuario ha de proporcionar información al SR sobre lo que le gusta y lo que no. Ésta es una tarea fundamental que es estrictamente necesaria apara proporcionar recomendaciones personalizadas.

- Auto-expresarse: A algunos usuarios no les importan tanto las recomendaciones como que se les permita contribuir con sus votos y expresar sus opiniones y creencias.

- Ayudar a otros: Algunos usuarios son felices contribuyendo con información (sus votos), ya que creen que la comunidad se verá beneficiada con su aportación. Ésta podría ser una gran motivación para introducir información en un SR que no se usa de forma habitual. Un ejemplo claro lo podemos encontrar en un sistema de recomendación de coches, en el que, si un usuario introduce su opinión sobre un coche, lo más probable es sea útil para otros usuarios, ya que él no volverá a usarlo hasta la siguiente vez que vaya a comprarse un coche nuevo.

- Influenciar a otros: En sistemas de recomendación basados en Web existen usuarios cuya finalidad principal es influenciar explícitamente a otros usuarios para comprar determinados productos. Por ejemplo votar alto (y muchas veces) un producto para que esté bien valorado y que le sea recomendado a otros usuarios.

Como podemos deducir de todos estos puntos, el rol de un SR en un sistema de información puede ser muy diverso, como también lo son las técnicas mediante las que los SR indentifican las recomendaciones correctas. Existen distintas taxonomías para clasificar los SR, pero la más extendida es la proporcionada por Burke [7], que distingue entre las siguientes seis clases:

- Basados en contenido: El sistema recomienda items similares a aquellos que al usuario le han gustado en el pasado. La similitud entre items se calcula basándose en las características que poseen los items comparados. Por ejemplo, si un usuario ha votado positivamente a una película que pertenece al género comedia, entonces el sistema puede aprender a recomendar otras películas de este género. En la sección 2.1 veremos en detalle este tipo de SR.

- Filtrado colaborativo: El sistema recomenda al usuario activo los items que a otros usuarios con gustos similares les han gustado en el pasado. La similitud entre usuarios se basa en el historial de voto. Es la técnica más popular y ampliamente implementada en los SR. En la sección 2.2 veremos en detalle este tipo de SR.

- Demográficos: Este tipo de sistemas recomienda items basados en el perfil demográfico del usuario. Se asume que, para nichos demográficos distintos, se generarán recomendaciones distintas. Por ejemplo, usuarios que son atendidos por una web en particular dependiendo del lenguaje o el país, o sugerencias que pueden modificarse dependiendo de la edad del usuario.

- Basados en conocimiento: Estos sistemas recomiendan items basándose en un matching entre las necesidades del usuario y un conjunto de opciones disponibles (mediante las características del ítem), esto es, cómo es de útil el ítem para el usuario. Existen varios tipos de SR basados en conocimiento. Un tipo son los basados en casos (case-based) donde se usa una función de similitud para estimar cuánto necesita el usuario (que se corresponde con la descripción del problema) la recomendación (la solución del problema). Podemos interpretar el valor de similitud como la utilidad de la recomendación para el usuario. Otro tipo de sistemas parecidos son los sistemas basados en restricciones. Estos sistemas se diferencian de los basados en casos en la forma en que se encuentran soluciones. Mientras que el primero se basa en métricas de similitud, los sistemas basados en restricciones utilizan bases de conocimiento predefinido que contienen reglas para relacionar las necesidades del usuario con las características del item.

- Basados en comunidad: Este tipo de sistemas recomienda items basándose en las preferencias de los usuarios amigos. Podríamos decir que esta técnica se basa en "Dime quiénes son tus amigos y te diré quién eres". Las evidencias sugieren que las personas tienden a creer más en las recomendaciones proporcionadas por sus amigos en vez de las realizadas por otras personas que, aun siendo similares a ellos, son desconocidos. Ésto, combinado con el gran auge de las redes sociales, está generando un gran interés en estos sistemas.

- Híbridos: Estos sistemas están basados en la combinación de las técnicas anteriores. El uso de un sistema híbrido de técnicas A y B trata de usar las ventajas de A para resolver las desventajas de B y viceversa. En la sección 2.3 veremos algo más en profundidad estos sistemas.

2.1. Sistemas de Recomendación basados en contenido

Los SR basados en contenido tienen su raíz en la Recuperación de Información [8] y utilizan muchas de sus técnicas. La filosofía con la que trabajan se puede resumir, coloquialmente hablando, en "recomiéndame productos como los que me han gustado en el pasado". En un SR basado en contenido, los productos de interés están definidos por sus características asociadas. Por ejemplo, en un sistema de recomendación de películas, se usarían como características a los actores, directores, productores, etc. Un sistema de este tipo aprende un perfil de usuario basándose en las características de los productos sobre los que el usuario ya ha mostrado interés. Los productos son recomendados por medio de una comparación entre las características que tienen y el perfil de usuario. Este perfil dependerá del algoritmo de aprendizaje que se emplee. A tal efecto, se han usado árboles de decisión, redes neuronales y representaciones basadas en vectores.

Formalmente, podemos definir un Sistema de Recomendación basado en contenido de la forma: Existe una gran cantidad m de items o productos $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$, que se describen por medio de un conjunto de l atributos o características, $\mathcal{F} = \{F_1, F_2, \ldots, F_l\}$. En el Cuadro 2.1 podemos ver una matriz en la que las filas representan a los items o productos y las columnas todas las características posibles para describirlos. En la matriz resultante, el valor de la tupla a, j será 1 en el caso que el producto I_a posea la característica F_j y 0 en caso contrario. Por otra parte, tenemos un conjunto de n usuarios, $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ junto con un conjunto de votos de cada usuario sobre sus items evaluados en $\mathcal{I}$.

$\mathcal{I}$ / $\mathcal{F}$	F_1	F_2	F_3	F_4	F_5	...
I_1	1	1	1	0	0	·
I_2	1	1	1	0	0	·
I_3	0	0	1	1	1	·
I_4	0	0	0	1	1	·
I_5	0	0	0	1	1	·
...	·	·	·	·	·	·

Cuadro 2.1: Base de Datos de descripción del contenido.

El principal problema de los Sistemas de Recomendación basados en contenido es que el uso de las características para representar el contenido de un ítem puede ser una representación pobre (mientras que podrían existir otras características que pudieran ser muy usables, pero que no pueden ser fácilmente estraidas); la dificultad de realizar buenas recomendaciones a usuarios con pocos votos y, finalmente, la sobreespecialización del sistema ya que tiende a recomendar items similares a aquellos que ya se han recomendado [9].

2.2. Sistemas de Recomendación colaborativos

La aproximación colaborativa para la recomendación es bastante diferente: En vez de recomendar items o productos porque son similares a otros que le gustaron al usuario, recomienda los que les han gustado a otros usuarios similares, es decir, en vez de calcular la similaridad entre items, usa la similaridad entre usuarios. A grandes rasgos, para cada usuario, obtendremos un conjunto de usuarios (usuarios vecinos) que se caracterizarán porque su patrón de voto está áltamente correlado con el del usuario. Así, para un producto no valorado por un usuario, obtendremos un valor para él basándonos en una combinación de los valores que han dado sus usuarios vecinos a ese item.

Formalmente, podemos definir un SR colaborativo de la forma siguiente: Tenemos un conjunto de m productos $\mathcal{I} = \{I_1, I_2, \cdots, I_m\}$ cuyo dominio puede ser variado: libros, películas, restaurantes, páginas web, etc. Tenemos un conjunto de n usuarios $\mathcal{U} = \{U_1, U_2, \cdots, U_n\}$. Un usuario U_i puede dar su opinión sobre cada uno de los productos asignándole un valor discreto s, $s \in \{1, 2, \cdots, \#r\}$. Los datos observados podemos verlos como una matriz $n \times m$ muy dispersa debido a que, normalmente, un usuario sólo vota una pequeña cantidad de productos. En la matriz R, $r_{a,j}$ representa el voto que el usuario U_a dio al producto I_j y asumimos que es cero si no lo votó. Por ejemplo, en el Cuadro 2.2 podemos ver una matriz en la que las filas representan a los usuarios, las columnas a los productos y $\#r = 2$.

$\mathcal{U}$	I_1	I_2	I_3	I_4	I_5	...
U_1	2	2	0	1	0	·
U_2	0	0	1	2	0	·
U_3	2	2	0	0	0	·
U_4	2	1	0	0	0	·
U_5	0	0	0	0	2	·
...	·	·	·	·	·	·

Cuadro 2.2: Base de Datos de votos.

Podemos agrupar los algoritmos usados en las recomendaciones colaborativas en dos clases [10, 9]: basados en memoria y basados en modelos.

SR Colaborativos basados en memoria.
Estos modelos son heurísticas que realizan la predicción basándose en las votaciones ya realizadas por los usuarios. Este tipo de sistemas se conocen también como SR basados en los vecinos más cercanos.

Existen dos formas de calcular el valor de un voto del sistema: Una mediante una predicción basada en usuario y otra basada en item. La predicción del voto basada en usuario $r_{a,j}$ para el usuario U_a del ítem I_j se calcula como una agregación de los votos de otros usuarios, los N más similares, para el mismo item. Por el contrario, las aproximaciones basadas en ítem usan los votos dados por el usuario U_a a los items más similares al actual I_j para calcular la predicción dada por el sistema.

Los métodos más comunes para el cálculo de la similitud entre usuarios y entre items son los métodos basados en correlación:

- Coeficiente de Correlación de Pearson (PCC): Mide el grado de asociación entre patrones de votos usando un valor entre -1 y +1. Un valor positivo significa que, votos altos para el usuario U se corresponden con votos altos del usuario V y que los votos bajos tienen la misma correspondencia (un valor negativo para la correlación implica una asociación inversa). Se calcula de la forma:

$$PCC(U,V) = \frac{\sum_j (r_{u,j} - \overline{r}_u) \cdot (r_{v,j} - \overline{r}_v)}{\sqrt{\sum_j (r_{u,j} - \overline{r}_u)^2 \cdot \sum_j (r_{v,j} - \overline{r}_v)^2}}, \tag{2.1}$$

donde las sumas sobre j se aplican sobre aquellos items que tanto U como V han votado. Si no existen items comunes entre ambos, entonces $PCC(U,V) = 0$ por defecto. $\overline{r}_u$ es el voto medio del usuario U.

La misma idea se puede aplicar para calcular el valor de similitud de Pearson entre dos items I y J:

$$PCC(I,J) = \frac{\sum_u (r_{u,j} - \overline{r}_j) \cdot (r_{u,i} - \overline{r}_i)}{\sqrt{\sum_u (r_{u,j} - \overline{r}_j)^2 \cdot \sum_u (r_{u,i} - \overline{r}_i)^2}}, \tag{2.2}$$

- Medida Coseno: Esta métrica define la similitud entre dos objetos A y B (usuarios o items) como el coseno del ángulo entre los vectores de votos, con valores entre 0 y 1. Un valor alto significa que la similitud es mayor (los dos vectores están muy próximos). La similitud Coseno entre un usuario U y otro V se define como:

$$COS(U,V) = \frac{\sum_{i \in \mathcal{R}_U \cap \mathcal{R}_V} r_{u,i} \cdot r_{v,i}}{\sqrt{\sum_{i \in \mathcal{R}_U} r_{u,i}^2 \cdot \sum_{i \in \mathcal{R}_V} r_{v,i}^2}} \tag{2.3}$$

Podemos encontrar distintos tipos de SR colaborativos basados en memoria según los recursos utilizados para determinar el vecindario, así como para determinar el voto propuesto por el sistema, es decir, existen distintos métodos para predecir los votos que los usuarios darían a los productos: KNN, Naive Bayes, árboles de decisión, redes neuronales y modelos probabilísticos entre otros.

SR Colaborativos basados en modelos.
En contraste con los métodos basados en memoria, los algoritmos basados en modelos usan la colección de votos para aprender un modelo (se realiza offline), que será usado posteriormente para realizar las predicciones (online). Constituyen un enfoque alternativo con el que se consigue la transformación de los items y usuarios al mismo espacio latente de factores (también se conocen como latent factor models), esto es, tratan de obtener los votos de

los usuario caracterizando los items y los usuarios en una serie de factores inferidos del feedback dado por los usuarios (sus votos).

Por ejemplo, para películas, los factores descubiertos pueden ser dimensiones obvias como la calificación para adultos, comedia contra drama, la cantidad de acción, etc.; otras menos definidas como la rareza o los giros inesperados en el argumento; y otras dimensiones totalmente ininterpretables.

Para ello se usan redes neuronales [11], pLSA (Probabilistic Latent Semantic Analysis) [12], Latent Dirichlet Allocation [13], y modelos que son inducidos por la factorización de la matriz de votos usuarios-items (conocidos como modelos SVD o métodos matriz de factorización), que han obtenido una gran popularidad debido a una buena precisión y escalabilidad:

Los modelos matriz de factorización mapean a los usuarios e items a un espacio de factores conjunto de dimensión f, de tal forma que las interacciones entre los usuarios y los items se modelan como el producto escalar en dicho espacio. Como consecuencia, cada ítem i está asociado con un vector $q_i \in \Re^f$ y cada usuario está asociado a un vector $p_u \in \Re^f$.

Para un ítem dado i, los elementos de q_i miden el grado en el que el ítem posee esos factores, de forma positiva o negativa. Para un usuario dado u, los elementos de p_u miden el grado de interés que el usuario tiene en items que tiene una gran correspondencia en los factores, también de forma positiva o negativa.

El resultante producto escalar $q_i^T p_u$ captura la interacción entre el usuario u y el ítem i (el interés global del usuario en las características del item). Esto aproxima el voto del usuario al item, r_{ui}, llegando a estimar $\hat{r}_{ui} = q_i^T p_u$.

Para aprender los vectores de factores (p_u y q_i), el sistema ha de minimizar el error cuadrático normalizado en el conjunto de votos conocidos:

$$\min_{q\cdot,p\cdot} \sum_{(u,i)\in K} (r_{ui} - q_i^T p_u)^2 + \lambda(||q_i||^2 + ||p_u||^2) \tag{2.4}$$

donde K es el conjunto de pares (u, i) para los que r_{ui} es conocido (el conjunto de entrenamiento).

El sistema aprende el modelo adaptando los votos previamente realizados.

La constante λ controla el grado de regularización de los parámetros y se determina normalmente mediante validación cruzada. Con ello evitamos la sobreadaptación de los datos observados.

La minimización se realiza normalmente usando dos tipos de algoritmos de aprendizaje: El descenso de gradiente estocástico o alternando los mínimos cuadrados (stochastic gradient descent o alternating least squares). Las técnicas de alternancia de mínimos cuadrados rotan entre fijar los p_u para resolver los q_i y viceversa, es decir, fijar los q_i para resolver los p_u.

El descenso de gradiente estocástico [14] itera a través de todos los votos del conjunto de entrenamiento. Para cada voto dado r_{ui}, se realiza una predicción $\hat{r}_{ui}$ y se calcula la predicción del error asociado $e_{ui} \overset{def}{=} r_{ui} - \hat{r}_{ui}$. Entonces, se modifican los parámetros por una magnitud proporciona a γ en dirección contraria al gradiente produciendo:

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

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

Uno de los beneficios de usar la aproximación matriz de factorización en el filtrado colaborativo es su flexibilidad tratando con varios aspectos de los datos y con otros requisitos específicos de la aplicación. Esto requeriría que se modificase la ecuación 2.4 para incorporar estos nuevos aspectos. Un ejemplo muy claro podría ser el prejuicio de algunos usuarios e items, esto es, tendencias sistemáticas para algunos usuarios a votar más alto que otros y, para algunos items, recibir votaciones más altas que otros. Otra opción es la de incorporar el feedback implícito, como puede ser el de los items votados (aparte del voto que se les ha dado, que sería el explícito) [15, 16, 17]. Por otra parte, podemos incorporar el dinamismo temporal, ya que la percepción de los productos y su popularidad está cambiando constantemente y surgen nuevas selecciones. De forma similar, las inclinaciones de los usuarios evolucionan, llevando a redefinir sus gustos. Así, el sistema debería tener en cuenta los efectos temporales reflejando el dinamismo de las interacciones usuario-item.

Los problemas principales que tienen los sistemas de recomendación colaborativos son: el inicio frío y el problema del usuario nuevo. El problema del inicio frío ocurre cuando hay pocas entradas registradas en la base de datos, es decir, para un producto determinado, hay pocos usuarios del sistema que han dado su opinión sobre el producto. El problema del usuario nuevo ocurre cuando las recomendaciones deben hacerse para usuarios que tienen pocos votos registrados [9, 18]. Aquí existe un problema, ya que cuanto mayor sea el número de votos realizados por el usuario, mejor será la asignación del grupo de usuarios similares. Adicionalmente, como un número relativamente alto de usuarios suelen votar a un número relativamente pequeño de items, es bastante complicado encontrar un grupo de usuarios con un patrón de voto similar significativo [9, 19]. Por esto, usuarios con gustos poco comunes no pueden ser alojados en grupos correctos y se obtienen para ellos recomendaciones pobres.

2.3. SR Híbridos

Debido a los problemas y limitaciones que tienen las aproximaciones basadas en contenido y colaborativas, surgen bastantes Sistemas de Recomendación que usan una aproximación híbrida.

Existen cuatro métodos para obtener la predicción para un usuario[9]:

- Combinando recomendaciones separadas: Se implementan dos modelos distintos, uno colaborativo y otro basado en contenido y la recomendación se obtiene combinando las salidas de ambos sistemas.
- Añadiendo características basadas en contenido a los modelos colaborativos: Se basa en técnicas tradicionales colaborativas pero, para cada usuario, se mantiene un perfil de usuario basado en contenido que es el que se utiliza para calcular la similitud entre dos usuarios.
- Añadiendo características colaborativas en modelos basados en contenido: La aproximación más popular en esta categoría consiste en obtener una vista colaborativa de una colección de perfiles de usuario.
- Desarrollando un único modelo unificado de recomendación: Existen distintos métodos como la creación de un clasificador simple basado en reglas, modelos de regresión bayesianos que emplean cadenas de Markov y Monte Carlo para estimar y predecir parámetros, métodos probabilísticos unificados, etc.

Capítulo 3

Redes Bayesianas: modelos canónicos

3.1. Redes Bayesianas

3.1.1. Definición

Una red bayesiana [2, 3, 4] consta de dos componentes. El primero de ellos, cualitativo, está representado por un grafo acíclico dirigido $G = (V, E)$ donde los nodos (el conjunto finito V) son variables aleatorias del problema, y los arcos ($E \subset V \times V$) indican relaciones entre variables. El segundo de ellos, cuantitativo, se trata de un conjunto de distribuciones de probabilidad condicionadas (una por nodo) donde la distribución en cada nodo está condicionada al posible valor de cada uno de los padres.

Aunque puede pensarse que una red bayesiana de nodos X_i, $i = 1, \ldots, n$ no es más que una expresión más cómoda para la distribución de probabilidad conjunta $p(x_1, \ldots, x_n)$, realmente es más que eso. Como veremos, su estructura permite determinar fácilmente relaciones de independencia entre diferentes variables, lo cuál puede ser útil a la hora de realizar cálculos.

3.1.2. Independencia e independencia condicional

Definición

Uno de los conceptos clásicos de la teoría de la probabilidad es el de *independencia de sucesos*; se dice que dos sucesos (representados por las variables aleatorias A y B) son independientes si $P(A = a, B = b) = P(A = a) \cdot P(B = b)$, o lo que es lo mismo, la probabilidad de que ocurra uno de los sucesos es independiente de que ocurra el otro. Dos variables aleatorias se dicen independientes si lo son para todos sus valores. Resulta fácil pensar en sucesos independientes, pues estamos rodeados de ellos. Por ejemplo, la probabilidad de que un día sea fiesta en Granada es independiente de que ese mismo día el precio del petróleo suba en Estados Unidos (siempre y cuando no exista una complicada conspiración que relacione ambos sucesos).

Para problemas con un amplio número de variables, aparece un fenómeno que relaciona a más de dos de ellas y que también es estudiado: el de *independencia condicional*. Sean tres variables aleatorias X, Y y Z. Decimos que X e Y son *condicionalmente independientes dada* Z si y sólo si todo par de valores x e y es condicionalmente independiente para cada z tal que $p(z) > 0$; es decir:

$$\forall x, \forall y, \forall z, p(z) > 0 \Rightarrow p(x, y|z) = p(x|z) \cdot p(y|z)$$

El concepto de la independencia condicional es extensible a conjuntos de variables.

Si existe una variable que está *observada*, es decir, que ha tomado un cierto valor de su conjunto de valores, y queremos comprobar independencias respecto a ella, basta con comprobar la independencia con dicho valor. Generalmente, al observar una variable, pueden cambiar las relaciones de independencia de un conjunto de variables.

Si un conjunto de variables $\{X_1, \ldots, X_n\}$ es independiente habiéndose observado otro conjunto $\{Y_1, \ldots, Y_n\}$, lo notaremos como:

$$I(X_1, \ldots, X_n|Y_1, \ldots, Y_n)$$

Obviamente, la independencia condicional es un caso general del concepto de independencia de variables anteriormente mencionado, ya que podemos decir que las variabes del conjunto $\{X_1, \ldots, X_n\}$ son independientes entre sí si $I(X_1, \ldots, X_n|\emptyset)$ (siendo esa una forma válida de notarlo).

Criterios gráficos

Resulta bastante sencillo encontrar criterios *gráficos* mediante los que podemos encontrar relaciones de independencia (condicional o no) entre conjuntos de variables tan solo observando el grafo correspondiente a la red bayesiana. El más importante es el llamado criterio de d-separación (véase [20] para un desarrollo más profundo del que se hace aquí).

> **Definición:** decimos que un nodo de una red bayesiana, para dos arcos concretos incidentes en él, es *cabeza-cabeza* si dichos arcos le apuntan. En caso contrario, diremos que es *no cabeza-cabeza* (ver figura 3.1 para más detalle).

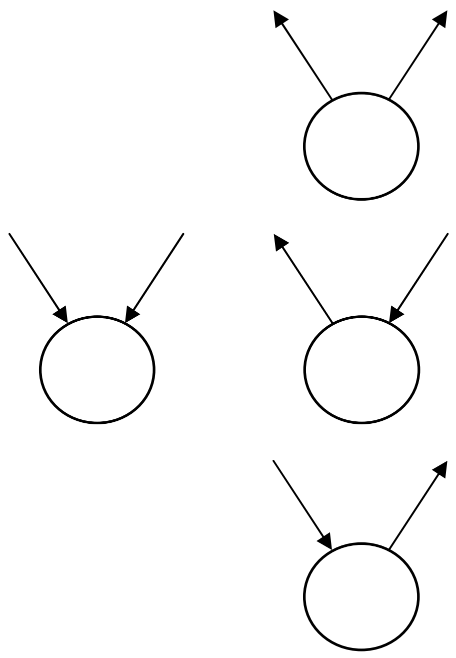


Figura 3.1: A la izquierda, nodo cabeza-cabeza. A la derecha, las tres posibles formas de nodo no cabeza-cabeza.

Definición (separación gráfica): decimos que X es independiente de Y dado $Z_1, \ldots, Z_k$ si todo camino (utilizando los arcos en ambas direcciones) entre X e Y está bloqueado en algún nodo debido a $Z_1, \ldots, Z_k$. Las formas de bloqueo (d-separación) de un nodo son dos:

- Un nodo no cabeza-cabeza observado en el camino.
- Un nodo cabeza-cabeza no observado, ni ninguno de sus descendientes no observados, en el camino.

Estas reglas son demostrables a partir de la definición de independencia, pero dicha demostración no la veremos aquí, pudiendo el lector remitirse a [3] para encontrarla.

3.2. Modelos canónicos: Modelo aditivo

3.2.1. Introducción

La construcción de modelos gráficos probabilísticos, tales como las redes bayesianas, requiere la especificación de gran cantidad de parámetros. Concretamente, para cada nodo X_i en la red, hay que concretar las distribuciones de probabilidad $Pr(x_i|y)$ donde $Y = \{Y_1, \ldots, Y_{|Pa(X_i)|}\}$, donde $Pa(X_i)$ representa el conjunto de nodos "padres" de X_i.

Para el caso de variables discretas, en las que cada nodo X_i toma valores del conjunto $S = \{1, 2, \ldots, r\}$ (y denotaremos $x_{i,s}$), y $|Pa(X_i)| = n$, el número de parámetros a estimar es $r^n - 1$, lo cual es bastante elevado. La solución a este problema es utilizar *modelos canónicos* [21].

Estos modelos cánonicos permiten, tras una serie de suposiciones, la construcción de redes bayesianas donde los nodos tienen un gran número de padres, sin necesitar la estimación de un número de parámetros exponencial en el número de variables (normalmente, sólo requieren estimar un número de parámetros lineal).

3.2.2. Modelo aditivo

Dado un nodo X_i debemos definir, para cada configuración $pa(X_i)$, la distribución de probabilidad condicionada $Pr(x_{i,j}|pa(X_i))$. Para ello, usamos el modelo canónico aditivo (estudiado en detalle en [5]):

Definición: Sea X_i un nodo en la red bayesiana, sea $Pa(X_i)$ el conjunto de padres de X_i y sea Y_k el *k-ésimo* padre de X_i en la red. Usando el modelo canónico aditivo, el conjunto de distribuciones de probabilidad condicionada para el nodo X_i pueden calcularse eficientemente de la forma:

$$Pr(x_{i,j}|pa(X_i)) = \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}) \tag{3.1}$$

donde $y_{k,l}$ es el valor que la variable Y_k toma en la configuración $pa(X_i)$ y $w(y_{k,l}, x_{i,j})$ son pesos que miden cómo el *l-ésimo* valor de la variable Y_k describe el *j-ésimo* estado del nodo X_i.

La única restricción que imponemos sobre los pesos es que sean un conjunto de valores positivos que verifiquen que:

$$\sum_{j=1}^{r} \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}) = 1, \forall\ pa(X_i)$$

Propagación

Cuando usamos el modelo canónico aditivo para representar la interacción entre variables, las probabilidades a posteriori se pueden obtener siguiendo el siguiente teorema (ver [5]) que muestra cómo calcular los valores exactos de probabilidad:

> **Teorema:** Sea X_a un nodo en la red bayesiana, sea m_{X_a} el número de padres de X_a, Y_j un nodo en el conjunto $Pa(X_a)$, y l_{Y_j} el número de estados que toma Y_j. Si las distribuciones de probabilidad condicionada pueden expresarse bajo las condiciones dadas en la Ecuación 3.1 y la evidencia se incluye sólo en ancestros de X_a, entonces las probabilidades a posteriori exactas pueden calcularse usando la fórmula siguiente:
>
> $$Pr(x_{a,s}|ev) = \sum_{j=1}^{m_{X_a}} \sum_{k=1}^{l_{Y_j}} w(y_{j,k}, x_{a,s}) \cdot Pr(y_{j,k}|ev).$$

Notar que la propagación puede realizarse en tiempo lineal con respecto al número de padres. Consideramos que este teorema es importante ya que expresa que la propagación exacta puede realizarse sin imponer restricciones de independencia entre padres de la variable X_a.

Capítulo 4

Contribuciones: Resumen, Discusión de los Resultados y Trabajo Futuro

A través de las contribuciones que vamos a presentar, creemos que queda demostrada la idoniedad del uso de las Redes Bayesianas para modelar las relaciones entre items y usuarios que se presentan en los problemas de la recomendación. La versatibilidad y potencia de las mismas nos ha permitido modelizar bajo un mismo formalismo distintos tipos de SR: colaborativos, basados en contenido, híbridos y grupales. Todos los sistemas desarrollados han sido evaluados utilizando colecciones de datos estándares en la literatura, obteniendo resultados competitivos con el estado del arte.

A continuación pasamos a detallar las soluciones propuestas para los distintos problemas abordados, así como las principales conclusiones que se obtienen.

4.1. Managing Uncertainty in Group Recommending Processes

4.1.1. Introducción al problema

La mayoría de Sistemas de Recomendación se han diseñado para uso individual, es decir, existe un usuario activo que recibe recomendaciones sobre productos concretos una vez que se ha identificado en el sistema. El problema surge en aquellos dominios en los que un grupo de personas participan en una actividad individual, por ejemplo, ver una película o ir de vacaciones, o incluso, en aquellas ocasiones en las que una persona ha de tomar una decisión

usuario	1*	2*	3*	4*	5*
A	0.2	0.2	0.2	0.19	0.21
B	0	0	0	0.1	0.9
C	0.49	0	0	0	0.51

Cuadro 4.1: Ejemplo: Probabilidad de voto para un determinado item.

en nombre de un grupo. En este trabajo se ha estudiado el problema de la recomendación a grupos, en el que el objetivo es obtener una recomendación para un grupo de personas [22]. De una u otra forma, la recomendación a grupos requiere la mezcla de distintas preferencias individuales. Es por esto que uno de los principales problemas en estos casos es la búsqueda de mecanismos de agregación que permitan obtener las recomendaciones para el grupo [23, 24, 25, 22]. En estos casos, se asume que la información con la que tratan las funciones de agregación es precisa y que la estrategia para mezclar esa información produce, por lo tanto, recomendaciones precisas. Esto no es necesariamente verdadero, especialmente si consideramos que las preferencias de usuario se determinan normalmente por medio de mecanismos automáticos. Como ejemplo podemos ver en el Cuadro 4.1 la probabilidad de voto (de 1 a 5 estrellas) para tres usuarios (A, B, y C) sobre un ítem determinado. En todos los casos podemos afirmar que el voto más probable es el 5, pero la certeza en cada situación no es la misma.

En este trabajo nos hemos centrado en el manejo de la incertidumbre en el proceso de decisión para grupos, estableciendo dos fuentes distintas: la incertidumbre al fijar las preferencias de usuario y la incertidumbre que es inherente al proceso de mezcla.

4.1.2. Solución propuesta

En este trabajo hemos investigado el valor del uso de las redes bayesianas para representar la interacción de los diferentes individuos pertenecientes a un grupo para tomar una decisión o recomendación. El modelo propuesto representa intuitivamente las relaciones entre usuarios y grupos y conseguimos representar la incertidumbre sobre la relevancia de un ítem para un usuario y la incertidumbre relacionada con los mecanismos usados por el grupo para agregar las preferencias individuales, mecanismos que se codifican mediante las distribuciones de probabilidad condicionada almacenadas en los nodos grupo. Esas distribuciones de probabilidad pueden ser consideradas como “funciones de valor social” y consideramos cuatro alternativas:

- Puerta Máximo: Modela aquellas situaciones en las que el voto del grupo es el mismo que aquel de la persona más satisfecha.
- Puerta Mínimo: El voto del grupo es el mismo que aquel de la persona más insatisfecha.
- Puerta Media: El voto del grupo es la media de los votos individuales de sus miembros.
- Puerta Mayoría: El voto del grupo es el voto mayoritario de los miembros del grupo.

4.1.3. Resultados obtenidos

Los resultados experimentales demuestran que, teniendo en cuenta la incertidumbre a la hora de realizar la agregación, obtenemos mejores resultados para los grupos. Además, los resultados obtenidos determinan otros factores que afectan al rendimiento del sistema, como la forma en que se crea el grupo, el número de individuos en el grupo, la función de agregación usada, etc. Por otra parte, el modelo propuesto es bastante genérico, de tal forma que puede ser aplicado en diferentes tareas de recomendación (por ejemplo, encontrar buenos items o predicción de votos) para un único ítem o un conjunto de ellos. Más aún, los resultados presentados pueden extenderse fácilmente a aquellas disciplinas donde la agregación de información representa un importante componente, y esas disciplinas incluyen estadística, teoría de la decisión, economía, ciencias políticas, psicología, etc.

4.2. Combining Content-based and Collaborative Recommendations: a Hybrid approach based on Bayesian networks

4.2.1. Introducción al problema

Los SR permiten a los usuarios encontrar y evaluar productos por medio de recomendaciones, es decir, enlazan usuarios a artículos que ellos posiblemente consumirían (comprar, ver, escuchar, leer, etc.), asociando automáticamente el contenido de esos productos (o las opiniones o acciones de otros individuos) a los consumidores potenciales. Éstos son capaces de filtrar la información disponible dependiendo de cada usuario específico para ayudarlos a tomar sus decisiones. La información necesaria para operar viene en forma de ranking de productos (por medio de votos que reflejan cuánto le ha gustado a alguien un item, o implícitamente mediante la observación de las acciones del usuario (estudiando los registros de compras, las páginas web visitadas, hábitos de navegación, etc.). También se puede requerir información demográfica (edad, sexo, área geográfica, etc.). Con esta información, los sistemas deben ofrecer una predicción, normalmente expresada mediante un valor numérico, que representa la opinión esperada sobre un producto. Además, pueden proporcionar una lista de productos que al usuario deberían gustarle más. Una razón por la que la predicción del voto no es una ciencia exacta es debido a la incertidumbre intrínseca asociada a las diferentes tareas que componen este campo de investigación:

- La información obtenida del usuario (de forma explícita e implícita) es una descripción vaga de la información real. Por varias razones, el usuario no puede obtener una representación exacta de lo que necesita (por ejemplo, no son capaces de expresar qué necesitan o navegan sin rumbo por los enlaces web). Las opiniones de los usuarios varían enormemente dependiendo de los productos observados.
- La construcción de la representación del producto es otro ejemplo, ya que es el resultado de una caracterización incompleta de su contenido, generalmente en forma de lista de palabras.
- La estimación del voto está también influenciada por la incertidumbre, derivada tanto de los dos puntos anteriores, como de la causada por los métodos para realizar la predicción en sí.

La inteligencia artificial ha centrado parte de sus esfuerzos investigadores en esos problemas en los que la incertidumbre es una característica importante, representando uno de los pocos campos en donde puede resolverse.

4.2.2. Solución propuesta

Las redes bayesianas, debido a su habilidad para funcionar bien en la solución de problemas con un alto nivel de incertidumbre, son un campo referencia en la inteligencia artificial. En este trabajo hemos realizado el diseño de un nuevo modelo de recomendación basado en redes bayesianas para intentar realizar predicciones más eficientes y correctas. Con este formalismo nuestro objetivo consiste en modelar la forma en que los diferentes participantes se relacionan: productos, usuarios, votos de usuario, usuarios similares, etc. La propuesta presentada es un Recomendador Híbrido que combina recomendaciones basadas en contenido y recomendaciones colaborativas.

~~Ambos métodos se combinan mediante una red bayesiana que es capaz de~~ representar las relaciones más importantes entre los elementos involucrados.

4.2.3. Resultados obtenidos

A partir del modelo presentado en este trabajo, se ha realizado una experimentación mediante la cual se han obtenido las siguientes conclusiones:

- El modelo está basado en una topología de capas representando todos los elementos que conforman el problema de la recomendación híbrida. El grado de participación de cada mecanismo de recomendación (colaborativo o basado en contenido) se selecciona automáticamente, adaptando el modelo a las condiciones específicas del problema. Hemos probado empíricamente que la combinación de la información colaborativa y de contenido ayudan a mejorar la precisión del modelo.

- Problemas como la dispersión de los datos o el hecho de que la clasificación debería calcularse en tiempo real han de tenerse en cuenta cuando nos centramos en los aspectos computacionales del proceso de recomendación. En particular, presentamos las directrices para estimar los valores de probabilidad de un conjunto de datos y hemos diseñado un algoritmo de propagación eficaz, basado en modelos canónicos.

- Siguiendo la clasificación de R. Burke [7], nuestra aproximación híbrida se encuentra en la clase “Feature Augmentation” ya que en una operación normal, las probabilidades obtenidas en la propagación de las capas variables envueltas en una recomendación basada en contenido son usadas para propagar probabilidades en las capas relacionadas en la recomendación colaborativa. Más aún, como existe un mecanismo para controlar la contribución de ambos elementos, puede clasificarse como “mixed”.

- El modelo propuesto es versátil: puede trabajar de forma exclusiva aplicando el filtrado colaborativo o basado en contenido y, además, puede aplicarse para resolver diferentes problemas de recomendación (como encontrar buenos items o predecir votos).

4.3. Using second-hand information in Collaborative Recommender Systems

4.3.1. Introducción al problema

El éxito de los SR Colaborativos depende de la disponibilidad de una gran cantidad de información. El problema surge cuando, para evaluar un usuario sobre un ítem concreto, las personas con gustos similares no poseen información sobre el ítem. En estos casos el sistema ofrecerá una predicción que no será lo suficientemente buena.

Existen estudios que tratan de resolver este problema mediante el uso de técnicas basadas en contenido. Una solución propuesta es el uso de aproximaciones híbridas que combinan el uso de técnicas colaborativas y basadas en contenido. En estos casos, cuando no existe información colaborativa disponible, las predicciones se realizan usando los votos dados por el usuario actual a ítems similares al ítem actual. Esta similitud entre ítems se calcula mediante la distancia Coseno entre el conjunto de características [26]. Otra aproximación [27] usa la similitud entre usuarios dependiendo de los perfiles basados en contenido en vez de comparar el estilo de votación. En [28] se propone el uso de una aproximación bayesiana que permite obtener buenas recomendaciones cuando no existe información colaborativa mediante el uso del algoritmo EM. Otra aproximación bayesiana la encontramos en [29] donde se propone una solución en la que, gracias a la topología del modelo propuesto, se usan los votos colaborativos dados a ítems similares al ítem actual. Otra solución totalmente distinta es el uso de un filtro colaborativo sobrealimentado (imputation-boosted) [19, 30], en el que eliminan el problema de la poca densidad de los conjuntos de datos insertando los votos predichos por un sistema puro basado en contenido. Esto habilita el conjunto completo de votos de todos los usuarios para intentar mejorar las predicciones.

4.3.2. Solución propuesta

Las propuestas para solucionar el problema dependen de la disponibilidad de la descripción del contenido de los ítems. Nosotros vamos a proponer una nueva aproximación que puede ser usada en aquellas situaciones en las que no existe información de contenido: el uso de lo que hemos llamado "información de segunda mano". Para ilustrar la idea, pensemos en la siguiente situación: José pregunta a sus amigos cercanos qué opinión tienen sobre una película en particular, pero casi ninguno de ellos la ha visto. En un intento de proporcionar su opinión sobre la película, sus amigos deciden preguntar a sus propios amigos. Esto es lo que hemos llamado "información de segunda mano”. En este artículo hemos estudiado si esas nuevas opiniones originan una mejora de las recomendaciones obtenidas para José. La implementación es sencilla: para aquellos usuarios similares que no han votado el ítem actual en el pasado, obtendremos nuevo conocimiento colaborativo usando los votos que el sistema predice para ellos, haciendo uso de la información de sus propios usuarios similares. Esta propuesta la vamos a implementar en dos SR: Uno propuesto por nosotros basado en redes bayesianas y otro basado en vecindario.

4.3.3. Resultados obtenidos

Gracias a la experimentación realizada se ha probado que obtener nueva información de segunda mano mejora las predicciones de los sistemas. Un hecho importante que contribuye al éxito, en términos de precisión, es que la información (en términos de nuevos votos) debe ser de calidad. Los resultados experimentales indican que hay dos situaciones en las que el uso de información de segunda mano no contribuye a la predicción del voto: primero, cuando la nueva información no se puede obtener de la base de datos de votos, como ocurre con ítems ’raros’ que apenas han sido votados. En esta situación, el uso de información de contenido (si está disponible) puede ser una buena solución para realizar las predicciones, como una aproximación híbrida. Por otra parte, el uso de información de segunda mano tampoco ayudará si ya existen suficientes votos de ’primera mano’. Aún así, en ambas situaciones, el funcionamiento del sistema no empeora cuando se incorpora la información de segunda mano. Las dos situaciones en las que nuestra propuesta puede ser provechosa son: por un lado, en aquellos casos en los cuales el ítem actual no es raro ni muy frecuente (que debería ser lo usual en la mayoría de aplicaciones). En estos casos, debería ser interesante la búsqueda de nueva información en la base de datos de votos. Por otra parte, esta propuesta podría ser útil en tiendas online (como Amazon o una aplicación basada en películas) donde, con bastante frecuencia, aparecen productos nuevos. En estas tiendas, los usuarios empiezan a votar después de incluir el nuevo ítem. Por consiguiente, al principio (por ejemplo semanas) es posible que, dado un usuario, pocos de sus vecinos lo hayan votado. En esta situación particular la información proveniente de vecinos de ’segunda mano’ es favorecedora.

4.4. Using past-predictions accuracy in Recommender Systems

4.4.1. Introducción al problema

Como ya hemos comentado en el Capítulo 2, existen dos clases de SR Colaborativos: basados en memoria y basados en modelos. Los basados en memoria usan el conjunto completo de votos para realizar las recomendaciones. Sin embargo, los algoritmos basados en modelos, las predicciones se realizan construyendo (offline) un modelo explícito de las relaciones entre los items. Este modelo es usado (online) para realizar las predicciones.

Centrándonos en las aproximaciones basadas en memoria, tenemos que las predicciones se obtienen considerando los usuarios/items más similares (por esto también se les conoce como modelos de recomendación basados en vecindario). Un paso crítico en estos modelos es la identificación de los vecinos, ya que la forma en la que se calcula la similitud tiene un impacto importante en el comportamiento del sistema. Hasta la fecha, los estudios e implementaciones utilizados en aplicaciones del mundo real se basan en medidas tradicionales de vector similitud como la Correlación de Pearson en sus distintas formulaciones, medida coseno, el Ranking de Spearman, etc. Este tipo de medidas de similitud miden la cercanía entre los votos de los usuarios pero no son capaces de caracterizar completamente sus relaciones.

4.4.2. Solución propuesta

En este trabajo se ha presentado una nueva métrica para medir la calidad de un vecino. Nuestra hipótesis es que, si un usuario U es bueno para predecir los votos ya realizados por el usuario activo, también lo será para las predicciones de un ítem no votado. La clave de esta aproximación es que vamos a considerar las predicciones de los votos en vez de los votos ya realizados. Por lo tanto, formalmente podemos decir que tendremos, por un lado, los votos dados por el usuario activo $\mathcal{R}_A = \{r_{a,1}, \ldots, r_{a,m}\}$ a sus m items votados y, por otra parte, los votos que serán propuestos (predichos) para cada ítem por el usuario U, $\hat{\mathcal{F}}_A(U) = \{\hat{r}_{a,1}, \ldots, \hat{r}_{a,m}\}$. Finalmente, una vez predichos todos los items votados por el usuario actual mediante el resto de usuarios, creamos una clasificación de dichos usuarios basándonos en la capacidad de predicción: cuanto mejor sea, mejor clasificado. En este momento nos quedamos con los N mejores para establecer el vecindario del usuario actual.

4.4.3. Resultados obtenidos

Mediante la experimentación realizada en este artículo, se ha demostrado que el uso de las probabilidades predictivas, esto es, cómo de probable es que el usuario activo vote el ítem actual con un valor, dado que conocemos los votos de los vecinos, puede usarse como estrategia para encontrar el vecindario (obteniendo resultados muy buenos) y para combinar las recomendaciones individuales (mediante su uso como peso que mide la fuerza de las relaciones entre un vecino y el usuario actual). Mediante el estudio de los vecindarios obtenidos usando la aproximación predictiva, hemos encontrado que el conjunto de usuarios obtenidos difiere con respecto al vecindario obtenido usando un criterio de correlación clásico, siendo tanto o más válidos que éstos. Más aún, hemos demostrado que los mejores resultados se obtienen mediante la combinación de las capacidades predictivas y la correlación, usando una aproximación conjunta.

4.5. Trabajo Futuro

De forma global, como trabajos futuros, nos proponemos dos grandes líneas de trabajo: por un lado, avanzar en el uso de los Modelos Gráficos Probabilísticos para modelar la problemática de los SR y por otro estudiar aquellos factores que permitan mejorar la calidad de las recomendaciones. Tomando como punto de partida los modelos desarrollados (basados en contenido, colaborativos e híbridos), pretendemos adaptarlos o crear nuevos modelos capaces de solventar los retos que nos proponemos:

- Por un lado, estudiar cómo se puede incluir más y mejor información en los procesos involucrados en la recomendación. Dos de los aspectos que pretendemos cubrir en esta línea son:
    - Dotar a los SR de mecanismos para manejar las relaciones entre los distintos componentes estructurales que definan los usuarios, los productos y las propias valoraciones de los usuarios. Mediante el análisis de estas relaciones debemos obtener mejores perfiles de usuario y, por tanto, ser más precisos a la hora de encontrar aquellos productos que realmente interesan.
    - Introducción del contexto a la hora de generar las recomendaciones. Entendemos el contexto en el amplio sentido de la palabra, pudiendo considerar elementos como criterios demográficos, tiempo, localización, clima, si el producto se va a compartir con otros

usuarios, etc. En este sentido podríamos permitir que las recomendaciones difieran según el contexto en el que se están realizando.

- Criterios temporales. Debemos permitir al sistema capturar las diferencias de gusto de un usuario a lo largo del tiempo y también las diferencias que existen en los votos para unos determinados ítems dependiendo de cuándo se realicen. Por ejemplo, considerando el contexto "tiempo", podemos hacer que los productos que se recomienden para un bebé varíen según la edad del mismo.

- Por otro lado, pretendemos mejorar los datos permitiendo que los mismos provengan de distintas fuentes o proveedores, por ejemplo utilizando las descripciones o valoraciones que, sobre un producto, se hagan en distintas fuentes de información (sitios web). Usaremos datos de interacción usuario-ítem, los datos del ítem, etiquetas sociales, redes sociales y estructuras de grafo.

- Estudio de mecanismos para mejorar la justificación que se da sobre la recomendación al usuario. Aún asumiendo que la recomendación es un proceso estadístico y por tanto, que necesariamente se cometerán errores, un SR no puede ser visto como una caja negra, pues difícilmente el usuario creerá en sus propuestas. Nuestro objetivo será el de dotar a los sistema de la capacidad de dar explicaciones sobre el proceso de decisión. Pretendemos que el sistema pueda interactuar con el usuario, y así podamos utilizar las opiniones que tiene el usuario sobre las explicaciones para mejorar su perfil. Pretendemos utilizar herramientas como abducción en redes bayesianas o técnicas basadas en el aprendizaje activo para este objetivo.

- Novedad (en entornos de recomendación de noticias). Cuando un usuario se asoma a la web con la intención de obtener información sobre las noticias del día, es fácil que se pierda entre cientos de resultados, muchos de ellos coincidentes. Un SR le puede ayudar a filtrar toda aquella información que no le es de interés, bien porque no encaje con su perfil, bien porque es redundante con otra información que ya conoce.

- Acceso a la información del grupo. Pretendemos definir modelos considerando la necesidad de información particular del grupo. Para ello estudiaremos mecanismos que puedan ayudar a los usuarios del grupo a encontrar información relevante, reduciendo así los esfuerzos individuales. Pretendemos definir modelos para determinar la relevancia de un producto para el grupo (como conjunto) y estudiar nuevas estrategias de agregación para determinar el perfil del grupo.

Capítulo 5

Publicaciones

5.1. Managing Uncertainty in Group Recommending Processes

- Revista: USER MODELING AND USER-ADAPTED INTERACTION
- Estado: Publicado
- Índice de Impacto (JCR 2010): 3.074
  Área de conocimiento: Computer Science, Cybernetics.
  Ranking: 2 / 19. Cuartil: Q1
- Índice de Impacto (SJR 2010): 0.069
  Área de conocimiento: Human-Computer Interaction.
  Ranking: 7 / 33. Cuartil: Q1

User Model User-Adap Inter (2009) 19:207–242
DOI 10.1007/s11257-008-9061-1

ORIGINAL PAPER

Managing uncertainty in group recommending processes

Luis M. de Campos · Juan M. Fernández-Luna · Juan F. Huete · Miguel A. Rueda-Morales

Received: 25 February 2008 / Accepted in revised form : 26 October 2008 /
Published online: 21 November 2008

Abstract While the problem of building recommender systems has attracted considerable attention in recent years, most recommender systems are designed for recommending items to individuals. The aim of this paper is to automatically recommend a ranked list of new items to a group of users. We will investigate the value of using Bayesian networks to represent the different uncertainties involved in a group recommending process, i.e. those uncertainties related to mechanisms that govern the interactions between group members and the processes leading to the final choice or recommendation. We will also show how the most common aggregation strategies might be encoded using a Bayesian network formalism. The proposed model can be considered as a collaborative Bayesian network-based group recommender system, where group ratings are computed from the past voting patterns of other users with similar tastes.

Keywords Group recommending · Management of uncertainty · Probabilistic graphical models

L. M. de Campos · J. M. Fernández-Luna · J. F. Huete (✉) · M. A. Rueda-Morales
Department of Computer Science and Artificial Intelligence, University of Granada,
18071 Granada, Spain
e-mail: jhg@decsai.ugr.es

L. M. de Campos
e-mail: lci@decsai.ugr.es

J. M. Fernández-Luna
e-mail: jmfluna@decsai.ugr.es

M. A. Rueda-Morales
e-mail: mrueda@decsai.ugr.es

1 Introduction

Recommender systems (RS) provide specific suggestions about items (or actions) within a given domain which may be considered of interest to the user (Resnick and Varian 1997). Depending on the information used when recommending, traditional RS are mainly classified into content and collaborative-based RS, although hybrid approaches do exist. The first type recommends a product by considering its content similarity with those products in which the user has previously expressed an interest. The second alternative attempts to identify groups of people with similar tastes to the user and to recommend items that they have liked. Most RS are designed for individual use, i.e. there is an active user who receives recommendations about certain products once they have logged on to the system.

In this paper, we will focus on the related problem of *group recommending* (*GR*), where the objective is to obtain recommendations for groups of people (Jameson and Smyth 2007). This kind of RS is appropriate for domains where a group of people participate in a single activity such as watching a movie or going on vacation and also in situations where a single person must make a decision on behalf of the group.

In one way or another, GR involves merging different individual preferences. In these situations, it is natural that one of the most important issues is the search for an aggregation mechanism to obtain recommendations for the group. According to Pennock and Wellman (2005) "... *there is nothing close to a single well-accepted normative basis for group beliefs, group preferences or group decision making.*", and many aggregation strategies can therefore be found in literature for group decisions (Masthoff 2004; Masthoff and Gatt 2006; Yu et al. 2006; Jameson and Smyth 2007). It is typically assumed that member preferences are given using a rating domain (let us say from 5*, *really like*, to 1*, *really hate*). An aggregation strategy is then used to determine the group rating. For example, let us consider a group with three individuals, *John, Ann* and *Mary*, where *John* rates a product 5*, *Ann* rates it 2*, and *Mary* rates it 5*. Following an average aggregation criterion, we could then say that the group rating for this product is 4*.

As in the previous example, the methods proposed in GR literature (see Jameson and Smyth 2007 for a review) do not deal with uncertainty. They assume that the inputs of the aggregation functions (i.e. user preferences) are precise and use a merging strategy to compute precise outputs. This assumption is not necessarily true, especially if we consider that the user's preferences are normally determined by means of automatic mechanisms. In these cases, a probability distribution over the candidate ratings might be used to express user likelihoods. For example, Table 1 shows the probability distributions representing the preferences of three users (A, B, and C). In this case,

Table 1 User ratings for a given item

User	1*	2*	3*	4*	5*
A	0.2	0.2	0.2	0.19	0.21
B	0	0	0	0.1	0.9
C	0.49	0	0	0	0.51

although 5* might be considered the most probable rating, we will not have the same confidence about every situation.

Surprisingly, little attention has been paid in GR literature to the problem of managing uncertainty although it has been well established in the general group decision framework (see Clemen and Winkler 1999; Genest and Zidek 1986 for a review). In this paper, therefore, we will focus on this particular problem. We maintain that two different sources of uncertainty can be found in group recommending processes: the uncertainty shown when user preferences are set, i.e. the user's personal opinion about an item or feature; and the uncertainty which is inherent to the merging process.

The purpose of this paper is to investigate the value of using Bayesian networks (BN) to represent how different individuals in a group interact in order to make a final choice or recommendation. In our approach, the BN formalism is used to represent both the interactions between group members and the processes leading to the final choice or recommendation. We will show how common decision rules in literature could be managed by adequately designing canonical models with the BN language, thereby shedding new light on the combination processes. Discussion about subjects such as how the groups are formed, how long they have existed, relationships between group members, how the group might interact to reach a consensus, etc. are beyond the scope of this paper. We shall assume that all the individuals use the same set of labels to express their preferences for an item, and that these preferences are represented by means of a probability distribution (probably estimated from a data set).

We consider BNs appropriate because they combine a qualitative representation of the problem through an explicit representation of the dependence relationships between items, users and groups, with a quantitative representation by means of a set of probability distributions to measure the strength of these relationships. Throughout the process, we must consider the computational aspects of the RS, where the sparseness of the data and the fact that the ranking should be computed in real time represent two challenges.

The second section of this paper briefly examines group recommender systems and related work. Section 3 presents the proposed BN-based model which enables the interaction between individuals to be represented. Section 4 examines how to represent the strength of the individuals' interactions (i.e. conditional probability distributions) and Sect. 5 discusses how inference is performed in order to make recommendations to the group. Section 6 examines the experimental framework. Section 7 discusses the experimental results obtained when considering uncertainty in individual ratings and in Sect. 8 we study those situations where the process behind the group rating is also uncertain. Finally, our conclusions and comments regarding further research are discussed in Sect. 9.

2 Classification of group recommender systems and related work

Although GR is quite a new research topic, many papers on this problem have already been published. The specific objectives of recommender systems in the research published so far are determined by the characteristics of the domain for which the system has been developed. These characteristics significantly affect the choice of design and

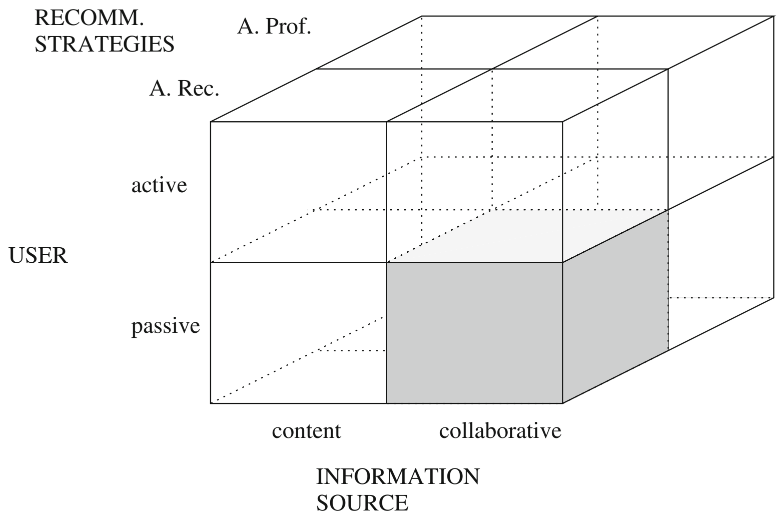


Fig. 1 Classification of Group Recommending Systems

each publication therefore focuses on a specific issue (from how to acquire information about group preferences or how the system generates and explains the recommendations to studying the mechanism used to reach a consensus (Jameson and Smyth 2007)). As a result, relating the different approaches is a difficult task.

In this section, we will present a new classification taxonomy for group recommending systems. This classification is based on three independent components of primary importance in the design of a group recommending system and not on the particular techniques used to solve each problem: the information source, the aggregation criterion used to make the recommendations, and the user's interaction with the system. Figure 1 shows a graphical representation of the proposed classification.

– *Source of information*: This classification criterion, which has been borrowed from classical RS literature (Adomavicius and Tuzhilin 2005), distinguishes between *content-based* (CB) and *collaborative filtering* (CF). In the first case, the recommended items are those which are similar to the ones that individuals have found interesting in the past. As a result, it is necessary to analyze the content's features for recommending.
  The second alternative considers that the recommendations for a target product have been obtained by considering how people with similar tastes rated a product in the past. These systems are based on the idea that people will agree in future evaluations if they have also agreed in their past evaluations. The information sources are therefore the preference ratings given by similar users.
  A new category can obviously be obtained if we consider hybrid approaches that combine both (collaborative and content-based) methods.[1]
– *Recommendation strategies*:
  Once we have the information to hand, the strategy used for aggregating this information is a central point in group recommending, and generally in any group decision process. In this case, two different approaches can be distinguished. The first

[1] Without loss of generality, we have decided not to include this category in our taxonomy since, to the best of our knowledge, no study has tried to combine both techniques in the group recommending framework.

approach, *aggregating recommendations* (AR), is a two-step strategy, where an individual recommendation is first obtained for each group member, and then a common recommendation is obtained by merging these individual recommendations. In the second approach, *aggregating profiles* (AP), the objective is to obtain a common profile by representing group preferences. This can be done explicitly, where the individuals use a common group account to give their preferences, or implicitly, by means of an aggregation mechanism for the different individuals' profiles or preferences.

– *Individual interactions*
  Finally, a group recommending system can also be categorized by considering the way in which the users interact with the system. The individuals can be dichotomized into *passive members* (PM) and *active members* (AM). Focusing on the *active* members, the final purpose is to reach a consensus between the group members and, like many decision support system approaches, it is necessary for the users to evaluate the system recommendations. In contrast, when the members are *passive*, the final purpose is only to provide a recommendation to the group, as might be the case when using an RS in a marketing campaign. In this situation, the individuals do not interact with the system in order to evaluate the proposed recommendations.

Since we use three non-overlapping criteria for classification purposes, a given GRS can be classified using three labels, one for each category. For instance, a GRS can be classified as CB+AP+PM if the group profile is obtained by combining the information about the content of the items which have been previously evaluated by each user. This profile will be used to send the final recommendations to the group.

2.1 Related work

Once the taxonomy has been presented, we will then go on to classify previously published GR systems.

– CB+AP+PM: most published GRSs might be included in this category. For example, let us consider MusicFX (McCarthy and Anagnost 2000). Given a database of member preferences for musical genres (each user rates each of the 91 genres on a five-point scale), the group profile is computed by summing the squared individual preferences. Using a weighted random selection operator, the next music station to be played is then selected. No interaction with the system is possible except by changing user preferences.
  The inputs in the case of group modeling (Masthoff 2004) are user preferences (ratings) for a series of programs, and in this paper we study the performance of several aggregation strategies. The article (Yu et al. 2006) presents various TV program recommendations for multiple viewers by merging individual user preferences on features (e.g. genre, actor, etc.) to construct a group profile. The aim of the aggregation strategy is to minimize the total distance in such a way that the merged profile is close to most user preferences, thereby satisfying most of the group.
– CB+AP+AM: The Travel Decision Forum (Jameson 2004) was developed to help a group of users agree on the desired attributes of a vacation. This system allows

group members to collaboratively specify their preferences and to reach an agreement about an overall solution. In this case, a group profile is obtained through the interaction of the members, taking into account the system's current recommendation which is obtained by aggregating individual preferences for each dimension. In the article (Kudenko et al. 2003), a system is presented to help a group of users reach a joint decision based on individual user preferences.

– CB+AR+PM: Intrigue (Ardissono et al. 2003) recommends tourist attractions for heterogeneous groups that include homogeneous subgroups where the members have similar preferences. In this system, the users record their preferences for a series of tourist attractions, and recommendations (obtained using a fuzzy AND) are then merged using a weighted scheme where each weight represents the relevance of the corresponding subgroup (for instance, a subgroup could be particularly influential since it represents a large portion of the group). Although the system explains their recommendations, it has no means of interacting with the user.
– CF+AR+PM: Polylens (O'Connor et al. 2001), an extension of MovieLens (Herlocker et al. 2004), recommends movies to groups of users. This system uses a nearest neighbor-based algorithm to find the individuals with the most similar tastes to those of each group member and to obtain recommendations for every user. The voting preferences of these individuals are then merged according to the principle of least misery (minimum criterion). Under the same classification, (Chen et al. 2008) uses genetic algorithms to learn the group rating for an item that best fits the existing ratings for the item given by the individuals and the subgroups. The idea is that it is possible to learn how the user interacts from the known group ratings. The proposed algorithm therefore recommends items based on the group's previous ratings for similar items.

2.1.1 The role of uncertainty

As far as the authors are aware, the role of uncertainty in group recommending processes has not been considered. Nevertheless, many papers have been published which tackle this problem when recommendations are made to individual users (Zukerman and Albrecht 2001; Albrecht and Zukerman 2007). Focusing on probabilistic approaches, those relating to the one presented in this paper include *content-based RSs* (Mooney and Roy 2000; de Campos et al. 2005), *collaborative filtering RSs* (Breese et al. 1998; Schiaffino and Amandi 2000; Butz 2002; Lekakos and Giaglis 2007; Miyahara and Pazzani 2000; Heckerman et al. 2001) and *hybrid* methods (Popescu et al. 2001; de Campos et al. 2006).

In terms of the group's process, the treatment of uncertainty is, however, a well-known problem in other disciplines and so in this section we will review those papers which focus on the combination of probabilistic information from a purely statistical approach (see Clemen and Winkler 1999; Genest and Zidek 1986). In general, we might consider these methods as analytical models operating on the individual probability distributions to produce a single "combined" probability distribution. These approaches can generally be further distinguished into axiomatic approaches (by considering a set of assumptions that the combination criteria might satisfy) and Bayesian approaches:

- Axiomatic approach: the following common functions deal with belief aggregation:
  - (i) *Linear Opinion Pool* where the group probability, $Pr(G)$, is obtained as the weighted arithmetic average over the individual probabilities, $Pr(V_i), i = 1, \dots, n$, i.e. $Pr(G) = \sum_{i=1}^{n} w_i Pr(V_i)$, with w_i being the weights totaling one.
  - (ii) *Logarithmic Opinion Pool* (weighted geometric average) defined as $Pr(G) = \alpha \prod_{i=1}^{n} Pr(V_i)^{w_i}$, with α being a normalization constant and the weights w_i (called expert weights) are typically restricted to total one. If the weights are equal to $1/n$, then the combined distribution is proportional to the geometric average.
- Bayesian Approach (Genest and Zidek 1986; Clemen and Winkler 1999): this has been used to combine expert information by taking into account the so-called Naive Bayes assumption. In our context, in order to obtain efficient combinations, individual opinions are assumed to be conditionally independent given the group vote.

3 Modeling group decision networks

The purpose of this paper is to develop a general methodology based on the Bayesian network (BN) formalism for modeling those uncertainties that appear in both the interactions between group members and the processes leading to the final choice or recommendation. For example, let us imagine that we want to advise a group of tourists to visit a particular monument or not. In such a situation, we should assume that the individuals in the group are unfamiliar with the monument (or item to be recommended). Each group member might speculate about their possible preference for visiting this monument and this is necessarily uncertain. Nevertheless, the group recommendations must be obtained by aggregating these preferences.

Individual preferences can be computed by considering two alternatives: the first considers content information (such as a description of the monument, location, etc.) and the second considers how people with similar tastes rated this monument in the past (for instance, `dislike` or `like`). This is the approach followed in this paper where the similarity between users will be computed by considering how common items have been rated. Following the classification presented in Sect. 2, our GRS can therefore be categorized as CF+AR+PM.

As a collaborative approach, our model will inherit most of the disadvantages of classical collaborative filtering approaches. For example, the system cannot draw any inferences about items for which it has not yet gathered sufficient information, i.e. we also have the First-Rater problem. Similarly, we also inherit the Cold-Start problem since it is difficult to recommend items to new users who have not submitted any ratings. Without any information about the user, the system is unable to guess user preferences and generate recommendations until a few items have been rated.

For our information sources, we will consider a database of ratings **R** (which is usually extremely sparse) to store user ratings for the observed items. For example, Table 2 shows the ratings given by each user U_i for an item I_j using the values $1 =$ `dislike` and $2 =$ `like` (the value $'-'$ represents the fact that the user has not seen the item).

Table 2 Database of user ratings

	U_0	U_1	U_2	U_3	U_4	U_5
I_1	1	1	2	2	1	1
I_2	1	–	2	–	2	2
I_3	1	1	2	1	1	2
I_4	2	–	1	–	1	2
I_5	2	2	1	1	1	1
I_6	2	2	–	2	2	2
I_7	2	–	–	–	1	2

In order to achieve this objective, our aim is to build a BN where two components might be considered. The first, described in Sect. 3.1 relates to the collaborative component of the recommender system. Both the topology of this component and the probability values will be learned from a set of past user ratings, and this will be used to compute a probability distribution representing the preferences of each group member for a given item. The second component will be used to merge these preferences in order to reach the final group opinion. This component is modeled using a BN with a fixed structure given the group members, and the weights will be computed based on the ratings provided by the group members (see Sect. 3.2).

3.1 BN-based collaborative component

In this section, we will briefly describe this component (those readers interested in further details can consult (de Campos et al. 2008)). Our objective is to model how each user should rate an item. In order to represent relationships between users, we shall include a node, U_i, for each user in the system. We use $\mathcal{U}$ to denote the set of user nodes, i.e. $\mathcal{U} = \{U_1, \ldots, U_n\}$. The user variable U_i will therefore represent the probability distribution associated to its rating pattern. For instance, using the data in Table 2, each node will store two probability values representing the probability of U_i liking ($Pr(U_i = 2)$) or disliking ($Pr(U_i = 1)$) an item.

In order to facilitate the presence of dependence relationships between individuals in the model (to avoid a possibly complex network topology), we propose that a new set of nodes $\mathcal{V}$ be included to denote collaborative ratings. There is one collaborative node for each user in the system, i.e. $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$. These nodes will represent a probability distribution over ratings, and they will therefore take their values in the same domain as $\mathcal{U}$.

3.1.1 Learning stage

Given an active user, the parent set of the variable V_a in the graph, $Pa(V_a)$, will be learnt from the database of votes, **R**. This set will contain those user variables, $U_b \in \mathcal{U}$, where U_a and U_b are most similar in taste, i.e. the best neighbors for the active user. Given a similarity measure, the set $Pa(V_a)$ can therefore be obtained by using a threshold or by only considering the first p variables in the ranking (see Fig. 2). It should be

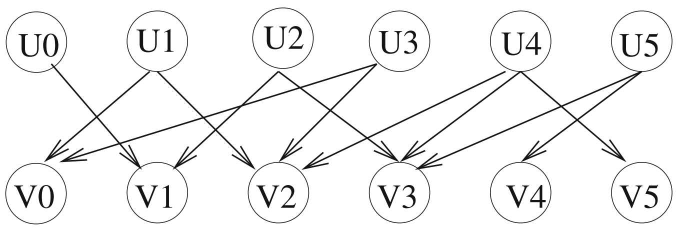


Fig. 2 Collaborative Recommending System Topology

noted that we do not include the links between $U_i \longrightarrow V_i, \forall i$, since we are modeling a collaborative rating scheme where (assuming that the item being recommended has not been observed by the active user) the predicted rating will only depend on those ratings given by its neighbors.

The similarity measure proposed in this paper is a combination of two different, but complementary, criteria: vote correlation between common items and the overlap degree, i.e.

$$sim(U_a, U_b) = abs(PCC(U_a, U_b)) \times D(U_a, U_b) \qquad (1)$$

The first criterion, which is normally used as the basis for calculating the weights in different collaborative systems, attempts to capture those similar users, i.e. those with the highest absolute value of Pearson's correlation coefficient defined as

$$PCC(U_a, U_b) = \frac{\sum_j (r_{a,j} - \overline{r}_a)(r_{b,j} - \overline{r}_b)}{\sqrt{\sum_j (r_{a,j} - \overline{r}_a)^2 \sum_j (r_{b,j} - \overline{r}_b)^2}} \qquad (2)$$

where the summations over j are over those items for which users U_a and U_b have recorded votes and $\overline{r}_a$ is the mean vote for user U_a. It should be noted that PCC ranges from $+1$ to -1: $+1$ means that there is a perfect positive linear relationship between users; -1 means that there is a perfect negative linear relationship; a correlation of 0 means that there is no relationship. Therefore, when there are no common items in U_a and U_b voting records, then $PCC(U_a, U_b) = 0$ by default. In our approach, by using the absolute value of PCC, $abs(PCC)$, we consider that both positively (those with similar ratings) and negatively correlated users (those with opposite tastes) might help[2] to predict an active user's final rating.

The second criterion tries to penalize those highly correlated neighbors which are based on very few co-rated items, which have proved to be bad predictors (Herlocker et al. 1999). We might therefore take into account the number of items that both U_a and U_b rated simultaneously, i.e. their overlap degree. In particular, we consider that the quality of U_b as the parent of variable U_a is directly related with the probability of a user U_a rating an item which has been also rated by U_b. This criterion can be defined by the following expression:

[2] For instance, if whenever U_b rates as `like` U_a rates with `dislike`, then knowing that U_b had rated an item with `like` provides information about U_a's possible rating.

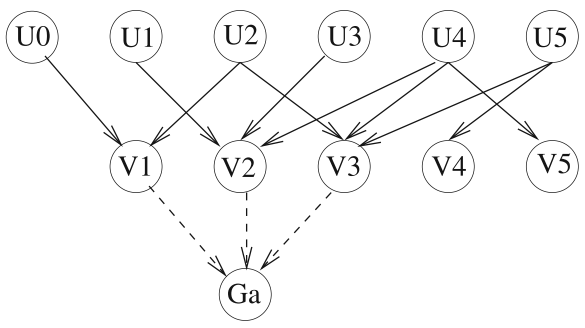


Fig. 3 Modeling groups

$$D(U_a, U_b) = \frac{|I(U_a) \cap I(U_b)|}{|I(U_b)|}.$$

where $I(U)$ is the set of items rated by user U in the data set. It should be noted that we are not considering the particular votes, merely whether the users rated an item or not.

3.2 Modeling the group component

As mentioned previously, since groups are usually created by their members, we shall not consider how groups are formed or how they are managed. We shall therefore assume that we know the composition of the groups, and our problem is to study how this information can be represented in the BN and also how to predict ratings for groups.

We propose to identify a group G as a new node in the BN. Since the recommendations are made by considering the preferences of its members, we propose that the parents ($Pa(G)$) of the group node (G) will be the set of nodes in $\mathcal{V}$ representing its individuals. In this case, we are modeling that the predictions of the group's ratings will depend on the collaborative predictions obtained for each of its members. Figure 3 illustrates a group G_a with three members: V_1, V_2, and V_3. We use dashed lines to represent user-group relations since we assume that the composition of the group is known.

In this paper, we will focus on how different aggregation strategies can be represented in our BN-based model. In order to maintain generality (so that the proposed aggregation mechanisms can be applied in more general situations[3]), we will use the following independence assumption: *given that we know the opinion (ratings) of all the group members, group opinion does not change (it is independent) if the state of any other variable in system X_i is known, i.e. $I(G, X_i|Pa(G)), \forall X_i \notin Pa(G_i)$*. It is important to remember that in certain domains this restriction might be very restrictive. For example, it might also be possible to consider other factors that would affect the group rating such as the context. Nevertheless, the study of how to include these factors in the model is beyond the scope of this paper.

[3] For example, it might be used to combine multiple classifiers (Kittler et al. 1998; Abellán and Masegosa 2007) where the new cases will be classified by considering all the results obtained by each classifier.

Table 3 Stored probability values

$P(U_0)$, $P(U_1)$, $P(U_2)$, $P(U_3)$, $P(U_4)$, $P(U_5)$				
$P(V_1\|U_0, U_2)$	$P(v_{1,1}\|1,1)$	$P(v_{1,1}\|1,2)$	$P(v_{1,1}\|2,1)$	$P(v_{1,1}\|2,2)$
$P(V_2\|U_1, U_3, U_4)$	$P(V_2\|1,1,1)$	$P(V_2\|1,1,2)$	$P(V_2\|1,2,1)$	$P(V_2\|1,2,2)$
	$P(v_{2,1}\|2,1,1)$	$P(v_{2,1}\|2,1,2)$	$P(v_{2,1}\|2,2,1)$	$P(v_{2,1}\|2,2,2)$
$P(V_3\|U_2, U_4, U_5)$	$P(v_{3,1}\|1,1,1)$	$P(v_{3,1}\|1,1,2)$	$P(v_{3,1}\|1,2,1)$	$P(v_{3,1}\|1,2,2)$
	$P(v_{3,1}\|2,1,1)$	$P(v_{3,1}\|2,1,2)$	$P(v_{3,1}\|2,2,1)$	$P(v_{3,1}\|2,2,2)$
$P(G_a\|V_1, V_2, V_3)$	$P(g_{a,1}\|1,1,1)$	$P(g_{a,1}\|1,1,2)$	$P(g_{a,1}\|1,2,1)$	$P(g_{a,1}\|1,2,2)$
	$P(g_{a,1}\|2,1,1)$	$P(g_{a,1}\|2,1,2)$	$P(g_{a,1}\|2,2,1)$	$P(g_{a,1}\|2,2,2)$

In order to complete the BN-based model, it is necessary to estimate the local probabilities that must be stored in the nodes. In particular, each node X_i has a set of conditional probability distributions, $P(x_i|pa(X_i))$ (except root nodes that store marginal probability distributions).[4] For each possible configuration $pa(X_i)$ of the parent set $Pa(X_i)$, these distributions quantify the effect that the parents have on the node X_i. In our case, these probabilities are used to encode both the strength of the user-user interactions and the processes leading to the final choice or recommendation for the group. In Table 3, we show those probability distributions stored in the example of Fig. 3, where for instance $P(V_2|1, 1, 2)$ represents $P(V_2|u_{1,1}, u_{3,1}, u_{4,2})$. The method of assessing the particular values will be discussed in Sect. 4.

3.3 How to predict the group rating: inference

Once the BN is completed, it can be used to perform inference tasks. In our case, we are interested in the prediction of the group's rating for an unobserved item, I. As evidence, we will consider how this product was rated in the past.[5] The problem therefore comes down to computing the conditional (posterior) probability distribution for the target group G_a given the evidence, i.e. $Pr(G_a|ev)$. For instance, let us assume that we want to predict the rating given by G_a in Fig. 3 to item I_7. If we look at Table 2, the evidences are $ev = \{U_0 = 2, U_4 = 1, U_5 = 2\}$ and the problem is to compute $Pr(G_a = 1|\, u_{0,2}, u_{4,1}, u_{5,2})$.

Since the BN is a concise representation of a joint distribution, we could propagate the observed evidence through the network towards group variables. This propagation implies a marginalization process (summing out over uninstantiated variables).

[4] Throughout this paper we will use upper-case letters to denote variables and lower-case letters to denote the particular instantiation. More specifically, we use v_i to denote a general value of variable V_i and $v_{i,j}$ to indicate that V_i takes the jth-value.

[5] It should be noted that we consider that no member of the group has observed the items beforehand and therefore the evidences are over the values taken by variables in $\mathcal{U}$. In the case of a group member (let us say U_i) having also previously rated I, we shall instantiate both node U_i and V_i to the value of the given ratings. The instantiation of V_i will imply that there is no uncertainty about its rating when the information is combined at a group level. Nevertheless, the computations are more complex in this situation.

A general scheme might be:

$$Pr(G_a = s|ev) = \sum_{\mathcal{V}} Pr(G_a = s|v_1, \ldots, v_k) Pr(v_1, \ldots, v_k|ev) \tag{3}$$

where $v_1, \ldots, v_k$ represents a given configuration of the collaborative variables (parent set) $Pa(G_a)$ and the sum is over the exponential number of possible configurations, and this requires an exponential time $O(r^{|Pa(G_a)|})$ with r being the number of candidate ratings. Considering that the evidence belongs to $\mathcal{U}$, the joint probability over the collaborative variables might be computed as

$$Pr(v_1, \ldots, v_k|ev) = \sum_{\mathcal{U}^-} \prod_{i=1}^{k} Pr(V_i = v_i|u^-, ev) Pr(u^-) \tag{4}$$

where the sum is over all the possible configurations, u^-, of the set of uninstantiated user variables, denoted by $\mathcal{U}^-$, also requiring an exponential time, $O(r^{|\mathcal{U}^-|})$.

These computations should be performed for each group variable when there is an item to be recommended. As there is usually a large set of groups in the system, this process becomes computationally expensive and reducing computational complexity becomes a key design parameter, especially if the objective is to obtain scalable strategies which can be implemented in real-time applications.

In order to tackle this problem, we propose the use of canonical models to represent conditional probability distributions. By means of these models, we can reduce the number of probability values stored and develop specific inference algorithms. In those cases where the computation of $Pr(V_1, \ldots, V_k|ev)$ is complicated, we also propose to approximate these values by using extra independence assumptions (see Sect. 5).

4 Estimating the strength of the users' interactions

In terms of assessing the probability values, we must distinguish between roots in the graph, nodes in $\mathcal{U}$, and the remaining nodes. In particular, for every user node U_k, we need to assess the prior probability distribution over the user's rating pattern, i.e. the probability of user U_k rating with a given value s, $1 \leq s \leq r$. For example, considering the relative frequency and the data in Table 2, we will obtain $Pr(U_3 = 1) = 2/4 = 0.5$ and $Pr(U_5 = 1) = 2/7 = 0.286$.

For each non-root variable, we must store an exponential number of conditional probability distributions: one probability distribution for each possible configuration of its parent set. The assessment, storage, and manipulation of these probability values can be quite complex, especially if we consider that the number of similar users (parents of the nodes in $\mathcal{V}$) and the size of the groups might be large when real group recommending applications are considered. We therefore propose the use of different canonical models to represent these conditional probabilities. By using this representation, it might be possible to reduce the problem of data sparsity (it is quite probable that many configurations lack data), leading to important savings in storage (we only need to store a linear number of probability values) and more efficient inference algorithms (see Sect. 5).

4.1 Probabilities of the collaborative component

The probabilities in the collaborative component (nodes in $\mathcal{V}$) might be estimated from the data set of past user ratings. For a given node V_i, we must define the conditional probability distribution $Pr(v_{i,j}|pa(V_i))$ for each configuration $pa(V_i)$. We propose the use of the following canonical model (studied in detail in de Campos et al. 2008) where an additive behavior of the collaborative nodes is assumed, thereby enabling the data sparsity problem to be tackled:

Definition 1 (*Canonical weighted sum)* Let X_i be a node in a BN, let $Pa(X_i)$ be the parent set of X_i, and let Y_k be the kth parent of X_i in the BN. By using a canonical weighted sum, the set of conditional probability distributions stored at node X_i are then represented by means of

$$Pr(x_{i,j}|pa(X_i)) = \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}) \tag{5}$$

where $y_{k,l}$ is the value that variable Y_k takes in the configuration $pa(X_i)$, and $w(y_{k,l}, x_{i,j})$ are weights (effects) measuring how this lth value of variable Y_k describes the jth state of node X_i. The only restriction that we must impose is that the weights are a set of non-negative values verifying that

$$\sum_{j=1}^{r} \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}) = 1, \quad \forall\ pa(X_i)$$

It is interesting to note that by defining how to compute the weights $w(y_{k,l}, x_{i,j})$, we can control individual bias[6] and the relative quality (importance) of the parents for the predicting variable, X_i.

The problem now is how to estimate those weights given by similar users, i.e. $U_b \in Pa(V_a)$. Following (de Campos et al. 2008), we consider that $w(u_{b,t}, v_{a,s})$ (i.e. the effect of user U_b rating with value t when it comes to predicting the rating of V_a) can be computed by means of

$$w(u_{b,t}, v_{a,s}) = w_{b,a} \frac{N^*(u_{b,t}, u_{a,s}) + 1/r}{N^*(u_{b,t}) + 1}, \quad 1 \le t, \quad s \le r. \tag{6}$$

where the value $N^*(u_{b,t}, v_{a,s})$ is the number of items from the set $I(U_a) \cap I(U_b)$ that having been voted with value t by user U_b have also been voted with value s by user

[6] Bias refers to a user's preference for a particular vote (some users tend to rate with high values whereas others prefer to use lower ones) and ability to predict X_i judgments.

U_a, and $N^*(u_{b,t})$ is the number of items in $I(U_a) \cap I(U_b)$ rated with value t by user U_b. In this expression, $w_{b,a}$ represents the relative importance of the parent. In this paper, we assume that all the users are equally important, i.e. $w_{b,a} = 1/|Pa(V_a)|$.

In our example, and focusing on node V_2, we must estimate 2^3 conditional probability distributions. Using Eq. 5, the probability $Pr(v_{2,2}|u_{1,1}, u_{3,1}, u_{4,2})$ is equal to $w(u_{1,1}, v_{2,2}) + w(u_{3,1}, v_{2,2}) + w(u_{4,2}, v_{2,2})$. Using the data in Table 2, these weights are $w(u_{1,1}, v_{2,2}) = 0.278$, $w(u_{3,1}, v_{2,2}) = 0.167$ and $w(u_{4,2}, v_{2,2}) = 0.25$ and therefore $Pr(v_{2,2}|1, 1, 2) = 0.695$.

4.2 Modeling social value functions

The objective of this section is to consider how conditional probability distributions for group nodes can be estimated. These distributions might be considered a "social value function", describing how member opinions affect the group's recommendation. For instance, let G_i be a group with six individuals rating with 1, 5, 5, 4, 5 and 4, respectively, using a rating domain from 1 to 5. In this case, the related configuration is $pa(G_i) = (v_{1,1}, v_{2,5}, v_{3,5}, v_{4,4}, v_{5,5}, v_{6,4})$.

Since we must assess the probability of G_i voting with a value k for each possible configuration $pa(G_i)$ (i.e. $P(G_i|pa(G_i))$) and taking into account that the size of the group might be large, we again propose the use of canonical models. By means of these models, the probability values needed will be computed as a deterministic function of the particular values of the configuration, thereby entailing an important saving in storage.

Definition 2 (*Canonical gate*) A group node G_i is said to represent a canonical combination criterion if given a configuration of its parents $pa(G_i)$ the conditional probability distributions can be defined as

$$P(G_i = k|pa(G_i)) = f(k, pa(G_i))$$

Following the ideas in (O'Connor et al. 2001; Masthoff 2004), in this paper we will consider four alternatives:

4.2.1 MAX and MIN gates

In terms of (O'Connor et al. 2001), Maximum (Minimum) gates can be used to model those situations where the group vote is equal to the vote of the most satisfied (or least satisfied, respectively) group member. Thus, considering the example configuration, the group rating is equal to 5 and 1 for the MAX and MIN gate, respectively. Although these gates correspond to extreme situations, it is quite common for small groups to take into account these criteria when making decisions (Masthoff 2004). More formally, these gates can be defined as

$$f(k, pa(G_i)) = \begin{cases} 1 & \text{if } k = \Phi(pa(G_i)) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

Table 4 Using canonical models to define the conditional probabilities for the node G_a in Fig. 3

$pa(G)$	AVG	MAJ	MAX	MIN
$v_{1,1}, v_{2,1}, v_{3,1}$	(1,0)	(1,0)	(1,0)	(1,0)
$v_{1,1}, v_{2,1}, v_{3,2}$	(0.67,0.33)	(1,0)	(0,1)	(1,0)
$v_{1,1}, v_{2,2}, v_{3,1}$	(0.67,0.33)	(1,0)	(0,1)	(1,0)
$v_{1,1}, v_{2,2}, v_{3,2}$	(0,33,0.67)	(0,1)	(0,1)	(1,0)
$v_{1,2}, v_{2,1}, v_{3,1}$	(0.67,0.33)	(1,0)	(0,1)	(1,0)
$v_{1,2}, v_{2,1}, v_{3,2}$	(0.33,0.67)	(0,1)	(0,1)	(1,0)
$v_{1,2}, v_{2,2}, v_{3,1}$	(0.33,0.67)	(0,1)	(0,1)	(1,0)
$v_{1,2}, v_{2,2}, v_{3,2}$	(0,1)	(0,1)	(0,1)	(0,1)

The pairs (x_1, x_2) represent the probabilities $Pr(g_{a,1}|pa(G))$ and $Pr(g_{a,2}|pa(G))$, respectively

where $\Phi(pa(G_i))$ is the $\max\{pa(G_i)\}$ and $\min\{pa(G_i)\}$ for the MAX and MIN gates, respectively. For example, Table 4 shows the probability distribution obtained using these canonical gates for the node G_a in Fig. 3.

4.2.2 MAJority gates

Our objective in this section is to model the Majority criterion where the final decision depends on a simple counting of the votes received for each rating from the individuals. The rating which receives the largest number of votes is then selected as the consensus (majority) decision, i.e.

$$f(k, pa(G_i)) = \begin{cases} \frac{1}{m} & \text{if } k = \arg\max_s count(s, pa(G_i)) \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $count(s, pa(G_i))$ is a function returning the number of occurrences of the state s in the configuration $pa(G_i)$, and m is the number of states where $count(s, pa(G_i))$ reaches the maximum value. It should be noted that we are assuming that all members have the same power (i.e. one-person-one-vote). In the previous configuration, the group rating will be 5 because this was the rating given by 3 out of 6 individuals. See Table 4 for an example.

4.2.3 AVeraGe gates

Our objective with this gate is to model those situations where the group rating can be considered as the average of individual ratings. This criterion can be modeled in a similar way to before, i.e.

$$f(k, pa(G_i)) = \begin{cases} 1 & \text{if } k = AVG(pa(G_i)) \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

where $AVG(pa(G_i)) = round\left(\frac{1}{|Pa(G_i)|}\sum_j v_j\right)$, and where v_j is the rating of the jth-parent of G_i. Thus, in the above example, the group rating is defined as $AVG(1, 5, 5, 4, 5, 4) = 4$.

We should mention that although this might be the formal definition of the average gate, in this paper we use a canonical weighted sum-based representation which eventually attempts to recommend the same rating to the group but using much more efficient inference mechanisms. In particular, the weights $w(v_{j,k}, g_{i,s})$ are defined as $1/|Pa(G_i)|$ if $k = s$ and 0 otherwise. An example is shown in the first column in Table 4.

One important fact is that under this representation, the predicted rating must be the group's posterior expected (mean) value which is defined as the sum of the posterior probability of each rating multiplied by the rating value, i.e. $\sum_{s=1}^{r} s \times Pr(G_i = s|ev)$ (see Appendix A for more details). The predicted rating for the example configuration is therefore computed as $1 \times 1/6 + 2 \times 0 + 3 \times 0 + 4 \times 2/6 + 5 \times 3/6 = 4$. At this point, we should mention that whenever we talk about the AVG gate in this paper we are considering this representation.

5 Inference with canonical models

In this section, we will present the specially designed propagation algorithms to ensure efficient computations. Since we consider past user ratings as evidence, a top-down propagation mechanism can be designed, starting with those nodes at the user layer where $Pr(U_i|ev)$ is computed. These probabilities are then used to compute the posterior probabilities at the $\mathcal{V}$ layer (see Eq. 4) which are eventually used to compute the posterior probabilities for the group layer, i.e $Pr(G_a|ev)$ (see Eq. 3). Our GRS therefore follows the typical performance of a collaborative RS since these probability values depend on how similar users rated the item.

We will distinguish between the different canonical models used. When using canonical weighted sums, we can compute exact probability values in polynomial time (see Sect. 5.1). When aggregating individual preferences by means of any other canonical model, it is necessary to use the independence assumptions below in order to reduce the computations needed (see Sects. 5.2 and 5.3).

Independence Assumption: The collaborative ratings are independent given the evidence, i.e.

$$Pr(V_1, \ldots, V_k|ev) = \prod_i Pr(V_i|ev). \tag{10}$$

In view of this assumption, and considering that we use a canonical weighted sum model at the nodes in $\mathcal{V}$, the joint probabilities in Eq. 4 are computed in linear time. Although this assumption might be very restrictive, in our experimentation (see Sect. 7) it has proved to be fruitful and has also been used successfully when combining information for other practical purposes (Clemen and Winkler 1999; Kittler et al. 1998). This performance leads us to believe that it does not matter how accurate the estimates of the posterior probabilities are as long as they help to predict the correct ratings.

5.1 Propagating with the canonical weighted sum

When the canonical weighted sum is used to represent the interaction between variables, the posterior probabilities can be obtained simply by applying the following theorem (see de Campos et al. 2008) which explains how to compute the exact probability values:

Theorem 1 *Let X_a be a node in a BN network, let m_{X_a} be the number of parents of X_a, Y_j be a node in $Pa(X_a)$, and l_{Y_j} the number of states taken by Y_j. If the conditional probability distributions can be expressed under the conditions given by Eq. 5 and the evidence is only on the ancestors of X_a, then the exact posterior probabilities can be computed using the following formula*:

$$Pr(x_{a,s}|ev) = \sum_{j=1}^{m_{X_a}} \sum_{k=1}^{l_{Y_j}} w(y_{j,k}, x_{a,s}) \cdot Pr(y_{j,k}|ev).$$

It should be noted that propagation can be performed in linear time with the number of parents. We consider this theorem to be important because it expresses the fact that exact propagation can be performed without imposing any independence restriction between the parents of variable X_a (see de Campos et al. 2008). Because the evidences are in nodes in $\mathcal{U}$ in our recommender system, we will therefore obtain the exact posterior probabilities in all the nodes where the conditional probabilities are represented by means of a canonical weighted sum, as will be the case of the nodes in $\mathcal{V}$ and also when modeling the average criterion in the group nodes.

For example, when we want to predict the rating given by G_a in Fig. 3 to item I_7, and considering the past ratings in Table 2, we must compute $Pr(G_a = 1| u_{0,2}, u_{4,1}, u_{5,2})$. In this situation, the exact posterior probabilities at the $\mathcal{V}$ layer are: $Pr(v_{1,1}|ev) = w(u_{0,2}, v_{1,1}) + Pr(u_{2,1}) \cdot w(u_{2,1}, v_{1,1}) + Pr(u_{2,2}) \cdot w(u_{2,2}, v_{1,1}) = 0.277$; $Pr(v_{2,1}|ev) = 0.341$ and $Pr(v_{3,1}|ev) = 0.36$. Similarly, if the group uses an AVG criterion to combine information (represented by a weighted sum gate), the exact posterior values at group nodes are $Pr(g_{a,1}|ev) = 0.326$ and $Pr(g_{a,2}|ev) = 0.674$ and the predicted rating is $round\ (1 \times 0.326 + 2 \times 0.674) = 2$.

5.2 Propagating with majority gates

One key idea behind the majority criterion is that the order in which the individuals are considered does not matter, and therefore there are many different configurations collapsing to the same situation. For example, let us consider that four individuals vote with 1 and one individual votes with 2. In this case, there are five different configurations representing the same situation, i.e. $pa_1(G_i) = \{2, 1, 1, 1, 1\}$, $pa_2(G_i) = \{1, 2, 1, 1, 1\}$, $pa_3(G_i) = \{1, 1, 2, 1, 1\}$, $pa_4(G_i) = \{1, 1, 1, 2, 1\}$ and $pa_5(G_i) = \{1, 1, 1, 1, 2\}$. It should be noted that since order is not a factor, we might talk about combinations with repetition,[7] denoted by δ. Therefore, the above configurations should be represented by $\delta(G_i) =< 1, 1, 1, 1, 2 >$.

In this situation, all the probabilities $Pr(G_i = s|pa_j(G_i))$, such that pa_j can be matched to the same combination δ, have the same values (in our case, $Pr(G_i = 1|pa_j(G_i)) = 1, \forall 1 \leq j \leq 5$). This can be exploited in order to efficiently perform the propagation processes in Eq. 3. In particular, the following theorem shows that we need only take into account those probabilities associated with combinations with repetition in order to propagate individual rating probabilities:

Theorem 2 *Let G_i be a group node in a BN whose conditional probability distributions are represented using a majority gate, let $\Delta(G_i)$ be the set of possible combinations repeating the values in its parent set, $Pa(G_i)$, then*

$$Pr(G_i = s|ev) = \sum_{\delta(G_i)\in\Delta(G_i)} Pr(G_i = s|\delta(G_i))Pr(\delta(G_i)|ev) \tag{11}$$

The proof of this theorem can be found in Appendix B. Although this theorem reduces the number of necessary computations in Eq. 3, an exponential number of computations will be needed in order to obtain the joint probabilities $Pr(\delta(G_i)|ev)$.

In order to ensure the scalability of the approach, we will approximate these values by considering that collaborative ratings are independent given the evidence (Eq. 10). In Appendix B, we show how to compute the required probabilities in a running time on the order of $O(rn^r)$. Taking into account that in many situations $r << n$, this entails important savings with respect to the $O(r^n)$ needed by the classical approach.

Following on with the previous example, we will have four possible combinations for V_1, $V2$ and V_3 where (assuming independence) the posterior probabilities are $Pr(< 1, 1, 1 > |ev) = 0.034$, $Pr(< 1, 1, 2 > |ev) = 0.215$, $Pr(< 1, 2, 2 > |ev) = 0.446$ and $Pr(< 2, 2, 2 > |ev) = 0.305$. Then, following a majority strategy, $Pr(G_a = 1|ev) = 0.034 + 0.215 = 0.249$ and $Pr(G_a = 2|ev) = 0.751$. Once again, the recommended rating is 2 as this is the most likely posterior probability.

5.3 Propagation with MIN and MAX gates

When propagating with MAX and MIN gates, we will also assume that a posterior probability for the collaborative nodes is independent given the evidence. Once these values have been computed, we still need to combine them with $Pr(g_{a,s}|pa(G_a))$ in order to obtain the final probability distributions. It can be proved that under the above independence assumption (Eq. 10), the probability distribution $Pr(G_a|ev)$ can be computed easily and efficiently (in a linear order to the number of group members).

Min-Gate Assume $1 < 2 < \cdots < r$
- $Pr(G_a = r|ev) = \prod_{i=1}^{m} Pr(V_i = r|ev) = \prod_{i=1}^{m} Pr(v_{i,r}|ev)$

[7] Since the number of parents in G_i is n and each parent has r different states, we find that the number of possible combinations with repetition is $CR_n^r = (n + r - 1)!/(n!(r - 1)!)$.

– $Pr(G_a = k|ev)$, for $k = 1, \ldots, r-1$, is equal to

$$\left(\prod_{i=1}^{m} Pr(V_i \geq k|ev)\right) - Pr(G_a > k|ev).$$

where $Pr(X \geq k) = \sum_{j=k}^{r} Pr(X = j)$.
Max-Gate Assume $1 < 2 < \cdots < r$
– $Pr(G_a = 1|ev) = \prod_{i=1}^{m} Pr(V_i = 1|ev) = \prod_{i=1}^{m} Pr(v_{i,1}|ev)$
– $Pr(G_a = k|ev)$, for $k = 2, \ldots, r$, is equal to

$$\left(\prod_{i=1}^{m} Pr(V_i \leq k|ev)\right) - Pr(G_a < k|ev)$$

where $Pr(X \leq k) = \sum_{j=1}^{k} Pr(X = j)$.

If we consider our example, the posterior probabilities for the MIN gate are $Pr(G_a = 2|ev) = \prod_{i=1}^{3} Pr(v_{i,2}|ev) = 0.305$; $Pr(G_a = 1|ev) = 0.695$ and the recommended rating is 1, whereas if we consider the MAX gate $Pr(G_a = 1|ev) = \prod_{i=1}^{3} Pr(v_{i,1}|ev) = 0.034$; $Pr(G_a = 2|ev) = 0.966$ and the decision is to recommend the rating 2.

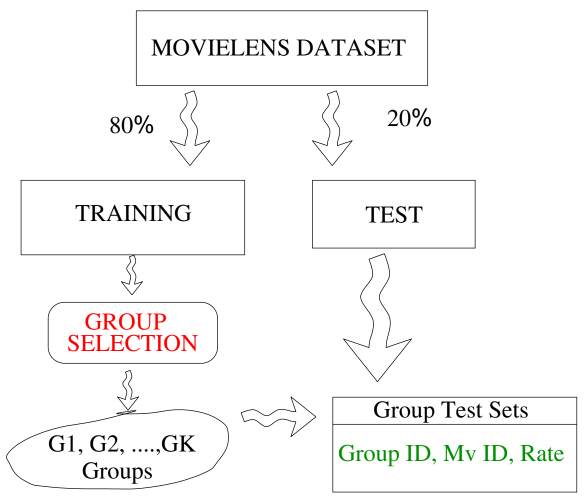
6 Experimental framework

There are various reasons why GRSs are difficult to evaluate. The first is that the evaluation criterion differs according to the goals for which the RS has been developed. We can find situations where the criteria used to measure system performance are user satisfaction or participation (O'Connor et al. 2001; Masthoff 2004; McCarthy and Anagnost 2000) whereas in other GRSs the objective is to explore the ability of the system to merge user profiles or reach a consensus (Jameson 2004; Yu et al. 2006).

The second reason why evaluation is difficult is the absence of public data sets for performing the evaluation. Most work into GRS evaluation has focused on live user experiments. In this case, we can distinguish between the work using controlled groups that have been (directly or indirectly) asked about the aggregation strategies they might use (Masthoff 2004; Masthoff and Gatt 2006) and those systems, such as Polylens (O'Connor et al. 2001) and MusicFX (McCarthy and Anagnost 2000), which have made the system available to a community of users. In this case, field studies were performed to evaluate system performance. The combination of these two factors makes it extremely difficult to compare the performance between different GRSs.

This situation clearly differs from that of recommending for a single user. In this context, the evaluation of system accuracy, i.e. the system's ability to predict individual ratings, has become a standard approach (Herlocker et al. 2004). Surprisingly, little attention has been paid in GRS literature to evaluating system accuracy. We believe that one of the main reasons for this is that it is difficult to access real group ratings since in many cases group composition is ephemeral, and, to the best of our

Fig. 4 Building the group data sets



knowledge, no data sets exist with this kind of information. On the other hand, we can consider that in real scenarios each group might rate using different aggregation strategies. For instance, some groups might use a *least misery strategy* whereas others might use an *average strategy*. In these situations, a blind algorithm, which does not take into account how the group rates an item, may be inappropriate for predicting group ratings.

Our goal in this experimentation is to discover how the use of the different uncertainties emerging in a group recommending process might affect system performance. We believe the best alternative for measuring system performance is to take into account accuracy criteria in an automatic evaluation over a semi-simulated data set (see below). We believe that by means of this evaluation, we can conduct large experiments that should validate our conclusions.

6.1 The data sets

We decided to use the MovieLens[8] data set. With the idea of using 5-fold cross validation, we have used 5 different data subsets, each obtained by splitting MovieLens into two disjoint sets, the first for training (with 80% of the data) and the second for testing (with 20% of the data).

Training data has been used for two different purposes. Firstly, we have used this data to learn the collaborative component of the system. Following (Herlocker et al. 1999), we have considered a fixed number of 10 parents (similar users) for each node in $\mathcal{V}$ (see Sect. 3.1). Using this component independently, we might compute for each member of a group, V_k, a probability distribution representing how this individual might rate an unseen movie, i.e. $Pr(V_k = s|ev)$. Training data has also been used to determine group composition, i.e. which individuals form a group (see Fig. 4).

[8] MovieLens was collected by the GroupLens Research Project at the University of Minnesota. The data set contains 1682 movies and 943 users, containing 100,000 transactions where a user rates a movie using a scale from 1 to 5.

We have used two different criteria in an attempt to capture different processes behind the creation of a group:

C1 Implementing the idea of *the group of my buddies*, we set each user as the group administrator and look for similar users (those who are positively correlated with the administrator in the training data set). We then select those groups for a fixed number of individuals (let us say n) with the only restriction being that they have at least rated (seen) one common movie (considering the experimental data, the members of the groups have rated a mean of 13.63 common movies). It should be noted that since similarities are not transitive, this criterion does not necessarily imply that groups have highly correlated members.

C2 Secondly, we have decided to fix a group (also with n individuals) with the only restriction being that all group members must rate at least four common movies in the training sets, independently of the given ratings. This alternative can be used to represent *circumstantial groups* such as those obtained by randomly selecting people as they leave a cinema. With this criterion, it is plausible that while individuals might watch a movie together, they might have different preferences.

The group test sets are obtained from each of the original MovieLens test sets, thereby ensuring that no data in the test sets has been used for learning purposes. Whenever we find a movie in an original MovieLens test set that has been rated by every group member, we include the tuple (*group ID, movie ID, group-rating*) in the respective group test data set. Since we know the real user ratings for this movie, the "true" group-rating will be obtained by considering the mechanism used by the group to aggregate the individuals ratings. This mechanism might be implemented by means of a deterministic function $\texttt{CombineRate}(r_1, \ldots, r_n)$.[9] For example, given a group of five individuals and the ratings $r_1 = 5, r_2 = 5, r_3 = 2, r_4 = 3, r_5 = 4$, then depending on the aggregation function used, the "true" group rating will be: 5 for the *majority* function, 2 for the *minimum* criterion, 4 if we consider an *average* strategy, etc.

By combining the strategy used by a group to rate a movie, i.e. average (AVG), majority (MAJ), maximum (MAX) and minimum (MIN), and the criterion used to form a group (C1 and C2), we therefore obtain 8 different data sets. More specifically, by fixing the group size to 5, we have found a mean of 108 (406) different groups and 115 (1752) group-movie pairs in the test set for C1 criterion (C2 criterion, respectively).

7 Experimentation: predicting a group's rating

The aim of this experimentation is to measure the effect of individual uncertainties on each different aggregation criterion. In order to focus on this aim, we shall assume that we know how the group combines individual ratings, i.e. we know whether the group rating is obtained by means of the *AVG, MAJ, MAX* or *MIN* of a set of individual ratings (what happens when this information is unknown will be discussed in Sect. 8).

[9] A similar mechanism for building the group rating has been used in Chen et al. (2008) to measure the effect of sparsity on the recommendations.

Table 5 Baseline and BN-based group aggregation mechanisms

Baseline	Using group layer
For each $V_k \in G$ do	$Pr(G\|ev) = \texttt{CombineProb}(Pr_1, \ldots, Pr_n)$
$r_k = \texttt{RateSelection}(Pr_k)$	$\texttt{G\_rate} = \texttt{RateSelection}(Pr(G\|ev))$
$\texttt{G\_rate} = \texttt{CombineRate}(r_1, \ldots, r_n)$	

The following method is proposed: for each group-item pair in the group test set, the first step is to instantiate those users who had rated the movie with the given rating (these ratings will be considered as the *evidence*, see Sect. 3.3). This evidence will be then propagated to the collaborative nodes, $\mathcal{V}$, representing the group's members. This information must finally be combined in order to obtain the predicted rating. At this point, it should be remembered that we assume that no member of the group has seen the movie and it is not possible to know how a node V_i might rate the movie. We will consider two different alternatives to combine this uncertain information:

– This alternative, which could be considered the *Baseline* (see left-hand side of Table 5), is a two-step approach: firstly, each group member makes a decision about the score that he or she might use to rate the movie (`RateSelection` process). These $r_1, \ldots, r_n$ precise ratings are then combined using the particular combination strategy (AVG, MAJ, MAX or MIN) used by the group when making decisions.
– The second method consists in first using canonical models to combine individual probability distributions into a single group distribution. This probability distribution ideally represents the group voting pattern. In the algorithm (see right-hand side of Table 5) this process is denoted as `CombineProb`. The group probability distribution is then used to select the final group rating (once again using the `RateSelection` process).

Finally, we will consider how the predicted rating might be selected, i.e. how `RateSelection` works. In this paper, we will explore two different alternatives:

– The first uses the raw probability values at group nodes (or collaborative nodes for the baseline approach). In particular, the group rating is defined as the posterior average rate, i.e. $rate = \sum_{k=1}^{r} k \times Pr(G = k|ev)$, for the AVG canonical model (see Sect. 4.2.3) and the maximum posterior probability, i.e. rate $= \arg\max_s \{Pr(G = s|ev)\}$, for MAX, MIN and MAJ canonical models.
– The idea behind the second alternative is to only take into account the new piece of evidence that each candidate rating receives. This criterion can be computed by taking into account the difference between the posterior and prior probability values,[10] i.e. $PD(G_a) = Pr(G_a|ev) - Pr(G_a)$. We consider only those ratings where the difference is positive, i.e. the evidence favors these ratings. We have to notice that PD is not a probability measure since $\sum_s PD(G_a = s) < 1$.

[10] It is worth remembering that posterior probabilities are obtained by instantiating the ratings previously given by similar users (the evidence belongs to the nodes in $\mathcal{U}$) and that prior probabilities are obtained by propagating in the network without evidence.

Table 6 Effect of user uncertainty on the combination strategies

Group rate	RS	C1				C2			
		Baseline		BN-Group		Baseline		BN-Group	
		%S	MAE	%S	MAE	%S	MAE	%S	MAE
AVG		62.111	0.387	44.945+	0.596+	63.730	0.369	59.037+	0.420+
	PD	**62.500**	**0.385**	62.060−	0.396−	64.045	0.365	**64.355**−	**0.363**−
MAJ		59.127	0.464	59.543+	0.430−	56.548	0.484	58.621+	0.441+
	PD	59.044	0.464	**59.745**+	**0.428**−	57.447	0.472	**58.926**+	**0.437**+
MAX		77.397	0.226	73.748−	0.285+	68.740	0.319	**75.353**+	**0.263**+
	PD	77.538	0.225	**78.991**+	**0.211**−	69.501	0.311	70.272+	0.300+
MIN		44.964	0.674	13.739+	1.648+	46.987	0.681	29.509+	1.032+
	PD	**46.195**	0.666	42.876+	**0.662**+	46.539	0.684	**49.780**+	**0.578**+

For example, let us consider that the prior probability values (in a rating domain from 1 to 3) are $P(g_{a,1}) = 0.1$, $P(g_{a,2}) = 0.2$ and $P(g_{a,3}) = 0.7$ and let us consider the following posterior values $P(g_{a,1}|ev) = 0.30$, $P(g_{a,2}|ev) = 0.25$ and $P(g_{a,3}|ev) = 0.45$. Thus, if we were to select the maximum posterior rating we would recommend the rating 3 whereas evidence seems to favor the rating 1. Therefore, considering PD, the new measure assigns a mass 0.20, 0.05 and 0 to ratings 1, 2 and 3, respectively. In order to standardize the recommending process, this new measure might be transformed into a probability by means of a proportional normalization process.

The effect of using uncertainty will be measured by considering the accuracy of the recommendations in the two previous situations. We have considered two different metrics (Herlocker et al. 2004): the percentage of success (%S), which measures the frequency with which the system makes correct predictions; and the mean absolute error (MAE), which measures the average absolute deviation between a predicted rate and the group's true rate.

Table 6 shows the results obtained for the different experiments. The first column represents the criterion used by the group to determine the group rate. Each row represents the results obtained when using this criterion to combine the information with the baseline model and when using the group layer in the BN, denoted by the BN-Group. The rows labeled PD in the second column represent the results obtained when considering the difference between prior and posterior probability distributions. We have highlighted in bold the best alternatives for each particular situation. We use the signs + and − throughout this paper to represent the fact that the results are significantly relevant or irrelevant, respectively, in relation to the baseline model, by using the paired Student's t-test (confidence level 0.05).

From this table, we can conclude that when combining uncertain information using BN at a group layer the best option is to use PD to correct the prior bias, particularly in those situations where the minimum and the average gates are used to merge individual ratings. The exception is the use of MAX gate (using the C2 data set) where

Table 7 Using a classical collaborative filtering algorithm

Group rate	C1		C2		Group rate	C1		C2	
	$\%S$	MAE	$\%S$	MAE		$\%S$	MAE	$\%S$	MAE
AVG	49.080+	0.538+	57.986+	0.422+	MAJ	52.713+	0.526+	56.562+	0.478+
MAX	74.655+	0.258+	67.920+	0.328+	MIN	38.255+	0.761+	39.258+	0.760+

the baseline model performs better. In terms of the use of PD in the baseline model, it seems that the results have not been affected significantly. We can also conclude that taking individual uncertainties into account helps to improve the recommendations (we obtain better MAE (%S) values in 7 out of 8 (6 out of 8) experiments). Additionally, the results in Table 6 show that it is possible to order the different aggregation strategies in relation to their accuracy: $MAX \prec AVG \prec MAJ \prec MIN$, where $X \prec Y$ means that X obtains better predictions than Y. We believe that this is due to the bias that MovieLens has towards high rating values (approximately 75% of the ratings have a value which is greater than or equal to 3) and that, in some respects, our collaborative model inherits this bias in the parameters' learning phase. Finally, we can conclude that the way groups are formed is relevant for prediction purposes. Therefore, MIN and AVG canonical models obtain better results when considering circumstantial groups (C2 group set) whereas MAX and MAJ canonical models obtain better results in the case of groups with related individuals (C1 group set).

Finally, and in order to validate our approach, we will compare our predictions with those obtained with a classical collaborative filtering algorithm (Herlocker et al. 1999) under the same conditions.[11] Following our baseline scheme, we first use the classical model to predict how individual users will rate specific items. These predictions are then aggregated by using the corresponding functions. Table 7 presents the obtained results for each execution. From this table, we can conclude that our model is an improvement on those obtained using a classical approach for recommending.

7.1 Effect of the group's size

We will now explore the effect of group size on the recommendations. The experiments were repeated but this time with groups comprising 3, 4, 5 and 6 individuals, except under criterion C1, *groups of my buddies*, where no groups with 6 members could be found. Table 8 presents the MAE metrics obtained after evaluating our models in the same conditions as before using C1 and C2 data sets, respectively. We only show the MAE values obtained using PD for each pair 'XXX#s', where XXX represents the aggregation strategy and #s the group's size, respectively.

[11] In this classical model, the similarity between users is computed using Pearson's correlation measure (Eq. 2). Once similarities are ready, predicting how user U_a will rate an item I_j can be calculated as $rate = \bar{r}_a + \frac{\sum_{h=1}^{m'} sim(U_a,U_h)(r_{h,j}-\bar{r}_h)}{\sum_{h=1}^{m'} |sim(U_a,U_h)|}$ where m' is the number of related users who have also rated item I_j.

Table 8 MAE results for different sized groups

C1 datasets

	AVG3	AVG4	AVG5	AVG6	MAJ3	MAJ4	MAJ5	MAJ6
Base	0.490	0.454	0.385		0.482	0.488	0.464	
BNPD	0.481	0.424	0.396		0.412	0.431	0.428	
	MAX3	MAX4	MAX5	MAX6	MIN3	MIN4	MIN5	MIN6
Base	0.407	0.310	0.225		0.738	0.742	0.666	
BNPD	0.389	0.308	0.211		0.663	0.649	0.662	

C2 datasets

	AVG3	AVG4	AVG5	AVG6	MAJ3	MAJ4	MAJ5	MAJ6
Base	0.469	0.417	0.365	0.327	0.477	0.497	0.472	0.427
BNPD	0.470	0.398	0.363	0.334	0.417	0.450	0.437	0.378
	MAX3	MAX4	MAX5	MAX6	MIN3	MIN4	MIN5	MIN6
Base	0.425	0.362	0.311	0.203	0.724	0.693	0.684	0.716
BNPD	0.398	0.341	0.300	0.199	0.644	0.607	0.578	0.604

It is possible to draw certain conclusions from this data. Firstly, we can say that the relative performance between both the baseline and the proposed BN-based group aggregation model is stable for almost all the experiments (27 out of 30). We can therefore say that it is preferable to combine uncertain information using BN, independently of the size of the group. Nevertheless, group size has an important impact on the quality of the model's predictions. This impact seems to be independent of the criteria used to create the group, C1 or C2, but not on the aggregation strategy used by the group to recommend the final rating. Accordingly, AVG and MAX obtain significantly better predictions (lower MAEs) as the group size increases (this also seems to be the case of MAJ when using C2 data sets) whereas the MIN criterion seems to be more or less stable. For AVG and MAJ combination criteria, this situation might be explained by the fact that as the number of members increases, there is a reduction in the impact of each member on the group prediction. In terms of the performance of extreme rating criteria such as MAX and MIN, this can be explained by considering the bias for rating with higher values in MovieLens, with it being easier to predict high rating values.

8 Uncertainties in the group's rating processes

In the previous section, we assumed that we know how a group combines the information and focused on the effect of uncertainty on user preferences when predicting the group's rating. In this section, we will consider that the process used by the group to rate a given product is also uncertain. In order to tackle this problem we will consider two different situations:

– The first is a situation of total ignorance, i.e. we do not know anything about how the group should combine the information.

Table 9 Predicting without information

Group rate	C1					C2				
	AVG+PD		MAJ+PD		LinOP	AVG+PD		MAJ+PD		LinOP
	Base	BN-G	Base	BN-G	–	Base	BN-G	Base	BN-G	–
AVG	0.385	0.396−	0.486	0.385+	0.386	0.365	0.363−	0.486	0.389+	0.388
MAJ	0.472	0.458−	0.464	0.428−	0.439	0.453	0.440+	0.472	0.437+	0.439
MAX	0.799	0.727+	0.740	0.709+	0.718	0.854	0.808+	0.793	0.788−	0.796
MIN	1.140	1.162+	1.192	1.167+	1.175	1.186	1.220+	1.283	1.243+	1.236
Mean	0.699	0.686	0.721	**0.672**	0.680	0.715	**0.708**	0.759	0.714	0.715
Dev	0.344	0.348	0.338	0.360	0.361	0.380	0.393	0.380	0.395	0.392

– The second considers that although we do not know exactly how a group combines the information, we have a database of previous group ratings. In this case, it might be possible to discover the mechanism used by a group to rate an item from the database of past ratings.

8.1 Total ignorance

Our objective is to study which of these proposed aggregation models is best if we do not know how the group combines the information. This situation, which might be related with the Cold-Start problem, is common when a new group is incorporated into the system and therefore the decision processes are blind. Nevertheless, we have tried to study whether the use of the proposed canonical model might be helpful or not under these circumstances.

Thus, given a new group, it might not be appropriate to determine the group rating by means of extreme canonical models such as MIN, MAX (as confirmed by preliminary experimental results). We will therefore attempt to determine the best option between combining the results using an average or a majority criterion (which in some respects relates to recommending the mean or mode value, respectively). We will also compare our results with those obtained using the classical linear opinion pool (LinOP) where all the users have been considered equivalent for prediction purposes.[12] In this case, the recommended rating is the one that obtains the maximum posterior probability.

In order to study the performance of MAJ and AVG in this situation, we have decided to measure the accuracy of MAJ+PD and AVG+PD when predicting group ratings under the four different combination mechanisms. As in Sect. 7, the experimentation has been conducted by considering the two criteria for forming the groups (groups of buddies and circumstantial users). Table 9 presents the results (we only

[12] It should be noted that this model is equivalent to considering a weighted sum canonical model using an unbiased uniform weighting scheme, i.e. the weights are defined as $w(v_{j,t}, g_{i,s}) = w_j$ if $t = s$, and 0 otherwise). The linear opinion pool can therefore be considered a particular case of the canonical weighted sum gate.

show the MAE metrics[13]). The last two rows of this table contains the mean values and standard deviation for the experiments.

From this table, we can conclude that it is safer to use the canonical models to combine information, and for majority gates, in particular, the MAE values are 6% better for both C1 and C2. Taking into account the uncertainties at an individual level therefore helps to improve the prediction. As before, the model performs differently when consideration is taken of how the groups are formed. On average, all the models obtain better results when groups of buddies (C1) are used than when groups of circumstantial members (C2) are considered. When using C1, the use of MAJ+PD criterion seems preferable to AVG+PD (statistical significance were found) and this might imply that in the case of groups with similar users it is better to use the mode value. When using C2, however, although it was better to use AVG+PD, there is no statistical significance with respect to MAJ+PD. Finally, we would like to mention that we have also studied the performance of the LinOP model when attempting to reduce the effect of the prior probabilities, but in this case no significant differences were obtained (the same mean results were obtained).

8.2 Learning how the groups rate

The aim of this section is to study the use of automatic learning algorithms to represent the group's profiles and their effect on the group's recommendations. There are two main points which must be considered: the first is that we are not imposing any restriction on how a group rates an item, and therefore the learning models might be general; and the second is related to efficiency and scalability since, as we have seen, these are both important in group recommending.

This paper will therefore explore the following two alternatives:

NB The first method uses a Naive Bayes (NB) classifier (Duda and Hart 1973) since it performs well on many data sets, and is simple and efficient. This modeling might be related to the classical Bayesian approach for combining probability distributions. In our context, predicting the group rating can be viewed as a classification problem where the group rating is the class variable and the individual ratings would be the attributes. NB assumes that individual ratings are independent given the group rating and has the advantage that no structural learning is required. In this case, the rating prediction comes down to finding the $rate$ such that

$$rate = \arg\max_s P(G = s) \prod_i Pr(v_i | G = s)$$

where v_i is the rating selected for the ith member of the group, V_i, by means of a `RateSelection(`$Pr(V_i|ev)$`)` process as in the previous baseline model (see left-hand side of Table 5).

[13] We have decided not to include the percentage of success in order to reduce the size of the tables. It should be noted that it appears to be correlated with the MAE values, as previously seen in the experiments.

The only parameters that must be estimated from the data sets of past group ratings are the group's prior probability, $P(G)$, and each member's rating conditional probability given that the group has rated with a certain value, $Pr(V_i|G=r)$. These probabilities can easily be estimated from the data sets using Add-$1/r$ smoothing such that

$$Pr(G_a = s) = \frac{N(g_{a,s}) + 1/r}{N(G_a) + 1}$$

where $N(g_{a,s})$ is the number of times that the group rating is s and $N(G_a)$ is the number of times that the group has rated.
For the nodes in $\mathcal{V}$, the conditional probabilities are estimated using

$$Pr(V_i = t|G = s) = \frac{N(v_{i,t}, g_{a,s}) + 1/r}{N(g_{a,s}) + 1}$$

where $N(v_{i,t}, g_{a,s})$ is the number of times that the user rate is t when the group rating is s .

WSG The second alternative consists in using the canonical weighted sum model presented in this paper (see Definition 1) to represent the combination process. In this case, we are assuming that the group rating is determined by aggregating the preferences of the group's members and the only independence restriction that we impose is that since we know how the group's members rate an item, the group rating is independent of the other information sources. As with NB, no structural learning is required in this model.
We must therefore determine each individual's effect on the group rating. In particular, looking at Eq. 5, we must estimate $w(v_{i,k}, g_{a,s})$, i.e. the effect that user V_i rating with the value k has on the group G_a rating with the value s. These weights might be defined as the ratio

$$w(v_{i,k}, g_{a,s}) = \frac{N(v_{i,k}, g_{a,s}) + 1/r}{N(v_{i,k}) + 1}.$$

The main advantages of both models rest on the assumptions used, which reduce the number of parameters to be estimated in several orders of magnitude and also facilitate the inference processes. Both assumptions are, however, the main drawbacks of the models since they are not realistic in this domain.

8.2.1 Experimentation: learning group rating pattern

In this experiment, we will focus on the ability of the proposed models to learn how the groups rate. We have therefore conducted the same experiments as before, but considering the learned group profiles. Table 10 presents the results obtained with both models when considering different combination mechanisms of the group. More specifically, we only show the results obtained using WSG with PD strategy (WSG+PD).

Table 10 Learning parameters from the database of cases

Group rate	C1				C2			
	Naive Bayes		WSG+PD		Naive Bayes		WSG+PD	
	%*S*	MAE	%*S*	MAE	%*S*	MAE	%*S*	MAE
AVG	55.995	0.574	57.456−	0.442+	57.727	0.471	62.222+	0.387+
MAJ	52.588	0.637	56.037−	0.488+	51.477	0.583	56.180+	0.487+
MAX	72.171	0.449	77.702+	0.244+	67.300	0.407	72.689+	0.284+
MIN	38.738	0.804	42.829+	0.710+	39.955	0.781	45.213+	0.663+
Mean	54.873	0.616	**58.506**	**0.471**	54.115	0.561	**59.076**	**0.455**
Dev	13.736	0.148	14.393	0.191	11.465	0.164	11.486	0.161

In addition, in terms of rate selection for WSG+PD, the average rating over the group probabilities is recommended,[14] i.e. $rate = round\left(\sum_{k=1}^{r} k \times Pr(G = k|ev)\right)$.

From this table, we can conclude that WSG+PD outperforms the Naive Bayes model in all the experiments. Finally, we will compare the results with those obtained in Table 6, which could be considered our goal since in this case we are using the same aggregation strategy as that used by the group. Although we always obtain better results as Table 6 shows, we can conclude that by learning the group rating pattern, we might make predictions which are almost ideal. For instance, the difference between the best MAE values are 0.046, 0.06, 0.033 and 0.048 (0.024, 0.05, 0.021 and 0.086) for AVG, MAJ, MAX and MIN for C1 data sets (C2 data sets, respectively). Although these differences are statistically significant, they give some idea of the ability of the WSG to map different aggregation criteria. This is important because it implies that WSG can be applied safely in those situations where the aggregation criterion used by the group is not known.

9 Conclusions

In this paper, we have proposed a general BN-based model for group recommending and this is an intuitive representation of the relationships between users and groups. The topology of the BN represents those dependence and independence relations considered relevant for modeling the group recommending processes. We should emphasize that only in situations where the required computation is quite complicated have we considered assumptions to reduce computational complexity. Although these assumptions might not be realistic in a given domain, they are necessary if we want to apply the proposed methodology in real applications. The experimental results show the viability of our approach.

With the proposed model we can therefore represent both uncertainty relating to the user's personal views about the relevance of an item, and also uncertainty

[14] The other alternatives were the combination of WSG without PD and the use of the most probable criterion to decide the predicted rating, but worse results were obtained.

relating to the mechanisms used by the group to aggregate individual preferences, mechanisms which are encoded by means of conditional probability distributions stored at the group nodes. In terms of efficiency, these distributions have been assessed by means of canonical models. The use of these models also allows the posterior probabilities to be computed in linear time and this is something which is necessary for deciding the recommended rate. Guidelines have been given for how to estimate the probability values from a data set and how the users interact with the RS. Experimental results demonstrate that by taking uncertainty into account at the individual level when aggregating, better prediction for the groups can be obtained. In addition, the results obtained determine other factors which affect system performance such as how the group is created, the number of individuals in the group, the aggregation function used, etc. It should be noted that the proposed model is quite general since it can be applied to different recommendation tasks (such as *find good items* or *predict ratings*) for a single item or for a set of items. Moreover, although this paper is set within the framework of the group recommending problem, the results presented can easily be extended to those disciplines where aggregating information represents an important component, and these disciplines include statistics, decision theory, economics, political science, psychology, etc.

By way of future work, we will attempt to incorporate mechanisms to enable consensus to be reached between group members. In this respect, we can say that selecting the rating r_i with probability 0.55 is not the same as selecting the rating r_i with probability 0.95. It seems clear that the use of these uncertainties will be helpful for reaching consensus in group decisions. We also plan to encode different strategies such as *ensuring some degree of fairness* (Masthoff 2004; Jameson and Smyth 2007) by means of a BN, for instance to ensure that at least 75% of users were satisfied. Another problem worthy of study is to determine in what circumstances it is possible to discover how a group rates a given item, such as for example how to consider the effect of context in the group rating pattern.

Acknowledgements This work has been jointly supported by the Spanish Ministerio de Educación y Ciencia, Junta de Andalucía and the research program Consolider Ingenio 2010, under projects TIN2005-02516, TIC-276 and CSD2007-00018, respectively.

Appendix

This appendix will include any technical results that are not necessary to understand the insights of the model but which are necessary to follow the mainstream of the paper.

A Modeling the average rating with canonical weighted sum gates

The average gate models those situations where the group rating can be considered as the average rating of its members. Although representing this combination criterion using this gate implies important savings in storage, a huge number of computations are required in the inference processes. In this appendix, we will illustrate how the

Table 11 Probability values for average gate and CWS-based criteria for configurations $c_1 = (1, 1, 2, 1, 1, 2)$, $c_2 = (1, 2, 2, 3, 3, 5)$ and $c_3 = (1, 5, 5, 4, 5, 4)$.

	AVG gate: $Pr(G = k\|c_i)$					CWS-based: $Pr(G = k\|c_i)$				
	1	2	3	4	5	1	2	3	4	5
$c1$	1	0	0	0	0	0.666	0.333	0	0	0
$c2$	0	0	1	0	0	0.166	0.333	0.333	0	0.166
$c2$	0	0	0	1	0	0.166	0	0	0.333	0.5

same recommended rating might be computed efficiently by considering a canonical weighted sum-based representation of the average rating in a two-step approach:

– Firstly, for a given configuration $pa(G_i)$, define the probability of the group rating with the value k as the ratio of the number of its members which would rate with k, i.e.

$$Pr(G_i = k|pa(G_i)) = \frac{\sum_{V_j \in Pa(G_i)} R(pa(G_i), V_j, k)}{|Pa(G_i)|}$$

with $R(pa(G_i), V_j, k)$ equal to 1 if user V_j rates with value k in the configuration $pa(G_i)$ and 0 otherwise. This definition is equivalent to aggregating individual ratings by means of a canonical weighted sum approach (see Definition 1) where all users are assumed to be equal for prediction purposes[15] with no bias in the user ratings. These weights might therefore be defined as follows

$$w(v_{j,t}, g_{i,k}) = \begin{cases} \frac{1}{|Pa(G_i)|} & \text{if } k = t, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

In Table 11, we present both the probability values for the AVG gate using Eq. 9 and those obtained using Eq. 12 (column labeled with CWS-based) for three example configurations.

– Secondly, determine the group rating as the one obtained by averaging over the group probabilities as

$$rate = round\left\{\sum_{k=1}^{r} k \times Pr(G_i = k|pa(G_i))\right\} \tag{13}$$

Thus, continuing with our example, for configuration c_1 the group rate is $round\{1.333\} = 1$, for configuration c_2 $round\{2.6666\} = 3$, and for configuration c_3 $round\{4\} = 4$, which are the same rating values as those that will be recommended using the average over individual ratings.

[15] It should be noted that in this case we do not consider situations where there are users with high quality opinions (experts). Nevertheless, these could easily be taken into account by adequately modifying the weights.

We should mention that by using a canonical weighted sum representation, we inherit the computational advantages of the canonical weighted sum models in the inference processes (see Sect. 3.3). When we refer to the average canonical model in this paper, denoted by AVG, we are therefore referring to the fact that the group rating is obtained using the conditional probabilities presented in Eq. 12 and that the selected rating is the one obtained using Eq. 13.

B Propagating with majority gates

In this section we will present the computations necessary to understand how the new evidences can be propagated efficiently using majority gates.

Since order is not a factor in the majority criterion, we might speak of combinations with repetition. We will use $\Delta(G_i)$ to denote the set of combinations with repetition from the individual votes in $Pa(G_i)$, and we use $\delta(G_i)$ or $<>$ to denote a single combination. We will say that a configuration pa belongs to the combination δ, denoted by $pa \in \delta$, if the combination δ can be obtained from configuration pa by removing the order constraints.

The following theorem shows that in order to combine the different individual ratings we only need to take into account the probability distributions associated to the set of combinations with repetition:

Theorem 2 *Let G_i be a group node in a BN whose conditional probability distributions are represented using a majority gate, let $\Delta(G_i)$ be the set of possible combinations with repetition of the values in its parent set, $Pa(G_i)$, then*

$$Pr(G_i = s|ev) = \sum_{\delta(G_i)\in\Delta(G_i)} Pr(G_i = s|\delta(G_i))Pr(\delta(G_i)|ev)$$

Proof Considering the independences in the model (see Sect. 3.2) we have that

$$Pr(g_{i,s}|ev) = \sum_{pa(G_i)} Pr(g_{i,s}|pa(G_i))Pr(pa(G_i)|ev)$$

If we consider the set of configurations that can be mapped to the combination $\delta(G_i)$, i.e. $pa(G_i) \in \delta(G_i)$, then

$$Pr(g_{i,s}|ev) = \sum_{\delta(G_i)} \sum_{pa(G_i)\in\delta(G_i)} Pr(g_{i,s}|pa(G_i))Pr(pa(G_i)|ev).$$

Since for the majority gate all configurations mapping to the combination $\delta(G_i)$ have the same conditional probability distribution, $Pr(g_{i,s}|\delta(G_i))$, the right-hand side of the above equality becomes

$$\sum_{\delta(G_i)} Pr(g_{i,s}|\delta(G_i)) \sum_{pa(G_i)\in\delta(G_i)} Pr(pa(G_i)|ev)$$

and finally

$$\sum_{\delta(G_i)} Pr(g_{i,s}|\delta(G_i))Pr(\delta(G_i)|ev). \qquad \square$$

This theorem shows that if we know $Pr(\delta(G_i)|ev)$, the information could be combined with a majority gate in a time which is proportional to the size of CR_n^r, i.e. $O(n^{r-1})$. Taking into account that in many situations $r << n$, this implies important savings in terms of considering the number of possible configurations, $O(r^n)$. For instance, if $n = 20$ and $r = 2$ then $CR_n^r = 21$ whereas the number of configurations (permutations) is more than a million.

B.1 Assuming independence to approximate $Pr(\delta(G_i)|ev)$:

In order to compute $Pr(\delta(G_i)|ev)$, however, we must sum over all the possible configurations in the combination, i.e. $\sum_{pa(G_i)\in\delta(G_i)} Pr(pa(G_i)|ev)$. We will see how by assuming that collaborative ratings are independent given the evidence these computations can be considerably reduced.

Firstly, and with the idea of being general, we will introduce some notation: let $X_1, \ldots X_n$ be a set of n independent variables and let π_n represent any configuration of these variables. As these variables are independent $Pr(\pi_n) = \prod_{i=i}^{n} Pr(x_{i,j})$, where $x_{i,j}$ is the value that variable X_i takes in the configuration π_n.[16] Let δ_k be a combination with repetition of a subset of k variables and let $s \in \delta_k$ represent the fact that the value s belongs to the combination δ_k. Additionally, we say that δ_{k-1} is a s-reduction of δ_k, denoted by $\delta_k^{\downarrow s}$, if δ_{k-1} can be obtained by removing a value s from the combination δ_k. The following theorem shows how $Pr(\delta_n)$ can be computed recursively:

Theorem 3 *Let δ_n be any combination with repetition from the set of $X_1, \ldots, X_n$. If X_i is independent of $X_j, \forall i \neq j$, the probability associated with the combination δ_n can then be computed as*

$$Pr(\delta_n) = \begin{cases} Pr(x_{1,t}) & \text{if } n = 1 \text{ and } t \in \delta_1 \\ \sum_{s\in\delta_n} Pr(\delta_{n-1}^{\downarrow s})Pr(x_{n,s}) & \text{if } n > 1 \end{cases} \tag{14}$$

Proof We know that $Pr(\delta_n) = \sum_{\pi_n\in\delta_n} Pr(\pi_n)$. Assuming independence between the variables, we have that $Pr(\pi_n) = Pr(\pi_{n-1})Pr(x_{n,s})$ where π_{n-1} is the configuration of the first $n-1$ variables in π_n and $x_{n,s}$ is the value of X_n in π_n. Grouping all the configurations with the same value for variable X_n we have that

$$Pr(\delta_n) = \sum_{s\in\delta_n} Pr(x_{n,s}) \sum_{\pi_{n-1}\in\delta_{n-1}} Pr(\pi_{n-1}) = \sum_{s\in\delta_n} Pr(\delta_{n-1}^{\downarrow s})Pr(x_{n,s}).$$

[16] It should be noted that in the case that all $Pr(X_a) = Pr(X_b), \forall a \neq b$, we will have a multinomial distribution of ratings, simplifying the estimation processes. Nevertheless, this situation should imply that all the collaborative nodes have the same rating probabilities, which is not a valid assumption in this domain.

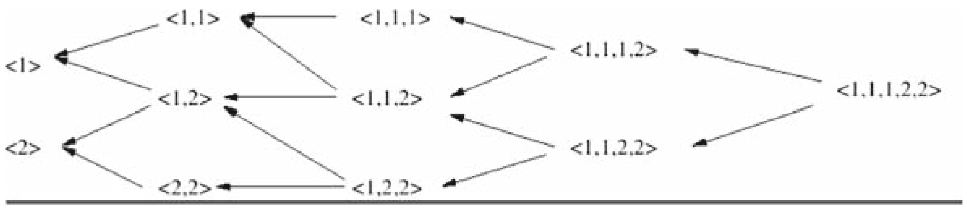


Fig. 5 Recursion graph for computing $Pr(<1, 1, 1, 2, 2>)$

Table 12 Algorithm for computing $Pr(\Delta)$

Computing $Pr(\Delta)$

```
$Pr(\delta_1) = Pr(X_1)$
for (k = 1; k < n; k++)
    for each $\delta_k \in CR_k^r$ do // each combination of size k
        for (s = 1; s <= r; s++) // values of $X_{k+1}$
            $Pr(\delta_{k \cup s}) += Pr(\delta_k) \times Pr(x_{k+1,s})$
```

A first idea would be to apply this result directly in order to compute $Pr(\delta(G_i)|ev)$. For instance, Fig. 5 shows the recursion graph for the computation of $Pr(<1, 1, 1, 2, 2>)$, where each different combination obtained after a reduction has been displayed only once. The key observation is that the number of (sub)combinations obtained after applying a reduction process is relatively small. A recursive algorithm may therefore encounter each one many times in different branches of its recursion graph. For example, Fig. 5 shows that the (sub)combination $Pr(<1, 1, 2>)$ should be computed twice and the (sub)combination $Pr(<1, 1>)$ three times. Moreover, some of these subproblems might also appear when computing different joint probabilities, such as $Pr(<1, 1, 2, 2, 2>)$. Applying Theorem 3 directly therefore involves more work than necessary.

We propose that every probability for a given subcombination be computed just once and its values saved in a table, thereby avoiding the work of recomputing this probability every time the subcombination is encountered.

The following algorithm (see Table 12) shows how to compute the joint probability distributions for all the possible combinations with replacement in the set Δ. We follow a bottom-up approach where we first compute the probabilities associated with the smallest (sub)combinations in terms of the number of variables used to form the combinations with repetition, and these probabilities will be used as the basis for calculating the largest combinations. Initially, when considering the first variable X_1, we have r different combinations with replacement, one for each possible value of the variable X_1. In a general stage, we then found that the probabilities associated with each combination δ_k of the first k variables are used in the computation of the probabilities of r different combinations with size $k+1$, one for each possible value of the variable X_{k+1}. Each of these combinations will be denoted by $\delta_{k \cup s}$ with $1 \leq s \leq r$.

An inspection of the algorithm yields a running time of $T(n) = \sum_{i=1}^{n} rCR_i^r$, i.e. $T(n) \in O(rn^r)$, which is much more efficient than applying the recursive algorithm from Theorem 3 directly. For example, in the case of bivaluated variables (as is usual in decision problems), we have a quadratic algorithm for combining the output of the different individuals. With respect to the memory needed to store the intermediate results, we find that the values in stage k are only used in stage $k-1$, and therefore the memory used is on the order of $O(CR_n^r)$.

References

Abellán, J., Masegosa, A.: Combining decision trees based on imprecise probabilities and uncertainty measures. In: Proceedings of the Ninth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty. Springer LNAI 4724, pp. 512–523. Hammamet, Tunisia (2007)

Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)

Albrecht, D., Zukerman, I.: Special issue on statistical and probabilistic methods for user modeling. User Model. User-Adapt. Interact. **17**(1–2) (2007)

Ardissono, L., Goy, A., Petrone, G., Segnan, M., Torasso, P.: INTRIGUE: personalized recommendation of tourist attractions for desktop and handset devices. Appl. Artif. Intell. **17**(8), 687–714 (2003)

Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: 14th Conference on Uncertainty in Artificial Intelligence, pp. 43–52. Madison, WI, USA (1998)

Butz, C.: Exploiting contextual independencies in web search and user profiling. In: Proceedings of World Congress on Computational Intelligence, pp. 1051–1056. Honolulu, HI, USA (2002)

Chen, Y.-L., Cheng, L.-C., Chuang, C.-N.: A group recommendation system with consideration of interactions among group members. Expert Syst. Appl. **34**, 2082–2090 (2008)

Clemen, R.T., Winkler, R.L.: Combining probability distributions from experts in risk analysis. Risk Anal. **19**, 187–203 (1999)

de Campos, L.M., Fernández-Luna, J.M., Huete, J.F.: Generalizing e-Bay.NET: an approach to recommendation based on probabilistic computing. In: 1st Workshop on Web Personalization, Recommender Systems and Intelligent User Interface, pp. 24–33. Reading, UK (2005)

de Campos, L.M., Fernández-Luna, J.M., Huete, J.F.: A Bayesian network approach to hybrid recommending systems. In: Eleventh International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 2158–2165. Paris, France (2006)

de Campos, L.M., Fernández-Luna, J.M., Huete, J.F.: A collaborative recommender system based on probabilistic inference from fuzzy observations. Fuzzy Sets Syst. **159**(12), 1554–1576 (2008)

Duda, R., Hart, P.: Pattern Classification and Scene Analysis. Wiley, New York (1973)

Genest, C., Zidek, J.: Combining probability distributions: a critique and annotated bibliography. Stat. Sci. **1**(1), 114–148 (1986)

Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., Kadie, C.: Dependency networks for inference, collaborative filtering, and data visualization. J. Mach. Learn. Res. **1**, 49–75 (2001)

Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: SIGIR 99: Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 230–237. University of California, Berkeley, USA (1999)

Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22**(1), 5–53 (2004)

Jameson, A.: More than the sum of its members: challenges for group recommender systems. In: AVI ’04: Proceedings of the Working Conference on Advanced Visual Interfaces, pp. 48–54. New York, NY, USA (2004)

Jameson, A., Smyth, B.: Recommendation to groups. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web: Methods and Strategies of Web Personalization, pp. 596–627. Springer Verlag LNCS 4321, (2007)

Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. IEEE Trans. Pattern Anal. Mach. Intell. **20**(3), 226–238 (1998)

Kudenko, D., Bauer, M., Dengler, D.: Group decision making through mediated discussions. In: Proceedings of the Ninth International Conference on User Modeling (UM 03), Springer LNAI 2702. University of Pittsburgh, Johnstown, USA (2003)

Lekakos, G., Giaglis, G.M.: A hybrid approach for improving predictive accuracy of collaborative filtering algorithms. User Model. User-Adapt. Interact. **17**(1–2), 5–40 (2007)

Masthoff, J.: Group modeling: selecting a sequence of television items to suit a group of viewers. User Model. User-Adapt. Interact. **14**, 37–85 (2004)

Masthoff, J., Gatt, A.: In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. User Model. User-Adapt. Interact. **16**, 281–319 (2006)

McCarthy, J.E., Anagnost, T.D.: MUSICFX: an arbiter of group preferences for computer supported collaborative workouts. In: CSCW '00: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, p. 348. New York, NY, USA (2000)

Miyahara, K., Pazzani, M.J.: Collaborative filtering with the simple Bayesian classifier. In: Pacific Rim International Conference on Artificial Intelligence, pp. 679–689. Melbourne, Australia (2000)

Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. In: DL '00: Proceedings of the Fifth ACM Conference on Digital libraries, pp. 195–204. New York, NY, USA (2000)

O'Connor, M., Cosley, D., Konstan, J.A., Riedl, J.: PolyLens: a recommender system for groups of user. In: ECSCW'01: Proceedings of the Seventh Conference on European Conference on Computer Supported Cooperative Work, pp. 199–218. Bonn, Germany (2001)

Pennock, D.M., Wellman, M.P.: Graphical models for groups: belief aggregation and risk sharing. Decis. Anal. **2**(3), 148–164 (2005)

Popescu, A., Ungar, L., Pennock, D., Lawrence, S.: Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: 17th Conference on Uncertainty in Artificial Intelligence, pp. 437–444. Seattle, Washington, USA (2001)

Resnick, P., Varian, H.R.: Recommender systems. Commun. ACM **40**(3), 56–58 (1997)

Schiaffino, S.N., Amandi, A.: User profiling with case-based reasoning and Bayesian networks. In: IBER-AMIA-SBIA 2000 Open Discussion Track, pp. 12–21. Atibaia, So Paulo, Brazil (2000)

Yu, Z., Zhou, X., Hao, Y., Gu, J.: TV Program recommendation for multiple viewers based on user profile merging. User Model. User-Adapt. Interact. **16**(1), 63–82 (2006)

Zukerman, I., Albrecht, D.: Predictive statistical models for user modeling. User Model. User-Adapt. Interact. **11**, 5–18 (2001)

Authors' vitae

Luis M. de Campos born in 1961, received his M.S. degree in Mathematics in 1984. He completed his PhD Thesis in 1988 and became Lecturer in Computer Science in 1991 at the University of Granada, Spain. He is currently Full Professor at the Department of Computer Science and Artificial Intelligence at the University of Granada. His current areas of research interest include Probabilistic Graphical Models, Machine Learning and Information Retrieval.

Juan M. Fernández-Luna received his Computer Science degree in 1994 at the University of Granada, Spain. In 2001 he got his PhD at the same institution, working on a thesis in which several retrieval models based on Bayesian networks for Information Retrieval where designed. This subject is still his current research area, although moving to structured documents. He has got experience organising international conferences and has been guest editor of several special issues about Information Retrieval. He is currently assistant professor at the Department of Computer Science and Artificial Intelligence at the University of Granada.

Juan F. Huete born in 1967, received his M.S. degree in Computer Science in 1990. He completed his PhD Thesis on "Learning Belief Networks: Non-Probabilistic Models" in 1995 and became Lecturer in Computer Science in 1997 at the University of Granada, Spain. His current areas of research interest are Probabilistic Graphical Models and its application to Information Retrieval, Recommender Systems and Text Classification.

Miguel Angel Rueda-Morales received his B.A. in Computer Science in 2003 from the University of Granada, Spain, and is currently a PhD candidate at its Department of Computer Science and Artificial Intelligence. His primary interests lie in the area of recommender systems using probabilistic models. His contribution is based on experiences gained both from his Ph.D. work as well as his current research on recommender systems.

5.2. Combining Content-based and Collaborative Recommendations: a Hybrid approach based on Bayesian networks

- Revista: INTERNATIONAL JOURNAL OF APPROXIMATE REASONING
- Estado: Publicado
- Índice de Impacto (JCR 2010): 1.679
  Área de conocimiento: Computes Science, Artificial Intelligence.
  Ranking: 37 / 108. Cuartil: Q2
- Índice de Impacto (SJR 2010): 0.069
  Área de conocimiento: Artificial Intelligence.
  Ranking: 31 / 106. Cuartil: Q2

International Journal of Approximate Reasoning 51 (2010) 785–799

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks [☆]

Luis M. de Campos, Juan M. Fernández-Luna [*], Juan F. Huete, Miguel A. Rueda-Morales

Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S.I. Informática y de Telecomunicación, CITIC-UGR Universidad de Granada, C.P, 18071 Granada, Spain

A R T I C L E I N F O

Article history:
Received 15 February 2008
Received in revised form 5 April 2010
Accepted 5 April 2010
Available online 11 April 2010

Keywords:
Content-based filtering
Collaborative filtering
Hybrid recommender systems
Bayesian networks
MovieLens
IMDB

A B S T R A C T

Recommender systems enable users to access products or articles that they would otherwise not be aware of due to the wealth of information to be found on the Internet. The two traditional recommendation techniques are content-based and collaborative filtering. While both methods have their advantages, they also have certain disadvantages, some of which can be solved by combining both techniques to improve the quality of the recommendation. The resulting system is known as a hybrid recommender system.

In the context of artificial intelligence, Bayesian networks have been widely and successfully applied to problems with a high level of uncertainty. The field of recommendation represents a very interesting testing ground to put these probabilistic tools into practice.

This paper therefore presents a new Bayesian network model to deal with the problem of hybrid recommendation by combining content-based and collaborative features. It has been tailored to the problem in hand and is equipped with a flexible topology and efficient mechanisms to estimate the required probability distributions so that probabilistic inference may be performed. The effectiveness of the model is demonstrated using the MovieLens and IMDB data sets.

1. Introduction

Recommender systems (RSs) attempt to discover user preferences, and to learn about them in order to anticipate their needs. Broadly speaking, a recommender system provides specific suggestions about items (products or actions) within a given domain, which may be considered of interest to the given active user [1]. Formally, in a hybrid recommending framework, there exists a large number m of items or products $\mathcal{I} = \{I_1, I_2, \dots, I_m\}$, which are described by a set of l attributes or features, $\mathcal{F} = \{F_1, F_2, \dots, F_l\}$, and each product is specified by one or several. There is also a large set of n users, $\mathcal{U} = \{U_1, U_2, \dots, U_n\}$ and for each user, a set of ratings about the quality of certain observed items in $\mathcal{I}$. Under this formulation we distinguish two different problems:

- Given an item not rated, predicting the rating that the user would give.
- Given a user, find the best items and their ratings for being recommended, showing the results ordered by predicted rating.

☆ Work jointly supported by the Spanish Ministerio de Educación y Ciencia (TIN2008-06566-C04-01) and the Spanish research programme Consolider Ingenio 2010: MIPRCV (CSD2007-00018).
* Corresponding author.
E-mail addresses: lci@decsai.ugr.es (L.M. de Campos), jmfluna@decsai.ugr.es (J.M. Fernández-Luna), jhg@decsai.ugr.es (J.F. Huete), mrueda@decsai.ugr.es (M.A. Rueda-Morales).

0888-613X/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ijar.2010.04.001

Although both notions are closely related, this paper deals with the first type, i.e. rating prediction. The usual formulation of the problem is then to predict how an active user might rate an unseen item.

Many different approaches to the recommender system problem have been published [2–4], using methods from machine learning, approximation theory, and various heuristics. Independently of the technique used and based on how the recommendations are made, recommender systems are usually classified [3] into the following categories: *Collaborative filtering systems* that attempt to identify groups of people with similar tastes to those of the user and recommend items that they have liked and *Content-based recommender systems* which use content information in order to recommend items similar to those previously preferred by the user.

Generally, collaborative systems report a better performance than content-based approaches, but their success relies on the presence of a sufficient number of user ratings [3,5,6,4,7]. Such systems have the drawback that they suffer from the item cold-start problems which occur when recommendations must be made on the basis of few recorded ratings [8,3]. These problems arise because the similarity analysis is not accurate enough. In these situations the use of a content-based approach appears as an alternative. Nevertheless, this approach has its own limitations. For example, the keywords used to represent the content of the items might not be very representative. Also, content-based approaches suffer the limitation of making accurate recommendations to users with very few ratings.

A common approach to solve the problems of the above techniques is to combine both content-based and collaborative information into a *hybrid recommender system* [9]. Different hybridization methods [3,6,9,10] have been proposed, such as the use of weighted criterion (the scores of different recommendation components are combined numerically), the use of a switching mechanism (the system chooses among recommendation components and applies the selected one) or even the presentation of the two recommendations together, leaving the decision in the user's hands. Nevertheless, a common problem with these methods is that the parameters controlling the hybridization have to be tuned.

This is the setting for the proposal presented in this paper, i.e. the design of a hybrid system with the aim of predicting how an active user should rate a given item. Particularly, we will explore the use of Bayesian network formalism to represent the relationships among users $\mathcal{U}$, items $\mathcal{I}$ and features $\mathcal{F}$, the elements involved in the recommendation processes. By using Bayesian networks, we can combine a qualitative representation of how users and items are related (explicitly representing dependence and independence relationships in a graphical structure) as well as a quantitative representation by means of a set of probability distributions, measuring the strength of these relationships.

In our proposal we shall distinguish two different parts: The first one is used to represent the knowledge that we have about how the active user rates the items, i.e. the user profile, which includes both content-based and collaborative information. The second component represents those relationships related to the target item. We would like to say that content-based information is not only used to improve the active user knowledge, but also this information has been used to improve the knowledge at the collaborative level. This is possible because we have a hybrid model where all the components are represented under the same formalism. By means of this fact we can explore the importance of the different elements in the quality of the predictions.

In order to present the model, this paper is structured in the following way. The following section introduces recommender systems and reviews the related work. Section 3 describes the model from a topological point of view. How to use the recommender model and how inference is performed efficiently are explained in Section 4. Section 5 discusses the probability distribution estimation. In order to determine the performance of the proposed model, it is evaluated in Section 6. Finally, Section 7 presents our conclusions and outlines future lines of research.

2. Related work and preliminaries

Based on how the recommendations are made, recommender systems are classified into:

- *Content-based recommender systems* that [3] store content information about each item to be recommended. This information will be used to recommend items similar to those previously preferred by the user, based on how similar certain items are to each other or the similarity with respect to user preferences (also represented by means of a subset of content features). Focusing on probabilistic approaches, learning as a constraint satisfaction problem is considered in [11], where the user profile is learnt by considering contextual independence. By assuming independence between variables, Bayesian classifiers have also been used in [12,13] to estimate the probability of an item belonging to a certain class (relevant or irrelevant) given the item description. Also, Bayesian networks [14,15] have been used to model the item's description.
- *Collaborative filtering systems* [3] attempt to identify groups of people with similar tastes to those of the user and recommend items that they have liked. According to [16], collaborative recommender systems can be grouped into *memory-based* and *model-based* approaches.

On the one hand, memory-based algorithms use the entire rating matrix to make recommendations. In order to do so, they use an aggregation measure by considering the ratings of the other users [17] (those most similar) for the same item. Different models can be obtained by considering different similarity measures and different aggregation criteria. Also *item-based* approaches, which take into account the similarity between items (two items are similar if they have been rated similarly) [18,19], appear as good alternatives to the user-based method.

On the other hand, in model-based algorithms, predictions are made by building (offline) an explicit model of the relationships between items. This model is then used (online) to finally recommend the product to the users. This kind of model ranges from the classical Naive–Bayes [16,20,21] to the use of more sophisticated techniques such as those based on aspect models [22–25]. Aspect-based models have been proposed as an approach to recommendation, for robust handling of the item cold-start problem. These models do not attempt to directly model pairwise interactions, instead they assume a latent or hidden variable that represents the different topics.

A good survey of the application of different machine learning approaches to the problem of collaborative filtering is [4].

- *Hybrid recommender systems* combine collaborative and content information. Depending on the hybridization approach different types of systems can be found [9]. Firstly we are going to consider those approaches that require building separate recommender systems using techniques that are specialized to each kind of information used, and then combine the outputs of these systems. For instance, the resulting scores can be combined using a weighted approach [26] or voting mechanisms [27], switching between different recommenders [28,29], and filtering or reranking the results of one recommender with another [9]. A different approach consists of combining both content and collaborative features. By means of this combination a single unified technique might be used regardless of the types of information used [16,7]. In such systems, a careful selection of the features is needed.

There have been some works on using boosting algorithms for hybrid recommendations [30,31]. These works attempt to generate new synthetic ratings in order to alleviate the cold-start problem. These new ratings can be obtained using various heuristics, based on content information (for instance according to who acted in a movie) or demographic information. After injecting these new ratings into the user-item matrix along with actual user ratings, a collaborative algorithm is used.

The use of aspect models [24] has been also extended to use many types of meta-data (e.g. actors, genres, and directors for movies) [32]. A similar approach has been also used for music recommendations [33] and online document browsing [34]. Also related, the hybrid Poisson-aspect model [35] approach combines a user-item aspect model with a content-based user cluster.

2.1. Canonical weighted sum: a gateway to solve complexity problems

Our hybrid proposal can be viewed as an extension of the BN-based collaborative model in [36], which will be discussed in more detail in this section. In terms of dependence relationships, this model considers that the active user's ratings are dependent on the ratings given by similar users in the system. The topology of the BN consists of a variable A, representing the active user, having as its parents those user variables, $U_i \in \mathcal{U}$, with most similar tastes. These parents are learned from the database of votes. As a similarity measure between the active user A and any other user U ($sim(A,U)$) a combination of two different but complementary criteria has been used: on the one hand, we use rating correlation (Pearson correlation coefficient, PC) between common items to measure the similarity between the ratings patterns. The second criterion attempts to penalize those highly correlated neighbours which are based on very few co-rated items, which have proved to be bad predictors [17]. In some way, we are measuring how the sets of ratings overlap:

$$sim(A,U) = abs(PC(A,U)) \times \frac{|I(A) \cap I(U)|}{|I(A)|}, \tag{1}$$

where I is the set of items rated by the user in the data set.[1] In our approach, by using the absolute value of PC, $abs(PC)$, we consider that both positively (those with similar ratings) and negatively correlated users (those with opposite tastes) might help to predict the final rating for an active user.

Taking into account the number of users involved in the predictions, in [36] we developed a canonical model to represent the conditional probability distributions: the *canonical weighted sum (CWS)* gate. When this model is assumed, we can factorize the conditional probability tables into smaller pieces (the weights describing the mechanisms) and use an additive criterion to combine these values. This model can be seen as an example of "independence of causal influence" [37,38], where causes lead to effect through independent mechanisms. Since the system presented in this paper also has to handle a large number of variables (users, items and features) we are going to briefly review this model.

Definition 1 (*Canonical weighted sum*). Let X_i be a node in a BN, let $Pa(X_i)$ be the parent set of X_i, and let Y_k be the kth parent of X_i in the BN. By using a canonical weighted sum, the set of conditional probability distributions stored at node X_i are then represented by means of

$$Pr(x_{i,j}|pa(X_i)) = \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}), \tag{2}$$

where $y_{k,l}$ is the value that variable Y_k takes in the configuration $pa(X_i)$, and $w(y_{k,l},x_{i,j})$ are weights (effects) measuring how this lth value of variable Y_k describes the jth state of node X_i. The only restriction that we must impose is that the weights are a set of non-negative values verifying that for each configuration $pa(X_i)$

[1] It should be noted that we are not considering the particular votes, merely whether the users rated an item or not.

Table 1
Example of matrices containing product descriptions, $\mathcal{D}$, and user ratings, **S**.

$\mathcal{I}/\mathcal{F}$	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
I_1	1	1	1	0	0	0	0	0
I_2	1	1	1	0	0	0	0	0
I_3	0	0	1	1	1	0	0	0
I_4	0	0	0	1	1	1	0	0
I_5	0	0	0	1	1	1	0	0
I_6	0	0	0	0	0	1	1	1
I_7	0	0	0	0	0	1	1	1
I_8	1	0	1	0	0	0	1	1
I_9	0	1	0	0	1	0	0	1
I_{10}	0	0	0	0	1	1	0	1

$\mathcal{U}/\mathcal{I}$	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	5	5	3	1	3	3	0	0	0	0
U_2	5	5	4	1	1	4	0	0	0	0
U_3	4	4	3	2	2	4	0	0	0	0
U_4	1	0	1	0	0	2	1	3	0	5
U_5	1	0	2	0	0	1	1	3	0	3
U_6	0	2	1	0	2	2	0	2	0	0
U_7	0	0	5	5	0	2	0	0	0	4
U_8	5	4	3	3	0	2	1	2	1	0
U_9	0	0	1	1	2	2	0	0	0	2
U_{10}	0	0	5	0	4	0	0	0	0	3

$$\sum_{j=1}^{r} \sum_{Y_k \in Pa(X_i)} w(y_{k,l}, x_{i,j}) = 1.$$

For example, assume that X_i has four parents, $Pa(X_i) = \{Y_1, \ldots, Y_4\}$, and that each variable takes its values in $\{a, b, c\}$. Then, given the configuration $pa(X_i) = (y_{1,c}, y_{2,a}, y_{3,c}, y_{4,b})$, we shall compute $Pr(x_{i,b} | y_{1,c}, y_{2,a}, y_{3,c}, y_{4,b})$ as the sum $w(y_{1,c}, x_{i,b}) + w(y_{2,a}, x_{i,b}) + w(y_{3,c}, x_{i,b}) + w(y_{4,b}, x_{i,b})$.

By means of this model we can tackle efficiently those complexity problems related to probability estimation, storage and inference. Thus, it is possible to estimate large conditional probability distributions since it is only necessary to estimate the weights involved in computing the conditional probability distributions in Eq. (2). Various additional advantages will also be obtained: firstly, since these weights can be computed independently (taking only a pair of variables into account), we reduce the problem of data sparsity; secondly, the parent set of X_i can be easily modified (for instance, including a new variable Y_{k+1} as the parent of X_i does not involve recomputing every conditional probability value); and thirdly, the use of this canonical model allows us to design a very efficient inference procedure (see Section 4).

The CWS gate has its own limitations since a general probability distribution cannot be represented by means of this gate. It only can represent properly those situation where the joint distribution can be computed by adding the individual's weights. Nevertheless, we believe that its use is appropriate in the recommender framework.

3. General description of the hybrid recommender model based on Bayesian networks

In this section we will describe the BN used to represent the hybrid system. This model represents how users, $\mathcal{U}$, items, $\mathcal{I}$, and features, $\mathcal{F}$, are related. Focusing on the input data, the content description of the items is usually expressed by means of a sparse binary matrix, $\mathcal{D}$, of size $m \times l$, where $d_{i,j} = 1$ when item i is described by feature j. When the entry is null, this relation is not established. An example of such a matrix is presented on the left-hand side of Table 1. Similarly, the ratings are also represented by means of a matrix, $\mathcal{S}$, of size $n \times m$, where users are represented in the rows and items in the columns. This matrix is usually sparse as users usually rate a low number of items. The value of the matrix, $s_{a,j}$ represents how user U_a has rated item I_j. We denote by $\mathcal{R}$ the rating's domain. When a user has not rated a product, the value is 0. The right-hand side of Table 1 shows an example of such a matrix.

3.1. Elements in a recommender context

Since our BN-based system should include information about users $\mathcal{U}$, items $\mathcal{I}$ and features $\mathcal{F}$, we are going to consider the domain of these variables:

- Features nodes: There will be an attribute node F_k for each feature used to describe a product. Each node has an associated binary random variable which takes its values from the set $\{f_{k,0}, f_{k,1}\}$, which means that the kth feature is not relevant (not apply), $f_{k,0}$, or is relevant (apply), $f_{k,1}$, for the description of the content of a product. [2]

[2] In our framework the term "relevant" expresses that it can help to (it is relevant for) predicting the target item's ratings.

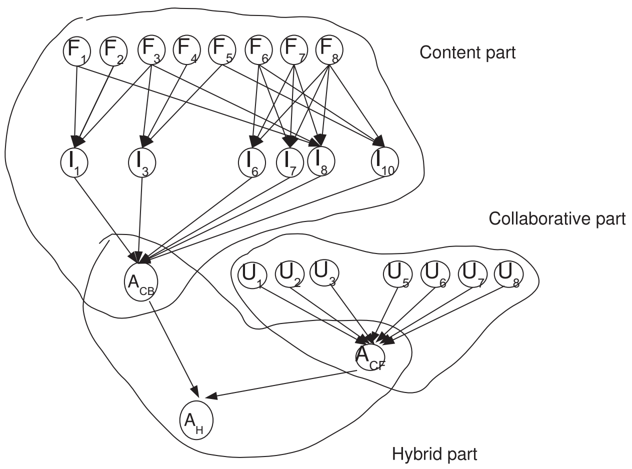

Content part
Collaborative part
Hybrid part
F_1 F_2 F_3 F_4 F_5 F_6 F_7 F_8
I_1 I_3 I_6 I_7 I_8 I_{10}
A_{CB}
U_1 U_2 U_3 U_5 U_6 U_7 U_8
A_{CF}
A_H

Fig. 1. The static subgraph of the hybrid Bayesian network.

- Item nodes: Similarly, there is a node I_j, for each item. The random variable associated with I_j will take its values from the set $\{i_{j,0}, i_{j,1}\}$ meaning that the item is not relevant (not apply) or is relevant (apply), respectively, when it comes to predicting the user's rating.
- User nodes: These U_i nodes will be used to predict the rating for the target item, particularly they should represent how probable is "the user rates with s an item". The domain of this variable is therefore the set $\mathcal{R} \cup \{0\}$. The additional value, 0, is included to model the lack of knowledge, i.e. the user has no useful information for predicting the target item's rating.

3.2. Topology of the model

Concerning the topology of our proposal, we shall distinguish two different parts: the first one is used to represent the knowledge that we have about how the active user would rate an item, i.e. the user profile. Since this component is centred on the user's perspective it might be static and could be built in an offline process. On the other hand, the second component represents those relationships related to the target item. As a consequence this part, which changes from one recommendation to another, has to be built dynamically. We are going to discuss these components in detail.

3.2.1. Static topology: representing the user profile

The user profile will be used to predict how the active user A would rate an item. In our hybrid approach, we use a BN to represent both content and collaborative components. Then, these components will be integrated in order to complete our hybrid system (see Fig. 1). The topology of the content part will be fixed (we only have to estimate the probability values from the data sets). We use a user variable A_{CB} gathering the information needed to perform content-based predictions. With respect to the collaborative component, we have to look for users similar to the active user (and therefore, a learning process becomes necessary). In this case, the collaborative information is also gathered in a user variable A_{CF}. To allow the combination of both components we use a variable A_H to encode the active user's predictions at the hybrid level. Following Burke's [9] ideas, the way in which we model the user profile can be considered as "mixed" since this variable encodes the mechanism controlling the contribution of both content and collaborative approaches.

Now, we are going to describe these components (for illustrative purposes, Fig. 1 shows the user profile associated to the user U_4, according with the data in Table 1):

CB *Content-based component:* We will consider that an item's relevance will depend on the relevance values of the features that define it. Therefore, there will be an arc from each feature node, F_i, to the nodes representing those items, I_j, which have been described with this feature. By directing the links in this way, we allow two items with a common subset of features to be dependent (except when we know the relevance values of these common features). For instance, using the data in Table 1, features F_1, F_3, F_7 and F_8 are connected to I_8.

In order to conclude this part, we must connect the nodes representing the items with the node representing the active user's predictions. The basic rule for performing these connections is simple: for each item I_j rated by the active user, add the arc $I_j \rightarrow A_{CB}$ to the graph.[3] Fig. 1 shows these arcs when the user U_4 plays the active user role.

CF *Collaborative component:* The collaborative component will comprise those people with similar tastes or preferences to the active user, represented by A_{CF}. These relations between users will depend on user ratings and so they must be

[3] The use of this model will imply that when the active user rates a new item, we must re-learn his or her conditional probability table. Nevertheless, by using the canonical weighted sum gate this process will be greatly simplified.

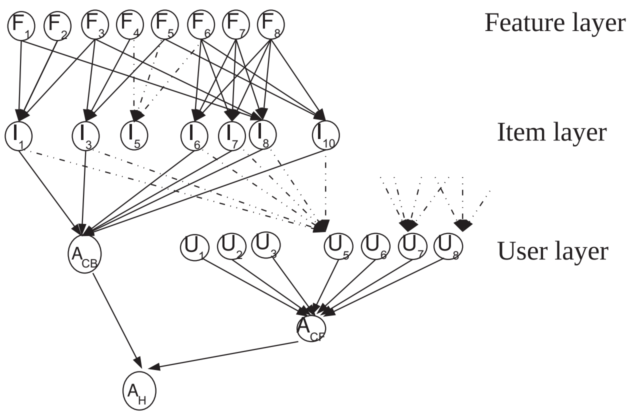

Feature layer
Item layer
User layer
F_1 F_2 F_3 F_4 F_5 F_6 F_7 F_8
I_1 I_3 I_5 I_6 I_7 I_8 I_{10}
A_{CB}
U_1 U_2 U_3 U_5 U_6 U_7 U_8
A_{CF}
A_H

Fig. 2. Extending the static component with the item-dependent relationships.

learnt from the database of votes, **S**. In this paper we use as a similarity measure the criterion previously discussed in Section 2.1 (see Eq. (1)). Regardless of the mechanism used to find these relationships, whenever a dependence (similarity) between the preferences of the active user and a given user U_i is found, an arc connecting both nodes should be included in the Bayesian network, $U_i \rightarrow A_{CF}$.

Following a common approach in traditional memory-based RS, we use a fixed neighbourhood (selecting the top-N most similar users). This approach simply selects more users so that the predictions might be based on a sufficient number of ratings. As a consequence, not all users in the selected neighbourhood have given a rating for the item that we want to predict. An advantage of this approach is that we have a neighbourhood (profile) which is independent of the target item. Therefore, an offline learning algorithm can be applied in order to update the system after new users or ratings arrive. This update can be done when the system has a low workload.

H *Hybrid component:* Given an active user A, we will have his or her own preferences about the relevance of a new item in node A_{CB} (representing the content-based component), and also the preferences borrowed from similar users in node A_{CF} (collaborative component). These two preferences must be combined in order to obtain the final prediction for the user. This can be easily represented in the BN-based model by including a new node, A_H, which has both content (A_{CB}) and collaborative (A_{CF}) information as its parents.

3.2.2. Dynamic topology: managing target item-dependent relationships

Given the active user profile, the purpose of the model is to predict the rating of an unobserved item. In order to take into account the information associated with this target item, we can enlarge both, content and collaborative components. The content-based component is enlarged by inserting a new node which represents the item itself. This node will be linked with all the features used to describe the item. For example, Fig. 2 illustrates this situation when we are trying to predict how U_4 should rate item I_5. In this figure we use dashed lines to denote the dynamic relationships.

Focusing on the collaborative component, it is possible to distinguish between those users who rated the target item I in the past ($\mathcal{U}_I^+$) and those who did not ($\mathcal{U}_I^-$). In the first case, we know exactly what the given ratings were. In the latter case, a first alternative might be not to use any information related to the users in $\mathcal{U}_I^-$ when recommending. This is common in a pure collaborative context since we do not have any more information. In a hybrid context, however, we might think about the use of content-based information in order to get some knowledge about how these users in $\mathcal{U}_I^-$ should rate an item. This kind of information can be included easily in our model by connecting each user U_i in $\mathcal{U}_I^-$ with the set of items previously rated by U_i. Continuing with our example $\mathcal{U}_I^- = \{U_5, U_7, U_8\}$ because they did not rate I_5. For clarity, in Fig. 2 we only show the links representing this content-based information for the user U_5.

With this approach, we allow not only for the active user to receive content information when recommending but also that those similar users in $\mathcal{U}_I^-$ might be favoured with this type of information in the collaborative component. Using Burke's classification [10], our hybrid approach could therefore also be placed in the "Feature Augmentation" class since we are combining both content and collaborative features when computing the probabilities in A_{CF}.

In the next section we are going to describe how the inference can be performed. Then, in Section 5 we consider how the particular weights in the CWS are assessed. We expect that this ordering will help the reader to get a better understanding about the assessment of the conditional probability distributions.

4. Inference mechanism: computing recommendations

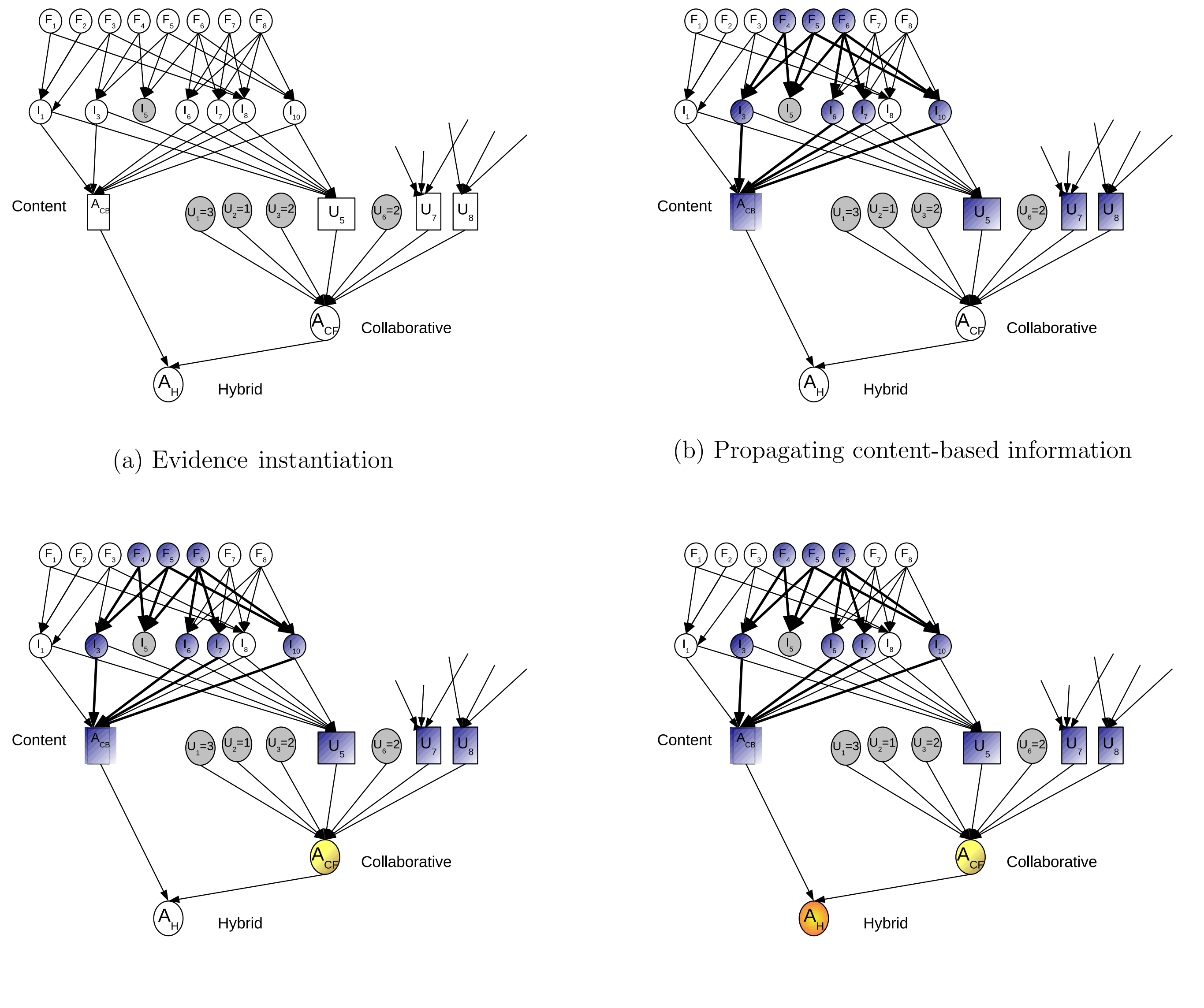
In this section we will see how the proposed topology and the use of canonical models to estimate the probability distributions enable very efficient inference mechanisms. Our goal is to compute how probable it is that the active user rates with a particular rating, given the evidence ev, i.e. $Pr(A_H = r|ev)$ for all $r \in \mathcal{R}$. In general terms, we have to instantiate the evidence in the BN and propagate towards the predictive nodes. Before studying how propagations should be performed, it is necessary to discuss how users should interact with the system.

4.1. Managing the evidence

In our framework, we will consider two different types of evidence given by content (ev_{cb}) and collaborative (ev_{cf}) information, i.e. $ev = ev_{cb} \cup ev_{cf}$.

- ev_{cb}: Focusing on the content component, we can consider two different approaches. On the one hand we can consider the item itself, in our example I_5, as evidence. So, we can instantiate this node I_5 to relevant, i.e. $ev_{cb} = \{i_{5,1}\}$. This approach will be denoted as *item instantiation*. The second alternative is to consider that the evidence comprises the features used to describe the item: F_4, F_5 and F_6 in the example. In this situation, we will instantiate all the features used to describe an item to relevant, $ev_{cb} = \{f_{4,1}, f_{5,1}, f_{6,1}\}$, and this is called *feature instantiation*.
- ev_{cf}: Focusing on the collaborative component, we know those users who rated the target item in the past, $\mathcal{U}_I^+$. Therefore, we can use the given rating as evidence. Continuing with our example, we know that I_5 was rated 3 by U_1, 1 by U_2, 2 by U_3, and 2 by U_6, i.e. the evidence is $ev_{cf} = \{u_{1,3}, u_{2,1}, u_{3,2}, u_{6,2}\}$.

Once the evidence is inserted in the model (Fig. 3a shows the instantiation of the evidence when predicting the rating for the item I_5), this information will be propagated through the network towards the predictive nodes.



(a) Evidence instantiation

(b) Propagating content-based information

(c) Propagating collaborative information

(d) Combining information

Fig. 3. Propagating the evidence towards predictive variables.

Table 2
Algorithm to compute $Pr(H_a|ev_{cb} \cup ev_{cf})$.

```
1. Content-based propagation:
    – If ($ev_{cb} == I_j$)// Item instantiation (see Fig. 3a)
        set $Pr(i_{j,1}|ev) = 1$
        Compute $Pr(F_k|ev)$ using Theorem 2//propagating towards features,
    else for each $F_k \in I_j$ set $Pr(F_k = 1|ev) = 1$.// Features Inst.
    – Propagate to items using Theorem 1.
    – Propagate to $A_{CB}$ and $U_i \in \mathcal{U}_I^-$ using Theorem 1.// (see Fig. 3b).
2. Collaborative propagation:
    – For each $U_k \in \mathcal{U}_I^+$ set $Pr(U_k = r_{k,j}|ev_{cf}) = 1$.// Collaborative evidence.
    – Propagate to $A_{CF}$ node using Theorem 1.// (see Fig. 3c)
3. Combine content-based and collaborative likelihoods at hybrid node $A_H$
4. Select the predicted rating.
```

4.2. Propagation processes

The aim of the inference process (as mentioned previously) is to estimate the rating of the active user A, given the evidence $Pr(A = s|ev)$. This propagation implies a marginalization process (summing out over uninstantiated variables) which requires an exponential time. Nevertheless, taking into account that:

(1) in a Bayesian network, a node is independent of all its ancestors given that the values taken by its parents are known,
(2) the conditional probabilities are represented using the canonical weighted sum gate (Eq. (2)),

the a posteriori probability distributions can be efficiently computed as a top-down inference mechanism. In abstract terms, the topology supporting the hybrid recommender model consists of three node layers (feature, item and user's layers) plus two more used to encode the active user predictions. Thus, starting from the first existing layer in the Bayesian network, the distributions of one layer are obtained using the a posteriori probabilities computed in the previous one. Fig. 3 illustrates the propagation process.

The following theorem (see [36]) explains how to compute the exact probability values. By means of this theorem, we express that each node collects the evidence from its predecessors and does not need to be distributed again. This is important because exact propagation can be performed in linear time with the number of parents (proof of this theorem can be found in the Appendix in [36]).

Theorem 1. *Let X_a be a node in a BN network, let m_{X_a} be the number of parents of X_a, Y_j be a node in $Pa(X_a)$, and l_{Y_j} the number of states taken by Y_j. If the conditional probability distributions can be expressed under the conditions given by* Eq. (2) *and the evidence is only on the ancestors of X_a, then the exact posterior probabilities can be computed using the following formula:*

$$Pr(x_{a,s}|ev) = \sum_{j=1}^{m_{X_a}} \sum_{k=1}^{l_{Y_j}} w(y_{j,k}, x_{a,s}) \cdot Pr(y_{j,k}|ev).$$

Focusing on the content-based component, the evidence would either comprise a set of features for an item (evidence in the first layer of the Bayesian network) or the item itself (in the second layer). In the first case, propagation is carried out directly as explained in Theorem 1. In the second case (instantiating items), the probability $Pr(F_k|i_{j,1})$ must be computed for each feature node F_k linked to the target item I_j. These posterior probabilities can then be incorporated into the propagation process. The following theorem (the proof is straight-forward) shows how to compute these values.

Theorem 2. *Let F_k be a parent node of I_j in a Bayesian network, with the former being a root node in the network. The a posteriori probability of relevance of the feature given the variable I_j playing the role of evidence is then computed as follows:*

$$Pr(f_{k,1}|i_{j,1}) = \begin{cases} Pr(f_{k,1}) & \text{if } F_k \notin Pa(I_j) \\ Pr(f_{k,1}) + \frac{w(f_{k,1},i_{j,1})Pr(f_{k,1})(1-Pr(f_{k,1}))}{Pr(i_{j,1})} & \text{if } F_k \in Pa(I_j). \end{cases} \tag{3}$$

where $Pr(i_{j,1}) = \sum_{F_k \in Pa(I_j)} w(f_{k,1}, i_{j,1}) Pr(f_{k,1})$.

The algorithm in Table 2 explains how the propagation process can be performed. In this algorithm, we consider that if U_k is a user who previously rated the target item I_j, then $r_{k,j}$ is the given rating.

5. Estimation of probability distributions

In order to complete the model's specification, the numerical values for the conditional probabilities must be estimated from the data sets. One important point to be considered is related to the size of the distributions that must be stored in a Bayesian network, exponential with the number of parents. Therefore, the assessment, storage and use of these large probability distributions can be quite complex.

As we said, we will use the canonical weighted sum model (see Section 2.1) to model item and user variables. When this model is assumed, we factorize the conditional probability tables into a set of weights and use an additive criterion to combine these values. We will now present various methods for estimating these weights.

(1) $\mathcal{F}$ *Feature variables:* Starting from the feature nodes (as these do not have parents) it is only necessary to compute the a priori probability distributions of relevance. In this paper, we propose two different alternatives for estimating these probabilities:
- *EP*: all features being equally probable, i.e. $Pr(f_{k,1}) = \frac{1}{l}$.
- *RF*: relative frequency, i.e. $Pr(f_{k,1}) = \frac{n_k+0.5}{m+1}$.

where l is the size of the set $\mathcal{F}$, n_k the number of times that feature F_k has been used to describe an item, i.e. the column sum of the left-hand side of Table 1, and m the number of items. The value $Pr(f_{k,0})$ is obtained as $Pr(f_{k,0}) = 1 - Pr(f_{k,1})$.

(2) $\mathcal{I}$ *Item Variables:* With respect to the item nodes, $I_j \in \mathcal{I}$, as these represent a binary variable, the only weights to be defined are those needed to compute $Pr(i_{j,1}|pa(I_j))$, since $Pr(i_{j,0}|pa(I_j)) = 1 - Pr(i_{j,1}|pa(I_j))$.

In order to assess these values we will consider the following idea: Assume that F_1 and F_2 are two features describing an item I_j, with F_1 being a common feature (in the sense that it has been used to describe many items) and F_2 a rare feature (it appears in few items). It is natural to think that when both features are relevant ($F_1 = f_{1,1}$ and $F_2 = f_{2,1}$) the contribution of F_2 on the I_j's relevance degree will be greater than the contribution of F_1. This idea has been widely used in the field of information retrieval [39,40] to consider the importance of a term in the entire document collection. Particularly, the concept of *inverted document frequency (idf)* is used to measure the term's importance. Therefore, using an *idf*-based approach we use the expression $\log((m/n_k) + 1)$ to measure the importance of a feature in the entire database. Obviously, when a feature is not relevant its weight is set to zero. Therefore, the weights will be computed as

$$w(f_{k,1}, i_{j,1}) = \frac{1}{M(I_j)} \log\left(\left(\frac{m}{n_k}\right) + 1\right) \quad \text{and} \quad w(f_{k,0}, i_{j,1}) = 0 \tag{4}$$

with $M(I_j)$ being a normalizing factor computed as

$$M(I_j) = \sum_{F_k \in Pa(I_j)} \log\left(\left(\frac{m}{n_k}\right) + 1\right). \tag{5}$$

For example, considering the item I_6 in Table 1 we have that $w(f_{6,1}, i_{6,1}) = 0.3$, $w(f_{7,1}, i_{6,1}) = 0.4$ and $w(f_{8,1}, i_{6,1}) = 0.3$. Thus, using the CWS (Definition 1) we have that $Pr(i_{6,1}|f_{6,1}, f_{7,1}, f_{8,1}) = 1$, $Pr(i_{6,1}|f_{6,0}, f_{7,1}, f_{8,1}) = 0.7$, $Pr(i_{6,1}|f_{6,0}, f_{7,1}, f_{8,0}) = 0.4$ and so on.

(3) $\mathcal{U}$ *User variables*: In this case we have to distinguish between those variables representing the content-based predictions (having items as their parents, i.e. A_{CB} and $U_i \in \mathcal{U}_I^-$) and the variable that combines collaborative information (having users as their parents, i.e. A_{CF}). Note that for those users in $\mathcal{U}_I^+$ the given rating is known, and therefore no probabilities have to be estimated.
- *Content-based predictions*: In this case, we must consider the influence of an item in the rating pattern of the user. To assess these weights we will consider two criteria: Firstly, for a given user $U_{CB} \in \{A_{CB}\} \cup \mathcal{U}_I^-$, whenever he or she rated an item I_k with the value s, then all the probability mass should be assigned to the same rating s at the user level. For example, since U_4 rates I_6 with 2 (see Table 1) we have that when I_6 is relevant, i.e. $I_6 = i_{6,1}$, then all the probability mass must be sent to the state (rating) 2 at the user node, $U_4 = u_{4,2}$. On the other hand, we will assume that all the items are equally important for predicting the active user's rating. Thus, taking into account these two ideas, and depending on whether the item I_k appears as relevant or not in the configuration $pa(U_{CB})$ ($I_k = i_{k,1}$ or $I_k = i_{k,0}$, respectively), these weights might be defined as follows:

$$\begin{aligned}
&w(i_{k,1}, u_{cb,s}) = \frac{1}{|I(U_{CB})|},\\
&w(i_{k,1}, u_{cb,t}) = 0, \quad \text{if } t \neq s, \quad 0 \leqslant t \leqslant \#r,\\
&w(i_{k,0}, u_{cb,0}) = \frac{1}{|I(U_{CB})|},\\
&w(i_{k,0}, u_{cb,t}) = 0, \quad \text{if } 1 \leqslant t \leqslant \#r.
\end{aligned} \tag{6}$$

Note that when an item is not relevant for predicting purposes, $I_k = i_{k,0}$, all the probability mass is assigned to the state 0 at the user level, representing the lack of knowledge. Thus, continuing with the example, $w(i_{6,1}, u_{4,2}) = 0.166$, $w(i_{7,1}, u_{4,1}) = 0.166$ and so on. Then, for example, given the configuration $pa(U_4) = \{i_{1,1}, i_{3,1}, i_{6,1}, i_{7,1}, i_{8,1}, i_{10,1}\}$ we have that $Pr(u_{4,1}|pa(U_4)) = 0.166 + 0.166 + 0.166 = 0.5$, $Pr(u_{4,2}|pa(U_4)) = 0.166$, $Pr(u_{4,3}|pa(U_4)) = 0.166$ and $Pr(u_{4,5}|pa(U_4)) = 0.166$. Note that when all the items are considered to be "relevant" (as before) the estimated distribution corresponds with the one that might be obtained using the maximum likelihood estimator from the U_4's ratings. This is because the assumption of "independence of the causal influences" holds. Similarly if $pa(U_4) = \{i_{1,0}, i_{3,0}, i_{6,1}, i_{7,1}, i_{8,0}, i_{10,0}\}$, we will have that $Pr(u_{4,2}|pa(U_4)) = 0.166$, $Pr(u_{4,1}|pa(U_4)) = 0.166$ and that $Pr(u_{4,0}|pa(U_4)) = 0.666$.

- A_{CF} *Collaborative-based predictions*: In this case, we must determine the weights reflecting the contribution of each similar user U_i in the prediction of the rating for the active user A. We will use similar ideas as in [36], but taking into account also the probability mass associated with the lack of knowledge at the parent nodes, i.e. the probabilities associated to the state $u_{i,0}$. To a certain extent, this mass captures the uncertainty about the rating to recommend. The particular weights are:

$$\begin{aligned} &w(u_{i,t}, a_{cf,s}) = RSim(U_i, A) \times Pr^*(A = s|U_i = t) \quad \text{if } 1 \leqslant t, s \leqslant \#r, \\ &w(u_{i,t}, a_{cf,0}) = 0 \quad \text{if } 1 \leqslant t \leqslant \#r, \\ &w(u_{i,0}, a_{cf,0}) = RSim(U_i, A), \\ &w(u_{i,0}, a_{cf,s}) = 0 \quad \text{if } 1 \leqslant s \leqslant \#r, \end{aligned} \tag{7}$$

As we can see, these weights have two components: on the one hand, we consider the relative quality (importance or similarity) of each parent in relation to the active user, defined as $RSim(U_i,A) = Sim(U_i,A)/\sum_{j\in Pa(A_{CF})} Sim(U_j,A)$; and on the other, we will consider the probability of A rating with a value s when U_i rated with t, $Pr^*(A = s|U_i = t)$. These probabilities are obtained from the data set of user ratings. In order to estimate these values we only consider those items which have been rated by both U_i and the active user A, i.e. the set $I(U_i) \cap I(A)$. Thus, $Pr^*(A = s|U_i = t) = \frac{N(u_{i,t},a_s)+1/\#r}{N(u_{i,t})+1}$ where $N(u_{i,t}, a_s)$ is the number of times from $I(U_i) \cap\ I(A)$ which have been rated t by U_i and also s by the active user A. In addition, $N(u_{i,t})$ is the number of items in $I(U_i) \cap I(A)$ rated with t by U_i.

In order to illustrate how these weights work, consider that the active user has three parents, U_x, U_y and U_z with $RSim$ values equal to 0.6, 0.3 and 0.1, respectively. Also assume that $Pr^*(A = 4|U_x = 5) = 0.9$, $Pr^*(A = 4|U_y = 1) = 0.5$ and $Pr^*(A = 4|U_z = 4) = 0.9$. Then, we have that $Pr(a_{cf,4}|u_{x,5},u_{y,1},u_{z,4}) = 0.78$ or that $Pr(a_{cf,4}|u_{x,0},u_{y,1},u_{z,4}) = 0.24$ and $Pr(a_{cf,0}|u_{x,0},u_{y,1},u_{z,4}) = 0.6$.

(4) A_H *Hybrid variable*: As A_H node has two parents, A_{CB} and A_{CF}, representing the content-based and collaborative predictions, we must assess the conditional probability values $Pr(A_H|A_{CB},A_{CF})$. These probabilities represent how to combine both types of information when predicting the active user's rating for the item I_j.

It is well known that the performance of the collaborative system improves as the information used for making recommendations increases. Inversely, the prediction for new or rare items (rated by a low number of similar users) becomes more difficult [26]. Taking this fact into account we propose that a parameter α_j, $0 \leqslant \alpha_j \leqslant 1$, be used to control the contributions of each component. Note that this parameter can vary from one recommendation to another.

$$\begin{aligned} &Pr(a_{h,s}|a_{cb,s}, a_{cf,s}) = 1 \\ &Pr(a_{h,s}|a_{cb,s}, a_{cf,q}) = \alpha_j, \quad \text{if } q \neq s \\ &Pr(a_{h,s}|a_{cb,t}, a_{cf,s}) = 1 - \alpha_j, \quad \text{if } t \neq s \\ &Pr(a_{h,s}|a_{cb,t}, a_{cf,q}) = 0 \quad \text{if } t, q \neq s \end{aligned} \tag{8}$$

Considering this equation, the higher the value of α_j, the greater the weights of the content-based nodes. For example, assigning $\alpha_j = 0$, the hybrid model tends to behave as a collaborative model, since only the information at the collaborative node A_{CF} is taken into account. With $\alpha_j = 1$, on the other hand, it will behave as a content-based model. With intermediate values, the recommendation may be performed by taking into account content-based and collaborative information, which gives expression to the hybrid model.

5.1. Determining the α parameter

In this section we will discuss the particular way in which this parameter is assessed. In the literature, a range of mechanisms to hybridize have been considered, from using a fixed value to a more sophisticated method which depends on the number of items rated by the active user [26,30,29]. Our initial hypothesis is that the parameter α_j might depend on our confidence on the results obtained in the collaborative component (which in some way depends on the number of parents of the active user A who have rated the item I_j in the past).

In order to illustrate our point of view, we will analyze the hybrid model in greater detail. Let U_i be a similar user who did not rate the target item, $U_i \in \mathcal{U}_I^-$. In this case, we use content-based information in order to predict how this user should rate I_j. This information is represented by $Pr(U_i = s|ev) = Pr(U_i = s|ev_{cb})$, with $s \in \mathcal{R} \cup \{0\}$. The state 0 represents the situation where we do not have information for recommending. For instance, if none of the items rated by U_i were relevant to the target item, we will have that $Pr(U_i = 0|ev_{cb}) = 1$. Moreover, looking at Eq. (7), this probability mass will be propagated towards the state 0 at the collaborative node, A_{CF}. In other words, the probability $Pr(A_{CF} = 0|ev)$ will reflect in some way how uncertain we are about the prediction at the collaborative level. For example, if all the similar users (parents) rated the item, we will have that $Pr(A_{CF} = 0|ev) = 0$ whereas this probability takes its maximum value when none of the similar users rated this item in the past.

Therefore, $Pr(A_{CF} = 0|ev)$ reflects our confidence degree in the collaborative recommendation, it can therefore be used to determine the extent to which we might consider each model when merging the recommendations. Taking into account that the higher the value of α_j, the greater the weights of the content-based nodes, in this paper we propose [4] to use $\alpha_j = Pr(A_{CF} = 0|ev)^2$.

6. Evaluation of the hybrid recommender model

This section establishes the evaluation settings (data set, evaluation measures and experimentation aims) and also presents the experimental results for the performance of the hybrid model.

6.1. Data sets

In terms of the test data set, we have decided to use MovieLens. It was collected by the GroupLens Research Project at the University of Minnesota during the seven-month period between 19th September 1997 and 22nd April 1998 with votes ranging from 1 to 5 stars (1 = `Awful`, 2 = `Fairly bad`, 3 = `It's OK`, 4 = `Will enjoy`, 5 = `Must see`) in a cinematographic context.

The data set contains 1682 movies rated by 943 users, and contains a total of 100,000 transactions on a scale of 1–5. In order to perform 5-fold cross validation, we have used the data sets u1.base and u1.test through u5.base and u5.test provided by MovieLens which split the collection into 80% for training and 20% for testing, respectively.

We decided to use MovieLens mainly for the following reasons: it is publicly available and has been used in many hybrid recommender systems. For these reasons, we believe that it is a good benchmark for our purposes. Moreover, it is especially interesting because it offers a content component, as the 1682 movies are classified into 18 genres (action, adventure, animation, children, comedy, crime, documentary, drama, fantasy, film-noir, horror, musical, mystery, romance, sci-fi, thriller, war and western). This allows us to perform recommendations by considering the content part of the data set.

In view of the fact that the movies included in the MovieLens database are only described using 18 genres, we have also extended the content description of the movies by means of the additional information provided by the *Internet Movie Database–IMDB–*.[5] More specifically, new information (e.g. directors, producers, plot keywords and cast) has been considered as part of the movie description in order to enrich it. With this expansion, the number of features increases from the original 18 to 17024. There are various movies from MovieLens that are not included in IMDB and these are only characterized by their genre. This extension is common practice when testing recommender systems with a significant amount of content. Some examples are [30,41,42] with the EachMovie dataset or [43,44,6,29] with MovieLens.

Therefore, in this experimentation we consider two different data sets: the first includes only the genre description using the original MovieLens dataset (denoted by ML), and the second one uses an extended version which considers more features from IMDB (denoted by ML + IMDB).

6.2. Evaluation measures

With respect to the second decision, in order to test the performance of our model, we shall measure its capability to predict a user's true ratings or preferences, i.e. system accuracy. Following [45], we propose to use the mean absolute error (MAE) which measures how close system predictions are to the user's rating for each movie by considering the average absolute deviation between a predicted rating and the user's true rating.

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} abs(p_i - r_i)}{N} \tag{9}$$

with N being the number of cases in the test set, p_i the vote predicted for a movie, and r_i the true rating.

6.3. Selecting the predicted rating

There is a key issue in a system's performance that has to be considered before presenting the experimental results. This issue consists of how to select one rating (the predicted rating) from a probability distribution over candidate ratings. There are several methods for computing this prediction. For instance, we can consider three different alternatives: the most probable rating, the expected rating and the median rating. Following [4], we will use the median prediction with all the models since it minimizes the mean absolute error.[6]

If we focus on the predictive variables in our models, A_{CB}, A_{CF} and A_H, we find that these variables take their values in $\mathcal{R} \cup \{0\}$. However, we must select a rating, *rate*, in $\mathcal{R}$. Before selecting the final rating, therefore, we must distribute the probability mass associated with the state zero, i.e. $Pr(A_\bullet = 0|ev)$. It should be remembered that this probability gathers all the mass associated with the lack of information in the recommending process. In this paper, we propose that a proportional

[4] We have also used a different approach with α being a function of the number of similar users who rated the item. This alternative gives worse results.

[5] http://www.imdb.com/.

[6] If the goal is to minimize the squared error, we should use the expected rating, and if the goal is to minimize the error rate, we should use the most probable rating.

Table 3
MAE values for *hybrid* model with MovieLens.

	ML (I_I)		ML(I_F)		ML + IMDB(I_I)		CF
NS	EQ	RF	EQ	RF	*EQ*	RF	
10	0.7293	0.7360	0.7813	0.7878	0.7254	0.7285	*0.7579*
20	0.7307	0.7393	0.7807	0.7876	**0.7198**	0.7277	0.7637
30	0.7330	0.7422	0.7808	0.7872	0.7207	0.7292	0.7681
50	0.7364	0.7467	0.7802	0.7879	0.7231	0.7328	0.7735
75	0.7401	0.7505	0.7804	0.7875	0.7252	0.7355	0.7784
CB	0.7837	0.7892	*0.7833*	0.7857	0.7908	0.7975	

criterion be used.[7] For a given active user A, we transform the posterior probability into a new one in the domain $\mathcal{R}$ using the following expression

$$Pr(A_\bullet = s|ev) = \frac{Pr(A_\bullet = s|ev)}{1 - Pr(A_\bullet = 0|ev)}, \quad \forall s \in \mathcal{R}.$$

Once we have the posterior probabilities in $\mathcal{R}$, the predictions (*rate*) for the active user A is the median rating:

$$rate = \{s|Pr(A_\bullet \leqslant s|ev) \geqslant 0.5,\ Pr(A_\bullet > s|ev) \geqslant 0.5.\} \tag{10}$$

6.4. Results for the hybrid model

In this section, we present the results obtained by the hybrid recommender model. The aim of this experimentation is to determine the validity of our approach and also to study the contribution of each component in the recommendation. In order to achieve this aim we have considered the following experimental conditions (Table 3 shows the obtained results):

Firstly, we have considered both MovieLens (ML) and the extension using IMDB (ML + IMDB) data sets. Secondly, we distinguish between item instantiation, I_I, and feature instantiation, I_F (see Section 4.1). Thirdly, we have considered the two different a priori values in the features nodes. The first, where all the features are equally probable, *EQ*, and the second which considers how frequently a feature has been used to describe a movie, *RF* (see Section 5). Finally, we also report the results obtained using different neighbourhood sizes (NS). To test the sensibility of the model with respect to the neighbourhood sizes, we have considered the most 10, 20, 30, 50 and 75 similar users.

The last row in Table 3 presents the results obtained by considering only the content-based component, i.e. the results obtained by predicting the rating using the probabilities in A_{CB}. Similarly, the last column presents the results obtained by considering *pure* collaborative information, i.e. the results obtained by first considering for all $U_i \in \mathcal{U}_I^-$ that $Pr(U_i = 0|ev_{cb}) = 1$ and then predicting the rating using the probabilities in A_{CF}.

Focusing on content-based predictions, we should highlight the worsening of performance of ML + IMDB in relation to the experiments using the original MovieLens dataset. The reason for this is clear since the efficiency of a content-based recommender system (which is in fact an information retrieval system) worsens when the number of terms increases, mainly because of the inclusion of non-significant features, as in the case of cast (i.e. IMDB considers all the actors and actresses with a role in the movie).

Focusing on collaborative predictions we can also observe that using a small number of neighbours tends to result in greater prediction accuracy. We should mention that this situation also appears when using a classical neighbour-based approach [17]. It seems that if there are many parents, some noise is introduced and the performance of the model is damaged. Nevertheless, a precision/recall tradeoff exists when using a small number of parents because the number of ratings predicted without collaborative information increases.

Now we will discuss the results obtained by the hybrid model. From the data in Table 3 we can see that the hybrid model using item instantiation performs much better than feature instantiation.[8] Moreover, with this combination the hybrid model outperforms the results obtained using collaborative or content-based recommendations isolately. Comparing these results with those obtained with our baselines, we can appreciate that significant improvements of around 6–8% have been achieved. In relation to the probabilities stored at the feature nodes, we can observe that it is better to consider that all the features are equally probable a priori. From these results it can be concluded that it is not very important that the recommendation process considers how relevant is a given feature to describe the target item.

Also, in a similar way to the pure collaborative model, we obtain better results using a small number of parents. The performance of the model worsens, in general, when the size of the parent set increases. This is true when considering item instantiation, but the performance is stable in the case of feature instantiation.

[7] We have also considered assigning the entire mass to the rating with the greatest posterior probability, but worse results were obtained.
[8] This also holds when considering ML + IMDB data set; accuracy using feature instantiation was similar to that obtained using only the genres (ML) with MAE values of around 0.785.

To conclude, under the same experimental conditions, the best results were obtained when the content description is extended with IMDB (ML + IMBD) data set. Taking into account that using IMDB worsens the performance in the content-based approach, we guess that the use of extra features might improve the recommendations in the collaborative component (via the variables in $\mathcal{U}_I^-$) and that these improvements are better in those items where the collaborative filtering does not work well, i.e. those items where the cold-start is an issue. Since usually these are rare items, the use of extra information seems to be beneficial. In order to corroborate this guess, we have run some experiments using content information at the nodes in $\mathcal{U}_I^-$. For example using I_I, EQ, $NS = 20$, and predicting the rating at the collaborative variable A_{CF} we have obtained a mean MAE of 0.7362. Comparing this result with the one obtained using the pure collaborative model (0.7637) we have that a significant improvement is obtained by using content-based information. Finally, if we compare this result with the 0.7198 in Table 3 (obtained at A_H variable), we might conclude that this last improvement (from 0.7362 to 0.7198) is mainly due to the way in which both content and collaborative information are mixed at the hybrid node A_H.

To conclude, we talk about the efficiency of the model. In this case, we say that the propagation process is quite efficient when trying to predict an active user's rating. In this case, since the model is learnt offline and the necessary a priori probability distributions can be computed and stored in a pre-processing step (offline), it is only necessary to compute those probability values affected by the evidence. Therefore, we only have to compute the posterior probabilities for those variables in the path from evidence nodes to the respective active user nodes. Moreover, since the computations of the necessary posterior probabilities at each node are linear with the number of parents, they can be computed efficiently. Thus, we conclude that the model is appropriate for being used in many real world applications when users are logged onto the system online, even when there is a large database of ratings.

6.5. Comparison with other models

In terms of the comparison of the performance of our proposal with other systems, it is worth mentioning that it is extremely difficult to find papers where the experimental setting is the same or at least reproducible in the context of hybrid systems. While there are changes in the goals of the papers, the sources of content data (differing on the use of product description, the use of social and/or demographic data about users) and also the way that training and testing, movie selection, user selection, feature selection, etc. are carried out. Therefore, and in order to compare the results of our model we have implemented several hybrid, collaborative or content-based recommender systems. Table 4 presents the MAE results obtained by each system. In this table, the row entitled with % presents the improvement percentage obtained with our model.

Firstly, we consider the hybrid model in [29] (noted by *switch* in Table 4) since its experimentation is similar to ours. In this model, a user-based collaborative filtering approach is used as the primary method and switches to content-based when collaborative predictions cannot be made (the number of neighbours for the collaborative model is fewer than five users).

We have performed a similar experiment, noted by *BN-switch* in Table 4, switching between our pure content-based and collaborative recommendations (A_{CB} and A_{CF} nodes) following the same criteria as [29], i.e. we selected the content-based recommendation when fewer than five users rated the movie and the collaborative recommendation otherwise. We have fixed the following experimental conditions: ML + IMDB, EQ, I_I and 20 parents.

We also show the results obtained using the imputation-boosted collaborative filtering [30] (named IBCF in Table 4). Firstly, we use the predictions obtained with a content-based recommender system to fill-in the sparse user-item rating matrix. Particularly, following the ideas in [30], we have implemented a bag-of-words (features) naive Bayesian classifier. Then we run a traditional Pearson correlation-based CF algorithm on this complete matrix to predict a novel rating.

Finally, we have also considered a model-based hybrid approach [33] which is an extension of the three-way aspect model [34], denoted by *ModelH* in Table 4. This model explains the generative mechanism for both content and collaborative data by introducing a latent variable, which conceptually corresponds to user types. Particularly, the probability distribution over users, items and features is decomposed into three conditional independent ones by introducing the latent variable. The interpretation is that a user type is selected according to the active user's preferences and the item features, then the prediction is obtained by conditioning to the user type. We have tuned this model and we report the best result, obtained when using 6 different states for the latent variable.

In order to quantify the improvement of the hybrid model, we should compare the results with those obtained using different content-based and collaborative models separately. In the first case, we borrow the results obtained by two classical content-based predictors [29] using ML + IMDB: the first is the *pure content-based (PCB)* predictor in which the cosine

Table 4
MAE values for other models.

	Hybrid				Collaborative			Content	
	Switch	BN-switch	IBCF	*ModelH*	Ubased	*Ibased*	*Triadic*	PCB	NB
MAE	0.7501	0.7498	0.7544	0.7405	0.7654	0.7604	0.7365	0.9253	1.2434
%	4.2	4.1	4.8	2.9	6.3	5.6	2.3	28.5	72.7

measure is used to calculate the similarity between two items; and the second one is an implementation of the content-based model using a *Naive Bayes (NB)* algorithm where the ratings are considered as the class labels.

With respect to collaborative filtering, we have used three different algorithms: firstly, we compare our baseline with the most classical *user-based* collaborative filtering model [17]. In this model, the similarity between users is computed using Pearson's correlation coefficient. In addition, the contribution of neighbours with fewer than 50 commonly-rated movies has been devaluated by applying a significance weight of $n/50$, where n is the number of ratings in common [17].

Also *item-based* approaches [18,19] appear as good alternatives to the user-based method. Item-based approaches look into the set of items the active user has rated and computes how similar they are to the target item, selecting the set of k most similar items. This item-based similarity is computed taking into account the ratings given by those users who have rated both of these items. Then the prediction is computed by taking a weighted average of the target user's ratings on these similar items. Particularly, in our experimentation we have used the adjusted cosine measure [18] and the result reported in Table 4 has been obtained when considering a neighbourhood size of 75.

Finally, we have considered the *Triadic* aspect model [23] (see Table 4) for pure collaborative predictions that considers a latent variable relating the triplet (user, item, rating). The purpose is to automatically look for the potential reasons that determine which subset of the causes are likely to be relevant for a specific item or a specific person, and in each individual case assigns a probability to the fact that a cause will be active for a given rating. If we compare this result with the ones obtained using *ModelH* we can see that extending the aspel model with content information does not lead to improvements. This result is similar to the one obtained in [32].

After evaluating these results, we could conclude that our model, reaching a best MAE of 0.7198, is competitive with the standards in the literature, producing better measures of quality.[9] We have also the advantage that these results are obtained using the same paradigm and that these facts might be exploited in order to give better explanations of the recommendations to the users.

7. Conclusions and further research

In this paper, we have proposed a hybrid recommender model based on Bayesian networks which uses probabilistic reasoning to compute the probability distribution over the expected rating. The model is founded on a layered topology representing all the elements involved in the hybrid recommendation problem. The participation degree of each recommending mechanism (content-based and collaborative) is automatically selected, adapting the model to the specific conditions of the problem. We have proved empirically that the combination of both content and collaborative information helps to improve the accuracy of the model.

Focusing on the computational aspects of the recommender process, problems such as data sparseness and the fact that the ranking should be computed in real time have been considered. In particular, guidelines for how to estimate the probability values from a data set are presented and an efficient propagation algorithm based on canonical models has also been designed.

Following Burke's classification [10], our hybrid approach could therefore be placed in the "Feature Augmentation" class since in normal operation the probabilities obtained in the propagation of the variable layers involved in a content-based recommendation are used to propagate probabilities in the layers relating to the collaborative recommendation. Moreover, as there is a mechanism to control the contribution of both elements, it may also be classified as "mixed".

It should be noted that the proposed model is versatile: it could work by exclusively applying content-based or collaborative filtering and it can also be applied to solve different recommendation tasks (such as *finding good items* or *predicting ratings*).

In terms of future research, we believe that there is room for improvement of the hybrid recommender model since there are several points that must still be researched:

- Design of new methods for estimating the weights stored in the nodes of the Bayesian network.
- Design of new feature selection methods (or the use of existing ones) to select only the best features so that the best performance may be achieved.
- Incorporation of relationships between features – this would involve introducing data mining techniques to find those features which might be related in terms of the classic co-occurrence measure of any other technique and would improve the expressiveness of the model.
- Change of the type of canonical model used in probability estimation and subsequently in the propagation-since the model uses sum gates, we could explore the possibility of applying either *And* or *Or* gates.

In the future, we therefore plan to study problems such as how our system can communicate its reasoning to users, the minimum amount of data (ratings or textual information) required to return accurate recommendations, and a more elaborate way of including item information.

[9] In order to show the variability in the conclusions we present the MAEs values per fold for the best model of our proposal (0.7304; 0.7206; 0.7069; 0.7201 and 0.7209) and the Triadic aspect model (0.75; 0.7369; 0.7306; 0.7328 and 0.7324).

References

[1] P. Resnick, H.R. Varian, Recommender systems, Communications of the ACM 40 (3) (1997) 56–58.
[2] S. Kangas, Collaborative filtering and recommendation systems, in: VTT Information Technology, 2002.
[3] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (6) (2005) 734–749.
[4] B. Marlin, Collaborative Filtering: A Machine Learning Perspective, Master's thesis, University of Toronto, 2004.
[5] H.J. Ahn, A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem, Information Sciences 178 (2008) 37–51.
[6] Q. Li, B. Kim, Clustering approach for hybrid recommender system, in: IEEE/WIC Proceedings of the International Conference on Web Intelligence, 2003, pp. 33–38.
[7] A. Gunawardana, C. Meek, A unified approach to building hybrid recommender systems, in: RecSys'09: Proceedings of the third ACM Conference on Recommender Systems, 2009, pp. 117–124.
[8] P. Melville, R. Mooney, R. Nagarajan, Content-boosted collaborative filtering, in: ACM SIGIR 2001 Workshop on Recommender Systems, 2001.
[9] R. Burke, Hybrid recommender systems: survey and experiments, User Modeling and User-Adapted Interaction 12 (4) (2002) 331–370.
[10] R.D. Burke, Hybrid Web recommender systems, Lecture Notes in Computer Science 4321 (2007) 377–408.
[11] C. Butz, Exploiting contextual independencies in web search and user profiling, in: Proceedings of the World Congress on Computational Intelligence, 2002, pp. 1051–1056.
[12] R.J. Mooney, L. Roy, Content-based book recommending using learning for text categorization, in: DL'00: Proceedings of the Fifth ACM Conference on Digital Libraries, ACM Press, 2000, pp. 195–204.
[13] M. Pazzani, D. Billsus, Learning and revising user profiles: the identification of interesting web sites, Machine Learning 27 (1997) 313–331.
[14] L.M. de Campos, J.M. Fernández-Luna, M. Gómez, J.F. Huete, A decision-based approach for recommending in hierarchical domains, Lecture Notes in Computer Science 3571 (2005) 123–135.
[15] L.M. de Campos, J.M. Fernández-Luna, J.F. Huete, Generalizing e-bay.net: an approach to recommendation based on probabilistic computing, in: First Workshop on Web Personalization, Recommender Systems and Intelligent User Interface, 2005, pp. 24–33.
[16] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: 14th Conference on Uncertainty in Artificial Intelligence, 1998, pp. 43–52.
[17] J. Herlocker, J. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in: SIGIR G 99: Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval, 1999, pp. 230–237.
[18] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the ACM World Wide Web Conference, 2001, pp. 285–295.
[19] G. Linden, B. Smith, J. York, Amazon.com recommendations: item-to-item collaborative filtering, IEEE Internet Computing 17 (6) (2003) 734–749.
[20] K. Miyahara, M.J. Pazzani, Collaborative filtering with the simple Bayesian classifier, in: Pacific Rim International Conference on Artificial Intelligence, 2000, pp. 679–689.
[21] V. Robles, P. Larrañaga, J. Peña, O. Marbán, J. Crespo, M. Pérez, Collaborative filtering using interval estimation Naive Bayes, in: Lecture Notes in Artificial Intelligence, vol. 2663, 2003, pp. 46–53.
[22] T. Hofmann, J. Puzicha, Latent class models for collaborative filtering, in: 16 Interantional Joint Conference on Artificial Intelligence, 1999, pp. 688–693.
[23] T. Hofmann, Learning what people (donG t) want, in: Machine Learning: ECML '01: Lecture Notes in Computer Science 2167, 2001, pp. 214–225.
[24] T. Hofmann, Latent semantic models for collaborative filtering, ACM Transactions on Information Systems 22 (1) (2004) 89–115.
[25] A.I. Schein, A. Popescul, L.H. Ungar, D.M. Pennock, Methods and metrics for cold-start recommendations, in: SIGIR '02: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2002, pp. 253–260.
[26] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, M. Sartin, Combining content-based and collaborative filters in an online newspaper, in: SIGIR99: Proceedings of the ACM SIGIR Workshop on Recommender Systems, 1999.
[27] M.J. Pazzani, A framework for collaborative, content-based and demographic filtering, Artificial Intelligence Review 13 (5–6) (1999) 393–408.
[28] D. Billsus, M.J. Pazzani, User modeling for adaptive news access, User Modeling and User-Adapted Interaction 10 (2–3) (2000) 147–180.
[29] G. Lekakos, P. Caravelas, A hybrid approach for movie recommendation, Multimedia Tools and Applications (36) (2008) 55–70.
[30] P. Melville, R. Mooney, R. Nagarajan, Content-boosted collaborative filtering for improved recommendations, in: Eighteenth National Conference on Artificial Intelligence (AAAI-02), 2002, pp. 187 – 192.
[31] S.-T. Park, D. Pennock, O. Madani, N. Good, D. DeCoste, Naïve filterbots for robust cold-start recommendations, in: KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006, pp. 699–705.
[32] A.I. Schein, A. Popescul, L.H. Ungar, D.M. Pennock, Generative models for cold-start recommendations, in: Proceedings of the 2001 SIGIR Workshop on Recommender Systems, 2001.
[33] K. Yoshii, M. Goto, K. Komatani, T. Ogata, H.G. Okuno, An efficient hybrid music recommender system using an incrementally trainable probabilistic generative model, IEEE Transaction on Audio, Speech and Language Processing 16 (2) (2008) 435–447.
[34] A. Popescu, L. Ungar, D. Pennock, S. Lawrence, Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments, in: 17th Conference on Uncertainty in Artificial Intelligence, 2001, pp. 437–444.
[35] C.-N. Hsu, H.-H. Chung, H.-S. Huang, Mining skewed and sparse transaction data for personalized shopping recommendation, Machine Learning 57 (1–2) (2004) 35–59.
[36] L.M. de Campos, J.M. Fernández-Luna, J.F. Huete, A collaborative recommender system based on probabilistic inference from fuzzy observations, Fuzzy Sets and Systems 159 (12) (2008) 1554–1576.
[37] C. Meek, D. Heckerman, Structure and parameter learning for causal independence and causal interaction models, in: Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence, 1997, pp. 366 – 375.
[38] N.L. Zhang, D. Poole, Exploiting causal independence in Bayesian network inference, Journal of Artificial Intelligence Research 5 (1996) 301–328.
[39] C.J.V. Rijsbergen, Information Retrieval, second ed., Butterworth, London, UK, 1979.
[40] G. Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, Inc., 1983.
[41] J. Basilico, T. Hofmann, Unifying collaborative and content-based filtering, in: ICML '04: Proceedings of the twenty-first International Conference on Machine Learning, 2004, pp. 9–16.
[42] K. Jung, D. Park, J. Lee, Hybrid collaborative filtering and content-based filtering for improved recommender system, Lecture Notes in Computer Science 3036 (2004) 295–302.
[43] N. Good, J.B. Schafer, J.A. Konstan, A. Borchers, B.M. Sarwar, J.L. Herlocker, J. Riedl, Combining collaborative filtering with personal agents for better recommendations, in: Conference of the American Association for Artificial Intelligence, 1999, pp. 439–446.
[44] B. Bezerra, F. de Carvalho, A symbolic hybrid approach to face the new user problem in recommender systems, Lecture Notes in Artificial Intelligence 3339 (2004) 1011–1016.
[45] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems 22 (1) (2004) 5–53.

5.3. Using second-hand information in Collaborative Recommender Systems

- Revista: SOFT COMPUTING
- Estado: Publicado
- Índice de Impacto (JCR 2010): 1.512
  Área de conocimiento: Computer Science, Artificial Intelligence.
  Ranking: 46 / 108. Cuartil: Q2
  Área de conocimiento: Computer Science, Interdisciplinary Applications.
  Ranking: 41 / 97. Cuartil: Q2
- Índice de Impacto (SJR 2010): 0,051
  Área de conocimiento: Artificial Intelligence.
  Ranking: 48 / 106. Cuartil: Q2
  Área de conocimiento: Computer Science Applications.
  Ranking: 69 / 187. Cuartil: Q2

Soft Comput (2010) 14:785–798
DOI 10.1007/s00500-009-0474-5

FOCUS

Using second-hand information in collaborative recommender systems

L. M. de Campos · J. M. Fernández-Luna ·
J. F. Huete · M. A. Rueda-Morales

Published online: 30 July 2009

Abstract Building recommender systems (RSs) has attracted considerable attention in the recent years. The main problem with these systems lies in those items for which we have little information and which cause incorrect predictions. One accredited solution involves using the items' content information to improve these recommendations, but this cannot be applied in situations where the content information is unavailable. In this paper we present a novel idea to deal with this problem, using only the available users' ratings. The objective is to use all possible information in the dataset to improve recommendations made with little information. For this purpose we will use what we call second-hand information: in the recommendation process, when a similar user has not rated the target item, we will guess his/her preferences using the information available. This idea is independent from the RS used and, in order to test it, we will employ two different collaborative RS. The results obtained confirm the soundness of our proposal.

Keywords Collaborative recommender systems · Bayesian networks · Neighbourhood · Second-hand information

L. M. de Campos · J. M. Fernández-Luna ·
J. F. Huete · M. A. Rueda-Morales (✉)
Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S.I. Informática y de Telecomunicación, CITIC-UGR, Universidad de Granada, 18071 Granada, Spain
e-mail: mrueda@decsai.ugr.es

L. M. de Campos
e-mail: lci@decsai.ugr.es

J. M. Fernández-Luna
e-mail: jmfluna@decsai.ugr.es

J. F. Huete
e-mail: jhg@decsai.ugr.es

1 Introduction

The Internet has become an indispensable tool for our day-to-day lives. Not many years ago, when seeking information on any subject, we had to resort to the use of encyclopedias, printed magazines, public libraries, etc. Since the rise of the Internet, access to information is much faster and easier. However, what appeared to be an advantage has become a big problem, due to the vast amount of information that exists. In an attempt to solve this problem, automated tools have been popularized to help users find the items they seek. Recommender systems (RSs) have emerged to address this issue. In general, a RS provides specific suggestions regarding items (or actions) within a given domain and which may be considered of interest to the user (Resnick and Varian 1997). There are different types of RSs (Resnick and Varian 1997; Kangas 2002) depending on the information that is used to make the recommendation:

- Content-based RSs: recommend similar items to those the user has rated positively in the past.
- Collaborative RSs: identify groups of people with similar tastes to active user and recommend those items they liked.

In the present paper we will focus on collaborative options, as they are very efficient and are easy to implement and to adapt to real systems. Collaborative filtering techniques match people with similar preferences in order to make recommendations. Their aim is to predict the utility of items for a particular user according to the items previously evaluated by other users. The big advantage of collaborative approaches in comparison with content-based ones is their outside the box recommendation ability (Burke 2002), i.e., the possibility of recommending items that do not evince content features expressed in the user profiles. For example, it may occur that listeners who enjoy hard rock also enjoy flamenco music, but a content-based recommender trained on the preferences of a hard rock fan would not be able to suggest items in the flamenco realm, since none of the features (performers, instruments, repertories) associated with items in the different categories would match. Only by looking outside the preferences of the individual, such suggestions can be made.

The success of these systems depends on the availability of a critical mass of information. The problem arises when, given an active user, the system requires information on a specific item, and people with similar tastes are not capable of offering this. Thus, in these situations, the system will offer a prediction that simply is not good enough.

One possible approach found in the literature with regard to solving this problem involves the use of content-based approaches. In this case, when no collaborative information is available, predictions are computed using the items' content description (Popescul et al. 2001; de Campos et al. 2006; Degemmis et al. 2007; Ali and van Stam 2004). Another possible solution found is to make use of the content information to fill the missing ratings in the dataset and then use it in a collaborative recommendation (Melville et al. 2001; Su et al. 2008).

These approaches obviously depend on the availability of content descriptions. In this paper, we explore a new approach to tackle this problem which can be used in situations where content description is unavailable: what we have called second-hand information. In order to illustrate the idea, let us assume the following situation: John asks his friends for their opinion about a particular movie but none, or few of them have seen it. In an attempt to provide their own opinion of the movie, his friends decide to ask their friends. This is what we call second-hand information. In this paper, we will study whether these second-hand opinions can be used to improve the recommendation given to John. This idea can be implemented easily: For similar users who did not rate the target item in the past, we will use the rating that the system might predict for them, making use of the information from their own similar users.

In order to test our proposal, we used two RSs. The first one is a classical neighbour-based and the second one is Bayesian-based. Through different experiments, we have shown the beneficial effect that the incorporation of new quality information has on the behaviour of the systems. To compare the results, we evaluated an imputation-boosted (Su et al. 2008) model.

The rest of the paper is organized in the following manner: some background information on RSs is presented in Sect. 2. In Sect. 3, we will discuss how to obtain the second-hand information. In Sect. 4, we will explain the models we will work with and will subsequently present the results achieved in some experiments in Sect. 5. In Sect. 6, we will discuss the accomplishments of our approach and finally present some conclusions and possible future work in Sect. 7.

2 Recommender systems

Recommender systems help people to find products they are interested in and which they would otherwise not be aware of, due to the vast amount of information on the Internet. Depending on the information used to make the recommendation, there are different types of RSs (Resnick and Varian 1997; Kangas 2002):

- Content-based RSs: recommend similar items to those the user has rated positively in the past. The content-based RS are rooted in Information Retrieval (Belkin and Croft 1992) and use many of its techniques. Their underlying philosophy can be summed up by "recommend me items similar to those that I liked in the past". In content-based RS, items of interest are defined by their associated features, such as actors, directors, producers, genres, etc, in a movie recommendation system.
- Collaborative RSs: recommend the items that other users with similar tastes considered to be good. Broadly speaking, for each user, we obtain a set of users (his/her neighbours) with a rating pattern that is highly correlated with him/her. Thus, given an item not rated by the user, we can predict a rating for it based on a combination of the known values given to the item by his/her neighbours. In this paper we will focus upon this type of RS. Collaborative RS are also known as collaborative filtering systems.
- Hybrids: the recommendation is made by combining collaborative and content-based approaches.

As in this paper we will focus on collaborative RSs, we will extend the explanation of this type of system: According to Breese et al. (1998), collaborative RSs can be grouped into memory-based and model-based approaches. Memory-based algorithms use the entire rating matrix to make recommendations. In order to do so, they use some kind of aggregation measure by considering the ratings of other users (those most similar) for the same item. Different models can be obtained by considering different similarity measures and different aggregation criteria (Konstan et al. 1997; Herlocker et al. 1999).

In model-based algorithms, on the other hand, predictions are made by building (offline) an explicit model of the relationships between items. This model is then used (online) to finally recommend the product to the users. In this approach, the predictions are therefore not based on any ad hoc heuristic, but rather on a model learnt from the underlying data using statistical and machine learning techniques: Clustering (OConnor and Herlocker 1999; Han et al. 2001), Naive Bayes (Miyahara and Pazzani 2000; Robles et al. 2003), and Probabilistic models (Breese et al. 1998; Marlin 2003; Hofmann and Puzicha 1999; Hofmann 2004) among others. A good survey of the application of machine learning to collaborative filtering is Marlin (2004).

The main purpose of a collaborative RS involves recommending items to users. Under this formulation, we distinguish two different problems:

- Given an item not rated, to predict the rating that the user would give.
- Given a user, to find the best items and their ratings for recommendation, showing the results ordered by predicted rating.

Although both situations are closely related, this paper deals with the first type.

With respect to the data used for recommendation in a collaborative framework, one can find, on the one hand, a large set of m items, $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$, whose domain can be diverse: books, movies, music, restaurants, web pages, etc. Moreover, there is a large set of n users, $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$. A user U_i can give his opinion about each item using a discrete rating $s, s \in \{1, 2, \ldots, \#r\}$. We can consider these ratings as a high sparse matrix R, of size $n \times m$, where users are represented in the rows and articles in the columns. This matrix is usually sparse, as users usually rate a low number of articles. The value of the matrix, $s_{a,j}$ represents how user U_a has rated item I_j. When a user has not rated a product, the value is 0. For example, Table 1 (left) shows an example of such a matrix.

Collaborative RSs present some well-known problems:

- Sparse rating problem: This problem arises because the number of available ratings previously obtained from users is usually very small compared to the number of ratings needed to achieve reliable predictions. The estimation of new ratings from a small number of examples is thus one of the critical issues in these systems.
- New user problem: When a new user enters the system, no personal ratings are available to him, and no proper recommendations can be made. As recommendations follow from a comparison between the target user and other users, based solely on the accumulation of ratings, if few ratings are available, it may become very difficult to categorize the user's interests.
- New item problem: This is the symmetric counterpart to the new user problem. When a new item is rated by an insubstantial number of users, the RS is unable to recommend it. Hence, a recent item that has not yet obtained many ratings cannot be easily recommended.

Associated with the new item problem and the sparse rating problem is the problem we attempt to solve in the present paper: given an active user, the system requires information on a specific item and people with similar tastes are unable to provide it. In this case, the system will offer an evaluation of the item that will surely be inappropriate. In situations in which no collaborative information exists, content-based approaches have been used in the literature as a possible solution. One approach found to attenuate this problem involves hybrid approaches combining the use of collaborative and content-based technicals. In this case, when no collaborative information is available, the predictions are computed using the ratings given by the active user to those items similar to the target one. The similarity between films is obtained taking into account the features of these items using, for instance, the cosine between the set of features (Ali and van Stam 2004). Another approach is found in Degemmis et al. (2007) in which these authors use similarities between users which rely on their content-based profiles rather than comparing their rating stiles. In Popescul et al. (2001) they use a hybrid Bayesian approach that allows for good recommendations where no collaborative information is available using the EM algorithm. Another bayesian approach is de Campos et al. (2006) in which thanks to the topology of their model, the problem is solved with the use of the collaborative rating to items similar to the target one. Another possible solution found is to use an imputation-boosted collaborative filter (Melville et al. 2001; Su et al. 2008). The aim of the model is to remove the sparseness of the data sets inserting the ratings predicted by a pure content system, i.e., to insert the ratings predicted by the system for every user and every movie that has not been rated. This enables the complete datasets of users' ratings to be used in order to improve predictions. In Table 1 (right), we can see an example of filling a dataset that

Table 1 Left: Data base of user's ratings. Right: imputation-boosted data

$\mathcal{U}$	I_1	I_2	I_3	I_4	I_5	…
U_1	2	2	0	1	0	·
U_2	0	0	1	2	0	·
U_3	2	2	0	0	0	·
U_4	2	1	0	0	0	·
U_5	0	0	0	0	2	·
…	·	·	·	·	·	·

$\Longrightarrow$

$\mathcal{U}$	I_1	I_2	I_3	I_4	I_5	…
U_1	2	2	1	1	1	·
U_2	1	1	1	2	2	·
U_3	2	2	2	1	2	·
U_4	2	1	1	2	1	·
U_5	2	2	1	2	2	·
…	·	·	·	·	·	·

contains four users and four movies. Once they have the entire ratings dataset, they build a collaborative RS using the new information available.

3 Using second-hand information

As we have shown, the approaches used to solve the sparsity and the new item problems depend on the availability of content descriptions. In this paper, we present a new possible approach for tackling this problem that can be used in situations in which content description is not available: What we have termed second-hand information. Let us consider the following illustrative example: Imagine that I am Mike and I want to know if I would like the Batman movie. I ask my group of friends (John, Lewis, Eli, Charles and Xavi) if they have seen it (see the modelling of the example in Fig. 1). I ask them because they are my best friends, they know my tastes about movies and we have very similar tastes, and they can therefore tell me whether I will like it or not. John and Lewis have seen the movie, so they can give me their opinion about it. But Eli, Charles and Xavi have not seen it. This is where we apply our idea, i.e., Eli, Charles and Xavi can ask their friends about the Batman movie and then, when we know if they will like the movie, they can give me their opinion about it. For Charles, all his friends have seen the film, and he will therefore obtain good information in this respect. The situation for Eli is the same, so she can obtain a good recommendation from her friends. But only one of Xavi's friends (Henry) has seen the movie. In this case, the information received by Xavi from his friends might not be very accurate. For this reason, Xavi might decide not to give me his opinion, as he considers that is not good enough, i.e., if Xavi thinks the information that he can give me is not quality information, he will not give it to me. Now, once I have all my friends' opinions regarding the Batman movie, I can form a more accurate opinion about whether I will like it or not.

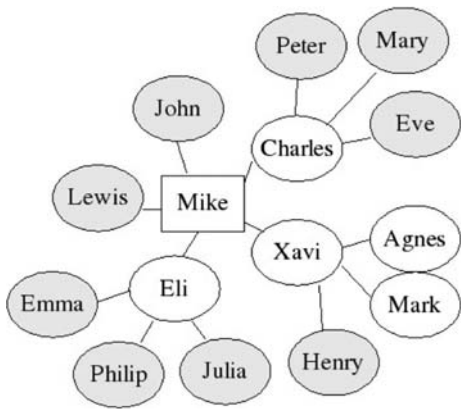


Fig. 1 Obtaining second-hand information

The idea is simple: For those neighbours who did not rate the target item in the past, we obtain new collaborative knowledge using the rating that might be predicted by the system (using the information available from their neighbours).

In order to test whether this idea works we used memory-based RSs. These are based on a two-step process:

1. *Neighbourhood selection*: For every user we obtain a neighbourhood using a metric that obtains the similarity between users. This metric usually depends on the cooccurrency of ratings in the training dataset. Particularly, the top-N users with greater values of similarity are selected as neighbours. We have to note that, for each user, the set of neighbours is fixed and does not depend on the particular item to be recommended.
2. *Computing the predictions*: Given a target item, we can compute the active user's prediction by aggregating in some way the set of ratings given by the neighbours in the past, i.e., $r_a = \texttt{Aggregate}(r_1, \ldots, r_N)$. Note that, since the set of neighbours is fixed, they do not all need to have rated the item previously.

With this recommending philosophy in mind, we can design a recursive[1] algorithm (see Table 2) that takes into account qualified second-hand information when it recommends. In the algorithm, U_i represents a neighbour of the active user A that has been obtained in the *neighbourhood selection* step. The rating prediction (line 9) for an item I by the active user A is made using both the ratings given by the neighbours if they have rated the item (line 3) and those ratings predicted by the system when a neighbour did not rate the item (only if the predicted ratings overcome a quality criteria) (lines 5–6). The function `Qualified` (line 10) returns if a prediction is good enough or not.

Although this approach might help to tackle the sparse rating problem and, in a certain way, the new item problem, its feasibility depends on the qualified second-hand predictions. Thus, even in those cases where we avail of sufficient information, the use of non-qualified information might lead to a worsening of the predictions. Therefore, when there are no available ratings for the target item, this kind of approach might not be helpful. However, since we are using only qualified information, we can expect the performance of the system not to worsen in these situations. Consequently, it might be difficult to recommend an item that has been rated by few users.

Finally, this approach implies a computational cost, as we have to compute the rating predictions for all the neighbours of the active user who have not rated the item. If we assume

[1] To clarify, we show a recursive version of the algorithm, but the implemented version is sequential.

Table 2 Recommending with second-hand information

```
Recommending( Item I, User A, iter it )
Input: I target Item, A active user, it iteration (first or second)
Output: pair<rating,bool> pred
            pred.first is the predicted rating
            pred.second represents the quality of the rating
1:  for (i = 1 to N ) do
2:    if user $U_i$ rated I
3:      $r_i$ = rating given by $U_i$
4:    else if (it == first)
5:      p = Recommending(I,$U_i$,second);
6:      if (p.second == true) $r_i$ = p.first
7:      else $r_i$ = unknown
8:    else $r_i$ = unknown              \\second iteration
9:  pred.first = Aggregate( $r_1,\ldots,r_N$) \\ using Eq.2 or Eq.5
10: pred.second = Qualified(pred.first,$r_1,\ldots,r_N$)
11: return pred
```

that the similarity weights calculations are computed offline, the final recommendations are obtained in $O(N^2)$, n being the maximum number of neighbours used for recommendation in the system. Nonetheless, and thanks to the offline computations, the approach can be used in online applications, even with a large number of users.

4 Collaborative RSs used

We have used two memory-based RSs: the first, a Pearson correlation-based model and the second, a Probabilistic model based on Bayesian networks (BNs), which we will introduce in Sect. 4.2.

4.1 Weighted average of deviations from the neighbour's mean

Based on the model proposed by Grouplens, weighted average of deviations from the neighbour's mean (Konstan et al. 1997; Herlocker et al. 1999), which hereinafter we call Average model, which is a collaborative RS that uses an algorithm based on neighbourhood.

1. *Neighbourhood selection*: To measure the similarity between users, used as the basis of weights in different collaborative systems, this model is based on the Pearson Correlation Coefficient (PCC), where U is the set of users and U_a two users from U. The PCC can be computed by means of the following formula:

$$\mathrm{PCC}(U_a, U_b) = \frac{\sum_j (r_{a,j} - \overline{r}_a) \cdot (r_{b,j} - \overline{r}_b)}{\sqrt{\sum_j (r_{a,j} - \overline{r}_a)^2 \cdot \sum_j (r_{b,j} - \overline{r}_b)^2}} \qquad (1)$$

where sums over j are applied to those items where users U_a and U_b have ratings, $I(U_a) \cap I(U_b)$ (where $I(U)$ is the set of items rated by the user U in the dataset). If there are no common items between U_a and U_b, then $\mathrm{PCC}(U_a, U_b) = 0$ by default. Furthermore, $\overline{r}_a$ is the average rating value for the user U_a, i.e.:

$$\overline{r}_a = \frac{1}{|I(U_a)|} \cdot \sum_{I_k \in I(U_a)} r_{a,k}.$$

The final value of similarity is computed by applying a correction factor that devalues those PCC values that have been obtained with fewer than 50 items in common (see Herlocker et al. 1999), i.e.:

$$\mathrm{sim}(U_a, U_b) = \mathrm{PCC}(U_a, U_b) \cdot \mathrm{CF},$$

with

$$\mathrm{CF} = \begin{cases} 1 & \text{if } k > 50 \\ \frac{k}{50} & \text{otherwise} \end{cases}$$

k being the number of items in common.

2. *Computing the predictions*: Once we obtain the neighbours for a user, to obtain the predicted rating for an item, the following formula is applied:

$$\mathrm{rate}_{a,i} = \overline{r}_a + \frac{\sum_{u=1}^{n} (r_{u,i} - \overline{r}_u) \cdot \mathrm{sim}_{a,u}}{\sum_{u=1}^{n} \mathrm{sim}_{a,u}}, \qquad (2)$$

where $\mathrm{rate}_{a,i}$ is the rate prediction for the active user A of the item i, n is the number of similar users which have rated the item, $r_{u,i}$ is the rate given by the neighbour U to item i, $\overline{r}_u$ is the average rate of the neighbour and $\mathrm{sim}_{a,u}$ is the similarity measure between the active user and the similar user u.

To finalise the presentation of this model, in this paper we have used a threshold on the number of neighbours used for prediction purposes. As Herlocker et al. (1999) indicates, the use of the best n neighbours performs relatively well without limiting the prediction coverage.

4.2 Probabilistic model: collaborative RS using Bayesian networks

In a certain way, our model is similar to the previously described one, but is based upon a Bayesian formalism in which we consider that all the users are represented as nodes in the BN (see Fig. 2). Particularly, we will include a node A to represent the active user and a subset of nodes in $\mathcal{U}$ to represent those users U_i similar to the active user. On the one hand, the states of each user node, $U_i \in \mathcal{U}$, are in $\{0, 1, 2, \ldots, \#r\}$. Note that state 0 occurs when the user has not rated the item.[2] On the other hand, since A represents the active user's predicted rating, it will take its values in the range of valid ratings, i.e., $\{1, 2, \ldots, \#r\}$. With regard to the aim of this paper, is unnecessary to fully understand the model and we will explain it with little detail (for more details, refer to de Campos et al. 2008).

[2] This is one of the differences from the reference model presented in de Campos et al. (2008), i.e., the inclusion of rating 0 in the performance of the system.

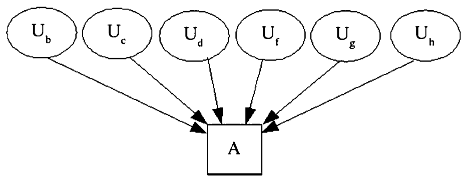


Fig. 2 Probabilistic recommender system topology

1. *Neighbourhood selection*: To measure the similarity between users, we propose a combination of two different but complementary criteria: On one hand, we use Pearson's correlation (see Eq. 1) to capture similarity between users and on the other, we use the overlap degree to penalize spurious correlations

$$\mathrm{sim}(A, U_b) = \mathrm{abs}(\mathrm{PCC}(A, U_b)) \times D(A, U_b).$$

The neighbourhood of the active user A will be obtained using the first n variables in the ranking. Note that we consider similar users with the highest absolute value of PCC. Therefore, both positively (those with similar ratings) and negatively correlated users (those with opposite tastes) will be used to predict the final rating for an active user.[3]

The second criterion attempts to penalize highly correlated neighbours based on very few co-rated items, which have been shown to be bad predictors (Herlocker et al. 1999). The way in which we compute this value varies from the Average model. In particular, we consider that the quality of U_b as the parent of variable A is directly related to the probability of a user U_b rating an item which has been also rated by A. This criterion can be defined by the following expression:

$$D(A, U_b) = \frac{|I(A) \cap I(U_b)|}{|I(U_a)|}.$$

where $I(U)$ is the set of items rated by user U in the dataset.

Learning the parameters: One important point to be considered relates to the size of the distributions to be stored in the BN. As the node A is related to a large number of users, we must assess large probability tables. To solve this problem, we propose the use of an additive canonical model (studied in detail in de Campos et al. 2008). When this model is assumed, we can factorize the conditional probability tables into smaller pieces (the weights describing the mechanisms) and use an additive criterion to combine these values.

[3] We have evaluated the system with only positive Pearson correlation and we have obtained worst results than using absolute value.

Definition Using the canonical additive model, the set of conditional probability distributions for the active user A can be calculated efficiently in the form:

$$\Pr(A = s|\mathrm{Pa}(A)) = \sum_{U_b \in \mathrm{Pa}(A)} w(U_b = t, A = s),$$

where t is the value that the user U_b takes in the configuration $Pa(A)$ and $w(U_b = t, A = s)$ are weights that measure how the tth value of the user U_b describes the rating sth of the active user A.

The particular way in which the necessary weights are defined is:

$$w(U_b = t, A = s) = \mathrm{RSim}(A, U_b) \cdot \Pr(A = s|U_b = t),$$

RSim being the relative importance of each parent in relation to the active user, defined as

$$\mathrm{RSim}(A, U_b) = \frac{\mathrm{sim}(A, U_b)}{\sum_{U_k \in \mathrm{Pa}(A)} \mathrm{sim}(A, U_k)}.$$

The term $Pr(A = s|U_b = t)$ represents how probable the A rating is with a value s when U_b rated with t. These probabilities are obtained from the dataset of user ratings. The particular way in which we estimate these probabilities will depend on whether U_b rated the target item or not:

- The user rated the target item with t, $U_b = t, t \neq 0$: In order to estimate this probability distribution, we only consider those items which have been rated by both U_b and the active user A, i.e., the set $I(A) \cap I(U_b)$. Particularly,

$$Pr(A = s|U_b = t) = \frac{N(U_b = t, A = s) + 1/\#r}{N(U_b = t) + 1},$$

$N(U_b = t, A = s)$ being the number of times from $I(A) \cap I(U_b)$ which have been rated t by U_b and also s by the active user U_a. In addition, let $N(U_b = t)$ be the number of items in $I(A) \cap I(U_b)$ rated with t by the user U_b.
- The user did not rate the target item, i.e., $U_b = 0$: In this situation we will explore two different options:

  V0E: All the ratings for the active user are equally probable, i.e.,

$$\Pr(A = s|U_b = 0) = \frac{1}{r}, 1 \leq s \leq r. \tag{3}$$

  V0A: The contribution to each possible rating will be determined by the a priori probability of the active user, A, i.e.,

$$\Pr(A = s|U_b = 0) = \Pr(A = s), 1 \leq s \leq r. \tag{4}$$

2. *Computing the predictions*: This model will be used to predict how the active user might rate a target item I. In the BN formalism, this problem is limited to computing the posterior probability distribution for A

given the evidence, i.e., $\Pr(A = s|ev)$ for all valid ratings, i.e., $\{1, 2, \ldots, \#r\}$. The problem now consists of determining what the evidence, ev, is, and how it should be included in the system. We can distinguish between users (parent of A in the BN) who rated the item in the past and those who did not rate it. In the first case, the evidence is the given rating, whereas in the second one, we have instantiate the node to the value 0 (unknown rating). For example, let us assume that $\{U_c, U_d, U_e, U_f\}$ is the set of neighbours of the active user A. Then, if U_c and U_e rated the target item I with 5 and 3, respectively, the evidence set will be $ev = \{u_{c,5}, u_{d,0}, u_{e,3}, u_{f,0}\}$.

We now have all the necessary information to compute the posterior probabilities at node A. This means that, using the advantages of this canonical model, the exact posterior probabilities for the active user (see de Campos et al. 2008) can be computed efficiently as

$$\Pr(A = s|ev) = \sum_{t=0}^{\#r} \sum_{U_b \in \mathrm{Pa}(A)} w(U_b = t, A = s) \cdot \Pr(U_b = t|ev).$$

In our case, as we know whether U_b rated the target item or not, the term $Pr(U_b = t|ev)$ takes only two values. In particular, $Pr(U_b = t|ev) = 1$ if in the evidence U_b rated with t the item, and 0 otherwise. Therefore, these probabilities can be calculated efficiently in the form:

$$\Pr(A = s|ev) = \sum_{U_b \in \mathrm{Pa}(A)} w(U_b = t, A = s). \qquad (5)$$

t being the value used to describe the rating of the user U_b to the target item.

Once the a posteriori probabilities have been computed, i.e., when we know $\Pr(A = s|ev)\ \forall s \in \{1, \ldots, \#r\}$, a key issue in the system's performance involves determining how to select the recommended ratings. In the present paper, we will consider two different alternatives for computing a prediction based on the distribution over the ratings (Marlin 2004):

MP: Select the most probable a posteriori rating:

$$\text{rate} = \arg_s \max\{\Pr(A = s|ev)\}.$$

MED: Select the median rating using the a posteriori probability in the form:

$$\text{rate} = \arg\min_s \sum_{i=1}^{s} \Pr(A = i|ev) \geq 0.5.$$

By way of an example, in the case of five possible ratings (ranging from 1 to 5), we obtain the posterior probability distribution $Pr(A = s|ev) = \{0.10, 0.15, 0.30, 0.35, 0.10\}$, i.e., $Pr(A = 1|ev) = 0.10$, $\Pr(A = 2|ev) = 0.15, \ldots, \Pr(A = 5|ev) = 0.10$. The ratings obtained using the different methods are: for MP the rating is 4 as it is the most probable and, for MED is 3 $(0.10 + 0.15 + 0.30 \geq 0.5)$.

4.3 Including second-hand information in the models

To include second-hand information in the models studied, we must change the way in which those models compute the predictions. There is a need to distinguish between the set of neighbours that have rated the movie (R) in the datasets and those for whom we have obtained a rating using second-hand information (NR):

- For the Average model presented in Sect. 4.1, we change Eq. 2 as follows:

$$\text{rate}_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{R} (r_{u,i} - \bar{r}_u) \cdot \text{sim}_{a,u} + \sum_{u=1}^{\mathrm{NR}} (\hat{r}_{u,i} - \bar{r}_u) \cdot \text{sim}_{a,u}}{\sum_{u=1}^{R} \text{sim}_{a,u} + \sum_{u=1}^{\mathrm{NR}} \text{sim}_{a,u}}$$

where $\hat{r}_{u,i}$ is the rate given by the neighbour U to item I using second-hand information.

- For the probabilistic model presented in Sect. 4.2, we change Eq. 5 as follows:

$$\Pr(A = s|ev) = \sum_{U_b \in \mathrm{Pa}_R(A)} w(U_b = t, A = s) + \sum_{U_b \in \mathrm{Pa}_{\mathrm{NR}}(A)} w(U_b = \hat{t}, A = s).$$

$\hat{t}$ being the value used to describe the rating of the user U_b to the target item using second-hand information.

5 Experimentation

The purpose of this experimentation is to study whether the use of second-hand information might be useful with regard to improve the performance of collaborative RSs. In this section, we will describe the evaluation criteria, the datasets used in the analysis and the particular experimental conditions. We then present and discuss the results regarding predictive accuracy, as well as several computational considerations.

5.1 Evaluation criteria

Our goal is to predict how a given user should rate an item. In this scenario, an individual item will be presented to the users, along with a rating indicating its potential interest. The performance of the model will therefore be evaluated by measuring prediction accuracy. In our paper, the following error measures will be considered, where N is the number of users, p_i the predicted rating and r_i the true rating.

- First, we measure the capacity of the system to predict the correct rating, i.e., the percentage of success of the systems (%S):

$$\%S = \frac{100\sum_{i=1}^{N}[p_i = r_i]}{N}.$$

- Furthermore, we consider the average absolute deviation between a predicted rating and the user's true rating, i.e., the mean absolute error (MAE) defined as

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N}\mathrm{abs}(p_i - r_i)}{N}.$$

- Also, we consider the coverage measure, i.e., the percentage of recommendations made by, at least, one neighbour.

$$\mathrm{coverage} = 100 - \frac{N_0}{N}.$$

Since one of the RS evaluated in this study, the Average model, obtains the ratings on a continuous scale, the percentage of success for this system is inappropriate and will therefore not be shown. Rather, the MAE criteria is obtained for the two models in all the experiments.

5.2 Datasets

Three different datasets were used to evaluate our algorithms. These datasets are available to the public for research purposes:

ML: Database Movielens,[4] containing 1,682 movies and 943 users who provide their ratings to films they have seen, giving rise to 100,000 ratings on a scale of 1–5.
MI-ML: Database Movielens containing 1 million ratings by 6,040 users for 3,900 movies.
JE: Jester Joke dataset.[5] This dataset contains 4.1 million continuous ratings (−10.00 to +10.00) of 100 jokes from 73,496 users. However, all the experiment results from the use of this dataset were scaled down to be equivalent with the other two datasets for easy comparison.

The purpose of this experimentation is to test our approach in conditions in which the predictions are computed, using a small number of ratings, but with available second-hand information. There are two possible situations in which this situation does not arise. First, when many users rated the items (as is the case of the Jester dataset) and, second, when it is very difficult to find similar users who rated the items, due to the sparseness of the data (Movielens datasets). Therefore, and in order to thoroughly explore the situation in which second-hand information is available, we simulated these conditions by randomly removing 50% of the ratings provided by the neighbours of the active user to the target item, i.e., we use the half of the first-hand information. It should be noted that each time we remove one rating, it ceases to be used in all predictions in which it intervened. Hereinafter, we will refer to these datasets as `reduced`.

To understand the impact of the reduction of the datasets, Table 3 shows for each model and each dataset (original and reduced) the number of times that each model predicts a rating, taking into account the past ratings available. The number of used ratings is split into different intervals (less than 6, between 6 and 12, …). As expected, the elimination of ratings increases the number of users in such a way that recommendations are obtained with a low number of ratings, i.e., for example, the numbers for the reduced dataset when neighbours are less than six are always larger than the original because we have removed some of first-hand information.

5.3 Experimental protocols and models' parameters

In the experimentation, we follow a classical protocol in the literature (Breese et al. 1998), where the available ratings for each user are split into an observed set and a held out set. The observed ratings are used for training and the held out ratings are used for testing the performance of

Table 3 Number of past ratings used in the predictions

Dataset	Num. ratings	Probabilistic		Average	
		Original	Reduced	Original	Reduced
ML	<6	4,296	18,865	5,823	18,427
	6…2	5,852	1,131	6,232	1,570
	13…18	5,364	4	4,940	3
	19…24	3,828	0	2,694	0
	>24	660	0	311	0
MI-ML	<6	83,355	187,163	83,355	178,607
	6…12	53,185	5,532	53,185	13,995
	13…18	35,100	15	35,100	108
	19…24	19,071	0	19,071	0
	>24	1,999	0	1,999	0
JE	<6	63	295,029	6,903	288,638
	6…12	2,482	52,400	23,621	58,413
	13…18	9,393	1,013	21,894	1,389
	19…24	179,473	0	156,858	2
	>24	157,031	0	139,166	0

[4] http://www.movielens.org.

[5] http://www.ieor.berkeley.edu/~goldberg/jester-data/.

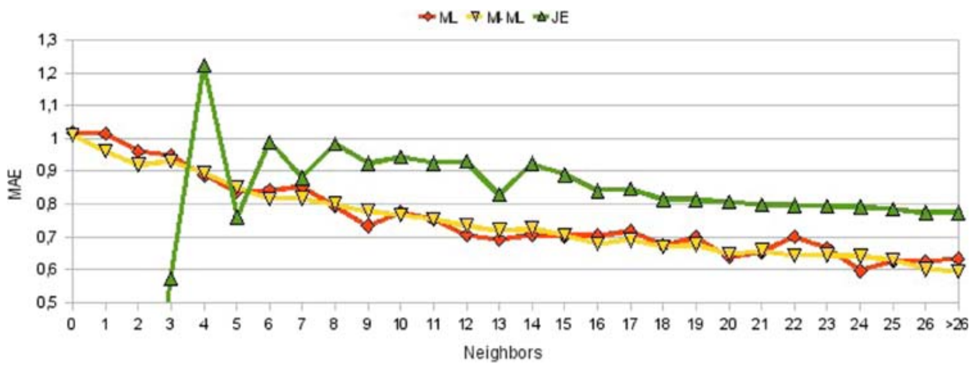


Fig. 3 MAE of the probabilistic model depending on the number of neighbours used to predict the rating

the method. Specifically, we divided the collections into 80% for training and 20% for testing. The final results were obtained by means of cross validation, showing the mean measures obtained across each multiple randomly selected set.

With respect to the parameters of the models, we first wish to discuss the neighbourhood size. On recommending with these models, the use of a small number of neighbours tends to result in greater prediction accuracy (Herlocker et al. 1999). It seems that if there are many parents, some noise is introduced and the performance of the model is damaged. Nevertheless, a precision/recall tradeoff exists when using a small number of parents, because the number of ratings predicted without collaborative information increases. Following the ideas of Herlocker et al. (1999) the neighbourhood size was fixed to 30 in both models, i.e., we consider only the best 30 neighbours for recommendation purposes.

A second parameter can also be discussed. It should be remembered that our aim in this paper is to study whether the inclusion of second-hand information improves the performance of the model. But, as has been pointed out, it seems natural that only those predicted ratings obtained with qualified information will be taken into account. But, what does "qualified information" mean? Our initial hypothesis is that the performance of the collaborative system improves with an increase in the information used for recommending. We have verified this hypothesis experimentally, as shown in Figs. 3 and 4. The abscissa represents the number of similar users who have rated the movie, nr and the ordinate shows the MAE values obtained by the models for each dataset. As can be seen, the behaviour of the systems is similar for the different datasets. It therefore seems reasonable to use a threshold in the number of neighbours who have rated the item as a quality criterion. Specifically, it can be said that a `qualityprediction` exists if it has been obtained using information from at least Q neighbours.

It should be noted that there is a tradeoff between this criterion and the amount of second-hand information used for recommending. Figures 5 and 6 show the percentage of qualified recommendations (coverage) that will be obtained using different number of similar users as the threshold Q. For example for $Q = 5$, i.e., those recommendations made with the use of at least five neighbours, the coverage is about 80% for ML, 90% for MI-ML and 100% for Jester. Looking at these figures, we can also observe a difference between the MovieLens and Jester datasets. It can be seen that the predictions obtained using the Jester dataset are well informed (this is because almost all users rated most of the jokes) where the sparseness of the MovieLens datasets (it is 97.5% sparse) implies that the predictions are obtained with less information. Availing of these data, in our paper we consider as quality ratings those obtained by means of information from at least 40% of neighbours (corresponding with $Q = 12$), because with this threshold we obtain good results in terms of MAE providing a reasonable coverage for both models. Note that the quality value comes into play only in evaluations used to add second-hand information, i.e., if a neighbour of the active user has not rated the movie, the recommended rating for this neighbour is only used if it is obtained by at least Q neighbours (second-hand neighbours).

5.4 Experimental results

5.4.1 Baselines

In this experimentation, our baseline results are those obtained without the use of second-hand information in the two models. Tables 4 and 5 show the results obtained using the Probabilistic-based model and the Average model.[6] We show the results considering the two criteria used to obtain the predicted rating with the Probabilistic model, particularly the most probable (MP) and the median rating (MED). As expected, MP maximizes the $\%S$ and MED minimizes the MAE values. Furthermore, for our probabilistic model, we show the results considering the two different alternatives for distributing the probability mass associated with

[6] Note that in this mode we do not show the success ratio as error measure because the predicted value is not an ordinal value.

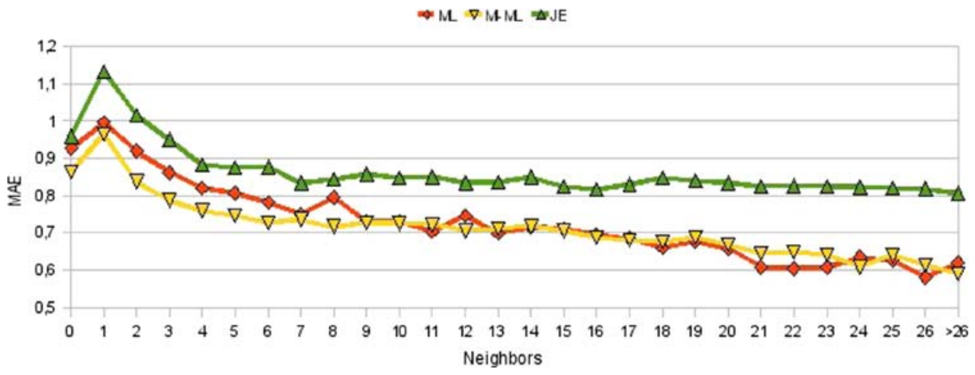


Fig. 4 MAE of the Average model depending on the number of neighbours used to predict the rating

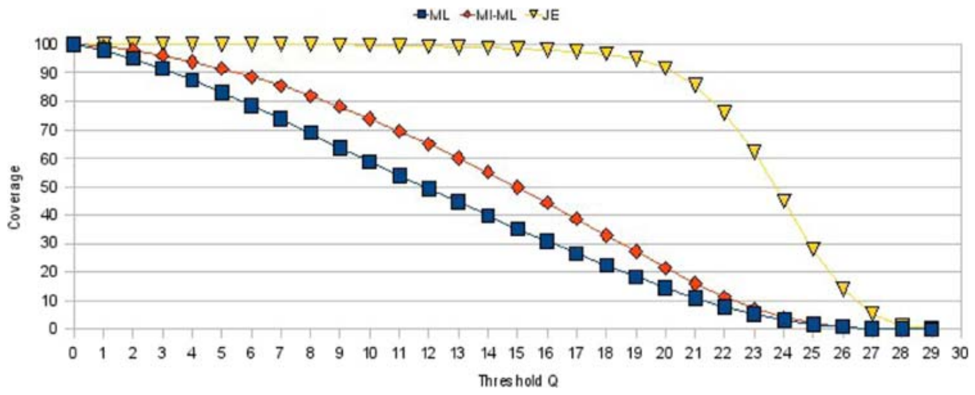


Fig. 5 Accumulated coverage of the Probabilistic model

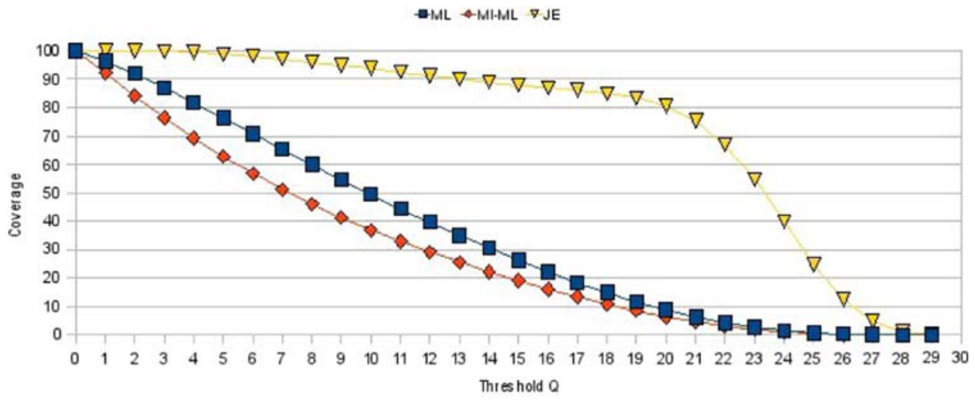


Fig. 6 Acumulated coverage of the Average model

situations in which the neighbours did not rate the target item (state 0): To use the prior probability of the active user (see Eq. 4), denoted as V0A, and to consider all the ratings being equiprobable (see Eq. 3), denoted as V0E. The best results for each dataset are highlighted in bold. Furthermore, Table 6 shows the coverage of the models for each dataset (original and reduced).

These data provide certain conclusions: First, we can see that the results obtained with the Average model are worse than those obtained by means of the Probabilistic model. Moreover, as might be expected, performance of both models presented poorer performance when predicting with less information (reduced datasets). Focussing on the Probabilistic model, significant differences were observed, depending on the method used to distribute the probability mass associated with the lack of information. Thus, with V0A, the performance of the reduced dataset declines by around 7%, whereas with V0E, the performance worsens significantly. Furthermore, the combination of MED + V0A provides the best results in terms of MAE for both datasets. Moreover, the best results in terms of percentage of success, *%S*, were obtained when predicting the most probable rating, but the criterion used to distribute the probability mass associated with the lack of information had a great impact on this metric. With respect of coverage values, we can see how the elimination of ratings cause a

Table 4 Probabilistic models

Dataset	Rate sel.	Original				Reduced			
		V0A		V0E		V0A		V0E	
		%*S*	MAE	%*S*	MAE	%*S*	MAE	%*S*	MAE
ML	MP	41.62	0.803	**42.38**	0.792	**39.51**	0.839	35.06	1.092
	MED	40.55	**0.755**	36.21	0.799	38.64	**0.791**	26.75	1.010
MI-ML	MP	42.86	0.773	**44.40**	0.737	**39.91**	0.831	28.83	1.425
	MED	41.37	**0.728**	38.37	0.749	38.69	**0.781**	26.19	1.017
JE	MP	45.56	0.803	**46.12**	0.850	**41.93**	0.977	37.77	1.158
	MED	41.25	**0.768**	39.57	0.795	37.66	**0.885**	24.96	1.078

Table 5 Average model

Dataset	Orig. MAE	Reduced MAE
ML	0.762	0.859
MI-ML	0.761	0.838
JE	0.828	0.962

Table 6 Coverage of the models

Dataset	Original		Reduced	
	Prob.	Average	Prob.	Average
ML	97.88	96.41	83.83	87.20
MI-ML	99.20	92.26	64.50	79.41
JE	100	99.98	85.34	87.59

decrease of coverage for each dataset been more clear for the MI-ML dataset. Moreover, we can see that the values obtained for the probabilistic model, in the original datasets, are higher than those obtained for the Average model. This result lead us to say that the neighbourhood obtained for every test user by the probabilistic model are more accurate than those obtained using the Average model.

5.4.2 *Using second-hand information*

Tables 7 and 8 show the results obtained after inserting qualified second-hand information into both models. In order to facilitate the comparison, we included a + or − symbol to indicate that the results are better or worse than those obtained in the baselines. If we focus on the results obtained with the use of the original dataset, we found that the results are quite similar to the ones obtained without the use of second-hand information. Thus, considering only the best results for the experiments (those in bold face), it can be seen that using Movielens, we obtained slight improvements in terms of MAE, whereas using Jester, worse results were provided. Moreover, in terms of percentage of success, we obtained slightly worse global results. On the other hand, the results reported from the reduced experiment appear to be quite conclusive. Using second-hand information, we might achieve a performance similar to the one obtained with all the available ratings for the active user (original datasets) which may be considered as the optimum. In a certain sense, it seems that we are capable of recovering the predictive capacity of the model. This conclusion is valid for the two collaborative filtering approaches. In Table 9, we can see a demonstration of both situations. If we use the original datasets, the coverage is almost equal to those evaluations without using second-hand information. In contrast, using the reduced datasets, we can see how the increase of coverage is clear respect those evaluations made without using second-hand information.

An in-depth study of both situations will help us to explain this difference in performance. In particular, there are two main situations in which second-hand information might be of no use: First, when the active user avails of sufficient information for recommending, i.e., most of his/her neighbours rated the item. In this case, the second-hand information does not contribute significantly to the computations of the predicted rating. As we discussed, this is the case of the Jester dataset. On the other hand, we may not include sufficient information after consulting our neighbours. This is the case of the original MovieLens datasets.

Table 10 presents the mean number of second-hand ratings added when the number of neighbours who rated the item (NR) ranges from 0 to 12, i.e., the table shows the mean number of second-hand ratings obtained for evaluations made with less than 12 neighbours. The second and third rows show these values when all the available data are used for prediction purposes. The fourth and fifth rows show these values when the artificially reduced dataset is used. We can see how, with the use of all the available ratings, it was difficult to include second-hand ones. On observing the MovieLens datasets, we found that this

Table 7 Inserting second-hand information in the Probabilistic model

Dataset	Rate sel.	Original				Reduced			
		V0A		V0E		V0A		V0E	
		%*S*	MAE	%*S*	MAE	%*S*	MAE	%*S*	MAE
ML	MP	41.61−	0.801+	**42.02**−	0.792	**41.54**+	0.805+	40.73+	0.856+
	MED	40.98+	**0.749**+	38.14+	0.778+	40.46+	**0.758**+	36.56+	0.805+
MI-ML	MP	42.97+	0.765+	**43.73**−	0.744−	**42.62**+	0.776+	42.25+	0.812+
	MED	41.66+	**0.723**+	40.29+	0.731+	41.25+	**0.732**+	38.80+	0.754+
JE	MP	45.79+	0.853−	**45.88**−	0.850	45.55+	0.866+	**46.09**+	0.849+
	MED	41.78+	0.787−	41.74+	**0.785**+	41.27+	0.794+	39.80+	**0.792**+

Table 8 Inserting second-hand information in the Average model

Dataset	Original MAE	Reduced MAE
ML	0.761+	0.792+
MI-ML	0.748+	0.773+
JE	0.836−	0.826+

Table 9 Coverage of the models inserting second-hand information

Dataset	Original		Reduced	
	Prob.	Average	Prob.	Average
ML	97.88	96.43	93.28	91.85
MI-ML	99.21	93.22	94.53	88.37
JE	100	100	100	100

situation holds for rare items (items rated by few people). For example, on average, those items with nr = 10 were rated by 10 and 6% of the users in ML and MI-ML, respectively, and when nr = 5 these items were rated by 5 and 3% of the users in ML and MI-ML, respectively. To the contrary, when the reduced datasets are used, the inclusion of second-hand information becomes more frequent, and as a consequence, the accuracy of the model is improved considerably.

These data lead us to conclude that the inclusion of second-hand information can be beneficial in a collaborative recommending process. When extra information can be included, we might expect better accuracy in the recommendations, whereas the performance is not damaged when this is not the case. This situation holds for the two collaborative approaches considered in the experiment.

5.4.3 Using imputation-based information

We also decided to compare our results with the ones obtained with the use of an imputation-boosted approach. We used the collaborative item-based Naive Bayes classifier described in Su et al. (2008) to fill the complete missing ratings: For each movie I_a, we obtain a classifier in which the class is the movie and we consider as attributes the remaining movies rated in the datasets. For each movie within the attribute set, we learn the weights in respect to the class, using the ratings within the training dataset given by users who have rated both films. Once a classifier for each movie has been created, we obtain the rating prediction for all users who have not seen this movie, using as evidence their ratings of the remaining movies.

Once we have the filled datasets, we use them to predict the ratings in the test sets, by means of our probabilistic model.

Table 11 shows the results obtained when on filling the datasets with the Naive Bayes classifier. The results were obtained using the ML dataset. As can be seen, the results are worse than when second-hand information is used. The main reason for these results might be due to the fact that the datasets are filled a priori, i.e., as we fill all missing ratings, the similarity between users can become distorted and the neighbourhood selection process therefore does not choose the best ones.

6 Discussion

Across the experimentation it has been proven that obtaining new second-hand information might improve the predictions of the systems. An important fact that contributes to the performance, in terms of accuracy, is that the information (in terms of new ratings) must be qualified.[7] This extra information is computed using the set of past ratings available.

The experimental results indicated that there are two possible situations in which the use of second-hand

[7] In order to test this fact we have also included all the ratings but it worsen the performance of the systems.

Table 10 Number of second-hand ratings included in the models

nr	0	1	2	3	4	5	6	7	8	9	10	11	12
ML	0	0.01	0.04	0.16	0.29	0.71	1.22	1.76	2.87	4.2	5.53	6.89	8.49
MI-ML	0.03	0.09	0.32	0.61	1.06	1.78	2.65	3.83	5.06	6.64	8.1	9.55	10.71
ML(R)	10.72	11.22	11.65	12.19	13.11	13.23	13.39	13.71	13.28	14.45	13.43	14	11
MI-ML(R)	17.54	15.47	15.4	15.72	16.04	16.45	16.19	15.93	15.37	14.67	14.16	12.97	13.9

Table 11 Imputation-boosted in the Probabilistic model filling the ratings using a NB classifier

	%S	MAE
MP	40.65	0.889
MED	39.58	0.786

information does not contribute to the rating prediction: First, when new information cannot be obtained from the database of ratings, as occurs with rare items. In this situation, content information (if available) can be used in order to make the predictions, as with hybrid approaches. Furthermore, the use of second-hand information will not help if sufficient first-hand ratings already exist. Nevertheless, in both situations, the performance of the system is not worsened when second-hand information is resorted to. We therefore consider the situations in which our approach might be useful. The following are two possible ones: On one hand, we consider the case in which the target item is neither rare nor highly frequent (this could be the common one in many applications). It might therefore be interesting to look in the database of ratings in search of extra information. Second, the approach can be useful in an online store (such as Amazon or a movie-based application) where new products frequently appear. In these stores, the users start to rate after the new item is included. Therefore, at the beginning (e.g., a few weeks) it is possible that, given a user, few of his/her neighbours have rated it. In this particular situation the information provided by second-hand neighbours is welcome (our experiments with the reduced dataset reinforce this idea). Therefore, as a final conclusion, as our approach is efficient (in time) and effective (it could improve the predictions), we believe that implementation thereof in real RSs can be beneficial.

7 Conclusions

We have presented a novel idea for application in collaborative RSs in order to improve the predictions of the system by increasing the available information in the datasets using the predictions made by the system. To test it, we used two RSs: A BN-based RS and a neighbourhood-based rating prediction.

We have proved that if we introduce quality second-hand information in the systems, their recommendations can also be improved. This situation is particularly beneficial when the amount of second-hand ratings included is large. Rather, when we use the original datasets, in predictions executed with little information, introducing quality ratings produces a low or almost null increase in information for such predictions. We have demonstrated that this is due to the existence of "rare" items rated by few users.

As future work, we will consider:

- Using additional databases as NetFlix to test our proposal.
- Performing a more exhaustive study of those evaluations in which few neighbours are used for the recommendation, in order to achieve better performance with the original datasets.
- Finding new methods to predict the rating as a mixing of the MP and MED criterion.
- Testing different methods to obtain the neighbourhood.
- Finding alternative criteria to define quality, such as using the final value of probability to assess this.
- Studying new ways to calculate the weights in order to improve overall system performance.
- Incorporating content information to the models to improve the new-items situations.

Acknowledgments This study has been jointly supported by the Spanish Ministerio de Educación y Ciencia under the projects TIN2005-02516 and TIN2008-06566-C04-01, and the research program Consolider Ingenio 2010 under the project CSD2007-00018.

References

Ali K, van Stam W (2004) Tivo: making show recommendations using a distributed collaborative filtering architecture. In: KDD '04: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 394–401. doi:10.1145/1014052.1014097

Belkin NJ, Croft WB (1992) Information filtering and information retrieval: two sides of the same coin? Commun ACM 35(12):29–38. doi:10.1145/138859.138861

Breese JS, Heckerman D, Kadie C (1998) Empirical analysis of predictive algorithms for collaborative filtering. In: 14th conference on uncertainty in artificial intelligence, pp 43–52. http://citeseer.ist.psu.edu/breese98empirical.html

Burke R (2002) Hybrid recommender systems: survey and experiments. User Model User Adapted Interact 12(4):331–370

de Campos LM, Fernández-Luna JM, Huete JF (2006) A Bayesian network approach to hybrid recommending systems. In: Eleventh international conference of information processing and management of uncertainty in knowledge-based systems. Paris, France, pp 2158–2165

de Campos LM, Fernández-Luna JM, Huete JF (2008) A collaborative recommender system based on probabilistic inference from fuzzy observations. Fuzzy Sets Syst. 159(12):1554–1576. doi:10.1016/j.fss.2008.01.016

Degemmis M, Lops P, Semeraro G (2007) A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. User Model User Adapted Interact 17(3):217–255. doi:10.1007/s11257-006-9023-4

Han S, Chee S, Han J, Wang K (2001) Rectree: an efficient collaborative filtering method. In: Lecture Notes in Computer Science: Data Warehousing and Knowledge Discovery, pp 141–151

Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. In: SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval. ACM, New York, pp 230–237. doi:10.1145/312624.312682

Hofmann T (2004) Latent semantic models for collaborative ltering. ACM Trans Inf Syst 22(1):89–115

Hofmann T, Puzicha J (1998) Latent class models for collaborative filtering. In: 16th international joint conference on artifical intelligence. Stockholm, Sweden, pp 688–693

Kangas S (2002) Collaborative filtering and recommendation systems. In: VTT information technology

Konstan JA, Miller BN, Maltz D, Herlocker JL, Gordon LR, Riedl J (1997) GroupLens: applying collaborative filtering to Usenet news. Commun ACM 40(3):77–87. http://citeseer.ist.psu.edu/konstan97grouplens.html

Marlin B (2004) Collaborative filtering: a machine learning perspective. University of Toronto

Marlin B (2003) Modeling user rating profiles for collaborative filtering. In: NIPS*17. MIT Press, Cambridge

Melville P, Mooney RJ, Nagarajan R (2001) Content-boosted collaborative filtering. In: Proceedings of the 2001 SIGIR workshop on recommender systems

Miyahara K, Pazzani MJ (2000) Collaborative filtering with the simple bayesian classifier. In: Pacific Rim International conference on artificial intelligence, pp 679–689. http://citeseer.ist.psu.edu/miyahara00collaborative.html

OConnor M, Herlocker J (1999) Clustering items for collaborative ltering. In: ACM SIGIR 99 workshop on recommender systems: algorithms and evaluation

Popescul A, Ungar L, Pennock D, Lawrence S (2001) Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: 17th conference on uncertainty in artificial intelligence. Seattle, Washington, pp 437–444. http://citeseer.ist.psu.edu/popescul01probabilistic.html

Resnick P, Varian HR (1997) Recommender systems. Commun ACM 40(3):56–58. doi:10.1145/245108.245121

Robles V, Larrañaga P, Peña J, Marbán O, Crespo J, Pérez M (2003) Collaborative filtering using interval estimation naive bayes. In: Lecture Notes in Artificial Intelligence: Advances in Web Intelligence, pp 46–53

Su X, Khoshgoftaar TM, Zhu X, Greiner R (2008) Imputation-boosted collaborative filtering using machine learning classifiers. In: SAC '08: Proceedings of the 2008 ACM symposium on applied computing. ACM, New York, pp 949–950. doi:10.1145/1363686.1363903

5.4. Using past-predictions accuracy in Recommender Systems

- Revista: INFORMATION SCIENCES
- Estado: Sometido
- Índice de Impacto (JCR 2010): 2.833
  Área de conocimiento: Computer Science, Information Systems.
  Ranking: 9 / 123. Cuartil: Q1
- Índice de Impacto (SJR 2010): 0,087
  Área de conocimiento: Computer Science Applications.
  Ranking: 25 / 187. Cuartil: Q1
  Área de conocimiento: Information Systems.
  Ranking: 13 / 129. Cuartil: Q1
  Área de conocimiento: Information Systems and Management..
  Ranking: 2 / 43. Cuartil: Q1
- **Nota:** Este artículo es una extensión de la publicación:
  Measuring predictive capability in Collaborative Filtering.
  Juan Francisco Huete Guadix; Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Miguel Ángel Rueda Morales.
  ACM Conference on Recommender Systems (3) (No 3. 2009), Proceedings of the third ACM Conference on Recommender Systems, ISBN:978-1-60558-435-5, Nueva York, EE.UU., 2009.

Manuscript Number: INS-D-11-948

Title: Using past-predictions accuracy in Recommender Systems.

Article Type: Full length article

Keywords: Recommending Systems; collaborative filtering; memory-based method; past preferences

Corresponding Author: Dr J.F. Huete,

Corresponding Author's Institution: University of Granada

First Author: J.F. Huete

Order of Authors: J.F. Huete; Juan M Fernández-Luna; Luis M de Campos; Miguel A Rueda-Morales

Cover Letter

Dear sir/madam

It is a pleasure for me to submit this paper, entitled "Using past-predictions accuracy in Recommender Systems", to the Information Sciences Journal

In this paper we present a new memory-based recommending system approach using as quality criteria the accuracy over the past predictions. This criteria has been tested on three datasets, MovieLens, Yahoo Music and Yahoo Movies.

I would like to note that:

a. This manuscript is the authors' original work and has not been published nor has it been submitted simultaneously elsewhere.
b. All authors have checked the manuscript and have agreed to the submission.

The contact author is
Juan F. Huete
Dpto. Ciencias de la Computacion e IA
Universidad de Granada
18071 Granada
email:jhg@decsai.ugr.es

Using past-predictions accuracy in Recommender Systems.

Juan F. Huete[a,*], J.M. Fernández-Luna[a], Luis M. de Campos[a], Miguel A. Rueda-Morales[b]

[a] *Departamento de Ciencias de la Computación e Inteligencia Artifi cial.*
E.T.S.I. Informática y de Telecomunicación,
CITIC-UGR Universidad de Granada,
C.P, 18071, Granada, Spain.
[b] *Oficina Transferencia Resultados de Investigación.*
Universidad de Granada

Abstract

This paper presents a new approach for memory-based Collaborative Filtering algorithms. In general terms, user-based rating prediction can be seen as a process where each neighbor suggests a rating for the target item and then these suggestions are combined by weighting the contribution of each neighbor. In this paper we will present a new alternative to perform these suggestions which is independent of the users' rating scale. These predictions will be based on what we call predictive probabilities. Also, in this paper we will explore how these probabilities can be used to select the nearest neighbors to perform the recommendations, being also able to integrate different kind of dependences presented in the ratings. The neighborhood-selection criterion will depend on the predictive capability of a user at predicting past ratings. Our hypothesis is that if a user was good when predicting the past ratings for an active user, then his/her predictions will be also helpful to recommend ratings in the future for that user.

Keywords:
Recommending Systems, collaborative filtering, memory-based method, past preferences

1. Introduction

Recommender systems (RSs) attempt to discover user preference, and to learn about them in order to anticipate their needs. The main task normally associated with a RS is to offer suggestions for items that may be useful to a user. For example, they can help the user by recommending a book, a movie, a hotel, a song, etc. From the service provider's perspective RSs help to increase the user satisfaction, and as consequence, it is also expected an increase of the sales. The study of RS emerged as an independent research area in the mid-1990s [9, 23], having its roots in related areas as databases or information retrieval. In recent years, the interest in recommender systems has dramatically increased, playing an important role in internet sites as, for example, Amazon.com, YouTube, Netflix, Yahoo, Tripadvisor, Last.fm, or IMDb.

The main tasks normally associated to a RS are, on the one hand, *rating prediction*, i.e. given an unseen item, to predict how much a user would like the item. In this case, the output of the RS is a rating, in a scale from one to five stars, for instance. On the other hand, the second task is to *find good items*:, to try to find the best items (and possibly their ratings) for being recommended to a given user. In this case, the results are ranked taking into account the predicted ratings. Although both notions are closely related, this paper deals with the first type, i.e. rating prediction.

Depending of the information used to recommend, RSs can be classified [1] into the following categories:

[*]Corresponding author
Email addresses: `jhg@decsai.ugr.es` (Juan F. Huete), `jmfluna@decsai.ugr.es` (J.M. Fernández-Luna), `lci@decsai.ugr.es` (Luis M. de Campos), `mruedam@ugr.es` (Miguel A. Rueda-Morales)

- *Content-based Recommender Systems* that store content information about each item. This information will be used to recommend items similar to those previously preferred by the user. This similarity can be based on how similar the content of the items is or taking into account the similarity with respect to user's preferences (also represented by means of a subset of content features).
- *Collaborative filtering* [23] attempts to identify groups of people with similar tastes to those of the user and recommend items that they have liked. The similarity in taste of two users is calculated based on the similarity in the rating history of the users or items.

Since in this paper we focus on Collaborative filtering, we are going to present this type of RSs in more detail. According to [5], collaborative Recommender Systems can be grouped into memory-based and model-based approaches. On the one hand, memory-based algorithms use the entire rating matrix to make recommendations. On the other hand, in model-based algorithms, predictions are made by building (offline) an explicit model of the relationships between items. This model is then used (online) to finally recommend the product to the users.

Because the predictions in memory-based approaches are obtained by considering the most similar users/items, these methods are also known as neighborhood-based recommendation models. A critical step in this methods is to identify the nearest-neighbors since the way in which the similarity is computed has a significant impact on the performance of the system. Studies and real-world implementations so far have relied on traditional vector similarity measures (say Pearson's correlation in different formulations, cosine, Spearman's Rank, etc. [10, 11, 2, 26, 7]). This type of similarity measures the closeness between users ratings but, as it is well known, they are not able to completely characterize their relationships. In this paper we will explore a new approach to perform the recommendation processes which is based on predictive criteria.

1.1. Motivation of a predictive approach

In order to illustrate our approach we will consider the set of ratings in Table 1 with the objective of computing Ann's prediction for the target item I_t.

Following [11], neighborhood-based methods can be separated into three steps.

1. Weight all users/items with respect to the selected similarity criterion. In our example we will consider the Pearson's correlation (PC) coefficient. Last column of Table 1 presents the PC values between each user's ratings and those of Ann.
2. Select a subset of users/items to use as a set of predictors. We will assume that only the best three neighbors, i.e. those three users having the greatest magnitude in the PC values are used.
3. Compute the prediction from a weighted combination of selected neighbors' ratings. For illustrative purposes, we will assume that the predicted rating is obtained by considering the average rating given to the target item by these neighbors[1].

Therefore, and in order to compute the prediction of Ann's rating, those ratings given by Sara, John and Sue to the target item I_t will be used (note that the same users will be selected using the cosine measure), i.e. $\hat{r}_{a,t} = (4 + 3 + 2)/3 = 3$. The idea behind this expression is that since, for instance, Sara's ratings are quite similar to Ann's, we can use them in the prediction of the rating given by Sara to the target item.

Looking at Table 1, we focus now on Alf and Nick ratings, which have been neglected by means of the previous selection criteria. Nick seems to be a user with opposite preferences to those of Ann, having a PC value equals to minus one. Whenever Nick increases his ratings, Ann decreases hers. Similarly, focusing on Alf's ratings, although his pattern of rating has no (linear) correlation with Ann's, there exists a strong dependence between them. Whenever Alf rates with a value Ann has rated with a different but fixed value, for example, if Alf rates with 1 Ann rates with 3.

[1]We would like to note that the use of this expression does not invalidate our subsequent argument, although it does not take into account facts as the degree of similarity between neighbors or the difference in the scale used to quantify their preferences.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_t	PC
Sara	2	2	2	3	3	3	4	4	4	5	5	5	4	1.00
John	1	2	1	2	3	3	3	3	3	4	4	5	3	0.91
Sue	1	3	1	3	3	3	2	2	4	2	4	4	4	0.51
Bill	1	5	1	5	1	5	1	5	1	5	1	5	1	0.15
Alf	2	2	2	4	4	4	1	1	1	3	3	3	1	0.00
Joe	1	2	3	1	2	3	1	2	3	1	2	3	3	0.00
Nick	4	4	4	3	3	3	2	2	2	1	1	1	2	-1.00
Ann	1	1	1	2	2	2	3	3	3	4	4	4	?	

Table 1: Measuring user similarities. The objective is to provide Ann's prediction for the target item I_t.

These type of dependencies are not properly captured by Pearson's correlation, neither by a vector based similarity measure. Moreover, in the case that during the selection process (step two of the algorithm) both users, Nick and Alf, were selected as Ann's neighbors, the use of their ratings, i.e., $\hat{r}_{a,t} = (4{+}3{+}2{+}\mathbf{2} + \mathbf{1})/5 = 2.4$, will probably worsen the predictions. In this case, instead of consider Nick and Alf's raw ratings to I_t (2 and 1, respectively), it could be more favorable to consider which should be the expected rating given by Ann to I_t after knowing Alf and Nick's ratings. Particularly, given that Alf rates with 1 we could expect that Ann rates with 3. Also, the same rating will be expected when using Nick's rating, and therefore $\hat{r}_{a,t} = (4 + 3 + 2 + \mathbf{3} + \mathbf{3})/5 = 3$.

Therefore, what we are proposing is the use of our beliefs about how Ann would rate I_t (in some sense a prediction of Ann's ratings) instead of the particular rating given by each one of her neighbors. When the users are strongly correlated they might coincide, but as we have shown, there are situations where this does not happen. In this paper we are going to explore this idea from two different perspectives: Firstly, as a way to compute the predictions and, secondly, as an alternative to measure the similarities between users (steps three and one of the previous sketch, respectively).

Focusing on the second objective, the idea will be to measure the capability of each user to predict the Ann's ratings. In this case, using the data in Table 1, we can observe that whenever Sara rates with a value, say x, Anns rates with a fixed value, but this also holds for Alf, and Nick. Nevertheless, by knowing their ratings for an item we know with 'certainty' which is Ann's rating. Therefore, for predictive purposes these three users are equivalent and they should be the ones selected when looking for the three nearest-neighbors of Ann. Our hypothesis is that if a user was good predicting the past ratings, then his/her predictions shall be also helpful to recommend ratings in the future.

By joining the two ideas we will get a new memory-based CF model, which as a whole is the main contribution of this paper. This model tries to predict the probability of the alternative ratings, being an example of Predictive modeling [8]. This paper is organized in the following way: Next section presents related work on memory-based recommender systems. Then Section 3 presents the proposed model, firstly by introducing how the individual suggestions can be combined (as a mixture of multinomial distributions) and secondly, by showing how the neighborhood is determined. Section 4 describes the proposed experimentation and the obtained results. Finally, Section 5 presents the concluding remarks.

2. Related Work: Neighborhood-based Recommender Systems

Previously to present some related work, we will introduce our notation. Formally, in a recommending framework there exists a large number m of items or products $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ and also a large set of n users, $\mathcal{U} = \{U_1, U_2, \ldots, U_n\}$ and for each user, a set of ratings about the quality of certain observed items in $\mathcal{I}$. Among the set of users we want to highlight the active user, U_a or A, which is the one that is interacting with the system, a general user will be denoted by capital letter U or V. Similarly, we also highlight an item I_t, the target item, that will be one for which we are making predictions, we use I to denote a general item. Finally, will denote $\mathcal{R}$ as the set of recorded ratings, $r_{u,i}$ being the rating given by user U to the item I and the ratings predicted by the system will be denoted as $\hat{r}_{a,t}$. Let $\mathcal{S}$ be the set of possible ratings (for instance, $\mathcal{S} = \{1, \ldots, 5\}$ or $\mathcal{S} = \{like, dislike\}$), when the user does not rate the item we consider that $r_{u,i} = \emptyset$. To identify the subset of items rated by the user U, we will use $\mathcal{R}_U$.

As we already mentioned, our work is framed into the neighborhood-based Collaborative Filtering approach, where the predictions rely on the ratings given by similar users in the system. The intuition is that if two users U and V have a similar pattern of rating then the rating given by U to the target item is likely to be similar to that of V. In this scenario the set of ratings, $\mathcal{R}$, are directly used to predict the ratings (in Section 2.5 we highlight the advantages of this approach). The predictions can be done in two different ways, depending on different views of the rating matrix: user-based methods [10, 25] that use an aggregation measure which considers the ratings given by similar users for the same item. Also item-based approaches [24, 6], which take into account the similarity between items (two items are similar if they have been rated similarly), appear as good alternatives to the user-based method. In this case, the predictions are generated considering those ratings given by the active user to similar items.

In a user-based recommender system we can distinguish two main steps: The computation of the active user's neighborhood and the processes involved in the rating predictions. Also, and in order to select the best neighbors how to compute the similarity between users is a critical element. In the following sections, we are going to consider these problems.

2.1. Similarity Measures

The three most common metrics in user-based RS [7] are:

- Pearson Correlation Coefficient (PC): This metric measures the degree of association between the ratings patterns using a value between -1 and +1. A positive value is an evidence of a general tendency that large ratings for user U are associated with large ratings of V and small ratings of U tends to be associated with small ratings of V (a negative value for the correlation implies an inverse association). The PC can be computed by means of the following formula:

$$PC(U,V) = \frac{\sum_j (r_{u,j} - \overline{r}_u) \cdot (r_{v,j} - \overline{r}_v)}{\sqrt{\sum_j (r_{u,j} - \overline{r}_u)^2 \cdot \sum_j (r_{v,j} - \overline{r}_v)^2}}, \tag{1}$$

  where sums over j are applied to those items that users U and V have rated, $\mathcal{R}_U \cap \mathcal{R}_V$. If there are no common items between U and V, then $PC(U,V) = 0$ by default. Furthermore, $\overline{r}_u$ is the average rating value for the user U, i.e. $\overline{r}_u = \frac{1}{|\mathcal{R}_U|} \cdot \sum_{I_k \in \mathcal{R}_U} r_{u,k}$.

  A related metric is the Spearman Rank Correlation, which tries to measure the similarity between users considering a ranking over the rated items instead of items ratings.

- Cosine Measure: This metric defines the similarity between two users as the cosine of the angle between the rating vectors, with values between 0 and 1. A larger value means that the similarity of the ratings increases (the two vectors are getting closer). The Cosine similarity of user U and user V is defined as

$$COS(U,V) = \frac{\sum_{i \in \mathcal{R}_U \cap \mathcal{R}_V} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in \mathcal{R}_U} r_{u,i}^2 \sum_{i \in \mathcal{R}_V} r_{v,i}^2}} \tag{2}$$

  Cosine similarity ignores the differences in rating scales between users. This fact can be taken into account using the adjusted cosine similarity, which has been found to perform even better than Pearson correlation within item-based collaborative filtering [19].

- Mean Square Difference: This is a distance measure (but the inverse can be considered as a similarity metric) which evaluate the distance between two users as the average squared difference between the rating given by the user to the same items, compute as:

$$MSD(U,V) = \frac{\sum_{i \in \mathcal{R}_U \cap \mathcal{R}_V} (r_{u,i} - r_{v,i})^2}{|\mathcal{R}_U \cap \mathcal{R}_V|}. \tag{3}$$

Different empirical analysis have been performed trying to determine the best similarity measure for computing the similarity weights [5, 10, 11, 14] with Pearson's correlation, PC, being found to perform better.

2.2. Selecting the neighbors

In order to perform recommendations for the target item I_t, only the users who have rated this item can be used. Among them, we have to select the best ones. Following [11], it has been demonstrated that it is better to set a fixed number of neighbors (potentially in a range from 20 to 60) than using a threshold on the similarity weights. Moreover, a significant gain in prediction accuracy is obtained by incorporating a significance weighting by devaluing correlations that are based on small numbers of co-rated items. Particularly, the best-n neighbors having larger values of the following similarity measure, denoted as $N_t(a)$, are used:

$$sim(U,V) = PC(U,V) \cdot CF, \tag{4}$$

with

$$CF = \begin{cases} 1 & \text{if } k > 50 \\ \frac{k}{50} & \text{otherwise} \end{cases}$$

k being the number of items in common.

2.3. Computing the Predictions

In this section we will present those methods used in the literature (see [7] for a good review) to compute the predicted rating, $\hat{r}_{a,t}$. As we said in Section 1.1, these methods use the ratings given to I_t by the best-n neighbors. Particularly, it can be considered two different approaches that mainly differ on whether the given ratings are normalized or not.

1. Raw ratings. In this case, it is considered that users might have a different similarity values, and a weight that reflects this similarity is added.

$$\hat{r}_{a,t} = \frac{\sum_{V \in N_t(a)} sim(A,V) r_{v,t}}{\sum_{V \in N_t(a)} sim(A,V)}. \tag{5}$$

2. Normalized ratings. By means of the normalization we are considering that users may use different rating scale to quantify the same level of preferences for an item. Different normalization strategies can be followed, but the most popular schemes are mean-centering and Z-score.
   The first one tries to consider that users may use different rating scale to quantify the same level of preferences, and the normalization is obtained by comparing the rating to the mean rating. In this case, the predicted rate is obtained as

$$\hat{r}_{a,t} = \overline{r}_a + \frac{\sum_{V \in N_t(a)} (r_{v,t} - \overline{r}_v) \cdot sim(A,V)}{\sum_{V \in N_t(a)} sim(A,V)}. \tag{6}$$

   The second ones, also consider the variance in the individual rating scales and the predicted rate is computed dividing the mean-centered rating by the standard deviation, σ:

$$\hat{r}_{a,t} = \overline{r}_a + \sigma_a \frac{\sum_{V \in N_t(a)} (r_{v,t} - \overline{r}_v)/\sigma_v \cdot sim(A,V)}{\sum_{V \in N_t(a)} sim(A,V)}. \tag{7}$$

The use of some kind of rating normalization has been found to improve the predictions [11, 14] although choosing one of them, mean-centering or Z-score, highly depends on the features of the rating matrix, $\mathcal{R}$.

In general terms, a user-based rating prediction can be seen as a process where each neighbor, $V \in N_t(a)$, suggests a rating for the target item, denoted by $f_A(V,t)$, and then the different proposals are combined by weighting the contribution of each neighbor by its similarity with respect to the active user, i.e.,

$$\hat{r}_{a,t} = \frac{\sum_{V \in N_t(a)} sim(A,V) f_A(V,t)}{\sum_{V \in N_t(a)} sim(A,V)}. \tag{8}$$

Thus, in the case of eq. 5 the suggestions are the neighbors' raw ratings $f_A(V,t) = r_{v,t}$, whereas in eqs. 6 and 7 these suggestions are normalized by taking into account how both users rated the observed items, by considering the mean-centering and Z-score, $f_A(V,t) = \overline{r}_a + (r_{v,t} - \overline{r}_v)$ and $f_A(V,t) = \overline{r}_a + \sigma_a(r_{v,t} - \overline{r}_v)/\sigma_v$, respectively.

2.4. Performance evaluation measures

In order to test the performance of the model one widely used metric is the system predictive accuracy, i.e. how well the system predicts the user's true ratings or preferences. Following [12], MAE is the metric typically used for this purpose, defined as the mean absolute error between the true test ratings, $\mathcal{R}$, and on the predicted ratings, $\hat{\mathcal{R}}$. Another accuracy metric is the root mean squared error (RMSE), which gives more strength to higher error, which computes the root of the expected squared loss in order to obtain the error in the same magnitude than ratings. This metric has been proved to be correlated with MAE metrics. In the literature it can be found classification accuracy metrics such as the percentage of success, $\%S$, which measures the number of correct predictions given by the system.

Obviously, depending on the recommending problem considered different metrics can be used. Thus, in the case of looking for an ranking over the set of recommended items, there are several ranking quality metrics which are commonly used to judge how good is the ranking proposed with respect to those items which has been really ranked by the user. For example, it can be found correlation-based metrics measuring the linear relationship between the rankings, those metrics which considers the precision on the top k predictions (P@k) or those metrics taking into account the position in the ranking of those items accessed by user, such as the normalized Discounted Cumulative Gain (nDCG).

2.5. Why memory-based approaches are worth of pursuit?

In this section we will present the advantages of using a neighborhood-based method. These methods have reached a great popularity because they are simple and intuitive on a conceptual level being also sufficient for many real-world problems [10, 24, 19, 26]. The other alternative is the use of a model-based approach [13, 4, 17, 18] where a model is built offline trying to reflect the relationships between users and items. In order to build this model several data mining techniques might be used, learning the model from the rating's matrix, $\mathcal{R}$. Then, the model is used (online) to finally recommend the product to the users.

Focusing on the task of predicting ratings, model-based approaches seems to outperform memory-based approaches [17], but there is an strong support about the fact that best accuracy can be obtained when combining different models that complement the shortcomings of each other, as Netflix Prize competition has shown [3]. For instance, good results have been obtained by using model-based methods to learn interpolation weights in neighborhood-based approaches.

Moreover, there are other factors beyond prediction accuracy which are important from the users perspective, where neighborhood-based approaches takes advantage [7] as *serendipity* (novelty) or *justifiability.* Related to serendipity, model-based predictions always tends to recommend items closer to the user's (latent) tastes, being difficult to recommend an item very different from his/her usual preferences. This problem is alleviated in neighborhood-based approaches because they capture local associations in the data. Also, the recommendations can be better explained (justified), helping the users to understand the recommendations. In commercial applications this understanding are finally related to customer's loyalty. These models have also some advantages from the perspective of the implementation in real system: they are simple (easy to implement, with fewer number of parameters to be tuned), efficient (no training is required and recommendations can be computed instantaneously using pre-computed nearest-neighbors) and scalable (little affected by the constant addition of users, items or ratings). Besides, there are others recommendation tasks, as group recommending, where memory-based approaches represents a good alternative [16, 21]. This is because, in this task, the groups are usually built from the scratch, being difficult to (previously) learn a model able to represent the unknown groups' preferences.

3. Predictive-based Model

As we said in Section 1.1, our objective in this paper is to study the capability of a user to predict a rating from two different perspectives: On the one hand, as a new alternative for computing the prediction of the rating for an unobserved item and, on the other hand, as the criterion used to select the best neighbors.

3.1. Selecting the predicted rating

As we said, predicting is a process where each neighbor, $V \in N_t(a)$, suggests a rating for the target item, $f_A(V,t)$, and these ratings have to be aggregated using a weighted function (Eq. 8). In this section we are going to present a different approach to perform these suggestions: Instead of considering the raw rating or a deviation from the mean rating we will consider how probably is that the active user rate with a value s, $r_{a,t} = s$, given that we know the value of the rating given by V to the target item, say $r_{v,t} = k$, i.e. $Pr(r_{a,t} = s | r_{v,t} = k)$. Our hypothesis is that these distributions can be used to characterize the user's rating pattern and could be estimated from the rating matrix $\mathcal{R}$ by relative counts over the set of items rated by either A and V, i.e. $\mathcal{R}_A \cap \mathcal{R}_V$. Following the example in Table 1, we can see that Sara rates the target item with 4, and that $Pr(Ann = 3 | Sara = 4) = 1$.

It is interesting to note that this kind of suggestions do not depend on whether the ratings are positive or negatively correlated neither the rating scale of the users. For example, suppose that each one of Sara's ratings were decreased by one. In this case, Sara would rate the target item with 3 and the probability distribution $Pr(Ann = s | Sara = 3)$ does not change. Similarly, there is no difference in the predictive probabilities if Bill alternates between 2 and 4, instead of 1 and 5 in Table 1. This represents an advantage over the methods in Section 2.3 since no normalization is required.

The main drawback of this approach is that the estimations are not reliable when ratings do not co-occur or they are obtained from a few number of common observations, i.e. the size of $\mathcal{R}_A \cap \mathcal{R}_V$ is small. In order to avoid this situation we consider that, a priori, all the ratings are equally probable. Particularly, we will use Laplace estimation which essentially starts by saying that every possible rating has occurred at least once. Therefore the predictive probabilities will be estimated by means of

$$Pr(r_{a,t} = s | r_{v,t} = k) = \frac{n(r_{a,.} = s, r_{v,.} = k) + 1}{n(r_{v,.} = k) + |\mathcal{S}|}, \tag{9}$$

where $n(r_{a,.} = s, r_{v,.} = k)$ represents the number of items in $\mathcal{R}_A \cap \mathcal{R}_V$ with user A rating with s and user V rating with k and $n(r_{v,.} = k)$ represents the number of times that user V rated with value k. Following the example in Table 1, we can see that Sara rates the target item with 4, and that $Pr(Ann = s | Sara = 4)$, with $s \in \{1, \ldots, 5\}$ are $0.125, 0.125, 0.5, 0.125, 0.125$, respectively.

Once each one of the neighbors' predictive probabilities have been estimated they must be combined. This combination represents also a difference with the classical methods, where an aggregation of a set of fixed valued is used. In our case we have to aggregate a set of probability distributions over the rating values. This aggregation can be seen as a mixture of different multinomial distributions [22]. As a result we will obtain a probability distribution characterizing our belief about the value of estimated rating, i.e.

$$Pr(r_{a,t} = s) = \frac{\sum_{V \in N_t(a)} sim(A,V) Pr(r_{a,t} = s | r_{v,t})}{\sum_{V \in N_t(a)} sim(A,V)}, \forall s. \tag{10}$$

Since the objective is the prediction of the rating that the active user should give to I_t, $\hat{r}_{a,t}$, a final decision becomes necessary. There are several methods for computing this prediction from a probability distribution over ratings. In this paper we will use three different alternatives: the most probable, the average and the median rating using the a posteriori probability in the form:

MP :
$$\hat{r}_{a,t} = \{r | \max\{Pr(r_{a,t} = r)\}. \tag{11}$$

MED :
$$\hat{r}_{a,t} = \{r | Pr(r_{a,t} < r) \leq 0.5, Pr(r_{a,t} > r) > 0.5\}. \tag{12}$$

AVG :
$$\hat{r}_{a,t} = \sum_{s} Pr(r_{a,t}) \cdot s. \tag{13}$$

Following with the example, we assume that $Pr(r_{a,t} = s)$ with $s \in \{1, \ldots, 5\}$ is equal to $0.10, 0.15, 0.30, 0.35, 0.10$, respectively. The ratings obtained using the different methods are: for MP the rating is 4 as it is the most probable, for MED is 3 ($0.10 + 0.15 + 0.30 \geq 0.5$) and for AVG is 3 ($(0.10 \cdot 1) + (0.15 \cdot 2) + (0.30 \cdot 3) + (0.35 \cdot 4) + (0.10 \cdot 5) = 3.2$).

3.2. Selecting the neighborhood using a predictive-based criterion

In neighborhood-based Collaborative Filtering the predictions rely on the ratings given by similar users in the system since it is assumed that users with similar tastes tends to rate similarly. Different measures has been used (see Section 2.1) trying to capture the *similarities* between those ratings given by both users to the set of observed items. As we said in Section 1.1, these measures perform well when the ratings are highly correlated, but they are not able to capture other dependencies between the ratings.

In this paper a new metric to measure the quality of a neighbor will be proposed. Our hypothesis is that if a user U was good at predicting the active user's past ratings, then his/her prediction for an unobserved item will be also good. [2] Note that the key aspect in this approach is that we will consider rating's predictions instead of raw ratings. Therefore, we will have, on the one hand, the ratings given by the active user, $\mathcal{R}_A = \{r_{a,1}, \ldots, r_{a,m}\}$, to his/her m rated items and, on the other hand, the ratings that will be suggested (predicted) for each item by the user U, denoted as $\hat{\mathcal{F}}_A(U) = \{\hat{r}_{a,1}, \ldots, \hat{r}_{a,m}\}$.

In order to determine the quality of the predictions, which measures in some sense the predictive capability of a user, we have to consider if the predicted ratings, $\hat{\mathcal{F}}_A(U)$, and the original ones, $\mathcal{R}_A$, are similar in absolute terms. Note that we are not interested in measuring whether the ratings vary in the same way or not. For example, a strong correlation does not necessarily implies good predictions, as it is the case where the predictions are equal to the original rating plus a constant value k. Then, the ratings are highly correlated (PC is equal to one) but the predictions worsen with k. Also, cosine measure is not appropriate for this purpose. It does provide an accurate measure of similarity but with no regard to magnitude, since the rating vectors are treated as unit vectors by normalizing them. But magnitude is a very important factor at determining the quality of the prediction. For example, suppose $\mathcal{R}_A = \{2, 2, 3, 3\}$ and the suggestions $\hat{\mathcal{F}}_A(U) = \{1, 2, 3, 4\}$ and $\hat{\mathcal{F}}_A(V) = \{6, 6, 7, 7\}$, with cosine values 0.967 and 0.993, respectively. In this case, the cosine criterion will say that V has better predictions than U, which is clearly erroneous.

In this paper, we propose the use of a loss function to measure the magnitude of difference between the two rating, $L(r_{a,i}, \hat{r}_{a,i})$, representing the cost incurred when the true rating $r_{a,i}$ and user U predicts the rating $\hat{r}_{a,i}$. Then, we will explore two different loss functions:

L_H: This function measures the number of correct prediction only, i.e. only considering whether we have a success or failure in the predictions:

$$L_H(r_{a,i}, \hat{r}_{a,i}) = \begin{cases} 0 & \text{if } r_{a,i} = \hat{r}_{a,i} \\ 1 & \text{otherwise.} \end{cases} \tag{14}$$

[2]We would like to note that we have explored [15] the idea of considering the quality of a user as a neighbor by measuring how likely were those ratings given by the active user when using the predictive probabilities. This probability were measured by considering the likelihood over the rating in $\mathcal{R}_{\mathcal{A}}$.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}
$\hat{\mathcal{F}}_A(U)$	1	1	2	2	2	3	4	3	3	5	5	5
$\hat{\mathcal{F}}_A(V)$	1	4	1	2	5	5	3	3	3	4	4	1
$\mathcal{R}_A$	1	1	1	2	2	2	3	3	3	4	4	4

Table 2: Candidate Ratings.

L_M: This loss functions measures how close the predictions are to the true ratings by considering the average absolute deviation between a predicted rating, $\hat{r}_{a,i}$, and the user's true rating, $r_{a,i}$:

$$L_M(r_{a,i}, \hat{r}_{a,i}) = abs(r_{a,i} - \hat{r}_{a,i}). \tag{15}$$

Therefore, the predictive capability of a user can be measured by considering the expected loss over the active user's past ratings, i.e.

$$EL(A, U) = \frac{\sum_{i \in \mathcal{R}_A} L(r_{a,i}, \hat{r}_{a,i}(u))}{|\mathcal{R}_\mathcal{A}|}. \tag{16}$$

This expected loss can be considered as a distance metric between the two vectors of ratings. Particularly, the use of L_H and L_M in the expected loss is related to the use of Hamming and Manhattan distances, respectively. For example, considering the data in Table 2 we have that $EL_H(A, U) = 6/12 = 0.5$, $EL_H(A, V) = 4/12 = 0.33$, $EL_M(A, U) = 6/12 = 0.5$ and $EL_M(A, V) = 12/12 = 1$.

Since we are considering a memory-based approach for recommending a rating for the target item I_t, we have to compute the $EL(A, U)$ for those users, U, who rated the target item I_t, i.e. U such that $r_{u,t} \in \mathcal{R}_U$. Then these users will be ranked in increasing order of their expected loss and the top N shall be selected as active user's neighbors.

3.2.1. Estimating the pasts rating

So, the problem is reduced to determine how a given user, say U, might predict the active user's ratings, i.e. how to compute $\hat{\mathcal{F}}_A(U)$. In order to capture different dependence criteria among ratings, these predictions will be based on the predictive probabilities discussed in Section 3.1, i.e. $Pr(r_{a,i}|r_{u,i})$. Then, the recommended rating will be obtained as the most probable (eq. 11), the median (eq. 12) or the average (eq 13) rating from $Pr(r_{a,i}|r_{u,i})$.

Let's suppose that U was asked about the rating that A would give an item $I_i \in \mathcal{R}_A$, i.e. we want to know $\hat{r}_{a,i}$. Obviously, it is assumed that when U have to suggest this rating he/she does not know the real rating give by A to the item I_i. Therefore, the information available for making his/her prediction is $\mathcal{R}_A \setminus \{r_{a,i}\} \cup \mathcal{R}_U$. Two different situations can be considered:

i) User U rated I_i, i.e. $r_{u,i} \in \mathcal{R}_U$: This is the same situation that we have when doing recommendations for an unobserved item. Thus, the predictive probabilities can be computed using Equation 9.

ii) User U does not rate I_i, i.e. $r_{u,i} \notin \mathcal{R}_U$: In this case, since U does not rate I_i, a naive approach might be to consider that all the ratings in $\mathcal{S}$ are equally probable. A second alternative might be to use the active user's average rating, $\overline{r}_a$, computed over $\mathcal{R}_A \setminus r_{a,i}$. In both cases, the prediction does not depend on U.
In this paper we will consider a third alternative which considers that knowing that U does not rate I_i might be helpful for predicting the A's rating, i.e. we are assuming a situation where a missing rating might be informative. For example, consider that A and U only rated 20 items in common (with 5 and 4 stars, respectively) and that the rest of the 20 ratings given by A got 1 star. In this case, if we ask U for a suggestion for an item I_i, knowing that I_i was rated by A and was not rated by U, $f_A(U, i)$, the value 1 will be selected probably, which differs of the A's average rating, 3.
Although this is a simplistic example, it can give as an idea that there exists situations where the mechanism for obtaining missing ratings cannot be ignored in general. Recent work by Marlin et al.

```
Inputs: $A$ active user, $I_k$ target item
Output: $\hat{r}_a^k$ predicted rating.
1. Neighbors Selection
  1.1 For each $U$ who rated the target item $I_t$
    1.1.1 Compute the conditional probabilities
        $Pr(r_a = x | r_u = y), \forall x \in \mathcal{S}, \forall y \in \mathcal{S} \cup \{\emptyset\}$
    1.1.2 Perform the prediction $\hat{\mathcal{F}}_A(U) = \{\hat{r}_{a,1}, \ldots, \hat{r}_{a,m}\}$
    1.1.3 Compute the $EL(A, U)$.
  1.2 Rank the users and select the best-N neighbors
2. Predictive Process
  2.1 Aggregate the predictive probabilities of each
      neighbor $V$
        $Pr(r_a = x) = \sum_V w(A, V) Pr(r_a = x | r_{v,t}), \forall x \in \mathcal{S}$
  2.2 Select a rating $\hat{r}_{a,t}$ from $Pr(r_a)$.
```

Table 3: Predictive Model

[20] provided evidence that the data typically used for training and testing recommender systems are Missing Not At Random. This may be a consequence of the fact that users are free to choose which items to rate, as for example a user having an strong preference for horror movies, which would not to be shared by most users.

Therefore, in order to compute the necessary predictive probabilities, $Pr(r_{a,i} = s | r_{u,i} = \emptyset)$, to perform the suggestion we will use again Laplace estimation, but in this case prior information will be taken into account,

$$Pr(r_{a,i} = s | r_{u,i} = \emptyset) = \frac{n^*(r_{a,\cdot} = s) + |\mathcal{S}| \times \alpha_s}{|\mathcal{R}_A \setminus \mathcal{R}_U| + |\mathcal{S}|}, \tag{17}$$

where $n^* = (r_{a,\cdot} = s)$ represents the number of items in $\mathcal{R}_A \setminus \mathcal{R}_U$ with user A rating with s and α_s represents the A's prior probabilities over the candidate ratings computed as the probability of A rating with value s from the set $\mathcal{R}_A$ and $|\mathcal{S}|$ will be used as the strength of the prior. Note that in the case the ratings were missing at random $Pr(r_{a,i} = s | r_{u,i} = \emptyset)$ must be quite similar to $Pr(r_{a,i} = s)$ and therefore the situation is equivalent to consider only the ratings of the active user to perform the predictions.

Finally, the suggested rating, $\hat{r}_{a,i}$ will be obtained as the most probable (eq. 11), the median (eq. 12) or the average (eq 13) rating.

3.3. Differences with vector-similarity based approaches

To conclude, the algorithm in Table 3 summarizes our recommendation model, where $w(A, V)$ represents the (normalized) weighted contribution of each neighbor. Now we are going to highlight the main differences with respect to classical user-based similarity approaches. The first one is that by means of our approach different types of relationships/dependencies between ratings beyond linear correlations or scale invariances can be considered. Also related, in order to determine the strength of the relationships we use ratings' predictions instead of raw ratings. Moreover, in order to perform these predictions our approach considers that missing ratings might be informative. Also, there exists some differences in the predictive step. For example, on the one hand we combine predictive probabilities instead of fixed suggestions and, on the other hand, by using these probabilities we are avoiding the problem of rating normalization.

From the efficiency perspective, we have to say that our approach is more expensive than vector-based measures since the predictions for the past ratings must be determined in order to compute the similarity. Nevertheless, it can be computed in a time linear with the number of items rated by the active user (it is necessary to iterate 2 times over the rated items, the first one to compute the conditional probabilities (step 1.1.1 in Table 3) and the second to compute the predictions and the EL (steps 1.1.2 and 1.1.3 in Table 3)).

4. Evaluation of the Recommender Model

This section establishes the evaluation settings and also presents the experimental results for the performance of the model. We want to express that in this work our purpose is to evaluate our approach as an alternative to memory-based recommended system, which complement and improves the existing state-of-art memory-based approaches.

4.1. Data set and experimental methodology

The objective of our experimentation is to measure the capability of the system at predicting the interest of the user for an unobserved item, i.e. we will consider the task of *rating prediction.* In order to measure the quality of our predictions we will use the MAE metric, typically used for this purpose, and the percentage of success in the predictions $\%S$ (see Section 2.4). In terms of the test data sets, we have decided to use three different datasets:

- *1M MovieLens (ML):* It was collected by the GroupLens Research Project at the University of Minnesota and contains 1,000,209 anonymous ratings (on a scale of 1 to 5) of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000. We have created randomly the test set using about 20% of ratings for every movie and using the rest of ratings for the training set.
- *Yahoo! Movies (YMv):* This data set contains ratings of 11,915 movies rated by 7,642 users also divided randomly into training and test sets with 211,231 and 10,136 ratings, respectively. In this case we have used the Yahoo! movies converted ratings (also, on a 1 to 5 scale).
- *Yahoo Music (YMs):* This dataset contains ratings for songs supplied by users during normal interaction with Yahoo! Music services collected between 2002 and 2006. The rating data includes approximately 300,000 user-supplied ratings (on a 1 to 5 scale) given by 15,400 users to 1000 songs, with at least ten ratings for each user. We have created randomly the test set using about 20% of ratings for every song (59.2502) and using the rest of ratings for the training set (52.454 ratings).

In order to validate our model, and similarly to most machine learning evaluation methodologies, we shall randomly divide our data sets into training (containing 80% of the ratings) and test set (containing 20%). To reduce variability on the results, we run 5-fold cross-validation over different partitions. The results presented in this experimentation are the average values obtained over the five rounds.

Finally, as baseline, we will considered the results obtained with a state-of-art user-based approach where the neighbors have been selected using Pearson's correlation (with significant weight correction, eq. 4) and the individual suggestions were combined using a mean-centering (eq. 6), which works well with different datasets.

4.2. Experimental Results

Previously to discuss the results, some considerations have to be done. Firstly, our model computes a probability distribution over candidate ratings, and this probability have to be converted in a rating, $\hat{r}$, in the range 1 to 5. The criterion used to obtain $\hat{r}$ will be related with the accuracy metric we want to optimize: if we predict the most probable rating we minimize the %S and if we predict the median rating we minimize the MAE . Secondly, and also related with the accuracy criterion, the loss function used to learn the neighborhood depends of the criterion we want to evaluate: when considering MAE or %S we will learn using the loss function EL_M and EL_H, respectively[3]. Finally, when aggregating the neighbors' probabilities (step 2.1 of the algorithm in Table 3) we will considered a normalized expected loss over the past predictions, denoted as wM and wH in Table 4,

From the results in Table 4 some conclusions can be obtained. The first one is that our proposals are competitive (our best results outperforms (5/6) with those obtained with the baseline). This fact indicates

[3]In our experimentation it has been proved that worse results are obtained with other combinations.

Table 4: Accuracy metrics

Neighbors	5	10	20	30	50
		MAE metric MovieLens DataSet			
Baseline	0.7464	0.7191	0.7077	0.7057	**0.7047**
$EL_M + wM$	0.7164	0.7081	0.7036	0.7014	**0.6967**
		MAE metric Yahoo! Movies			
Baseline	0,7530	**0.7453**	0.7477	0.7502	0.7543
$EL_M + wM$	0.7476	**0.7439**	0.7555	0.7606	0.7687
		MAE metric Yahoo! Music			
Baseline	0,9701	0,9377	0,9212	**0,9181**	1,1235
$EL_M + wM$	1,0600	1,0491	1,0385	1,0289	**1,0109**
		%S metric MovieLens DataSet			
Baseline	42.731	43.903	44.380	44.559	**44.570**
$EL_H + wH$	43.409	44.108	44.240	44.550	**44.581**
		%S metric Yahoo! Movies			
Baseline	45,9747	46,9021	47,0501	**47,4152**	47,4053
$EL_H + wH$	52,3086	53,3248	**53,7194**	53,6997	53,6504
		%S metric Yahoo! Music			
Baseline	35,04	35,29	35,39	**35,42**	35,14
$EL_H + wH$	39,5781	40,7105	41,1409	**41,3215**	40,8017

Table 5: Number of common neighbors: Baseline vs. Expected Loss

	5	10	20	30	50
MovieLens	29.59	34.77	39.81	41.31	38.41
Yahoo! Movie	75.37	65.49	55.86	50.15	42.60
Yahoo! Music	64.47	61.35	54.48	48.33	39.98

that there exists many users which are not captured by correlation criteria that might be good for predictive purposes. In order to illustrate this fact, we show in Table 5 the percentage of common neighbors between both approaches when considering the mean average error as the loss function, i.e. $P_B \cap P_L$. From this table we can conclude that Yahoo! databases and MovieLens behaves differently for this purpose. Thus, it seems that both criteria to select the neighborhood have greater agreement with Yahoo! movies than with MovieLens.

We want to note that our method for selecting the neighborhood considers how well a user predicts all the past ratings. Moreover, in our preliminary experimentation, we have found that by using any criteria to give more strength to those neighbors which a large number of co-rated items does not improve the neighborhood's quality. We consider this fact important since it indicates that our approach represents a robust selection criterion.

We have also try to study the effect of considering the strength of the number of co-rated items when combining the predictive probabilities (step 2 of the algorithm in Table 3). In this sense, the results obtained by considering the number of co-rated items in combination with the normalized expected loss were not conclusive, but good results have been obtained when combining this factor with how the ratings given to these common items correlate, in absolute terms. Particularly we use

$$w(A,U) = \frac{1}{K} abs(PC(A,U)) \times \frac{|\mathcal{R}_A \cap \mathcal{R}_U|}{|\mathcal{R}_A|} \quad (18)$$

Table 6: Combining using correlations and predictive capabilities.

Neighbors	5	10	20	30	50
		MAE metric $EL_M + wPr$			
MovieLens	0.6843	**0.6798**	0.6799	0.6805	0.6842
Yahoo Movies	0.6849	**0.6804**	0.6808	0.6879	0.6934
Yahoo Music	0,9297	**0,9278**	0,9465	0,9660	0,9795
		%S metric $EL_H + wPr$			
MovieLens	44.480	44.794	**44.824**	44.676	44.378
Yahoo Movies	53,4530	54,0549	54,0746	**54,2423**	54,0943
Yahoo Music	47,5190	**47,6844**	46,8658	45,7165	44,6970

being K a normalization factor. In our experiments this correction-factor gives better results than the one used in eq. 4. Table 6 shows the obtained results. Some conclusions can be also obtained from this table. Firstly, that better results have been obtained by using together predictive capability and ratings' correlations, which indicates that both metrics reinforce each other. Also, the number of neighbors necessary to obtain the best predictions (the neighborhood's size) has been reduced considerably (surprisingly, with only 10 neighbors we obtain good predictions with all the metrics), showing the ability of the combination at finding the very-best neighbors.

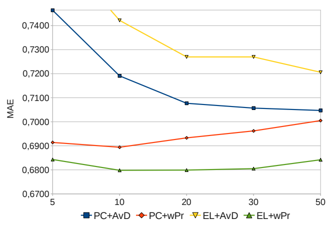
4.2.1. Combining the approaches

In a user-based recommendation system there are two components which might be considered orthogonal: neighborhood selection (N_Sel) and rating prediction (*Pred*). Thus, on the one hand, there exists two alternatives for neighborhood selection, i.e. considering weighted Pearson's correlations between users ratings (eq. 4), denoted as PC, and the other which tries to minimize the expected loss in the predictions, particularly in this section we only consider the mean average error (eq. 15) and will be denoted as EL. On the other hand, with respect to the combination of the neighbors suggestions in the prediction step, we shall consider the case where the neighbor's suggestions are obtained by using the raw differences with respect to the average rating (eq. 6), denoted as AvD, and the case where the user suggestions are weighted predictive probabilities (step 2.1 of the algorithm in Table 3, with the weights in eq. 18), denoted as wPr. Figure 1 shows the MAE values obtained when combining different strategies for doing these tasks[4] and Table 7 summarizes the best results, being NN the optimum number of neighbors.

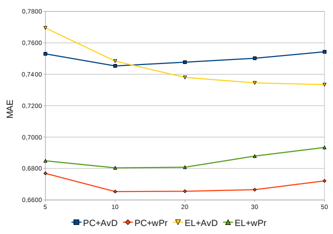
From these figures some conclusions can be obtained: Firstly, combining both approaches (correlations and predictive probabilities) seems to be an alternative worth of pursuit, significant improvements can be achieved. Thus, these results represent improvements of 3.53%, 8,71% and 2.70% with respect to baseline, for MovieLens, Yahoo! Movies and Yahoo! Music, respectively (these results are statistically significant using the adjusted t-test). Also, this experimentation shows that the best results were obtained combining predictive probabilities in the prediction processes. This fact implies that combining predictive probabilities is a robust strategy, independently of the criterion used to select the neighborhood. In this sense, it is remarkable the bad performance of EL+AvD combination (particularly, compared with EL+wPr). Our hypothesis is that this situation can be explained because there must be some selected neighbors whose ratings do not correlates with the active user's ratings, and therefore the use of AvD does not have to be a good approach. With respect to the size of the neighborhood, an important conclusion is that using *wPr* as combination strategy the optimal number of neighbors is around 10, whereas AvD requires large values. Finally, and as global conclusion, this paper shows that there exists neighbors with a good predictive capability which are neglected by correlation-based approaches.

[4]We only show MAE values because it is the metric typically used to measure the performance of this systems, but the performance obtained when considering %S is similar.

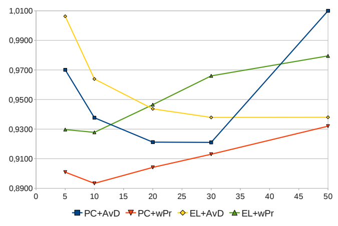
Figure 1: Comparing different models



MovieLens



Yahoo! Movies



Yahoo! Music

Table 7: Best Experimental Results.

DataSet	MAE	NN	N_Sel	Pred.
MovieLens	0.6798	10	EL	wPr
Yahoo Movies	0.6653	10	PC	wPr
Yahoo Music	0.8933	10	PC	wPr

5. Conclusions

In this paper, we shown that the use of predictive probabilities (how probable is that the active user rates the target item with a value given that we know its neighbors ratings) is a valid strategy to both, finding the neighborhood and combining their individuals suggestions. By using a predictive-based approach we have found that there exists many users which, being good for predictive purposes, are not captured by classical correlation criteria. Moreover, we have also shown that the best neighbors can be obtained by combining correlations and predictive capabilities, as in an ensemble-based approach.

In this sense, and as future work, we will explore how the proposed technique can be integrated with model-based approaches, for instance the learned model can be used to learn interpolation weights while using neighborhood, independently of the criteria used to select this neighborhood. Also, we are working on studying different mechanisms to incorporate additional knowledge in order to give a context-based strength to the individual suggestions. By means of this knowledge a user might be consider a good candidate for predicting the rating for some items, for instance sports news or horror movies, whereas the same user can be discarded when suggesting the rating for economy news or family movies. Similarly, by means of this context we can represent that new ratings might be considered more important than the old ones.

Acknowledgments

Work jointly supported by the Spanish Ministerio de Educación y Ciencia (TIN2008-06566-C04-01), the Spanish research programme Consolider Ingenio 201 0: MIPRCV (CSD2007-00018) and the Junta de Andalucía Excellence Project: TIC-04526.

References

[1] G. Adomavicius, A. Tuzhilin. *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.* IEEE Transactions on Knowledge and Data Engineering 17 (6) (2005) 734-749.
[2] H.J. Ahn. *A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem.* Information Sciences 178:37-51, 2008.
[3] R. Bell and Y. Koren. *Lessons from the Netflix Prize Challenge*, SIGKDD Explorations 9, 75-79. 2007.

[4] D.M. Blei, A.Y. Ng and M.I. Jordan *Latent Dirichlet Allocation.* Journal of Machine Learning Research 3, 993-1022. 2003.

[5] J. S. Breese, D. Heckerman, C. Kadie. *Empirical analysis of predictive algorithms for collaborative filtering.* In: 14th Conference on Uncertainty in Artificial Intelligence, pp. 43-52. 1998

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

[7] C. Desrosiers and G. Karypis *A comprehensive survey of neighborhood-based recommendations methods* In Recommender System Handbook, (F. Ricci, L. Rokach, B. Shapira and P. Kantor editors) Springer. pp. 107-144, 2011.

[8] S. Geisser (1993) *Predictive Inference: An Introduction.* New York: Chapman & Hall.

[9] D. Goldberg, D. Nichols, B.M. Oki and D. Terry. *Using collaborative filtering to weave an information tapestry* Commun. ACM 35(12), 61-70 (1992)

[10] J. L. Herlocker, J. A. Konstan, A. Borchers, J. Riedl. *An algorithmic framework for performing collaborative filtering.* Proc. ACM SIGIR 99, pp. 230–237. 1999

[11] J.L. Herlocker, J.A. Konstan, and J.T. Riedl. *An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms.* Information Retrieval, 5(4):287-310. 2002.

[12] J.L. Herlocker, J.A. Konstan, L.G. Terveen and J.T. Riedl. *Evaluating collaborative filtering recommender systems.* ACM Trans. Inf. Syst. 22, 1. 2004, pp. 5-53.

[13] T. Hofmann. *Latent semantic models for collaborative filtering.* ACM TOIS V22 (1) pp 89–115. 2004

[14] A.E. Howe and R.D. Forbes *Re-Considering Neighborhood-Based Collaborative Filtering Parameters in the Context of New Data* Proceeding of the 17th ACM conference on Information and knowledge management, CIKM'08, pp. 1481–1482, 2008

[15] L.M. de Campos, J.M. Fernández-Luna, J.F. Huete and M.A. Rueda-Morales. *Measuring Predictive Capability in Collaborative Filtering* RecSys '09 Third ACM Conference on Recommender Systems, pp. 313–316. 2009

[16] L.M. de Campos, J.M. Fernández-Luna, J.F. Huete and M.A. Rueda-Morales. *Managing uncertainty in group recommending processes.* User Modeling and User-Adapted Interaction, Vol. 19, No. 3. (21 August 2009), pp. 207-242.

[17] Y. Koren,R. Bell and C. Volinsky. *Matrix factorization techniques for recommender systems.* IEEE Computer 42(8), 30–37. 2009

[18] Y. Koren and R. Bell *Advances in Collaborative Filtering* In Recommender System Handbook, (F. Ricci, L. Rokach, B. Shapira and P. Kantor editors) Springer. pp. 145–186, 2011.

[19] G. Linden, B. Smith, J. York. *Amazon.com Recommendations: Item-to-Item collaborative filtering* IEEE Internet Computing, pp. 76-80. 2003.

[20] B. Marlin and R. Zemel. *Collaborative prediction and ranking with non-random missing data.* In ACM Conference on Recommender Systems (RecSys), 2009.

[21] J. Masthoff *Group Recommender System: Combining Individual Models* In Recommender Systems Handbook, F. Ricci and L. Rokach (Eds.). pp. 677-704. Springer 2011.

[22] G.L. McLachlan, and D. Peel *Finite Mixture Models.* Wiley. 2000.

[23] P. Reskick, H.R. Varian. *Recommender systems.* Communications of the ACM, 40(3):56–58, 1997.

[24] B. Sarwar, G. Karypis, J. Konstan, J. Reidl. Item based collaborative filtering recommendation algorithms, Proc.10th Int. Conf. on WWW, p.285-295

[25] U. Shardanand, P. Maes. *Social information filtering: Algorithms for automating "word of mouth".* In Proc. of the SIGCHI Conf. on Human factors in Computing Systems, pp. 210-217. New York, NY, USA. 1995

[26] J.Wang, A.P. de Vries, M.Reinders *Unified Relevance Models for Rating Prediction in Collaborative Filtering* ACM TOIS, 26:3. Article 16. 2008

Capítulo 6

Otros trabajos publicados

- *Measuring predictive capability in Collaborative Filtering.*
  Juan Francisco Huete Guadix; Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Miguel Ángel Rueda Morales.
  ACM Conference on Recommender Systems (3) (No 3. 2009), Proceedings of the third ACM Conference on Recommender Systems, ISBN:978-1-60558-435-5, Nueva York, EE.UU., 2009.

- *Hierarchical Naive Bayes Models for Representing User Profiles* Juan Francisco Huete Guadix; Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Miguel Ángel Rueda Morales.
  Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (31) (No 31. 2008). Proceedings of the 31ST Annual International ACM SIGIR Conference, ISBN: 978-1-60558-164-4. Singapur. 2008.

- *Using structural knowledge in a content-based Recommender System.*
  Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Juan Francisco Huete Guadix; Miguel Ángel Rueda Morales.
  FLINS Conference on computational intelligence in decision and control (8) (No 8. 2008). Proceedings of the 8th international FLINS Conference on computational intelligence in decision and control. Madrid, Spain. 2008.

- *Usando información de segunda mano en un Sistema de Recomendación colaborativo*
  Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Juan Francisco Huete Guadix; Miguel Ángel Rueda Morales.
  ESTYLF 2008: XIV congreso español sobre tecnologías y lógica fuzzy

(14) (No 14. 2008). ISBN: 978-84-691-5807-4. Cuencas mineras asturianas (Langreo-Mieres). 2008.

- *Group recommending: a methodological approach based on bayesian networks.*
  Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Juan Francisco Huete Guadix; Miguel Ángel Rueda Morales.
  IEEE international workshop on web personalization, recommender systems and intelligent users. Interfaces (3) (No 3. 2007). Proceedings of the IEEE third international workshop on web personalization, recommender systems and intelligent users interfaces. ISBN: 1-4244-0832-6. Estambul. 2007.

- *Using structural content information for learning user profiles*
  Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Juan Francisco Huete Guadix; Miguel Ángel Rueda Morales.
  Information retrieval and graphical models workshop (1) (No 1. 2007).Proceedings of the information retrieval and graphical models workshop. Amsterdam. 2007.

- *Average and Majority Gates: Combining Information by means of Bayesian Networks*
  Luis Miguel de Campos Ibañez; Juan Manuel Fernández Luna; Juan Francisco Huete Guadix; Miguel Ángel Rueda Morales.
  Symbolic and Quantitative Approaches to Reasoning with Uncertainty. Lecture Notes in Computer Science. Volume 4724/2007, 572-584. 2007.

Bibliografía

[1] Resnick, P., Varian, H.R.: Recommender systems. Commun. ACM **40**(3), 56–58 (1997). DOI http://doi.acm.org/10.1145/245108.245121

[2] de Campos, L.M.: Modelos de recuperación de información basados en redes de creencia (2003)

[3] Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)

[4] Jensen, F.V.: An Introdution to Bayesian Networks. UCL Press (1996)

[5] de Campos, L.M., Fernández-Luna, J.M., Huete, J.F.: A collaborative recommender system based on probabilistic inference from fuzzy observations. Fuzzy Sets Syst. **159**(12), 1554–1576 (2008). DOI http://dx.doi.org/10.1016/j.fss.2008.01.016

[6] Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22**(1), 5–53 (2004). DOI http://doi.acm.org/10.1145/963770.963772

[7] Burke, R.: The adaptive web. chap. Hybrid web recommender systems, pp. 377–408. Springer-Verlag, Berlin, Heidelberg (2007). URL `http://portal.acm.org/citation.cfm?id=1768197.1768211`

[8] Belkin, N.J., Croft, W.B.: Information filtering and information retrieval: two sides of the same coin? Commun. ACM **35**(12), 29–38 (1992). DOI http://doi.acm.org/10.1145/138859.138861

[9] Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING **17**(6), 734–749 (2005)

[10] Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. pp. 43–52 (1998). URL `citeseer.ist.psu.edu/breese98empirical.html`

[11] Salakhutdinov R., M.A., G., H.: Restricted boltzmann machines for collaborative filtering. In: 24th Annual International Conference on Machine Learning, pp. 791–798 (2007)

[12] T., H.: Latent semantic models for collaborative filtering. ACM Transactions on Information Systems (22), 89–115 (2004)

[13] Blei D., N.A., M., J.: Latent dirichlet allocation. Journal of Machine Learning Research (3), 993–1022 (2003)

[14] Koren, Y.: Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, pp. 426–434 (2008)

[15] Y., K.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2008)

[16] A., P.: Improving regularized singular value decomposition for collaborative filtering. In: KDD Cup and Workshop (2007)

[17] R., S., A., M.: Probabilistic matrix factorization. Advances in Neural Information Processing Systems (20), 1257–1264 (2008)

[18] Li, Q., Kim, B.M.: Clustering approach for hybrid recommender system. In: in: IEEE/WIC Proceedings of the International Conference on Web Intelligence (2003)

[19] Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering. In: In Proceedings of the 2001 SIGIR Workshop on Recommender Systems (2001)

[20] Castillo, E., Gutierrez, J., Hadi, A.: Expert Systems and Probabilistic Networks Models. Springer-Verlag (1997)

[21] Díez, F.J., Druzdzel, M.J.: Fundamentals of canonical models. In: Proc. of IX Conferencia de la Asociacion Espanola para la Inteligencia Artificial (CAEPIA-TTIA 2001), pp. 1125–1134 (2001)

[22] Jameson, A., Smyth, B.: Recommendation to groups. In: P. Brusilovsky, A. Kobsa and W. Nejdl (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Springer Verlag LNCS 4321. pp. 596–627 (2007)

[23] Masthoff, J.: Group modeling: Selecting a sequence of television items to suit a group of viewers. User Model User-Adap Inter. **14**, 37–85 (2004)

[24] Masthoff, J., Gatt, A.: In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. User Model User-Adap Inter. **16**, 281–319 (2006)

[25] Yu, Z., Zhou, X., Hao, Y., Gu, J.: Tv program recommendation for multiple viewers based on user profile merging. User Modeling and User-Adapted Interaction **16**(1), 63–82 (2006). DOI http://dx.doi.org/10.1007/s11257-006-9005-6

[26] Ali, K., van Stam, W.: Tivo: making show recommendations using a distributed collaborative filtering architecture. In: KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 394–401. ACM, New York, NY, USA (2004). DOI http://doi.acm.org/10.1145/1014052.1014097

[27] Degemmis, M., Lops, P., Semeraro, G.: A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. User Modeling and User-Adapted Interaction **17**(3), 217–255 (2007). DOI http://dx.doi.org/10.1007/s11257-006-9023-4

[28] Popescul, A., Ungar, L., Pennock, D., Lawrence, S.: Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In: 17th Conference on Uncertainty in Artificial Intelligence, pp. 437–444. Seattle, Washington (2001). URL `citeseer.ist.psu.edu/popescul01probabilistic.html`

[29] de Campos, L.M., Fernández-Luna, J.M., Huete, J.F.: A bayesian network approach to hybrid recommending systems. In: Eleventh International Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp. 2158 – 2165. Paris, Francia (2006)

[30] Su, X., Khoshgoftaar, T.M., Zhu, X., Greiner, R.: Imputation-boosted collaborative filtering using machine learning classifiers. In: SAC '08: Proceedings of the 2008 ACM symposium on Applied computing, pp.

949–950. ACM, New York, NY, USA (2008). DOI http://doi.acm.org/10.1145/1363686.1363903