UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
Departamento de Ing. Civil Informática y
Ciencias de la Computación

Profesor Patrocinante:
Gonzalo Rojas Durán

Comisión:
Andrea Rodríguez
Diego Seco

MODELADO DE SISTEMAS DE RECOMENDACIÓN CON INTEGRACIÓN DE REDES SOCIALES

POR ISMAEL ALEJANDRO GARRIDO CHARLÍN

Informe de Memoria de Título presentado a la
Facultad de Ingeniería de la Universidad de Concepción para optar al
título profesional de Ingeniero Civil Informático

Diciembre de 2015
Concepción, Chile.

Índice general

Índice de figuras

Índice de cuadros

Resumen

Para hacer frente al continuo y acelerado aumento de la información que circula en Internet, se han desarrollado una gran cantidad de herramientas que facilitan la navegación de los usuarios. Una de estas herramientas son los sistemas de recomendación, los cuales utilizan información vinculada a sus usuarios para sugerirles elementos que probablemente sean de su interés. Es aquí donde las redes sociales se vuelven una potencial fuente de información, capaz de complementar los sistemas de recomendación y agilizar su desarrollo. Sin embargo, no existe un marco de trabajo que permita hacer uso de esta información, por lo que cualquier aproximación queda exclusivamente en manos de un eventual desarrollador.

Esta Memoria de Título propone el uso de modelos para guiar el desarrollo de sistemas capaces de generar recomendaciones a partir la información obtenida de alguna red social. Para esto, utiliza modelos UML que describen el comportamiento de las redes sociales, lo cual facilita la generación automática de código y permite obtener resultados en pocos minutos.

Capítulo 1

Introducción

En este capítulo se presenta una descripción general del problema a tratar y de cómo se abordó, se describen los objetivos propuestos y se detalla la estructura establecida para esta Memoria de Título.

1.1. Descripción general del Problema

Debido al continuo y acelerado crecimiento de la cantidad de información que circula en Internet, se hace imperativo reducir la sobrecarga de ésta que aqueja a los usuarios a la hora de buscar contenido través de la web. Para solucionar este problema, se han propuesto distintos métodos que facilitan la experiencia de navegación del usuario, pero uno de los métodos más utilizados, y además con un gran potencial de explotación, son los sistemas de recomendación.

Los sistemas de recomendación se encargan principalmente de sugerir a los usuarios ítems[1] que son destacables del resto en base a algún criterio. Estos ítems usualmente poseen características asociadas y pueden ser evaluados de alguna manera por los usuarios. De esta forma, es común que para las recomendaciones se seleccionen los ítems más populares, los mejor evaluados por la comunidad o los que más se adecuen a las preferencias del usuario. La idea detrás de todo esto es dar a conocer a un usuario, de forma personalizada y automática, aquellos ítems que posiblemente le interesen pero que probablemente no conozca o aún no ha evaluado.

Para que lo anterior sea posible, es necesario contar con un mínimo nivel de conocimiento sobre cada usuario y sus preferencias. Por esta razón, aunque existen métodos que permiten recopilar información sobre las preferencias del usuario de forma implícita [2] (por ejemplo, midiendo el tiempo o la frecuencia con que un usuario lee un determinado documento), la manera más frecuente de obtener esta información es de forma explícita a través del mismo sistema. En otras palabras, cada sistema de recomendación es el encargado de pedir a sus usuarios que expresen sus preferencias o que evalúen un determinado set de ítems antes de poder realizar algún tipo de recomendación.

No obstante, progresivamente los sistemas de recomendación han incorporado nuevas características con el fin de mejorar la calidad de las recomendaciones, donde la precisión de éstas no es el único atributo a considerar. Dado que existen diversos problemas que deben enfrentar los sistemas de recomendación [1, 3, 4], considerar aspectos como el contexto o características adicionales del usuario abre la puerta al uso de nuevas herramientas en pos de solucionarlos. Es por esto que las acciones realizadas y la información disponible en redes sociales como *Facebook*, *Twitter*, *Foursquare*, etc. son potenciales fuentes de información sobre las preferencias del usuario.

Actualmente, las redes sociales son utilizadas por algunos sistemas de recomendación (y muchos otros sistemas), pero casi únicamente como método de autenticación, por lo que el aprovechamiento de la información de usuario a la que ofrecen acceso (tanto preferencias como vínculos sociales) es aún incipiente. Adicionalmente, aunque las redes sociales facilitan medios de acceso a la información de sus usuarios, no se identifican marcos de trabajo que promuevan su explotación para complementar otros sistemas, por lo que cualquier aproximación queda exclusivamente en manos de un eventual desarrollador.

En capítulos posteriores de esta Memoria de Título, se explicará en forma más detallada el funcionamiento de los distintos métodos utilizados por los sistemas de recomendación, lo cual permitirá comprender de mejor manera la gran cantidad de opciones y mejoras que ofrece esta nueva fuente de información.

[1] En adelante se denominará como ítem a cualquier elemento que forme parte de un dominio (como películas, reportajes, artículos de oficina, páginas web, libros, documentos, etc.).

1.2. Descripción de la Propuesta

Tal como existe una gran variedad de dominios en los que un sistema de recomendación puede desenvolverse, también existe una gran variedad de algoritmos que permiten implementarlos, cada uno con sus respectivos pros y contras. Típicamente existen dos grandes tipos de algoritmos, los llamados Métodos Basados en Contenido (CB, por sus siglas en inglés, Content-Based) [1, 3, 5] y Métodos de Filtrado Colaborativo (CF, por sus siglas en inglés, Collaborative Filtering) [1, 3, 6, 7]. En términos generales, los métodos CB miden la similitud del contenido de un ítem que no ha sido evaluado por el usuario objetivo[2], con el de los demás ítems que sí ha evaluado. Esta similitud se obtiene generalmente comparando las preferencias personales del usuario con los atributos asociados al respectivo ítem. Por otro lado, los métodos CF buscan usuarios que posean gustos similares al usuario objetivo comparando sus evaluaciones anteriores con las evaluaciones hechas por los demás usuarios.

Aunque estos algoritmos son la base de la mayoría de los métodos utilizados por los sistemas de recomendación, constantemente aparecen nuevos enfoques y paradigmas que buscan mejorar su desempeño. Es así como se han desarrollado sistemas de recomendación basados en confianza (*Trust-Aware Recommender System*) [3, 8], en reputación [9] o en conocimiento (*Knowledge-based*) [3]; aplicaciones particulares de CF como las basadas en usuario (*User-based*) [8], en ítems (*Item-based*) [8], de Ítem a Ítem (*Item-to-Item*) [10], organizado por modelos [8] o basados en redes sociales (*Social Network-based*) [1], entre muchos otros.

En esta Memoria de Título se propone identificar las distintas formas en que se puede enriquecer o hacer posible el funcionamiento de los algoritmos de recomendación, anteriormente mencionados, usando información disponible en redes sociales. Para esto, es necesario conocer tanto el tipo y cantidad de información que ofrecen las distintas redes sociales, como los requisitos básicos de los métodos de recomendación para su implementación. En consecuencia, se desarrolló un estudio tanto de los algoritmos de recomendación más representativos, como de algunas APIs[3] o bibliotecas de recomendación disponibles en la actualidad, con el fin de identificar el tipo de información que se requiere para su funcionamiento. Adicionalmente, se llevó a cabo un estudio de las APIs distribuidas por algunas de las redes sociales más utilizadas a nivel nacional para identificar a qué tipo de información ofrecen acceso.

Los resultados de los estudios mencionados abrieron las puertas para proponer modificaciones a algunos algoritmos de recomendación existentes o posibles soluciones a problemas típicamente asociados a otros. Por ejemplo, en los sistemas basados en CF se debe lidiar constantemente con el problema del *cold-start* [1, 3, 4] o partida en frío, también conocido como problema del usuario nuevo, el cual ocurre cuando un usuario ingresa por primera vez a un sistema, por lo que no se cuenta con información sobre él y, por con consiguiente, no se le puede realizar una recomendación. En este caso, contar con la información relativa a las preferencias del usuario, que ya ha sido expresada en alguna red social, podría permitir que el sistema de recomendación genere automáticamente las primeras recomendaciones.

Una vez identificadas las características de las redes sociales y de los algoritmos/APIs de recomendación, se desarrolló un modelado de sistemas de recomendación en diagramas

[2]Usuario al cual se busca entregar una recomendación.
[3]Application Programming Interface (en español Interfaz de Programación de Aplicaciones).

UML[4], con lo que se busca integrar las características identificadas en las redes sociales, lo cual, entre otros beneficios, permitiría:

- **Generación automática de código:** El uso de modelos facilita el traspaso de modelo a código en forma automática, liberando a los desarrolladores de trabajo que, de otra forma, tendría que hacerse en cada aplicación.
- **Facilitar la comprensión:** Un modelo que describe el funcionamiento del sistema facilita su entendimiento por parte de todos los miembros del equipo de desarrollo.
- **Abstracción de la red social:** Un modelo que se adapte a la información de cualquier red social permitiría que los desarrolladores destinen la mayor parte de sus esfuerzos a los objetivos principales de su proyecto (por ejemplo, a la experimentación) en vez de al almacenamiento o administración de la información.

El modelo resultante fue implementado a través de un prototipo de sistema de recomendación que usa la información identificada y fue comparado con las capacidades de otras aproximaciones de desarrollo (por ejemplo, bibliotecas de algoritmos de recomendación).

1.3. Objetivos

1.3.1. Objetivo general

Formular una propuesta de modelado conceptual para sistemas de recomendación que considere la información de usuario disponible en redes sociales.

1.3.2. Objetivos específicos

- Plantear opciones de enriquecimiento para los sistemas de recomendación a partir de la información de usuario disponible en las redes sociales.
- Proponer modelos de sistemas de recomendación que integren las características identificadas en las redes sociales, utilizando diagramas de clases de UML.
- Desarrollar un prototipo de sistema de recomendación, basado en los modelos desarrollados.
- Evaluar la factibilidad del modelo propuesto y comparar sus capacidades con las de otras herramientas de desarrollo.

1.4. Metodología de trabajo

Para alcanzar los objetivos presentados en esta memoria, fue necesario llevar a cabo una serie de investigaciones que tienen aplicación directa en el modelado e implementación señalados. Es por esto que el informe se desarrolló en forma simultánea al proceso de investigación y desarrollo, ya que de esta manera fue posible registrar detalladamente tanto los conceptos teóricos necesarios para la comprensión del tema, como los procesos ligados a la obtención y manejo de la información.

[4]Unified Modeling Lenguage (en español, Lenguaje Unificado de Modelado). Lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad.

1.5. Estructura del informe

1.5.1. Capítulo 2: Marco Teórico

En este capítulo se describen los conceptos teóricos básicos y se identifican las principales características presentes en los métodos de recomendación más utilizados actualmente en la industria. Adicionalmente, se describen las principales categorías y componentes presentes en las redes sociales.

1.5.2. Capítulo 3: Trabajos Relacionados

En este capítulo se describen algunas aproximaciones que han demostrado que hacer uso de la información de usuario disponible en las redes sociales permite mejorar la calidad de los sistemas de recomendación.

1.5.3. Capítulo 4: Descripción de la solución propuesta

En este capítulo se explican todos los pasos efectuados y se justifican todas las decisiones tomadas para llevar a cabo esta Memoria de Título, desde el estudio de las redes sociales hasta la experimentación con la implementación del modelo propuesto.

1.5.4. Capítulo 5: Análisis de Redes Sociales

Durante este capítulo se presentan las principales herramientas utilizadas por las redes sociales *Facebook*, *Twitter*, *Linkedin*, *Intagram* y *Foursquare*. Adicionalmente, se detalla qué tipo de información, relevante para ser usada por sistemas de recomendación, es posible obtener a través de las APIs que ofrecen estas redes sociales. Finalmente, se proponen distintas formas en las que se podrían enriquecer los sistemas de recomendación si integraran la información suministrada por los usuarios en las distintas redes sociales.

1.5.5. Capítulo 6: Definición e implementación del modelo

En este capítulo se muestra el modelo de sistema de recomendación desarrollado a partir de la información disponible en las redes sociales y se describen sus componentes más importantes. Adicionalmente, se detalla cómo llevar a cabo su implementación según las distintas herramientas disponibles.

1.5.6. Capítulo 7: Experimentos y aplicación

En este capítulo se explican los experimentos realizados para evaluar los resultados obtenidos al implementar sistemas de recomendación. Adicionalmente, se describe la información de usuario utilizada en estos experimentos y cómo es posible obtenerla de las redes sociales.

1.5.7. Capítulo 8: Conclusiones

En este capítulo se resume el todo el trabajo realizado y se explica cómo en base a este se consiguieron los objetivos presentados en la sección 1.3. Adicionalmente, se presentan las limitaciones identificadas en el trabajo propuesto y los posibles trabajos futuros.

1.5.8. Capítulo 9: Anexos

En este capítulo se profundiza en la documentación disponible sobre diversos tipos de información y procedimientos mencionados en capítulos anteriores, pero que por razones de continuidad y brevedad no fueron tan extensamente detallados. Aquí se puede encontrar información relacionada a los datos obtenidos de las redes sociales y a las herramientas, procedimientos y códigos utilizados durante la implementación de los sistemas de recomendación.

Capítulo 2

Marco Teórico

En este capítulo se describen los conceptos teóricos básicos y se identifican las principales características presentes en los métodos de recomendación más utilizados actualmente en la industria. Adicionalmente, se describen las principales categorías y componentes presentes en las redes sociales.

2.1. Sistemas de Recomendación

2.1.1. ¿Qué son los sistemas de recomendación?

Son sistemas desarrollados para aliviar la sobrecarga de información que afecta a los usuarios a la hora de buscar contenido a través de la web, lo que resulta de gran utilidad para sitios web dedicados al comercio electrónico (como Amazon.com, Netflix.com, etc.), ya que les permite ofrecer sus productos y/o servicios de forma personalizada y automática para cada uno de sus clientes.

Aunque existe una diversa variedad de métodos utilizados para implementar sistemas de recomendación, todos ellos comparten la necesidad de poseer algún tipo de información acerca de los usuarios. Pese a que la mayoría se puede clasificar como Método de Filtrado Colaborativo (CF) [1, 3, 6, 7] o Método Basado en Contenido (CB) [1, 3, 5], constantemente se desarrollan nuevas aproximaciones que mezclan las características de estos o que utilizan nuevas fuentes de información en pos de generar mejores sistemas de recomendación.

En las siguientes subsecciones de este capítulo se profundiza en el funcionamiento de los métodos de recomendación más relevantes, haciendo énfasis en aquellos que potencialmente pueden hacer uso de la información de las redes sociales, con el objetivo de identificar cuáles son los requerimientos necesarios para la utilización de cada uno de ellos.

2.1.2. Métodos de Filtrado Colaborativo

La idea básica del CF es que si dos usuarios mostraron intereses similares en el pasado, estos tendrán gustos similares en el futuro [1, 3]. Por ello, las recomendaciones recibidas por el usuario objetivo no sólo se basan en sus preferencias personales, sino también en las preferencias de otros usuarios.

La forma en que se representan estas preferencias puede variar de un método a otro, pero principalmente se identifican dos formas:

- **En forma implícita:** Las preferencias del usuario se obtienen haciendo un seguimiento de su comportamiento y típicamente son recolectadas por el sitio web o aplicación donde está incrustado el sistema de recomendación. De esta forma, dentro de los sitios web es posible encontrar distintas fuentes implícitas de información. Por ejemplo, medir el tiempo que un usuario dedica a leer un determinado documento [2], registrar los ítems a los que se ha accedido (como en el caso de *Youtube*), la frecuencia con que se accede a un determinado tipo de información o la compra en línea de un determinado ítem pueden interpretarse como una evaluación positiva.

  Aunque el principal beneficio de estas evaluaciones implícitas es que no requieren esfuerzos adicionales por parte del usuario (ya que pueden recolectar datos continuamente y en forma automática), no se puede estar seguro que la interpretación de un determinado comportamiento sea la correcta.

- **En forma explícita:** Las preferencias del usuario son directamente declaradas por él mismo (por lo que se ha propuesto [3] que probablemente es la forma más precisa de

obtener su opinión). Para esto, los sistemas cuentan con herramientas que permiten asociar algún tipo de evaluación a los respectivos ítems y, aunque existen diversas formas de evaluar un ítem, hay tres que son las más utilizadas:

- Unaria: El usuario puede evaluar los ítems solo positivamente. Actualmente *Facebook* utiliza este tipo de evaluación, a través de su implementación de me gusta.
- Binaria: El usuario puede evaluar los ítems tanto positiva como negativamente. Particularmente utilizado por *Youtube*, con su implementación de me gusta/no me gusta.
- Escalar: El usuario puede asignar una calificación numérica a cada ítem, determinando así su nivel de agrado o desagrado. Usualmente se establece un intervalo de valores, donde el valor menor representa el máximo nivel de desagrado y el valor mayor representa el máximo nivel de agrado por un determinado ítem. Cabe destacar que la implementación de este tipo de evaluación puede ser hecha para distintos intervalos de valores y las calificaciones asignadas por los usuarios pueden tomar valores discretos o continuos. Algunas implementaciones observables están en *Amazon* (intervalo de 1 a 5) y en *IMDB* (intervalo de 1 a 10).

Ofreciendo un sistema de evaluación se busca cumplir el requisito principal para que un sistema de recomendación basado en CF pueda funcionar: cada usuario debe estar asociado a una lista de ítems sobre los cuales ha expresado, explícita o implícitamente, sus preferencias.

Una de las principales ventajas del CF tradicional es que evita la costosa tarea de proveer detalles y actualizaciones a la descripción de los ítems manejados por el sistema, ya que son vistos como símbolos sin estructura interna [8]. Sin embargo, la mayoría de los métodos basados en CF presentan problemas relacionados con escalabilidad y *cold-start*. Por un lado, el *cold-start* (o partida en frío) ocurre en dos situaciones: cuando se ingresa al catálogo un nuevo ítem sin evaluaciones asociadas y cuando un usuario ingresa por primera vez a la aplicación y éste no ha realizado las suficientes evaluaciones para que el sistema de recomendación pueda identificar una similitud con otros usuarios. Por otro lado, presenta problemas de escalabilidad debido que a medida que aumenta el número de ítems en el catálogo y crece la cantidad de usuarios, aumentan los cálculos necesarios para encontrar usuarios con preferencias similares. Sin embargo, cada una de las aproximaciones basadas en CF intenta integrar nuevas técnicas o fuentes de información con el fin de aliviar o solucionar los problemas mencionados.

En general, los métodos basados en Filtrado Colaborativo se pueden dividir en dos categorías principales [11]: los basados en modelos (*model-based* CF) y los basados en memoria (*memory-based* CF). Los métodos basados en modelos utilizan todos los usuarios e ítems registrados en el sistema para crear un modelo que permita hacer predicciones. En cambio, los métodos basados en memoria se preocupan de predecir la calificación que el usuario objetivo le daría a un ítem en base a las calificaciones anteriores hechas por él y el resto de los usuarios. En consecuencia, los métodos basados en modelos reportan mayor precisión en las recomendaciones que generan, sin embargo son demasiado costosos computacionalmente. En lo que sigue de esta Memoria de Título se le dará un mayor énfasis a los métodos basados en memoria (en lo que se refiere a CF), dado que las aproximaciones más relevantes están basadas en ellos y ofrecen una mayor variedad de herramientas para los distintos escenarios.

Filtrado Colaborativo basado en Usuarios

El llamado *User-based CF method*, asume que: las preferencias del usuario permanecen consistentes en el tiempo y, si usuarios tuvieron gustos similares en el pasado, también tendrán gustos similares en el futuro [3]. Este método trabaja buscando un conjunto de usuarios que posean gustos o preferencias similares al usuario objetivo, para así, ofrecer a este último recomendaciones basadas en las preferencias históricas de los otros usuarios. En general su funcionamiento es el siguiente:

1. Identifica el conjunto de calificaciones del usuario objetivo.
2. Compara las calificaciones de (1) con las de los demás usuarios e identifica los usuarios más similares al usuario objetivo (vecinos).
3. Identifica los ítems que más le gustan a los vecinos y que el usuario objetivo no ha calificado.
4. Genera una predicción, predice la nota que le pondría el usuario objetivo a cada uno de los ítems identificados en (3).
5. Recomienda una lista con los n-mejores ítems identificados en (4).

Aunque se han propuesto distintas formas de calcular la similitud entre usuarios (*Adjusted cosine similarity*, *Spearmans Rank correlation coefficient*, etc.), estudios empíricos [12] han demostrado que para el caso de sistemas de recomendación basados en usuario el Coeficiente de Correlación de Pearson obtiene mejores resultados que los otros métodos. Sin embargo, se debe tener la precaución de elegir la cantidad adecuada de usuarios que conforman la vecindad del usuario objetivo, ya que si se eligen demasiados usuarios solo aportarán ruido, mientras que si la vecindad es muy pequeña (por ejemplo, menos de 10 usuarios), se puede ver afectada la calidad de la recomendación.

Filtrado Colaborativo basado en Ítems

Conocido como *Item-based CF method*, tiene como idea principal predecir las calificaciones que haría el usuario objetivo usando la similitud entre ítems en vez de la similitud entre usuarios. Los métodos basados en ítems se desarrollaron para solucionar el principal problema que presentan los métodos tradicionales de CF: su desempeño decrece considerablemente al enfrentarse a volúmenes demasiado grandes de ítems y usuarios (millones/millones). A grandes rasgos, los métodos basados en ítems funcionan de la siguiente manera:

1. Identifica los ítems que el usuario objetivo ha evaluado anteriormente.
2. Calcula qué tan similares son (1) con los demás ítems del catálogo, comparando las calificaciones asignadas por otros usuarios.
3. Selecciona los k-ítems más similares obtenidos en (2).
4. Predice la nota que el usuario objetivo daría a (3) en base al valor de la calificaciones de (1).

Al igual que en el caso de los métodos basados en usuarios, se debe definir una medida de similitud entre los ítems a partir de la correlación entre sus calificaciones. En el caso de las recomendaciones basadas en ítems la Similitud Ajustada de Coseno ha probado entregar los mejores resultados [3].

Por otro lado, para solucionar el problema de escalabilidad se recurre a pre calcular los datos en forma *offline*. La idea detrás de esto es construir con anterioridad una matriz ítem/ítem que describa la similitud entre todos los pares de ítems, para que luego, al ejecutarse el sistema de recomendación, se haga la predicción de calificaciones simplemente seleccionando los ítems más similares a los ítems calificados anteriormente por el usuario objetivo. De esta forma, los cálculos computacionalmente más costosos se llevan a cabo fuera de la percepción del usuario y la predicción es fácilmente calculada en un tiempo adecuado para aplicaciones *online*.

Cabe mencionar que la pre computación *offline* también es posible para los métodos basados en usuarios, pero en el caso de métodos basados en ítems, los resultados son más estables y la precisión de la recomendación no se ve afectada.

A modo de ejemplo, se presenta una aplicación particular de este método, llamado Filtrado Colaborativo Ítem a Ítem o *Item-to-Item Collaborative Filtering* [10]. Este método fue desarrollado por la tienda de comercio electrónico *Amazon* debido a que no existían algoritmos capaces de escalar a datos masivos de varios millones de clientes/productos y que, además, generaran recomendaciones de calidad en tiempo real. Se caracteriza principalmente por considerar que la compra de un determinado ítem representa una evaluación positiva para este (adicionalmente a las calificaciones que tenga asociadas) y por utilizar todas las evaluaciones hechas por la comunidad para elaborar una matriz ítem/ítem que registra la relación que existe entre todos los ítems que los usuarios tienden a comprar juntos. A grandes rasgos, la principal diferencia del método *Item-to-Item* con el CF tradicional, es que presenta un buen desempeño incluso con información limitada, pudiendo generar recomendaciones de alta calidad a usuarios que hayan evaluado hasta dos o tres ítems.

Sistema de recomendación basado en confianza

TARS (por sus siglas en inglés, *Trust-Aware Recommender System*), es un método *User-based* propuesto en [13] como un sistema de recomendación en el que un usuario, además de evaluar los ítems, puede asignar calificaciones a los demás usuarios para representar su grado de confianza hacia ellos. De esta forma, se genera una red de confianza entre todos los usuarios del sistema que permite mejorar la calidad de las recomendaciones.

La idea básica del método es: los usuarios confiarán más en recomendaciones que estén basadas en las preferencias de usuarios que consideran confiables. Para llevar esto a la práctica, al momento de calcular las similitudes se le asigna un peso a las opiniones de los usuarios, el cual es proporcional al grado de confianza que cada uno de ellos genere en el usuario objetivo. Por otro lado, TRAS considera que la confianza es transitiva, pero no simétrica. En consecuencia, este método es capaz de aliviar el *cold-start*, ya que aunque el usuario objetivo no haya realizado evaluaciones, solo requiere haya declarado su confianza hacia otro usuario para que las recomendaciones se hagan en base a las preferencias de este último y sus vecinos.

Adicionalmente, existen otras aproximaciones [9] que relacionan la confianza con los sistemas de recomendación. En este caso se propone una nueva forma de seleccionar las vecindades a través de la reputación de cada usuario, la cual es proporcional a la cantidad de veces que se ha recomendado exitosamente en base a su información.

Otras aproximaciones de Filtrado Colaborativo [3]

Hasta ahora sólo se han descrito los métodos de CF más importantes, pero existe una gran cantidad de otros métodos que los especializan, mejoran y/o adaptan. En lo que sigue de esta sección se describen tres métodos que vale la pena tener en consideración.

Métodos basados en pre-procesamiento. La información básica de un sistema de recomendación (usuarios, ítems y las evaluaciones que los relacionan) suele representarse directamente a través de una matriz usuarios/ítems, pero cuando estos sistemas son de gran tamaño se puede recurrir a una factorización de la matriz para mejorar la precisión de las recomendaciones, encontrando, a través de un cálculo *offline* previo, factores latentes que pueden caracterizar a los usuarios e ítems. En consecuencia, las recomendaciones se concretan cuando un ítem y un usuario son similares con respecto a estos factores.

Reglas de asociación. La idea principal es buscar patrones entre los usuarios y/o entre los ítems del catálogo para establecer reglas que las relacionen. De esta forma es posible identificar comportamientos como: dado que los ítems i_1, i_2 e i_3 generalmente se compran juntos, si el usuario objetivo está comprando los ítems i_2 e i_3, entonces es probable que también quiera comprar el ítem i_1. Estas reglas pueden variar su complejidad según la calidad de información con la que se cuente, por lo que es posible que incluso a partir de una sola preferencia se obtengan variadas posibilidades de recomendación.

Aunque este enfoque es muy sencillo de implementar y ejecutar en forma *online*, está asociado a grandes costos al momento de obtener y definir las reglas, ya que es necesario contar con un amplio historial del dominio de aplicación y analizar la totalidad de esos datos para poder calcular una lista de reglas capaces de entregar resultados precisos.

Recomendaciones probabilísticas. En este tipo de sistemas, el objetivo es predecir la calificación más probable que un usuario le dará a un ítem, en base a sus evaluaciones anteriores y las de los demás usuarios. Para esto, se visualiza el problema de CF como un problema de clasificación, ya que generalmente es necesario encasillar a los usuarios/ítems en una o varias categorías predefinidas. Aunque existe una variada cantidad de opciones para su implementación (como el uso de Redes Bayesianas), la principal desventaja de este enfoque es que es computacionalmente muy complejo para grandes volúmenes de datos y no presenta un buen desempeño en volúmenes de datos pequeños o dispersos.

2.1.3. Métodos Basados en Contenido

Conocidos también como *Search-based methods*, los métodos CB [1,3,5] tienen como idea básica el recomendar ítems similares a aquéllos que el usuario objetivo ha preferido anteriormente, por lo que se enfocan más en hacer uso de la información asociada a los ítems que a las opiniones o interacciones de otros usuarios.

Para poder hacer uso de la información mencionada y generar una recomendación, en primera instancia es necesario llevar a cabo un análisis del contenido de los ítems del dominio respectivo, ya que si su información no está estructurada (por ejemplo, en forma de texto), es necesario realizar un pre-procesamiento para poder obtener la información relevante (palabras clave, conceptos, etc.). Una vez que se identifica el contenido relevante de los ítems, se establecen un conjunto de categorías con el fin de poder clasificar los ítems en una o varias de ellas. Luego, se debe generar un perfil de usuario para cada miembro del sistema de recomendación, en el cual se registran cuáles son las categorías que prefiere. Finalmente, se predice si un determinado ítem será del interés del usuario objetivo comparando las categorías registradas en su perfil (obtenidas en base a las evaluaciones hechas anteriormente) con las categorías asociadas al ítem. De esta forma, es posible ofrecer en una lista un subconjunto de los ítems más afines al usuario.

Este método fue pensado originalmente para obtener características cualitativas de dominios subjetivos (como e-mails, sitios web, documentos, etc.) con el fin de poder filtrar o recomendar sus ítems. La forma más común de hacerlo es a través de aprendizaje de máquina (*machine learning*), ya que permite obtener automáticamente las categorías necesarias. Considerando estos antecedentes, es posible identificar varias ventajas en el uso de este método:

- Funciona en forma independiente para cada uno de sus usuarios, ya que sólo requiere conocer las evaluaciones hechas por el usuario objetivo para generar recomendaciones. Considerando que adicionalmente sólo requiere las características de los ítems para generar recomendaciones, es capaz de solucionar el *cold-start* asociado a la inclusión de un ítem nuevo (y que aun nadie ha evaluado).
- Ofrece la posibilidad de explicar más fácilmente al usuario objetivo por qué se le está recomendando un determinado ítem. A diferencia de los métodos de CF, en los cuales las recomendaciones se basan en las preferencias de usuarios desconocidos para el usuario objetivo, con el método CB es sencillo mostrar la relación entre las preferencias del usuario y los atributos del ítem que se le está recomendando, a través de sus características o descripciones. De esta forma, si un usuario ha declarado, por ejemplo, que le gustaban los géneros de películas “acción”, “ciencia ficción”, “terror” y “aventura”, al recomendarle la película Mad Max: Furia en el Camino (que está vinculada a los géneros “acción”, “ciencia ficción” y “aventura”) se le podrá mencionar que esa sugerencia se debe a que los géneros asociados a la película coinciden con sus preferencias personales.

Sin embargo, presenta una serie de desventajas que han contribuido a su limitado uso, asociadas principalmente a la complejidad de su implementación:

- Es necesario que el contenido asociado a los ítems pueda ser clasificado (puede ser complicado en ítems como ideas u opiniones) y se debe confiar en que esa información sea correcta, lo que muchas veces implica confiar en fuentes de información externas al sistema de recomendación.
- Es difícil evitar la sobre especialización. Uno de los objetivos principales de los sistemas de recomendación es sugerir ítems novedosos e interesantes para el usuario objetivo, sin embargo el método CB tradicional es incapaz de hacerlo si éste solo evaluó un determinado tipo de ítems. Por ejemplo, en un sistema de recomendación de películas,

si un usuario únicamente ha preferido películas de ciencia ficción, únicamente recibirá recomendaciones de ese tipo de películas. Mientras que este problema tiene mayor probabilidad de ocurrir cuando la cantidad de ítems evaluados por el usuario es demasiado pequeña, también se presentan complicaciones cuando la cantidad de ítems es muy grande, ya que es impracticable ejecutar consultas que consideren todas las preferencias involucradas. Para lidiar con este problema se debe usar un subconjunto de los ítems evaluados, lo que inevitablemente reduce la calidad de las recomendaciones.

- Sufre de un tipo de *cold-start* distinto al presentado para los métodos de CF. Mientras que en los métodos de CF es necesario contar con una lista de ítems evaluados para poder comparar las preferencias del usuario objetivo con las de los demás usuarios, en los métodos CB es necesario que los usuarios cuenten con un perfil que almacene sus preferencias. Es por esto que si un usuario es nuevo, usualmente se le solicita que señale un conjunto de categorías o palabras clave con el fin que el algoritmo entienda sus preferencias y entregue una recomendación precisa.

Pese a que presentan una gran cantidad de dificultades, los métodos CB ofrecen herramientas para construir sistemas de recomendación sobre dominios en los que otros métodos no son capaces de actuar. De todas formas es posible apoyar o mejorar su funcionamiento a partir de nuevos enfoques y/o fuentes de información, por ejemplo: desde la aparición de la Web 2.0 son los mismos usuarios los que cada vez más están dispuestos a complementar el contenido asociado a los ítems, lo cual aliviaría en gran medida la carga de un sistema de recomendación.

2.2. Redes Sociales

En los últimos años la masificación de las redes sociales ha demostrado que éstas son capaces de expandir los límites de aplicación para los que fueron concebidas. Además de permitir la comunicación casi en tiempo real con otras personas y ofrecer herramientas para la publicación de contenido, las redes sociales se han transformado en un nexo entre las personas y las empresas, en un medio para que las empresas se den a conocer, en una fuente de información, en una enciclopedia de experiencias y, principalmente, en un unificador de la sociedad en torno a determinados intereses. Por otro lado, han escalado hasta transformarse en una herramienta de apoyo y mejoramiento de otros servicios; por ejemplo, se han desarrollado variados métodos de autenticación, los cuales liberan a los sitios web que los implementan de toda tarea relativa al registro de nuevos usuarios. Siguiendo esta idea, luego de identificar el gran potencial existente en las redes sociales (debido a la enorme cantidad de información que almacenan), se desarrollaron sus respectivas APIs. De esta forma, cada red social ofrece herramientas que pueden ser libremente implementadas en aplicaciones particulares.

Dado de que cada red social es distinta a otra en mayor o menor medida, en la siguiente subsección se describirán los tipos de redes sociales disponibles hoy en día, con el fin de tener una noción del tipo de información que manejan y sus características en general.

2.2.1. Tipos de Redes Sociales

La principal razón de que las redes sociales hayan tenido un crecimiento y masificación tan grande es que pueden ser aplicadas sobre cualquier actividad que implique una interacción entre personas, lo cual se ha fomentado cada vez más desde la aparición de la Web 2.0.

Gracias a una extensa exploración y estudio de variadas redes sociales es posible notar que, aunque existe una gran cantidad de ellas con una serie de características distintivas propias, la mayoría hace uso del mismo conjunto de herramientas. En consecuencia, se propone la siguiente clasificación, la cual permite identificar los componentes más visibles de todas las redes sociales:

- **Perfil de usuario:** Todas las redes sociales permiten al usuario crear y administrar su propio perfil, incluyendo información e imágenes personales.
- **Blog y Microblog:** Son servicios que se enfocan en la publicación de contenido en forma de texto (generalmente información personal sobre algún tema en particular). Usualmente son implementados dentro de las redes sociales en forma de un muro personalizado, en el cual se despliega todo el contenido publicado por el usuario, pero adicionalmente existen configuraciones que también despliegan el contenido publicado por otros usuarios, ya sea basado en interés o en cercanía.
- **Vínculos sociales:** Se ofrece la posibilidad de declarar explícitamente vínculos sociales entre usuarios. De esta forma, se pueden establecer relaciones virtuales con familiares, amigos, compañeros de trabajo, etc. a través de listas de amigos, círculos sociales y seguidores, entre otros. Las relaciones entre dos usuarios pueden declararse con el consentimiento de uno de ellos o de ambos, lo que depende principalmente de las características de seguridad particulares de la red social.
- **Compartir contenido:** Se permite a los usuarios subir contenido a la web, ya sea como imagen, video u otro medio, de tal forma que sea visible por otros usuarios a través de su muro u otra característica.
- **Foros:** Ofrece espacios de discusión donde los usuarios son libres de intercambiar opiniones en forma pública a través de mensajes. La principal diferencia con los blogs o microblogs es que es posible intercambiar opiniones con usuarios con los cuales se ha establecido ningún tipo de relación.
- **Marcadores y novedades sociales:** Los marcadores sociales permiten guardar y organizar enlaces a diversos recursos o sitios web, de manera que éstos sean más fáciles de buscar y/o compartir. Por otro lado, las novedades sociales permiten publicar ítems o enlaces por los cuales se puede votar de manera pública o restringida, de tal forma que la comunidad es la que decide qué ítems serán los más accesibles para los demás usuarios.
- **Etiquetas o Tags:** Son ampliamente utilizados al momento de publicar contenido, de manera que se establecen explícitamente palabras claves o conceptos que describen el contenido compartido. Los tags son especialmente útiles para agrupar contenido (ya que el trabajo de clasificación es realizado por los mismos usuarios) y facilitar búsquedas.

Adicionalmente, existe una infinidad de otros complementos disponibles, los cuales permiten personalizar en mayor o menor medida la navegación de cada usuario. Es así como

algunas redes sociales permiten el envío de mensajes privados entre los usuarios, declarar explícitamente sus gustos sobre ítems u otros usuarios, etc.

Por otro lado, aunque haya una enorme variedad de redes sociales enfocadas en distintos ámbitos, este memorista también propone que es posible clasificarlas en base al propósito para el cual se desarrollaron. De esta forma podemos encontrar:

- **Redes sociales personales:** Son aquellas redes sociales enfocadas en mantener y generar vínculos sociales a través del intercambio de opiniones y contenidos personales. Algunas redes sociales presentes en esta categoría son Facebook, Foursquare y Google+.
- **Redes sociales de intercambio de contenido:** Están destinadas a facilitar el intercambio de contenido, ya sea recreacional, académico o profesional. Se presentan como una de las mejores alternativas si lo que se busca es dar a conocer de forma pública algún tipo de contenido, por lo que usualmente las restricciones de acceso son mínimas. Algunas redes sociales presentes en esta categoría son Twitter, Instagram, Youtube y Pinterest.
- **Redes sociales de intereses comunes:** A diferencia de las otras redes sociales que permiten a los usuarios establecer los temas de interés, este tipo de redes sociales se establece sobre un tema determinado y son los usuarios los que acceden a la red para relacionarse en torno a éste. De esta forma, es posible encontrar redes sociales donde solo se tratan temas relacionados con el medio ambiente, enfocadas al intercambio de experiencias profesionales o de acceso restringido únicamente de los miembros de una empresa. Algunas redes sociales presentes en esta categoría son Linkedin y Zyncro.

2.3. Conclusión

A lo largo del capítulo 2 se describieron los principales métodos de recomendación sobre los cuales se basan la mayoría de las demás aproximaciones. Con esto se busca establecer las bases teóricas para los análisis y conclusiones propuestas en los capítulos posteriores de esta Memoria de Título. Adicionalmente se presentó, a grandes rasgos, las principales características y herramientas identificables en las redes sociales como una forma de mostrar que la información manejada por cada una de ellas depende directamente tanto de su propósito como de las herramientas que utiliza. En consecuencia, se propone una relación entre las redes sociales y los sistemas de recomendación, donde el foco de interés se centra en hacer uso de la información administrada por las redes sociales dentro de los sistemas de recomendación con el fin de mejorar el rendimiento de estos últimos.

Capítulo 3

Trabajos Relacionados

En este capítulo se describen algunas aproximaciones que han demostrado que hacer uso de la información de usuario disponible en las redes sociales permite mejorar la calidad de los sistemas de recomendación.

Los trabajos descritos a continuación se encontraron principalmente en documentos presentes en las plataformas web *IEEE Xplore Digital Library* y *Google Scholar*, dentro de las cuales las búsquedas más importantes incluyen conceptos relacionados con: *Recommender Systems*, *Social Network*, *User Similarities* y *Collaborative Filtering*.

3.1. Redes sociales y métodos de recomendación

Gracias a la publicación voluntaria de preferencias personales y vínculos sociales por parte de los usuarios de redes sociales, es que éstas han llamado la atención de los sistemas de recomendación, los cuales constantemente buscan nuevas fuentes de información que faciliten las recomendaciones o mejoren su calidad. En lo que resta de este capítulo se describen algunas aproximaciones que han hecho uso de nuevos enfoques para obtener información relevante o que han desarrollado nuevos métodos para complementar la información disponible y generar mejores recomendaciones.

3.1.1. Sistema de Recomendación basado en Redes Sociales

La propuesta ofrecida por el método de *user-based CF* Sistema de Recomendación basado en Redes Sociales (SNRS, por sus siglas en inglés, *Social Network-based Recommender System* [1]) toma como base el siguiente escenario (ilustrado en la Figura 3.1): Ya sea para buscar restaurantes, elegir una película, comprar una casa o buscar trabajo, todas las decisiones de la vida diaria están influenciadas por tres factores: nuestras preferencias personales, la opinión del resto de la comunidad y la opinión de nuestros amigos. Si alguno de estos tres factores reporta una evaluación negativa sobre un determinado ítem, es probable que nuestro interés por ese ítem disminuya.

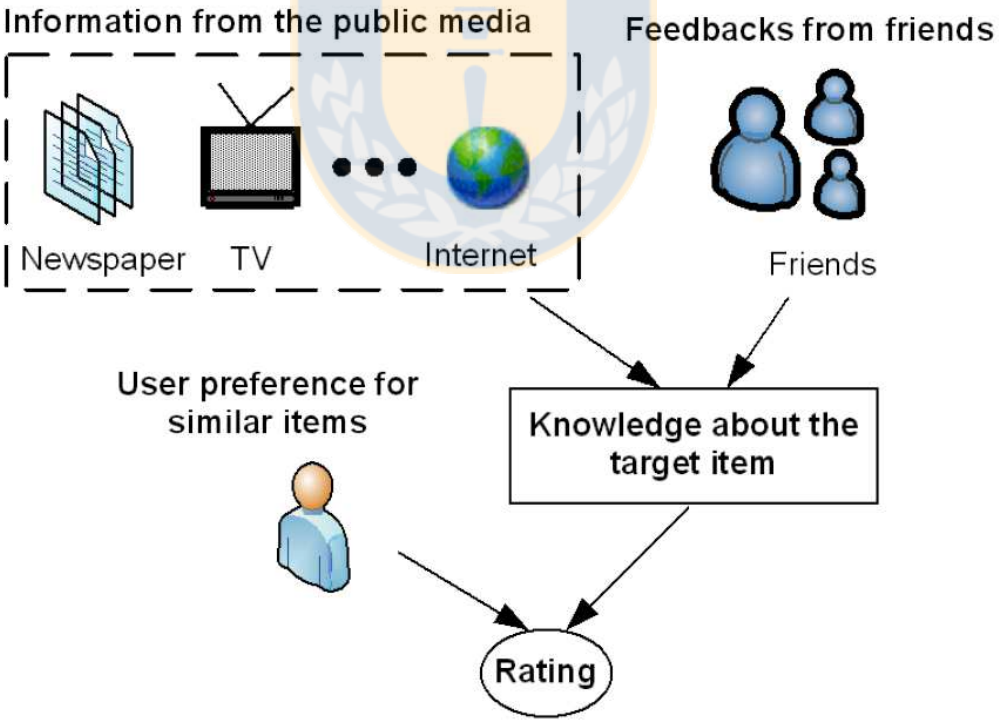

Figura 3.1: Los tres factores que influencian las decisiones de un usuario según SNRS [1].

De esta forma, para predecir la nota que daría el usuario objetivo a un determinado ítem, SNRS utiliza las preferencias expresadas anteriormente por el mismo usuario, las calificaciones declaradas por otros usuarios similares al usuario objetivo y las calificaciones declaradas por un conjunto de sus amigos inmediatos o *immediate friends* (usuarios con los que el usuario objetivo tiene alguna relación social directa). Sin embargo, puede darse el caso que los

amigos inmediatos de un usuario no hayan evaluado los mismos ítems que él, por lo que no se podría establecer una relación entre sus preferencias y considerar su influencia sólo perjudicaría la calidad de la recomendación. Por lo tanto, para manejar estas eventualidades SNRS sustituye la influencia de los amigos inmediatos por la de amigos distantes o *distant friends* (usuarios que no tienen alguna relación social directa con el usuario objetivo, pero sí con alguno de sus amigos).

Una de las principales ventajas que presenta este método es que no sufre de *cold-start*, ya que para los usuarios nuevos que aún no han declarado sus preferencias puede generar recomendaciones a través de las preferencias de sus amistades (basta con que un usuario lo haya invitado a la aplicación). Sin embargo, es necesario tomar cierto tipo de precauciones que mejoran considerablemente el desempeño del método. Para que la influencia social sea realmente efectiva, los grupos de amigos deben seleccionarse adecuadamente según los tipos de ítems que se estén manejando (por ejemplo, dos amigos que comparten gustos musicales pueden opinar completamente diferente con respecto a tipos de comida), por lo que se recomienda aplicar un filtro semántico al momento de definir el grupo de amigos inmediatos/distantes.

3.1.2. Filtrado Colaborativo entre personas

El Filtrado Colaborativo entre personas (o *CF for People to people* [11]) es un método propuesto específicamente para ser usado en redes sociales. A diferencia del CF tradicional, donde solo se toman en cuenta los gustos de los usuarios y los ítems como tales pasan a segundo plano, en el CF entre personas se considera que los usuarios también son ítems a ser recomendados y que además participan activamente en interacciones sociales, las cuales generalmente deben ser definidas en función del dominio en que se utilicen.

Con el fin de entregar un mayor entendimiento de este método, a continuación se describe desde un alto nivel de abstracción el algoritmo SocialCollab [11] y su aplicación particular sobre una red social dedicada a reunir gente y programar citas. Este algoritmo está basado en la similitud entre usuarios y su objetivo es predecir cuáles usuarios pueden ser del interés del usuario objetivo. Para esto establece relaciones entre usuarios a través de interacciones, las cuales pueden ser exitosas o no: una interacción es exitosa si un usuario u_1 contacta a un usuario u_2 y este último responde en forma positiva, mientras que una interacción no es exitosa si el usuario u_2 responde de forma negativa. Adicionalmente, para determinar la similitud entre usuarios hace uso de los conceptos de gusto y atractivo, con lo que se considera que dos usuarios pueden tener gustos similares (si anteriormente han preferido a un grupo en común de usuarios) o ser similares en atractivo (si hay un conjunto de usuarios que los ha preferido a ambos). En consecuencia, para modelar el comportamiento dentro de la red social asume:

- Si a personas con gustos similares al usuario A le gusta el usuario B, entonces A se interesará por B.
- Si un usuario A le gusta la gente con atractivo similar al usuario B, entonces a A le interesará B.

Finalmente, SocialCollab genera una recomendación bidireccional (predice que una interacción será exitosa) en los casos donde el atractivo de un usuario u_1 coincide con los gustos

de un usuario u_2 y los gustos de u_1 coinciden con el atractivo de u_2.

3.1.3. Filtrado basado en contenido en redes sociales

Aunque en la mayoría de los casos lo que se busca es complementar los sistemas de recomendación con la información en las redes sociales, también es posible complementar las redes sociales utilizando métodos de recomendación. Es así como, considerando que la mayor parte de la información presente en las redes sociales está en forma de texto y que ésta principalmente se encuentra disponible en los muros de los usuarios, se desarrolló el sistema *CB Filtered Wall* [14] con el objetivo de filtrar automáticamente los mensajes no deseados por el usuario objetivo. Lo anterior se origina en que algunas redes sociales permiten restringir el acceso al muro a un grupo determinado de usuarios, pero no ofrecen apoyo cuando se trata de restringir la publicación de cierto tipo de contenido.

El sistema *Filtered Wall* permite a los usuarios definir un conjunto de reglas de filtrado según el tipo de mensajes que se deseen omitir. Para esto, el sistema comienza por consultar un componente llamado *Short Text Classifer* (STC), el cual clasifica los mensajes de acuerdo a un set de categorías obtenido a través de aprendizaje de máquina. Luego, recurre a un componente de filtrado llamado *Content Based Messages Filtering* (CBMF) que decide cuáles mensajes se mostrarán en el muro del usuario objetivo a partir de las reglas declaradas por el usuario y los resultados obtenidos por el STC.

3.1.4. Recomendaciones basadas en el contenido de etiquetas sociales

Existen sistemas en los cuales los usuarios asignan libre y voluntariamente etiquetas a los distintos ítems. Estas etiquetas se utilizan principalmente para agrupar contenido, además de facilitar la búsqueda de ítems de interés para la comunidad. En general, los usuarios suelen asignar etiquetas a ítems que de alguna forma son relevantes para ellos, por lo que en algunos estudios [15] se ha considerado que éstos describen sus gustos, intereses y necesidades. En base a lo anterior, se ha considerado que entre más utilice un usuario una determinada etiqueta, más representativa será de su perfil. En forma análoga, se considera que, como las etiquetas asociadas a un ítem describen su contenido, entre más usuarios asignen una determinada etiqueta a un ítem, mejor describirá su contenido.

De esta forma, a través de las etiquetas asignadas en el sistema es posible obtener las preferencias de cada usuario y, de manera automática, las categorías en las que es posible clasificar los ítems. En consecuencia, es posible aplicar métodos CB tradicionales simplemente llevando a cabo un pre-procesamiento de la información.

3.2. Conclusión

Como se ha mencionado anteriormente, la calidad de los sistemas de recomendación no pasa únicamente por la precisión con la que se entregan las recomendaciones, también existe una gran variedad de aspectos a considerar, tales como la usabilidad del sistema y el aprovechamiento de la información para generar recomendaciones. Es por esto que durante este capítulo se presentaron algunos de los métodos de recomendación más novedosos en el uso de la información disponible en las redes sociales donde, por ejemplo, es posible obtener recomendaciones a partir de la interacción del usuario con el sistema o aumentar la cantidad de ítems recomendados a un usuario a partir de las relaciones sociales declaradas por él y otros usuarios.

Para el desarrollo de esta Memoria de Título se busca identificar nuevas fuentes de información provenientes de distintos tipos de redes sociales, de manera que se simplifique la implementación de los sistemas de recomendación, aliviando en mayor o menor medida algunos de los problemas típicos identificados en los métodos anteriormente vistos (como es el caso del *cold-start*). De esta forma, se propone hacer uso paradigmas similares al utilizado en el caso del *Social Network-based Recommender System*, donde toda la información necesaria para las recomendaciones es obtenida del perfil del usuario.

Capítulo 4

Descripción de la solución propuesta

En este capítulo se explican todos los pasos efectuados y se justifican todas las decisiones tomadas para llevar a cabo esta Memoria de Título, desde el estudio de las redes sociales hasta la experimentación con la implementación del modelo propuesto. Se debe señalar que en muchos puntos la descripción se realiza en forma general, para llevar a cabo una descripción más detallada en capítulos posteriores.

4.1. Descripción General

Como se presentó en el capítulo anterior, se han desarrollado variados métodos de recomendación [1, 11, 14, 15], todos enfocados en mejorar la calidad de los sistemas de recomendación utilizando la información que los usuarios ofrecen en las redes sociales. Sin embargo, ninguno de estos métodos hace uso de modelos para la implementación de sus experimentos. Por otro lado, aunque se han desarrollado modelos que describen sistemas de recomendación [16], estos no consideran el uso de información de redes sociales. En consecuencia, en cada aplicación que busque explotar el potencial de las redes sociales serán los desarrolladores los encargados de levantar los requerimientos, destinando tiempo a tareas que generalmente no son el foco del proyecto.

El objetivo central de esta Memoria de Título es proponer un modelo de sistemas de recomendación que integre la información disponible en las redes sociales, lo cual, entre otros beneficios, permitiría:

- **Generación automática de código:** El uso de modelos facilita el traspaso de modelo a código en forma automática, liberando a los desarrolladores de trabajo que, de otra forma, tendría que hacerse en cada aplicación.
- **Facilitar la comprensión:** Un modelo que describe el funcionamiento del sistema facilita su entendimiento por parte de todos los miembros del equipo de desarrollo.
- **Abstracción de la red social:** Un modelo que se adapte a la información de cualquier red social permitiría que los desarrolladores destinen la mayor parte de sus esfuerzos a los objetivos principales de su proyecto (por ejemplo, a la experimentación) en vez de al almacenamiento o administración de la información.

En lo que sigue de este capítulo se detalla en profundidad los pasos necesarios para conseguir el objetivo mencionado anteriormente.

4.2. Redes Sociales

El primer paso para desarrollar un modelo para sistemas de sistema de recomendación es identificar cómo funcionan. En este punto no es necesario comenzar desde cero, ya que se cuenta con el modelo abstracto para sistemas de recomendación propuesto en [16]. Una vez que se tiene la base del modelo, se debe identificar la información de las redes sociales que será administrada por el sistema. De esta forma, el siguiente paso es el estudio y análisis de la información a la que las redes sociales ofrecen acceso.

Considerando que uno de los principales beneficios que se buscan del modelo mencionado es que se pueda abstraer de la red social que se elija utilizar, es necesario identificar cuáles son las redes sociales más importantes y representativas para llevar a cabo un estudio sobre ellas (existe una cantidad tan grande de redes sociales que resulta impracticable realizar un estudio sobre todas ellas). Es por esto, y considerando la clasificación de redes sociales propuesta en la sección 2.2.1, que se llegó a la conclusión de que cinco redes sociales era una cantidad suficiente para tener alcance sobre la diversidad existente. De esta manera, los estudios se realizaron sobre dos redes sociales de tipo Personal, dos redes sociales de tipo Intercambio de Contenido y una de tipo Intereses comunes. Se debe tener presente que las redes sociales de tipo Intereses comunes muestran un tipo de funcionamiento diferente a las

demás debido a que giran en torno a un tema especifico y deben adaptarse a éste, por lo tanto un estudio sobre un mayor número de éstas no ofrecería suficiente información relevante.

4.2.1. Selección de redes sociales

Sabiendo la cantidad y tipo de redes sociales que deben ser estudiadas, resulta necesario realizar una selección de aquellas más utilizadas a nivel mundial. Esta selección se basó principalmente en los resultados entregados por Alexa Internet (www.alexa.com), la cual es una subsidiaria de la compañía Amazon.com que provee información acerca de la cantidad de visitas que recibe un sitio web y permite su clasificación. Con la ayuda de esta herramienta se seleccionaron las siguientes redes sociales:

Redes sociales personales:

- Facebook:[1] Es la red social más popular del mundo y se encuentra en el puesto número 2 entre los sitios web con mayor cantidad de visitas a nivel mundial.
- Foursquare:[2] Pese a que no se encuentra entre las redes sociales más populares, se optó por escogerla debido a la diversidad de dominio que aporta. Se debe considerar que la mayoría de las redes sociales de tipo Personales son sumamente similares a *Facebook*, por lo que no aportarían demasiado al objetivo planteado.

Redes sociales de intercambio de contenido:

- Twitter:[3] Es la segunda red social más popular del mundo y se encuentra en el puesto número 9 entre los sitios web con mayor cantidad de visitas a nivel mundial.
- Instagram:[4] Se encuentra en el puesto 26 del ranking de *Alexa Internet* entre todos los sitios web del planeta, sin embargo existen otras redes sociales enfocadas al intercambio de contenido que resultan más populares. Este es caso de *Pinterest*, la cual fue considerada mejor opción por tratarse de una red social más popular y con un manejo de más variedad de información, pero debió ser descartada debido a que su API aún se encontraba en desarrollo.

Redes sociales de intereses comunes:

- Linkedin:[5] Es la tercera red social más popular del mundo y se encuentra en el puesto número 14 entre los sitios web con mayor cantidad de visitas a nivel mundial. Lo interesante de esto es que las redes sociales enfocadas a intereses comunes no suelen ser masivamente utilizadas, al tratarse de un tipo muy específico de dominio.

[1] https://www.facebook.com/
[2] https://foursquare.com/
[3] https://twitter.com/
[4] https://instagram.com/
[5] https://www.linkedin.com/

4.2.2. Acceso a la información

El siguiente paso es identificar qué información se puede obtener de cada red social, para lo cual es necesario llevar a cabo un estudio sobre la documentación de las APIs disponibles. Se debe recordar que las APIs (*Application Programming Interface* o Interfaz de Programación de Aplicaciones), son un conjunto de procedimientos y métodos que cumplen una serie de funciones que facilitan la comunicación entre módulos de distintos sistemas. En el último tiempo ha aumentado notoriamente el uso de este tipo de herramientas debido a que ofrece una serie de beneficios para las entidades que se relacionan con ellas:

- **Sistemas:** Las APIs desarrolladas por los distintos sistemas les permiten ofrecer acceso a la información que administran de forma segura y controlada. Esta funcionalidad fomenta el interés de los desarrolladores en relacionarse con el sistema, lo que conlleva una mayor publicidad y alcance sobre sus usuarios. Por ejemplo, un usuario de *Facebook* podría, además de acceder a todas las características propias de la red social, acceder gracias a ella a otro tipo de aplicaciones que requieran tener cuentas de usuario en determinadas redes sociales.

- **Desarrolladores:** Las APIs les ofrecen una guía sobre qué información está disponible, qué acciones pueden realizar y cómo deben hacerlo. De esta forma pueden complementar sus aplicaciones con información obtenida de las redes sociales, haciendo su trabajo más atractivo para los usuarios.

- **Usuarios:** Las APIs permiten a los usuarios compartir su información personal en forma segura y controlada con otras aplicaciones a cambio de mejores y/o nuevos servicios.

Pese a que en todos los sistemas varía la forma en que se deben utilizar sus APIs, en aspectos generales éstas funcionan de la misma manera. A continuación se describe el procedimiento completo para obtener la información de un usuario, desde la creación de la aplicación hasta el acceso a los datos.

Paso 1: Crear una aplicación

Si un desarrollador desea crear una aplicación que acceda a la información que los usuarios ofrecen en un determinado sistema, primero debe registrarse como desarrollador. En nuestro caso, es posible hacerlo en los siguientes sitios web:

- Facebook: https://developers.facebook.com/
- Foursquare: https://developer.foursquare.com/
- Twitter: https://dev.twitter.com/
- Instagram: https://instagram.com/developer/
- Linkedin: https://developer.linkedin.com/

Una vez registrado como desarrollador, se debe registrar una nueva aplicación especificando su nombre y los permisos que se requieran. Estos permisos especifican a qué tipo de información de usuario se quiere tener acceso. De esta forma, cuando el usuario intente utilizar la aplicación por primera vez, se le solicitará que acepte estos permisos, siendo especificado qué tipo de información será compartida en caso de aceptar.

Una vez registrada la aplicación, se obtienen los códigos de acceso que permiten acreditar más adelante que la aplicación alojada en un determinado sitio web corresponde a la aplicación registrada en el sistema.

Paso 2: Comunicación con la API

Ya que en general casi todas las APIs funcionan de la misma manera, a continuación se describe su funcionamiento utilizando como ejemplo un diagrama de secuencia publicado en la documentación de la API *Gateway* del sitio web de *Oracle*[6] (Figura 4.1).

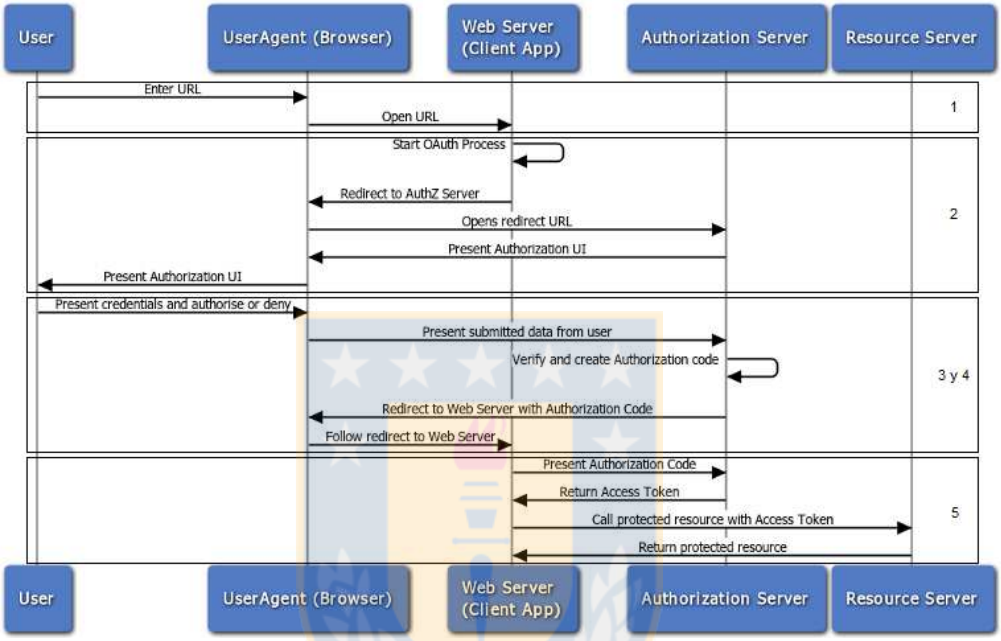

Figura 4.1: Diagrama de secuencia de la API *Gateway* de *Oracle*.

1. *Ingreso a la aplicación.* El usuario ingresa la dirección web de la aplicación a través su navegador.

2. *Autenticación de la aplicación.* La aplicación utiliza los códigos de acceso suministrados por la API para comprobar su autenticidad con respecto a los registros del sistema. Una vez terminado el proceso de autenticación, se le muestra al usuario que él debe autenticarse y aceptar los permisos solicitados por la aplicación en caso de no haberlo hecho en oportunidades anteriores.

3. *Autenticación del usuario.* Si es la primera vez que un usuario intenta acceder a la aplicación, se le muestra qué tipo de información será compartida en caso de aceptar los permisos requeridos por esta. Si el usuario ya ha aceptado los permisos en alguna oportunidad anterior, podrá ingresar directamente. Normalmente si el usuario deniega las solicitudes, no se le permite acceder a la aplicación.

4. *Permisos sobre el usuario.* Cuando el usuario accede a la aplicación, esta recibe un código que le permite acceder a la información del usuario. Esta información siempre será limitada por los permisos de acceso aceptados por el usuario.

[6]http://www.oracle.com/

5. *Consultas al servidor*. Con el código de acceso asociado al usuario es posible efectuar consultas al servidor sobre su información personal. Todas las APIs cuentan con una lista de las posibles consultas a realizar, las cuales van desde acceder a la información personal del usuario hasta crear nuevo contenido en su nombre. Adicionalmente, existen características asociadas a algunas APIs que permiten obtener información no sólo del usuario autenticado, sino de la comunidad de usuarios o de otras entidades participantes (como lugares, empresas, etc.).

4.2.3. Selección de información

Una vez que la aplicación puede comunicarse con el servidor (o en nuestro caso con la red social) a través de la autenticación de un usuario, se puede tener acceso a una enorme cantidad de información. Es por esto que para identificar qué tipo de información es de alguna manera útil para los sistemas de recomendación, es necesario llevar a cabo un estudio sobre la documentación ofrecida por las APIs de las distintas redes sociales.

El principal criterio de selección de información es que permita usar de alguna manera los métodos de recomendación mencionados en el capítulo 2, esto es: información asociada a los intereses y preferencias del usuario, a las relaciones del usuario con otros miembros de la comunidad, a las interacciones entre el usuario y otras entidades, etc. El detalle de esta información se muestra en el siguiente capítulo.

4.3. Modelo

Una vez que se ha identificado toda la información relevante a la que es posible acceder, se está en condiciones de desarrollar un modelo que describa el funcionamiento del sistema de recomendación. Como se mencionó anteriormente, este modelo no se desarrolló desde cero, ya que se cuenta con el modelo abstracto de un sistema de recomendación propuesto en [16] para ser usado como base (ver Figura 4.2).

Este modelo abstracto consideran los aspectos más importantes de un sistema de recomendación, independiente de su dominio de aplicación. De esta forma, se cuenta con un catálogo que contiene todos los ítems del sistema, con que cada ítem puede tener una serie de ítems similares en base a algún algoritmo, que las preferencias establecen una relación entre los ítems y los usuarios (las cuales pueden ser de tipo unaria, binaria o escalar), que existe un grupo de usuarios que contiene a todos los miembros de la comunidad y que cada usuario puede tener una serie de usuarios similares. Adicionalmente, la Figura 4.2 muestra un ejemplo de cómo se puede especializar el modelo a un dominio particular como es la recomendación de películas, donde se consideran aspectos como que un ítem es una película y que está asociado a actores, géneros y etiquetas.

En consecuencia, ya que se cuenta con los aspectos más importantes de un sistema de recomendación genérico, se deben identificar las características adicionales necesarias para modelar el contenido presente en las redes sociales.

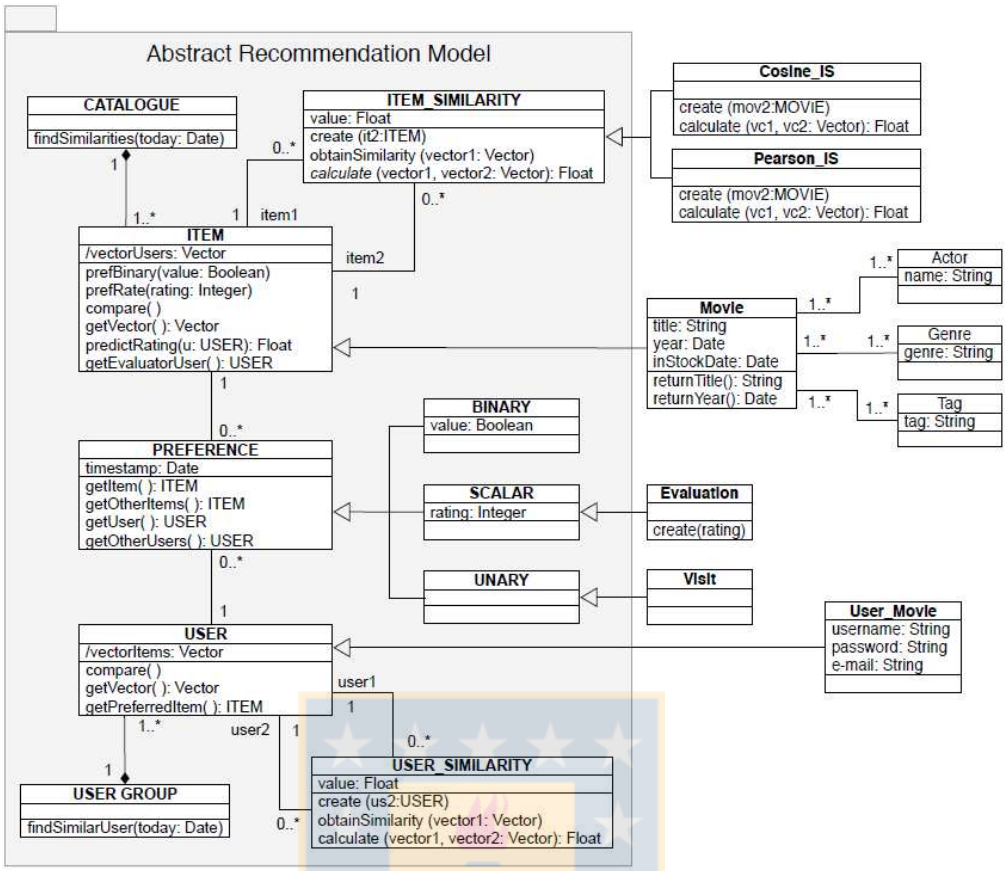

Figura 4.2: Modelo abstracto de un sistema de recomendación.

4.4. Implementación

El siguiente paso es utilizar este modelo para generar un sistema de recomendación, para lo cual existen principalmente dos herramientas capaces de generar código a partir de un modelo UML. La primera es la transformación de modelo a código en forma automática utilizando el *plugin* del IDE[7] Eclipse, Java Code Generation[8], el cual permite obtener instantáneamente el código de un modelo en el lenguaje de programación Java. La segunda herramienta es la transformación de modelo a código utilizando el complemento de Eclipse, Acceleo[9], el cual permite establecer manualmente las reglas de transformación de modelo a cualquier lenguaje de destino. Pese a que el uso de Acceleo requiere disponer de una mayor cantidad de tiempo debido a que se deben establecer las reglas de conversión, para esta Memoria de Título se cuenta con un *plugin* para Eclipse llamado *RecommendationSupport*[10], que ya contiene las reglas de transformación establecidas y tiene otro importante beneficio, fue desarrollada específicamente para la implementación de sistemas de recomendación y cuenta también con una serie de algoritmos de recomendación disponibles para generar recomendaciones automáticamente. Cabe señalar que, aunque es posible generar recomendaciones utilizando una gran variedad de métodos de recomendación sobre la diversa información presente en las redes sociales, los estudios realizados en esta Memoria de Título se llevaron a cabo utilizando únicamente Métodos de Filtrado Colaborativo, debido principalmente a que los

[7]Integrated Development Environment o Ambiente de Desarrollo Integrado
[8]`https://wiki.eclipse.org/Java_Code_Generation/`
[9]`https://eclipse.org/acceleo/`
[10]http://adweb.inf.udec.cl/rectool/

requisitos para implementar estos métodos permiten sacar el mayor provecho a los aspectos más característicos de las redes sociales: una gran cantidad de usuarios interactúa con una serie de elementos expresando su opinión sobre ellos en forma explícita (*me gusta*, *favoritos*, asignando calificaciones, etc.) e implícita (al responder publicaciones, *retweetear*, etc.).

En consecuencia, la implementación del modelo propuesto se llevó a cabo haciendo uso de las dos herramientas mencionadas anteriormente. De esta forma, no sólo es posible demostrar que el modelo propuesto permite la generación de un sistema de recomendación que se complemente de las redes sociales, sino también es posible comparar los resultados obtenidos a partir de las distintas formas de implementación.

4.5. Experimentos

El último paso es evaluar el desempeño de los procedimientos establecidos, poniendo principal atención en el tiempo que toma generar un sistema de recomendación al utilizar cada uno de ellos. Para esto, se compararon los resultados obtenidos con otras formas de generar recomendaciones, como utilizar sólo librerías de recomendación o crear un modelo desde cero para una red social específica.

Finalmente, se presenta el análisis de los resultados obtenidos y un ejemplo de cómo sería un uso real de los prototipos en un sistema de recomendación ya existente.

4.6. Conclusión

En este capítulo se presentaron, en forma general, todas las etapas que fueron requeridas durante el desarrollo de esta Memoria de Título para poder implementar un sistema de recomendación que integre la información de las redes sociales y se fundamentaron las decisiones más relevantes para llevarlo a cabo.

En los capítulos posteriores se abordan los mismos pasos que acaban de ser introducidos, pero en forma más detallada.

Capítulo 5

Análisis de Redes Sociales

Durante este capítulo se presenta el resultado de un diagnóstico de la información presente en las redes sociales *Facebook*, *Twitter*, *Linkedin*, *Instagram* y *Foursquare*. Adicionalmente, se detalla qué tipo de información, relevante para ser usada por sistemas de recomendación, es posible obtener a través de las APIs que ofrecen estas redes sociales. Finalmente, se proponen distintas formas en las que se podrían enriquecer los sistemas de recomendación si integraran la información suministrada por los usuarios en las distintas redes sociales. De esta forma, este diagnóstico es la base para diseñar la propuesta de trabajo descrita en capítulos posteriores.

5.1. Descripción General

Existe una gran variedad de herramientas utilizadas por las redes sociales para potenciar sus características y su llegada a los usuarios. Aunque en la sección 2.2.1 se propuso una lista de las herramientas más utilizadas, cada red social tiene completa libertad de adaptar estas herramientas a sus necesidades o simplemente desarrollar otras totalmente nuevas e innovadoras. Cada una de estas herramientas permite que el usuario declare y/o reciba distintos tipos de información, por lo que cada red social administra distintos tipos de datos y de diferentes formas. A esto último se debe que todas las APIs sean muy variadas con respecto a la información que administran.

En lo que sigue de este capítulo se muestran las principales herramientas de las redes sociales seleccionadas en el capítulo 4, la información relevante a la que ofrecen acceso y una serie de propuestas sobre cómo se podrían enriquecer los sistemas de recomendación utilizando esta información.

5.2. Facebook

Facebook es la red social más popular del planeta, principalmente por las posibilidades que ofrece a sus usuarios y se considera una red social de tipo Personal (según la clasificación propuesta en el capítulo 2.2.1) debido a que su funcionalidad más visible es facilitar la comunicación e interacción entre sus usuarios. Para esto *Facebook* ha implementado una gran variedad de herramientas, pero dentro de las más importantes se pueden identificar:

- **Vínculos sociales:** Cada usuario tiene la posibilidad de ponerse en contacto con otros usuarios que también utilicen *Facebook*. Para esto posible buscarlos por medio de su nombre y enviarles una *solicitud de amistad*, la cual notifica al otro usuario sobre quién desea estar en contacto con él. En caso que el segundo usuario acepte la solicitud, pasan a considerarse *amigos* dentro de la red social, lo que les permite acceder al *perfil de usuario* de su *amigo* y ver el contenido que publique.

- **Perfil de usuario:** Ofrece a sus usuarios la opción de mostrar su información personal. Esta puede incluir datos como fecha de nacimiento, historial académico, historial laboral, relaciones familiares declaradas en el sitio y foto personal, entre otros. Adicionalmente, permite configurar la privacidad del *perfil de usuario* para restringir quiénes pueden acceder a éste.

- **Muro o Biografía:** Muestra una lista de publicaciones hechas por el propio usuario y/o por sus *amigos*, las cuales pueden contener videos, imágenes, enlaces, noticias, texto, etc. Es uno de los aspectos más característicos de *Facebook* debido a que permite visualizar en tiempo real y en un solo sitio todas las publicaciones realizadas por cada una de las *amistades* del usuario. Adicionalmente, estas publicaciones pueden ser compartidas (equivalente a publicar en el propio *muro* el contenido compartido por otro usuario, pero mencionando que se obtuvo a través de él) y comentadas.

- **Etiquetas:** En cada una de las publicaciones es posible *etiquetar* a varios usuarios, lo que les notifica que fueron mencionados y fomenta su participación.

- **Envío de mensajes:** Permite el intercambio instantáneo de mensajes privados entre usuarios.

- **Declarar preferencias:** Para cada elemento disponible, *Facebook* ha implementado la opción de declarar "me gusta" como un tipo preferencia unaria. De esta forma permite a los usuarios saber qué tan bueno o popular es algún contenido en base a la cantidad de *me gusta* que posea. Adicionalmente, se permite asignar preferencias unarias y escalares a determinado tipo de contenido. Por ejemplo, un usuario puede contar con una lista de los libros a los que marcó con *me gusta*, otra con los libros que ha marcado "por leer" y otra con los libros que ha declarado "leídos" (si un libro se declara como leído, también es posible asignarle una calificación en una escala de 1 a 5).

5.2.1. Información relevante

Tal como es esperable de una red social de tipo personal, la información de *Facebook* disponible a través de su API se centra principalmente en las preferencias declaradas e interacciones realizadas por los usuarios. Sin embargo, también ofrece acceso a determinados tipos de ítems como videos, música, libros, etc. En el caso particular de *Facebook*, para acceder a la información relacionada con un usuario este debe haber aceptado las solicitudes realizadas por la aplicación que recopila la información.

A continuación se presenta una lista de la información más relevante disponible a través de la API de *Facebook*:

- **Información personal:** Cada usuario de *Facebook* tiene asignado un número identificador, a través del cual se llevan a cabo las consultas sobre la información personal más general declarada por el usuario. De esta forma es posible obtener datos como nombre, correo electrónico, educación cursada, puestos de trabajo ocupados, lugar de nacimiento, etc. Para información más detallada, ver Anexos (Tabla 9.1).

- **Relaciones con usuarios:** Se puede obtener la cantidad de usuarios que son amigos de un determinado usuario, los grupos de amigos de los cuales forma parte, las relaciones explícitas que ha declarado y sus mensajes privados. Se debe tener presente que para acceder a la información asociada a otros usuarios, estos también deben haber aceptado los permisos solicitados por la aplicación que realiza las consultas. Para información más detallada, ver Anexos (Tabla 9.3).

- **Publicaciones.** Es posible acceder a cada publicación realizada en el *muro* del usuario autenticado, tanto por él como por otro usuario. Cada una de estas publicaciones tiene asociado un número identificador que permite obtener más información, tal como el número de veces que ha sido compartido, los elementos contiene (fotos, enlaces, etc.), el número de veces que ha sido marcado con *me gusta* o el lugar y fecha donde fue publicado. Para información más detallada, ver Anexos (Tablas 9.4 y 9.2).

- **Preferencias Explícitas:** *Facebook* ha dispuesto la creación de páginas que representan un determinado tipo de contenido. En otras palabras, una página puede representar, por ejemplo, a un artista, donde toda la información administrada y las preferencias recibidas por aquellas página se atribuyen a aquel artista. De esta forma, existe una serie de categorías como libros, música y películas que agrupan páginas a las cuales se puede asignar una preferencia escalar en el rango de 1 a 5. Sin embargo, también existe una enorme cantidad de otras categorías (relacionadas con productos, empresas, animales, etc.), cada una con su propio conjunto de datos y que es posible marcarlos como *me gusta*. Para información más detallada, ver Anexos (Tablas 9.5, 9.6 y 9.7).

5.2.2. Posibilidades de recomendación

Una vez que se ha identificado el tipo de información de usuario a la que se puede tener acceso, es posible identificar posibilidades de complementar los métodos de recomendación mencionados en el capítulo 2 utilizando información que anteriormente no era considerada o que era utilizada de otra forma. A continuación se propone una lista de posibilidades que permitirían mejorar un sistema de recomendación.

1. Actualmente la similitud entre usuarios (en los sistemas de recomendación) se obtiene a través de las preferencias explícitas que éstos han declarado. Esto podría complementarse utilizando la similitud entre usuarios que es posible rescatar de *Facebook* de la siguiente manera:

   - Considerar que las publicaciones son ítems y que comentarlas, darles *me gusta* y compartirlas es un tipo de evaluación unaria.
   - Considerar que los usuarios también pueden ser tratados como ítems y que las relaciones de amistad representan una evaluación unaria.

   En consecuencia, es posible obtener directamente similitudes y recomendaciones, tanto sobre usuarios como sobre ítems, utilizando Métodos de Filtrado Colaborativo (ver capítulo 2).

2. Ya que existen preferencias declaradas explícitamente sobre algunos ítems como películas, música y libros, estas pueden exportarse directamente a sistemas de recomendación que trabajen sobre estos dominios, facilitando así el ingreso de nuevos usuarios al sistema. Para esto, se requiere establecer una relación entre los ítem presentes en la red social con su ítem equivalente en el sistema de recomendación, para lo cual se pueden comparar identificadores (por ejemplo, en el caso de películas: si las dos partes cuentan con el identificador utilizado por IMDB[1] para una determinada película) o hacer uso de análisis de texto en caso de comparar nombres.

   Adicionalmente, se puede acceder a todos los elementos de *Facebook* a los cuales se les ha asignado *me gusta*, con lo cual se pueden obtener todas las categorías a las cuales estén asociados. En consecuencia, es posible utilizar estas categorías para construir un perfil que represente todos las preferencias del usuario y utilizar Métodos Basados en Contenido (ver capítulo 2) para generar recomendaciones.

3. Acceder a las conversaciones privadas de los usuarios permite utilizar la cantidad de mensajes intercambiados como un indicador de interacción y cercanía entre los distintos usuarios.

4. Utilizar la cantidad de veces que han sido compartidas o la cantidad de *me gusta* que han recibido las publicaciones de un usuario como un indicador de la confianza que genera en el resto de la comunidad. Esta información puede complementar los sistemas de recomendación basados en confianza como el mencionado en el capítulo 2.

5. Es posible utilizar las publicaciones en las que se menciona a un usuario y que además referencian un lugar, como una preferencia unaria. Esto sería de gran utilidad para complementar sistemas que recomienden lugares y facilitar el ingreso de nuevos usuarios.

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

5.3. Twitter

Twitter es una red social de intercambio de contenido (según la clasificación propuesta en el capítulo 2.2.1) y su creciente éxito radica principalmente en que permite que sus usuarios publiquen en mensajes de texto cortos (no más de 140 caracteres) de manera instantánea. De esta forma, como todo el contenido está resumido, *Twitter* se convierte en una forma rápida y fácil de estar informado en todo momento sobre temas de interés personal. En sentido contrario, se transforma en una herramienta para dar a conocer ideas o acontecimientos instantáneamente y desde cualquier lugar, gracias a su amplia llegada a dispositivos móviles. *Twitter* destaca por ser una red social simple, sin demasiadas funcionalidades, pero que ha sabido personalizar sus herramientas para conseguir lo que sus usuarios necesitan. Dentro de estas herramientas es posible encontrar:

- **Perfil de usuario:** Aunque es bastante simple y limitado, cada usuario posee una sección en la que puede declarar información personal, tal como nombre, ciudad en la que vive y una pequeña descripción. Por defecto este perfil puede ser visto por cualquier usuario, pero existen configuraciones de privacidad que permiten regularlo.

- **Vínculos sociales:** *Twitter* ofrece un tipo de relación unidireccional, donde un usuario puede "seguir" a otros usuarios (no es necesaria su aprobación para seguirlos, a menos que declaren su cuenta como privada), pero estos no necesariamente lo *siguen* a él. En consecuencia, cada vez que un usuario publique algún contenido, este será visible en los *timelines* de todos los usuarios que lo *sigan*.

- **Compartir contenido:** El aspecto más característico de *Twitter* es la forma en que se publica el contenido. Todas las publicaciones realizadas siguen el formato de una cadena de texto de, a lo más, 140 caracteres, los cuales reciben el nombre de "tweet". Sin embargo, cada *tweet* también puede adjuntar enlaces, imágenes y/o videos. Adicionalmente, cada uno de estos *tweets* puede ser respondido, "retweeteado" (otro usuario publica el mismo *tweet* como propio, pero referenciando al usuario que lo publicó originalmente) y/o marcado como *favorito* (una forma de preferencia unaria).

- **Etiquetas:** Por un lado, en cada *tweet* es posible agregar una serie de etiquetas simplemente concatenando el caracter # al comienzo de una palabra. Esto permite agrupar y buscar *tweets* a partir de las eqiquetas (o *hashtag*) que contenga. Por otro lado, también es posible referenciar a otros usuarios concatenando el caracter @ al comienzo del nombre de sus cuentas. Esto permite que un usuario le avise a otro que el contenido de su *tweet* está relacionado con él o que puede ser de su interés, además también facilita seguimiento de los *tweets* que son en respuesta a otro.

- **Timeline (Microblog):** El *timeline* de un usuario es una sección destinada a mostrar todos los *tweets* realizados por todos los usuarios a los que sigue. Sin embargo, es posible personalizar otras secciones para visualizar subconjuntos de estos *tweets*.

- **Envío de mensajes:** Permite el intercambio instantáneo de mensajes privados entre usuarios.

5.3.1. Información relevante

Dado que *Twitter* es una red social enfocada al intercambio de contenido, la información a la cual se puede acceder a través de sus APIs está relacionada principalmente con los *tweets*

publicados. Sin embargo, también ofrece acceso a otro tipo de información como mensajes privados y listas de usuarios.

A continuación se presenta una lista con la información más relevante disponible a través de la API de *Twitter*:

- **Información de usuario:** Cada usuario de *Twitter* tiene asignado un número identificador a través del cual es posible realizar consultas sobre su información personal. En el caso de *Twitter* es posible acceder a la información de cualquier usuario mientras se conozca su número identificador (o nombre de su cuenta) y no haya restringido su privacidad. De esta forma se puede acceder a datos como nombre del usuario, nombre de cuenta, descripción personal, número de usuarios *seguidores*, número de usuarios *seguidos*, etc. Para información más detallada, ver Anexos (Tabla 9.8).
- **Información de cada *tweet*:** Los *tweets* son la característica más importante de *Twitter*, por lo que cada uno de ellos también posee un número identificador que facilita la obtención de su información. Así, la API permite consultar datos como: si el *tweet* fue publicado en respuesta a otro, cuál fue el medio por el que se publicó (qué tipo de dispositivo), número de veces que ha sido marcado como *favorito*, número de veces que ha sido *retweeteado*, etc. Para información más detallada, ver Anexos (Tabla 9.9).
- **Relaciones personales:** Dentro de las relaciones declaradas por los usuarios se pueden obtener la información personal de todos los usuarios seguidos y seguidores de un determinado usuario, búsquedas guardadas, los miembros de las listas de usuarios personalizadas, etc. Para información más detallada, ver Anexos (Tabla 9.10).
- **Información sobre publicaciones:** Adicionalmente a la información asociada a cada *tweet* es posible obtener información relacionada conjuntos de estos. De esta manera, es posible acceder a la lista de los *tweets* más recientes en los que se ha mencionado un determinado usuario, los *tweets* más recientes que ha publicado, los *tweets* y *retweets* mostrados en el *timeline* del usuario, etc. Para información más detallada, ver Anexos (Tabla 9.11).

5.3.2. Posibilidades de recomendación

Una vez que se identifica el tipo de información a la que se puede tener acceso, es posible proponer formas en las que se pueden enriquecer los sistemas de recomendación complementando los métodos mencionados en el capítulo 2.

1. En forma análoga a lo visto en *Facebook*, es posible complementar la similitud de usuario usada en los sistemas de recomendación a través la similitud de usuario obtenida de las interacciones dentro de *Twitter*. Esta similitud dentro de la red social puede obtenerse de la siguiente manera:
    - Considerar que los *tweets* son ítems y que marcarlos como *favorito*, hacerles *retweet* o comentarlos es una forma de evaluación unaria.
    - Considerar que los usuarios también pueden ser tratados como ítems y que *seguir* a un usuario representa una forma de evaluación unaria.

En consecuencia, es posible obtener directamente similitudes y recomendaciones, tanto sobre usuarios como sobre ítems, utilizando Métodos de Filtrado Colaborativo (ver capítulo 2).

2. Es posible crear un perfil para cada usuario que incluya todos los *hashtag* que ha utilizado. De esta manera se puede hacer uso de Métodos Basados en Contenido (ver capítulo 2) para generar recomendaciones.

3. Utilizar la cantidad de veces que han sido *retweeteadas* o la cantidad de *favoritos* que han recibido las publicaciones de un usuario como un indicador de la confianza que genera en el resto de la comunidad. Esta información puede complementar los sistemas de recomendación basados en confianza como el mencionado en el capítulo 2, en el cual se pueden evaluar también a los usuarios, de tal forma que las recomendaciones basadas en un usuario confiable serán mejor recibidas.

4. Acceder a las conversaciones privadas de los usuarios permite utilizar la cantidad de mensajes intercambiados como un indicador de interacción y cercanía entre los distintos usuarios.

5. Es posible utilizar los *tweets* en los que se menciona a un usuario y que además se referencia un lugar como una forma de preferencia unaria para ese lugar. Esto sería de gran utilidad para complementar sistemas que recomienden lugares, facilitando el ingreso de nuevos usuarios.

5.4. Linkedin

Aunque el hecho de ser la tercera red social más popular del planeta es en sí un logro llamativo, lo verdaderamente destacable de *Linkedin* es que lo consiga siendo una red social de tipo Intereses comunes (según la clasificación propuesta en el capítulo 2.2.1). El tema central sobre el que se desarrolla *Linkedin* está estrictamente ligado al ámbito laboral. Esta red social está diseñada para facilitar a sus usuarios una forma de hacer visibles sus antecedentes laborales, pudiendo ser accedidos por otros usuarios o por empresas en búsqueda de nuevos miembros. En otras palabras, su principal característica es poder conectar a sus usuarios con las organizaciones en las que presenten interés o viceversa.

En consecuencia, *Linkedin* ha puesto a disposición de sus usuarios herramientas que principalmente les permitan declarar sus estudios, virtudes y experiencias. Dentro de las herramientas más relevantes que se pueden identificar se encuentran:

- **Vínculos sociales:** Como toda red social, *Linkedin* permite a sus usuarios interactuar entre sí, para lo cual es necesario que estos estén *conectados*. La conexión entre usuarios es un vínculo bidireccional que permite realizar acciones que involucran a otras personas, tal como recomendar un usuario a otro o validar algunas características personales declaradas.

- **Perfil de usuario:** Es la principal herramienta de *Linkedin* y está diseñada para que el usuario detalle cada aspecto relacionado su vida laboral. De esta forma, es posible declarar todos los estudios realizados, un historial detallado de experiencias laborales, listar conocimientos sobre temas específicos y mencionar virtudes, entre muchos otros. Dada la naturaleza de esta red social, esta información suele ser visible para cualquier miembro de la comunidad.

- **Inicio (Microblog):** Ofrece una sección que reúne todas las publicaciones hechas por otros usuarios, empresas o páginas de interés. Así mismo, facilita la publicación y difusión de contenido para el usuario.

5.4.1. Información relevante

Dado que *Linkedin* es una red social enfocada en dar a conocer cierto tipo de características personales de sus usuarios, la información a la que se puede acceder a través de su API está principalmente relacionada con el perfil declarado por sus usuarios.

A continuación se se presenta una lista con la información más relevante disponible a través de la API de *Linkedin*:

- **Información de usuario:** Cada usuario de *Linkedin* tiene asociado un número identificador a través del cual es posible realizar consultas sobre su información personal. De esta forma, es posible acceder a datos como nombre del usuario, título profesional, industria en la que de desenvuelve, especialidades y habilidades declaradas (en forma de etiquetas), cursos tomados, voluntariados, etc. Para información más detallada, ver Anexos (Tabla 9.12).
- **Información de compañías:** Es posible acceder a datos asociados a las distintas compañías que se exhiben dentro de *Linkedin*, tal como nombre, tipo de compañía, industria a la que pertence, especialidades, etc. Para información más detallada, ver Anexos (Tabla 9.13).

5.4.2. Posibilidades de recomendación

A continuación se proponen algunas formas en que se podría enriquecer un sistema de recomendación utilizando la información mencionada en la sección anterior.

1. Considerando que la mayor parte de la información disponible tiene relación con las características de los usuarios, se pueden rescatar estos datos y almacenarlos en un perfil de usuario adicional para implementar Métodos Basados en Contenido (ver capítulo 2). De esta forma, se puede calcular la similitud entre distintos usuarios en base a características declaradas por ellos mismos, como sus habilidades, voluntariados, etc.
2. Pese a que es posible calcular un valor de similitud entre usuarios utilizando la información de los usuarios con los que están relacionados y Métodos de Filtrado Colaborativo (ver capítulo 2), no resulta recomendable debido a la naturaleza de la red social: las *conexiones* no representan necesariamente cercanía o similitud. Sin embargo, esta información podría ser utilizada para complementar los resultados obtenidos en base al punto anterior.

5.5. Instagram

Instagram es una red social de tipo Intercambio de contenido (según la clasificación propuesta en el capítulo 2.2.1) que se especializa en la publicación personalizada de imágenes y videos de corta duración . De esta forma, *Instagram* permite a sus usuarios visualizar contenido compartido por otros usuarios e interactuar con él.

Pese a que esta red social se caracteriza por ser simple, es posible encontrar las siguientes herramientas:

- **Vínculos sociales:** Cada usuario puede *seguir* a otros usuarios. Esto representa una relación unidireccional que asegura el acceso a las publicaciones del usuario *seguido*.
- **Inicio o Muro (Blog):** Es la sección más importante de *Instagram*. Aquí el usuario puede visualizar, en orden cronológico y en tiempo real, todo el contenido publicado por los usuarios que *sigue*.
- **Compartir contenido:** Todos los usuarios pueden publicar contenido, pero debe estar restringido a imágenes y videos de corta duración. Adicionalmente, este contenido puede ser acompañado de un texto que permite citar a otros usuarios o indicar el lugar donde se capturó el contenido. Finalmente, los usuarios pueden realizar comentarios en respuesta a las diversas publicaciones o marcar que les han gustado.
- **Perfil de usuario:** Cada usuario puede declarar su información personal básica para que sea visible por otros usuarios.
- **Etiquetas:** Dentro de cada publicación o comentario puede agregarse una etiqueta concatenando el caracter # al comienzo de de una palabra. Esto permite agrupar el contenido que contenga estas etiquetas y facilitar su búsqueda. Por otro lado, es posible referenciar a otros usuarios concatenando el caracter @ al comienzo del nombre de su cuenta. Esto suele usarse para fomentar la participación del usuario referenciado o para darlo a conocer a otros usuarios.

5.5.1. Información relevante

Dado que *Instagram* es una red social más enfocada a compartir contenido que la información de sus usuarios, a través de su API se obtiene acceso a información que está principalmente relacionada con el contenido publicado por los usuarios.

A continuación se presenta una lista con la información más relevante disponible a través de la API de *Instagram*:

- **Información de usuario:** Cada usuario cuenta con un número identificador que permite realizar consultas sobre éste a la API. De esta forma se pueden obtener datos personales básicos como el nombre del usuario, número de usuarios *seguidos*, número de medios digitales publicados, una pequeña descripción, etc. Para información más detallada, ver Anexos (Tabla 9.14).
- **Publicaciones:** Todas las publicaciones realizadas en *Instagram* deben contener un medio digital, ya sea en forma de imagen o de video. De estas publicaciones se pueden extraer datos como los comentarios recibidos, una lista de los usuarios que han sido citados, una lista con las etiquetas utilizadas, etc. Para información más detallada, ver Anexos (Tabla 9.15).
- **Relaciones Personales:** Se obtiene acceso a las listas con todos los usuarios *seguidos* o que *siguen* a un usuario en particular. Para información más detallada, ver Anexos (Tabla 9.16).

5.5.2. Posibilidades de recomendación

A continuación se proponen algunas formas en que se podría enriquecer un sistema de recomendación utilizando la información mencionada en la sección anterior:

1. Pese a que las relaciones establecidas en *Instagram* son sumamente impersonales y que, por consiguiente, *seguir* a alguien no representa necesariamente cercanía, es posible establecer una similitud entre usuarios basándose en los usuarios en común a los que *siguen*. Para esto, es necesario considerar a los demás usuarios también como ítems y a la acción de *seguir* como una evaluación unaria.

2. La acción de referenciar a otros usuarios en alguna publicación o comentario puede tomarse como un indicador de cercanía y/o intereses comunes. Adicionalmente, contabilizar la cantidad de *me gusta* entre los usuarios puede usarse para complementar las similitudes obtenidas por otros medios.

3. Crear un perfil para cada usuario que agrupe las etiquetas que ha utilizado y que represente sus intereses permitiría encontrar similitudes con otros usuarios a través de Métodos Basados en Contenido (ver capítulo 2).

5.6. Foursquare

Foursquare es una red social de tipo Personal (según la clasificación propuesta en el capítulo 2.2.1) que se especializa en la exhibición y recomendación de lugares, los cuales pueden pertenecer a diversas categorías, desde *restaurantes* hasta *vida nocturna*. El funcionamiento de esta red social no se basa tanto en la interacción directa entre usuarios, sino más bien en un tipo de interacción indirecta a través de su plataforma. En otras palabras, los usuarios que han tenido la oportunidad de conocer un determinado lugar publican *tips* (comentarios) sobre cómo fue su experiencia allí, mientras que por otro lado, los usuarios que desean mayor información sobre cómo es un determinado lugar pueden leer los *tips* que ha recibido.

Esta red social se caracteriza por tener también una aplicación móvil que facilita la publicación de los *tips*, sin embargo comparte la mayor parte de sus funcionalidades con su versión web. Dentro de las herramientas más importantes que ha implementado *Foursquare* podemos encontrar:

- **Perfil de usuario:** Cada usuario cuenta con una sección en la que se muestra su información personal, los *tips* que ha publicado y las relaciones establecidas con otros usuarios.

- **Sugerencias y *tips* (Microblogs):** Cada usuario puede acceder a una lista de lugares sugeridos según las categorías que haya señalado como de interés. En esta sección se presenta una lista de sugerencias que ordena una gran variedad de lugares según su puntuación y muestra un resumen de cada uno. Adicionalmente, cada lugar cuenta con una lista de los *tips* que le han publicado los usuarios.

- **Vínculos sociales:** Los usuarios pueden establecer relaciones con otros usuarios, lo que facilita obtener recomendaciones por parte de la red social y permite al usuario visualizar *tips* directamente de personas por las que ha declarado interés.

- **Mapa:** Junto con la lista de lugares sugeridos *Foursquare* despliega un mapa que permite visualizar la ubicación de cada uno de ellos.
- **Checkin:** Todos los usuarios pueden hacer *checkin* a través de un dispositivo móvil. En otras palabras, un usuario puede publicar que actualmente se encuentra en un determinado lugar.

5.6.1. Información relevante

Pese a que *Foursquare* se trata de una red social de tipo Personal (debido a que se centra principalmente en los gustos y preferencias de sus usuarios), su funcionalidad más importante se desarrolla en torno a la información suministrada por sus usuarios sobre los distintos lugares de interés. Es por esto que la mayor parte de la información disponible a través de su API está relacionada con *tips* y lugares.

A continuación se presenta una lista con la información más relevante disponible a través de la API de *Foursquare*:

- **Información de usuario:** Cada usuario cuenta con un número identificador a través del cual se le pueden efectuar consultas a la API. De esta forma es posible acceder a datos como nombre, ciudad de residencia, una pequeña descripción, etc. Para información más detallada, ver Anexos (Tabla 9.17).
- **Información de lugares:** Todos los lugares también pueden ser consultados por medio de un número identificador, a través del cual se puede acceder a datos como nombre, ubicación, categorías, horario de atención, descripción, etc. Para información más detallada, ver Anexos (Tabla 9.18).
- **Publicaciones:** Como publicaciones se pueden encontrar tanto los *checkin* como los *tips*. Mientras que de los *checkin* se obtienen datos como fecha, ubicación y comentarios recibidos, de los *tips* se obtienen datos como fecha de creación, contenido (texto), lugar comentado, etc. Para información más detallada, ver Anexos (Tablas 9.19 y 9.20).

5.6.2. Posibilidades de recomendación

A continuación se proponen algunas formas de enriquecer un sistema de recomendación utilizando la información mencionada en la sección anterior:

1. Se puede calcular una similitud entre usuarios considerando las evaluaciones escalares declaradas sobre los diversos lugares a través de Métodos de Filtrado Colaborativo (ver capítulo 2).
2. También es posible calcular la similitud entre usuarios usando las evaluaciones unarias declaradas sobre distintos elementos de la red social, ya sea *checkins* sobre los lugares, *me gusta* sobre los *checkins* u otros.
3. Al reunir información sobre las preferencias establecidas implícitamente por los usuarios, como las categorías de los lugares que frecuentan y las etiquetas utilizadas, es posible generar un perfil que para obtener similitudes entre usuarios a través de Métodos Basados en Contenido (ver capítulo 2).

5.7. Conclusión

Durante este capítulo se describieron las características, herramientas e información disponible de cada una de las redes sociales en estudio. Se debe tener presente que aunque en algunos casos el tipo de información pueda parecer muy similar, la forma en que los usuarios utilizan las distintas herramientas suministradas se traduce en un significado diferente u otro grado de relevancia para la información. Por ejemplo, las amistades establecidas en *Facebook* tienen asociado un grado de cercanía mucho mayor que el visto en *Instagram*, donde usualmente no se conoce a las personas que se *siguen*.

Una vez establecidos y detallados los componentes y características de las redes sociales analizadas, es posible identificar una serie de aspectos comunes, dentro de los que destaca principalmente que independiente del grado de cercanía que representen las relaciones entre usuarios, es posible obtener un valor de similitud considerándolos como otro ítem por el que se puede declarar una preferencia (esto no está necesariamente relacionado con que se busque recomendar usuarios, sino con identificar una similitud entre usuarios en base a los *amigos* que tienen en común). Pese a que cada red social ofrece nuevas y distintas oportunidades de enriquecer un sistema de recomendación, es esta característica común la que lleva a resultados más inmediatos en caso de su aplicación, debido que las APIs de todas las redes sociales facilitan el acceso a los datos asociados a las relaciones entre usuarios.

En el siguiente capítulo se muestra cómo es posible utilizar la información obtenida por medios de las APIs de redes sociales para generar similitudes y sugerencias que complementen y/o enriquezcan otros sistemas de recomendación.

Capítulo 6

Definición e implementación del modelo

En este capítulo se muestra el modelo de sistema de recomendación desarrollado a partir de la información disponible en las redes sociales y se describen sus componentes más importantes. Adicionalmente, se detalla cómo llevar a cabo su implementación según las distintas herramientas disponibles.

6.1. Descripción del modelo

Como se detalló en el capítulo anterior, todas las APIs de las redes sociales ofrecen variada información sobre sus usuarios y otros elementos adicionales. Una vez que se tiene esta información es posible generar un modelo de sistema de recomendación que contenga las características más importantes. Para esto se utilizó un componente de código abierto de la extensión Model Development Tools[1] del IDE[2] Eclipse llamado Papyrus. Esta herramienta provee un entorno que facilita el desarrollo y edición de modelos UML. De esta forma, a través de Papyrus se desarrolló un diagrama de clases que puede administrar la información de cualquier red social (ver Figura 6.1).

En este modelo se pueden identificar todas las características asociadas a usuarios, ítems y preferencias presentes en las redes sociales.

- **User**. Los usuarios pueden estar asociados a una serie de características (*Features*). De esta forma se pueden administrar todo el contenido que puede ser utilizado en Métodos Basados en Contenido, como las etiquetas (en el caso de *Facebook*, *Twitter*, *Instagram*), habilidades (en el caso de *Linkedin*), ciertas categorías preferidas (en el caso de *Facebook*, *Foursquare*), etc. Adicionalmente, un usuario puede crear publicaciones o responder/comentar publicaciones hechas por él mismo u otros usuarios.
- **Item**. Dentro de las redes sociales se pueden identificar dos tipos de ítems principales: *Posts* (o Publicaciones) y *Digital Media* (o Medios Digitales). Sin embargo, considerar a los usuarios también como ítems permite generar recomendaciones sin importar la red social en que se trabaje.
  - **Post**. Es todo el contenido que ha sido publicado por un usuario y que, generalmente, puede ser comentado/respondido por él mismo u otros usuarios a través de otro *post*. Adicionalmente, este tipo de ítem puede contener una serie de elementos adicionales como imágenes, enlaces y medios digitales. Algunos ejemplos de estos ítems son los *tweets* de *Twitter*, las fotos de *Instagram* y los *tips* de *Foursquare*.
  - **Digital Media**. Es todo el contenido existente en la red social, pero que no fue publicado por un usuario de forma convencional. Adicionalmente, se considera a los lugares como una especialización de este tipo de ítem ya que su única diferencia es que estos se encuentran asociados a un lugar físico. Algunos ejemplos de estos ítems son los locales de *Foursquare*, las películas de *Facebook* y las empresas de *Linkedin*.
  - **User**. Considerar a los usuarios como ítems permite implementar Métodos de Filtrado Colaborativo para obtener similitudes entre usuarios directamente. De esta forma se aprovecha una característica inherente de todas las redes sociales, el hecho de establecer relaciones con otros usuarios.
- **Preference**. Dentro de las redes sociales se pueden encontrar los tres tipos de evaluaciones sobre ítems: unaria, binaria y escalar. El uso de cada una se rige directamente por la naturaleza de la red social. De esta manera, es posible identificar evaluaciones escalares hechas por usuarios sobre películas o comentarios (asignándoles una nota), una evaluación unaria sobre publicaciones (*me gusta* o *favorito*) o binaria en el caso de otras redes sociales.

[1] Herramientas para el desarrollo de modelos
[2] Integrated Development Environment o Ambiente de Desarrollo Integrado

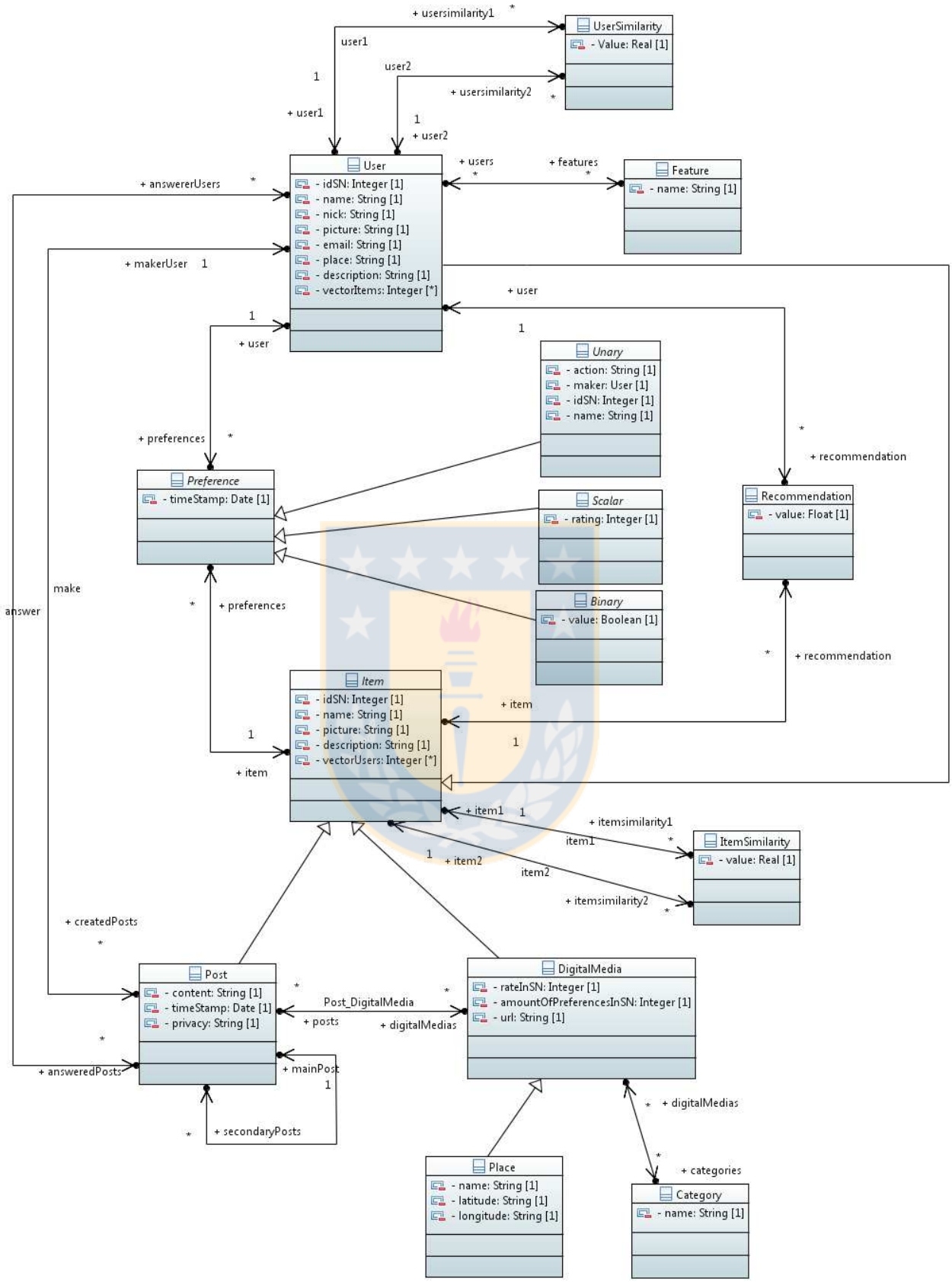

Figura 6.1: Diagrama de clases de un sistema de recomendación que usa información de redes sociales.

6.2. Implementación

Tal como se mencionó en el capítulo 4, la implementación del modelo se llevó a cabo utilizando dos tipos de herramientas de Eclipse, el *plugin* RecommendationSupport y el *plugin* Java Code Generator.

Independientemente de cuál herramienta se utilice, el sistema de recomendación a implementar se encuentra dividido en tres módulos (ver Figura 6.2). El primer módulo está relacionado con las APIs de las redes sociales y es el encargado de conectarse con la respectiva red social, facilitar el ingreso de los usuarios y obtener la información requerida por el sistema generado. El segundo módulo está encargado de administrar la información obtenida mediante las API de las redes sociales, facilitando sus consultas y la conexión con otros módulos. Finalmente, el tercer módulo es el encargado de generar las recomendaciones a partir de los datos obtenidos de las redes sociales utilizando algoritmos de recomendación.

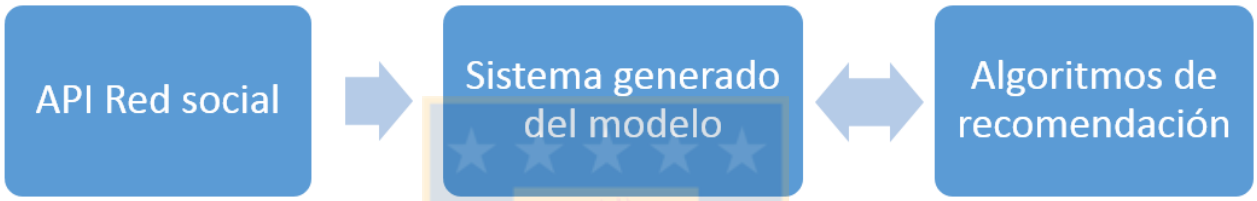

Figura 6.2: Módulos principales del sistema de recomendación.

En consecuencia, se observa que el primer módulo es común para las distintas formas de implementación debido a que es la única forma de obtener los datos de los usuarios, que el segundo módulo se obtiene generando código automáticamente del modelo propuesto a través de los *plugins* mencionados anteriormente y que la implementación del tercer módulo depende de la herramienta utilizada para generar el segundo.

A continuación se explican con mayor detalle los pasos necesarios para obtener el sistema de recomendación definitivo según la herramienta seleccionada y la arquitectura que lo describe. Adicionalmente se debe señalar que, dado que el primer módulo del sistema se obtiene de la misma manera en los dos casos, en la descripción se considera que los datos de los usuarios ya fueron obtenidos.

6.2.1. Arquitectura de la solución

En la arquitectura de la solución propuesta, visible en la Figura 6.3, se muestran las interacciones entre sus distintos elementos:

- **Modelo abstracto**. Describe el funcionamiento genérico de un sistema de recomendación que integre la información de redes sociales (ver Figura 6.1).
- **Modelo específico**. Describe el funcionamiento específico de una red social. Ahí es donde los desarrolladores deben describir las características de una red social en particular que no estén consideradas en el modelo abstracto. En Anexos (Sección 9.2) se muestran algunos ejemplos.
- **API Red social**. La información disponible en las redes sociales se guarda en una base de datos.

- **Algoritmo de Filtrado Colaborativo**. Los requisitos para el algoritmo de Filtrado Colaborativo que se utilice debe concordar con la información administrada por el modelo abstracto.
- **Código de algoritmos de recomendación**. Se genera el código correspondiente a los algoritmos de Filtrado Colaborativo, los cuales permiten obtener las recomendaciones para el sistema.
- **Código del sistema de recomendación**. A partir del modelo abstracto y del modelo específico se obtiene todo el código que permite la implementación de un sistema de recomendación.
- **Definición de ejecución**. Es el código encargado de obtener la información de la base de datos y entregarla a los algoritmos de recomendación utilizando el código del sistema de recomendación. Adicionalmente, está encargada de administrar los resultados obtenidos y guardarlos en la base de datos.

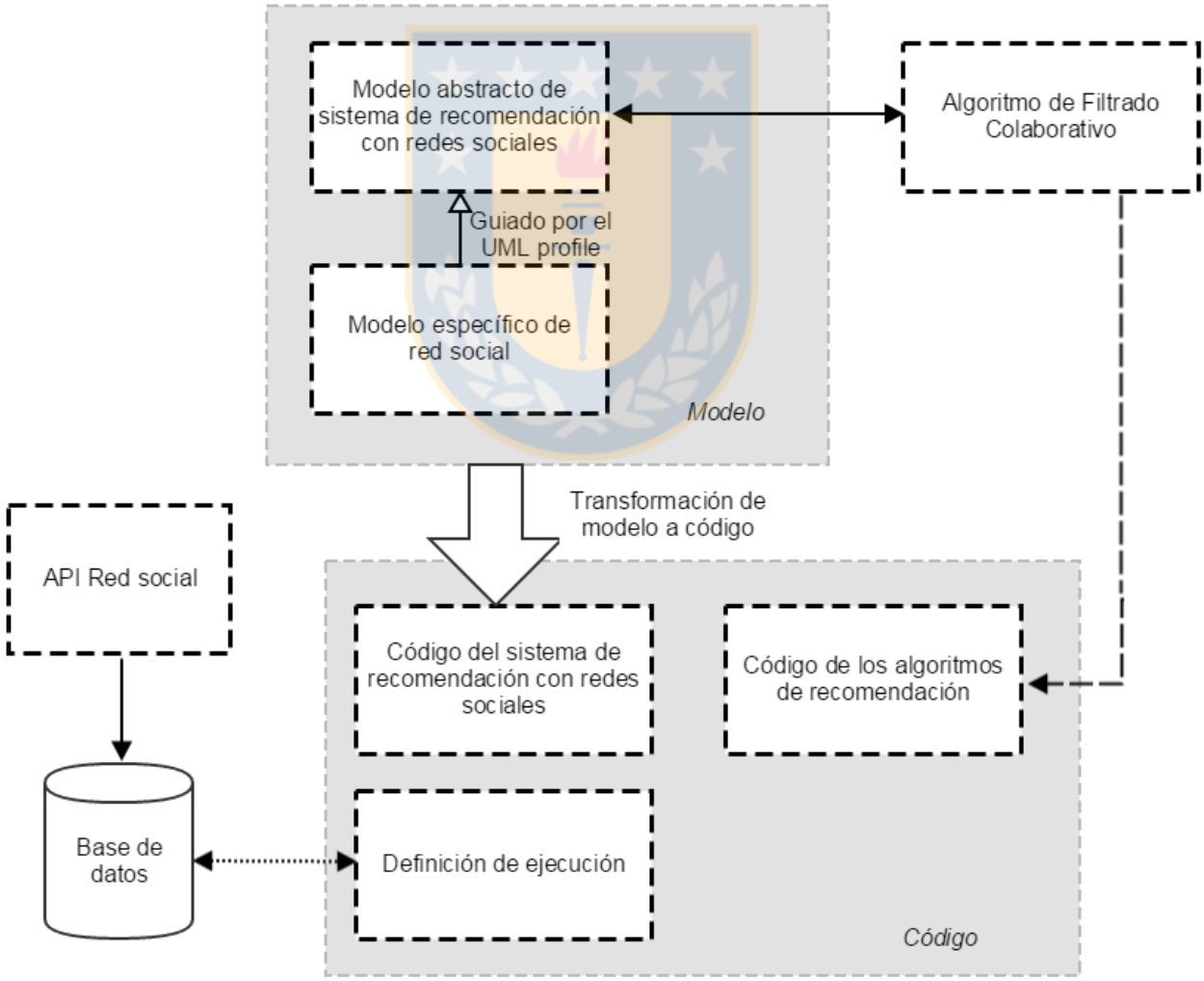

Figura 6.3: Arquitectura de la solución propuesta para generar sistema de recomendación basados en modelos.

6.2.2. Plugin RecommendationSupport

Este *plugin* fue desarrollado en base al modelo abstracto de la Figura 4.2 y su principal objetivo es el de facilitar la implementación de sistemas de recomendación a través de la generación automática de código. Adicionalmente, cuenta con una serie de algoritmos de Filtrado Colaborativo que permiten obtener recomendaciones sin necesidad de otras herramientas. Sin embargo, al ser desarrollado exclusivamente para ese modelo, el modelo propuesto en la Figura 6.1 debe ser adaptado para que pueda ser compatible. De esta forma, se extendió el modelo anterior a un nuevo modelo abstracto que, además de incluir la información identificada en las redes sociales, contiene las clases requeridas por el *plugin* para la generación de recomendaciones (ver Figura 6.4):

- **UserGroup**. Clase abstracta para administrar a todos los usuarios registrados.
- **Catalogue**. Clase abstracta para administrar a todos los ítems registrados.
- **Similarity y algoritmos**. Clases abstractas necesarias para comunicarse con los algoritmos del *plugin*.

Con estas modificaciones ya es posible generar todo el código del sistema. Adicionalmente, el *plugin* RecommendationSupport permite obtener las recomendaciones llamando a los algoritmos de recomendación a través de unas cuantas líneas de código. En Anexos (sección 9.2) se describen todos los pasos necesarios para generar las recomendaciones y se explica el código desarrollado para implementar un prototipo del sistema de recomendación.

Por otro lado, la arquitectura de la solución también se ve modificada al utilizar esta herramienta (ver Figura 6.5). En ella podemos encontrar que ahora los algoritmos de Filtrado Colaborativo también se encuentran en forma de modelo y la generación de su código está a cargo del *plugin*.

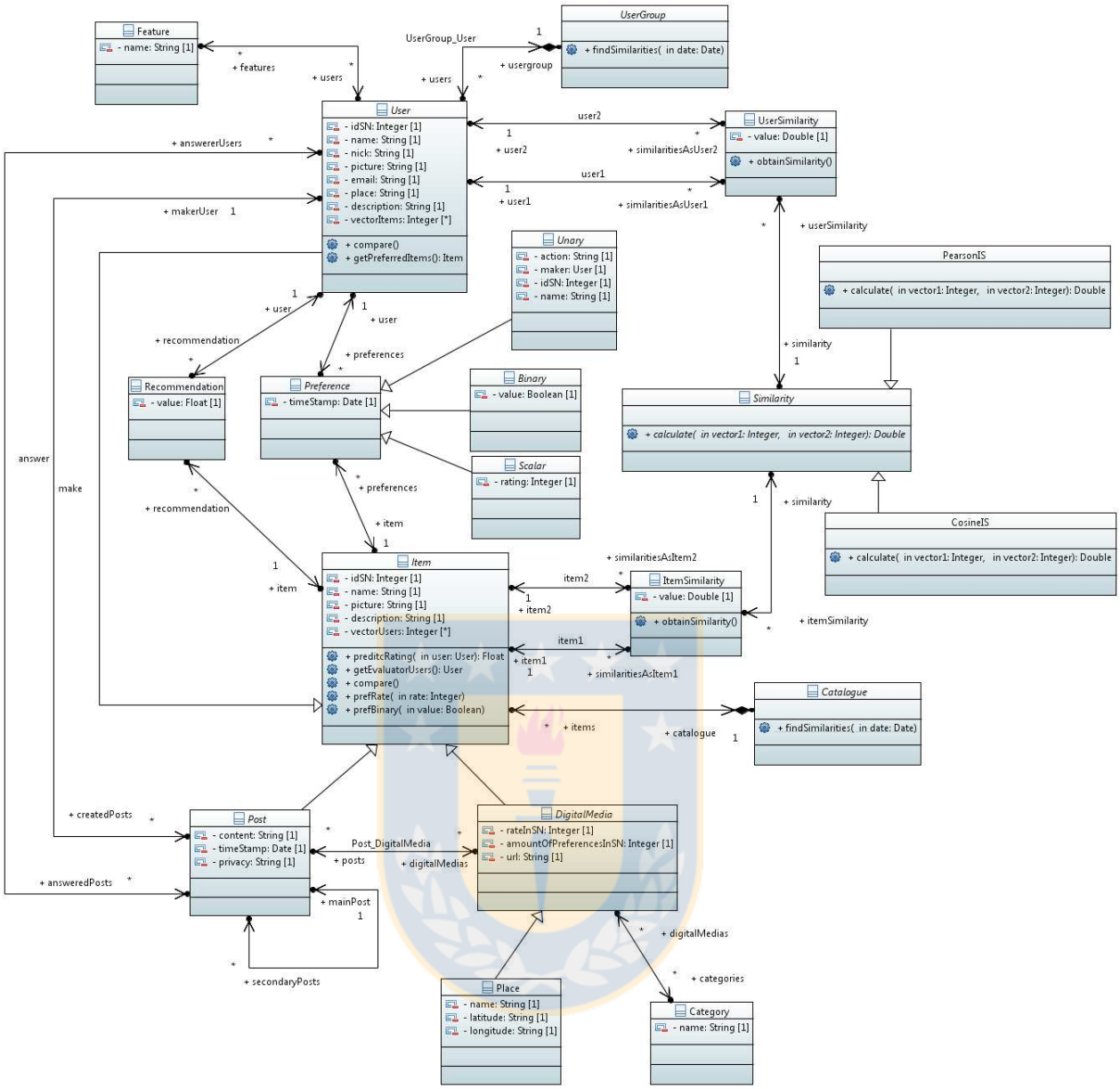

Figura 6.4: Modelo abstracto adaptado para el uso del *plugin RecommendationSupport*.

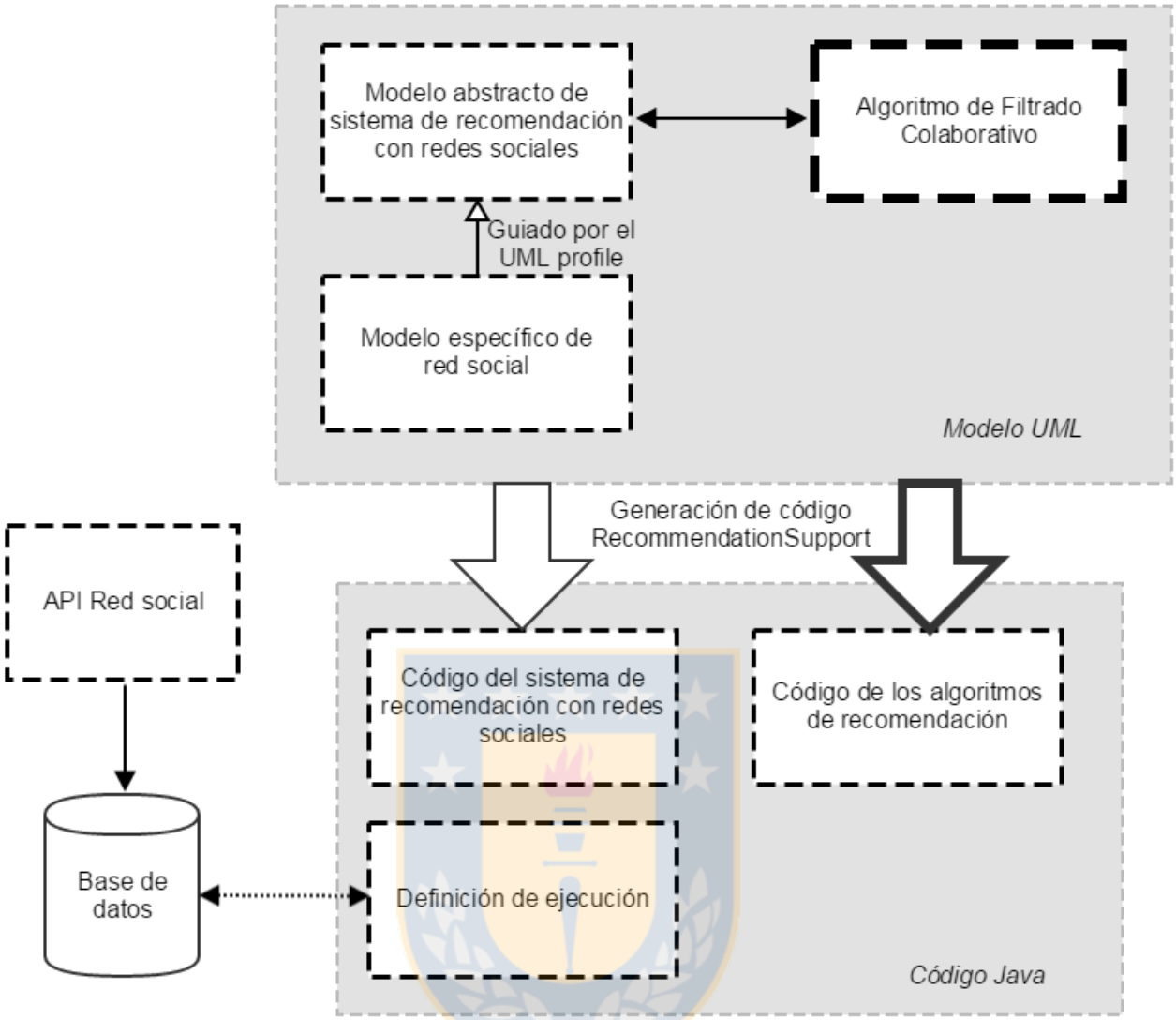

Figura 6.5: Arquitectura de la solución propuesta utilizando el *plugin* RecommendationSupport.

6.2.3. Plugin Java Code Generation

El *plugin* Java Code Generation[3] fue desarrollado por Eclipse principalmente para facilitar la obtención automática del código, en lenguaje Java, de un modelo UML. En este caso basta con utilizar el modelo asbtracto propuesto en la Figura 6.1 y el modelo específico de alguna red social para generar su código automáticamente a través esta herramienta. Sin embargo, como se mencionó anteriormente, de esta forma sólo es posible conseguir el segundo módulo. En consecuencia, para poder hacer uso de los algoritmos de recomendación imprescindibles para un sistema de recomendación, es necesario apoyarse de APIs o librerías externas. Pese a que existe una gran variedad de librerías disponibles para facilitar el desarrollo de sistemas de recomendación, para esta Memoria de Título se utilizaron los algoritmos ofrecidos por *Apache Mahout*[4], debido a que se trata de un proyecto de *Apache Software Fundation* enfocado principalmente en algoritmos de aprendizaje de máquina en áreas de Filtrado Colaborativo.

En Anexos (sección 9.3) se describen todos los pasos necesarios para generar recomendaciones usando estas dos herramientas y se explica el código desarrollado para implementar un prototipo de sistema de recomendación.

Finalmente, la arquitectura de la solución propuesta presenta otro tipo de cambios (ver Figura 6.6). En ella podemos encontrar que ahora la generación de código se utiliza sólo sobre el modelo abstracto y el modelo específico, ya que los algoritmos son incluidos directamente en el sistema de recomendación a través las librerías de Apache Mahout. En consecuencia, al definir la ejecución del sistema se debe establecer una comunicación entre la base de datos y las librerías, utilizando el código generado.

6.3. Aplicación práctica

Hasta ahora se mostró como generar un sistema de recomendación que entregue recomendaciones basándose en la información que se encuentra disponible en las redes sociales, sin embargo su aplicación práctica está vinculada a enriquecer las recomendaciones ya presentes en otros sistemas de recomendación y a aliviar algunos de los problemas detallados en el capítulo 2, como la partida en frío.

A continuación se presentan todos los pasos que debería seguir un sistema de recomendación ya existente para sacar provecho de las redes sociales. Adicionalmente, se explica cómo se podrían aliviar problemas como la partida en frío y cómo el uso de algunas redes sociales puede suponer beneficios adicionales, como la evaluación automática ítems.

6.3.1. Complementar un sistema de recomendación

A modo de ejemplo se asumirá que se cuenta con un sistema de recomendación ya implementado y que ofrece recomendaciones por medios propios, como *Netflix*[5].

En caso de que *Netflix* quisiera utilizar la información de las redes sociales para enriquecer su sistema de recomendación, debería:

[3] `https://wiki.eclipse.org/Java_Code_Generation/`
[4] http://mahout.apache.org/
[5] http://www.netflix.com/

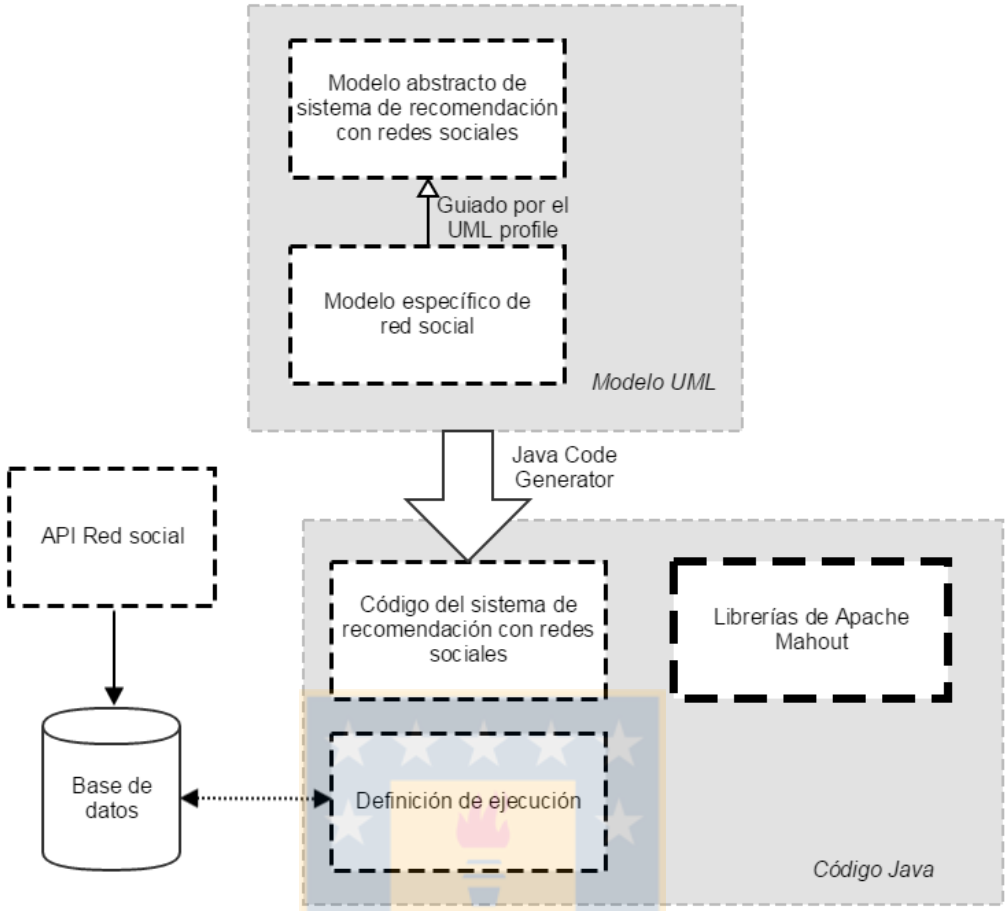

Figura 6.6: Arquitectura de la solución propuesta utilizando el *plugin* Java Code Generation.

1. Llevar a cabo un estudio de las redes sociales más utilizadas por sus usuarios.
2. Estudiar la información disponible a través de la API de la red social seleccionada.
3. Crear una base de datos que almacene la información relevante de la red social, con todo lo que ello implica. Por ejemplo, en el caso de una base de datos relacional, crear un Modelo Entidad Relación, obtener su modelo relacional, generar el código SQL, etc.
4. Crear una aplicación que recolecte los datos de los usuarios utilizando la API de la red social.
5. Almacenar la información que considere relevante en una base de datos.
6. Desarrollar un sistema capaz de administrar toda la información almacenada.
7. Realizar los cálculos asociados a los algoritmos de recomendación, ya sea utilizando los que ya se encuentran en el sistema o algún medio externo.

Por otro lado, en el caso de que *Netflix* utilizara el trabajo desarrollado en esta Memoria de Título para enriquecer su sistema de recomendación, debería:

1. Llevar a cabo un estudio de las redes sociales más utilizadas por sus usuarios.
2. Utilizar los modelos propuestos permite saber cuál es la información más relevante que debe ser consultada a través de la API de la red social seleccionada.

3. También pueden usarse los modelos propuestos junto con herramientas de transformación de modelo a texto para generar automáticamente el código SQL necesario para una base de datos relacional.

4. Crear una aplicación que recolecte los datos de los usuarios utilizando la API de la red social.

5. Poblar la base de datos con la información obtenida de la API.

6. Generar un sistema de recomendación capaz de administrar toda la información disponible, implementando los modelos propuestos a través de las herramientas mencionadas en la sección 6.2 .

7. Generar automáticamente las recomendaciones a través de las herramientas mencionadas.

Una vez generadas las recomendaciones a partir de los datos obtenidos de las redes sociales, queda en manos del sistema de recomendación principal la forma en que se utilizarán, ya que depende directamente de los algoritmos de recomendación que se encuentren implementados. Sin embargo, un posible uso se ilustra a través del siguiente ejemplo:

Ana es usuaria de *Netflix*. Allí se le recomienda una lista de películas en base a las preferencias que ha expresado con anterioridad. Estas preferencias se comparan con las de los demás usuarios a través de métodos como el Filtrado Colaborativo y se obtienen las películas que se le deben sugerir. Sin embargo, para generar estas recomendaciones no se considera si Ana conoce o no a los usuarios comparados. Por otro lado, Ana también es usuaria de *Facebook*, donde tiene una gran cantidad de amigos. Al usar los prototipos propuestos en esta Memoria de Título se puede obtener la similitud que existe entre Ana y cada uno de sus amigos en base a las acciones que han efectuado en la red social. Ahora, si alguno de los amigos más similares a Ana, Pedro, también fuese usuario de *Netflix*, *Netflix* podría utilizar esta similitud para asignar una mayor influencia a las preferencias de Pedro dentro de los algoritmos de recomendación.

6.3.2. Aliviar la partida en frío

Como se mencionó en el capítulo 2, el *cold-start* (o partida en frío) es un problema propio de los Métodos de Filtrado Colaborativo y ocurre cuando un usuario (o ítem) ingresa por primera vez a un sistema de recomendación y no se cuenta con información sobre él. Dado que el usuario no ha expresado preferencias previas sobre algún ítem, no es posible comprarlo con otros usuarios y sugerirle recomendaciones. Usualmente, para solucionar este problema se le pide al usuario que evalúe un conjunto de ítems como una forma de suplir su historial de preferencias ausente. Sin embargo, muchas veces esto resulta tedioso para el usuario, el cual busca obtener recomendaciones de la forma más simple posible.

Para aliviar este problema, se puede utilizar el trabajo propuesto en esta Memoria de Título y las similitudes obtenidas a partir de las redes sociales. De esta forma, cuando un usuario entre por primera vez a un sistema de recomendación, se pueden analizar inmediatamente los datos disponibles en las redes sociales e identificar sus amistades más similares. En consecuencia, si alguno de esos usuarios también forma parte del grupo de usuarios del sistema de recomendación, al usuario nuevo se le pueden recomendar, por ejemplo, los ítems que mejor ha evaluado su amigo.

6.3.3. Beneficios adicionales

Existen algunas oportunidades de apoyar aun más los sistemas de recomendación, pero estas oportunidades dependen directamente de que las características la red social y del sistema de recomendación que se utilicen sean compatibles. Un ejemplo de esta compatibilidad es la recomendación de libros de *Amazon*[6] con los libros registrados por *Facebook*.

Dentro de *Facebook* un usuario puede declarar libremente sus preferencias sobre una serie de libros, ya sea marcándolos con *me gusta* o asignándoles una calificación. Así, si *Amazon* utilizara los prototipos propuestos en esta Memoria de Título, podría conocer inmediatamente las preferencias de sus usuarios nuevos sobre los ítems que administra, aliviando el *cold-start* en particular para ese dominio. Sin embargo, se debe considerar que los nombres de los ítems registrados en *Facebook* no siempre presentan su nombre oficial (por ejemplo, "Moby-Dick" en vez de "Moby Dick"), por lo que debe efectuarse una verificación de correspondencia. Esto se puede realizar comparando identificadores únicos de cada dominio. Por ejemplo, en el caso de los libros, se puede buscar una correspondencia a partir de su ISBN o en el caso de películas, el identificador asignado por IMDB. Sin embargo, esta correspondencia está sujeta a que ambos sistemas cuenten con esa información, por lo que en caso de no existir, es necesario recurrir a herramientas de análisis de texto que permitan encontrar y asociar los componentes más importantes de, por ejemplo, el título de los libros.

6.4. Conclusión

En este capítulo se describieron todos los elementos necesarios para generar un prototipo de sistema de recomendación que integre la información obtenida de las redes sociales. Para esto se mostró el modelo propuesto a partir de la información recopilada en el capítulo 4, se explicó la información que es capaz de administrar y cómo este puede ser implementado a través de dos herramientas distintas para generar recomendaciones, el *plugin* RecommendationSupport y el *plugin* Java Code Generation, con apoyo de las librerías de recomendación de Apache Mahout. Se debe recordar que el único requisito necesario para poder generar recomendaciones, teniendo implementado el sistema de recomendación, es contar con información de usuarios que pueda interpretarse como una forma de evaluación sobre otro elemento.

Adicionalmente, se describieron los pasos a seguir para hacer uso de la información disponible en las redes sociales en un sistema de recomendación, tanto con las herramientas propuestas como sin ellas. De aquí, se pudo observar que hacer uso de los procedimientos descritos a lo largo de esta Memoria de Título, permite facilitar en gran medida la implementación de este tipo de sistemas de recomendación.

En el siguiente capítulo se detallan los experimentos llevados a cabo para evaluar el desempeño obtenido por las distintas implementaciones.

[6] https://www.amazon.com/

Capítulo 7

Experimentos y aplicación

En este capítulo se explican los experimentos realizados para evaluar los resultados obtenidos al implementar sistemas de recomendación a través las dos herramientas mencionadas en el capítulo anterior. Adicionalmente, se describe la información de usuario utilizada en estos experimentos y cómo es posible obtenerla de redes sociales como *Facebook* y *Twitter*.

7.1. Descripción general

Como se mostró en el capítulo anterior, se desarrollaron dos formas en las que se puede implementar un prototipo de sistema de recomendación utilizando diferentes herramientas de generación de código y de algoritmos de recomendación. Sin embargo, aunque esto demuestra de forma práctica que es posible utilizar los modelos propuestos para obtener un sistema de recomendación, no muestra qué tan viable es su uso para un desarrollador. Es por esto que se llevaron a cabo tres experimentos que permiten evaluar el proceso de implementación de un sistema de recomendación que utilice información de redes sociales, considerando como principal indicador el tiempo dedicado a esta tarea.

7.2. Experimentos

- Experimento 1: Comparar los resultados obtenidos al usar el *plugin* Recommendation-Support en el modelo propuesto en la Figura 6.1 con los resultados obtenidos utilizando sólo las librerías ofrecidas por Apache Mahout.
- Experimento 2: Comparar los resultados obtenidos al usar el *plugin* Recommendation-Support en el modelo propuesto en la Figura 6.1 con los resultados obtenidos al crear un modelo desde cero (específico para una red social) y generar tanto su código como las recomendaciones con las funcionalidades presentes en el *plugin.*
- Experimento 3: Comparar los resultados obtenidos al usar el *plugin* Recommendation-Support en el modelo propuesto en la Figura 6.1 con los resultados obtenidos al usar el *plugin* Java Code Generator y las librerías ofrecidas por Apache Mahout en el modelo propuesto en la Figura 6.4.

Se debe tener en cuenta que el objetivo principal de desarrollar estos sistema de recomendación es facilitar la implementación del segundo y tercer módulo propuestos en la sección 6.2, por lo que para estos experimentos se consideró que toda la información necesaria ya fue obtenida de una red social y se almacenó adecuadamente en una base de datos. Por otro lado, para analizar el tiempo asociado a cada experimento se asume que todos complementos y *plugins* se encuentran correctamente instalados y que tanto librerías como los códigos necesarios para ejecutar los prototipos se encuentran en manos del desarrollador.

7.2.1. Datos utilizados

Durante el desarrollo de los experimentos se utilizaron los *datasets* ofrecidos por la Universidad de Stanford[1], donde se consultaron los datos disponibles a cerca de las redes sociales *Facebook*[2] y *Twitter*[3]. La información obtenida de este conjunto de datos se centra principalmente en las relaciones establecidas por los usuarios y algunas de sus características personales. Sin embargo, para proteger la información de estos usuarios, se han modificado los ID que mantenían dentro de la red social, se ha omitido su nombre y las características fueron estandarizadas. Por ejemplo, en vez de que la característica "amistoso" esté asociada a un grupo de usuarios, se puede identificar que la característica "atributo1" está asociada a ellos. Adicionalmente, no se cuenta con información acerca de ítems dentro de las redes

[1] https://snap.stanford.edu/data/index.html
[2] https://snap.stanford.edu/data/egonets-Facebook.html
[3] https://snap.stanford.edu/data/egonets-Twitter.html

sociales (como publicaciones), ya que por un lado *Facebook* mantiene privada ese tipo de información y por el otro *Twitter* solicitó directamente a la Universidad de Stanford que quitaran la información relacionada con *tweets*. Sin embargo esto no resulta un problema, debido a que el modelo propuesto permite utilizar a los propios usuarios como ítems para calcular similitudes y generar recomendaciones.

Pese a lo anterior, y con el fin de demostrar que los sistemas de recomendación obtenidos también generan recomendaciones sobre ítems, como publicaciones, en Anexos (sección 9.4) se muestra a modo de ejemplo un conjunto de datos obtenidos de *Facebook* y se especifica el código a través del cual fueron consultados a la API.

Adicionalmente, en Anexos (sección 9.5) se muestra un ejemplo del código utilizado para realizar las consultas a la base de datos una vez que se ha almacenado la información de los *datasets* mencionados anteriormente.

7.2.2. Experimento 1

Para el primer experimento se propuso tomar los procedimientos establecidos para implementar el prototipo propuesto en la sección 6.2.1 (el modelo propuesto en la Figura 6.4 apoyado con la generación de código y algoritmos de recomendación del *plugin* RecommendationSupport) y compararlos con lo que pasaría si se intentara desarrollar un sistema de recomendación que use información de las redes sociales utilizando sólo las librerías ofrecidas por Apache Mahout.

Para llevar este experimento a cabo, se debió consultar a la base de datos la información asociada a las relaciones de los usuarios y pasarla directamente a las librerías de Apache Mahout, para lo cual se utilizó el código presentado en Anexos (sección 9.6.1). De esta forma, se contrastaron los resultados obtenidos por la solución propuesta en la sección 6.2.2 con la solución descrita por la arquitectura de la Figura 7.1, para contabilizar el tiempo dedicado a esta tarea y analizar el sistema de recomendación resultante.

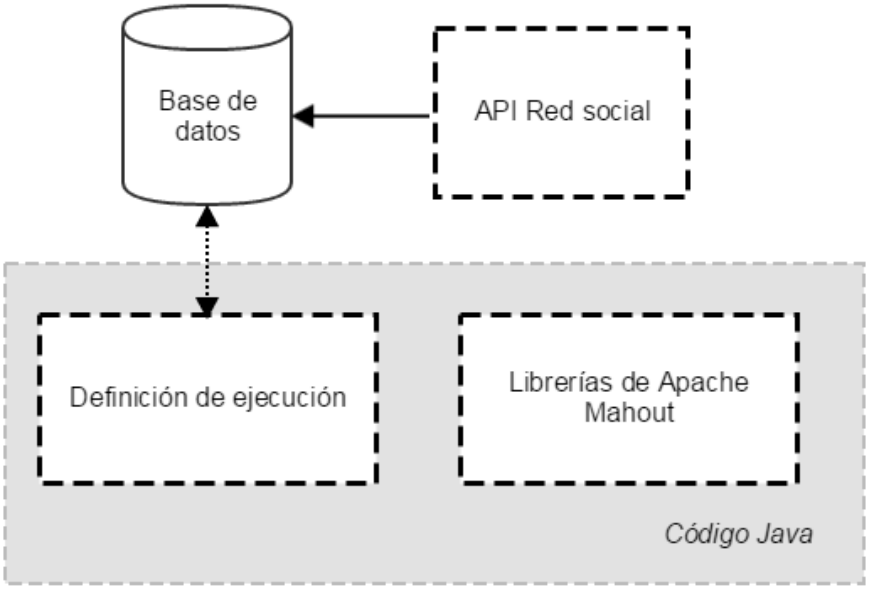

Figura 7.1: Arquitectura de una solución que sólo utiliza la librerías de Apache Mahout.

7.2.3. Experimento 2

Para el segundo experimento se propuso tomar los procedimientos establecidos para implementar el prototipo propuesto en la sección 6.2.1 (el modelo propuesto en la Figura 6.4 apoyado con la generación de código y algoritmos de recomendación del *plugin* RecommendationSupport) y compararlos con los resultados registrados al observar qué ocurriría al intentar desarollar un modelo exclusivamente para *Facebook*, pero siendo apoyado por la generación de código y algoritmos de recomendación presentes en el *plugin* RecommendationSupport. De esta forma, se contrastaron los resultados obtenidos por la solución propuesta en la sección 6.2.3 con la solución descrita por la arquitectura de la Figura 7.2.

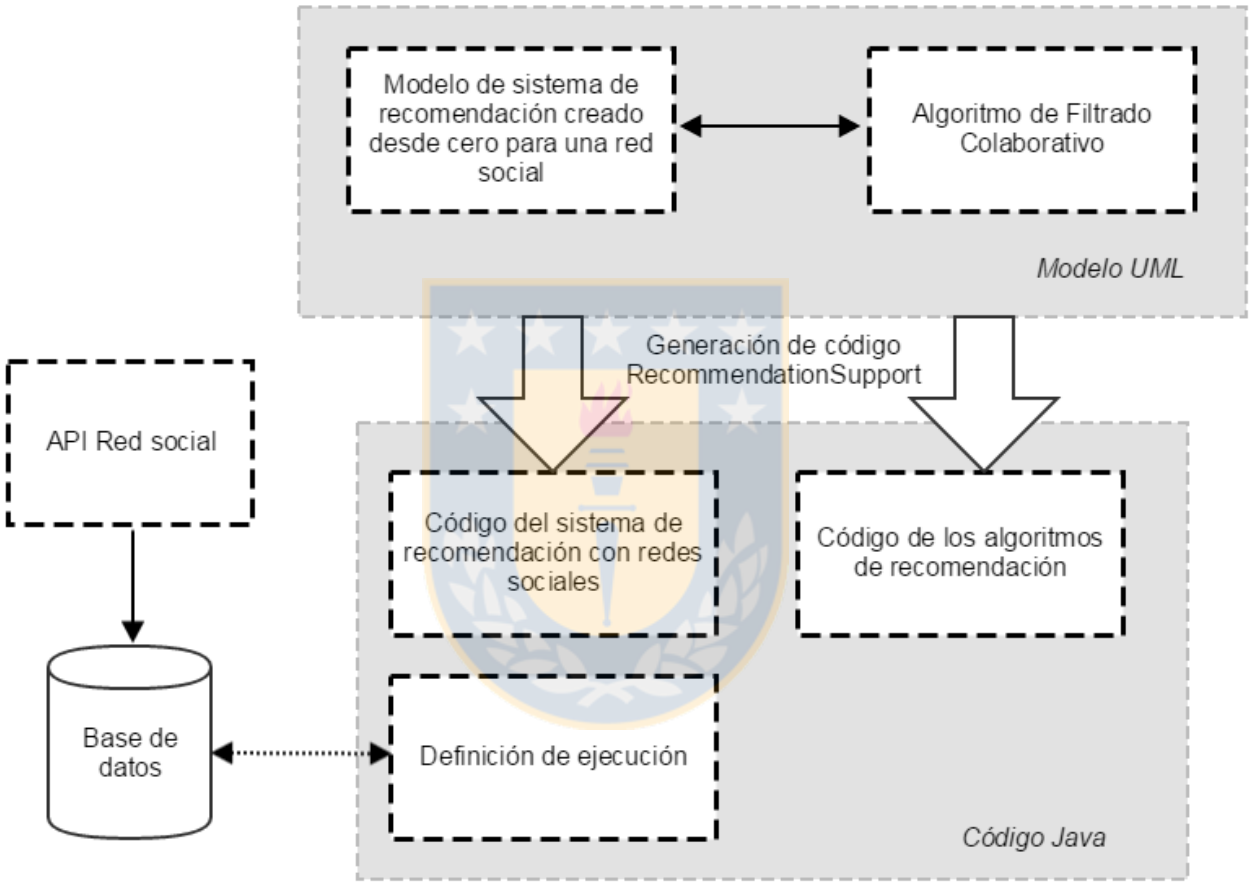

Figura 7.2: Arquitectura de una solución que crea un modelo desde cero para una red social en particular.

Para llevar a cabo este experimento se contabilizó el tiempo requerido para desarrollar desde cero el modelo necesario para generar un sistema de recomendación. En consecuencia, se consideró que, como mínimo, el diagrama debe presentar la forma propuesta en la Figura 7.3.

7.2.4. Experimento 3

Para el tercer experimento se propuso comparar los procedimientos establecidos para implementar los dos prototipos propuestos en la sección 6.2. Por un lado, utilizando el *plugin* RecommendationSupport para generar el código del modelo propuesto en la Figura 6.4 y los algoritmos que tiene incorporados para generar las recomendaciones. Y por el otro lado,

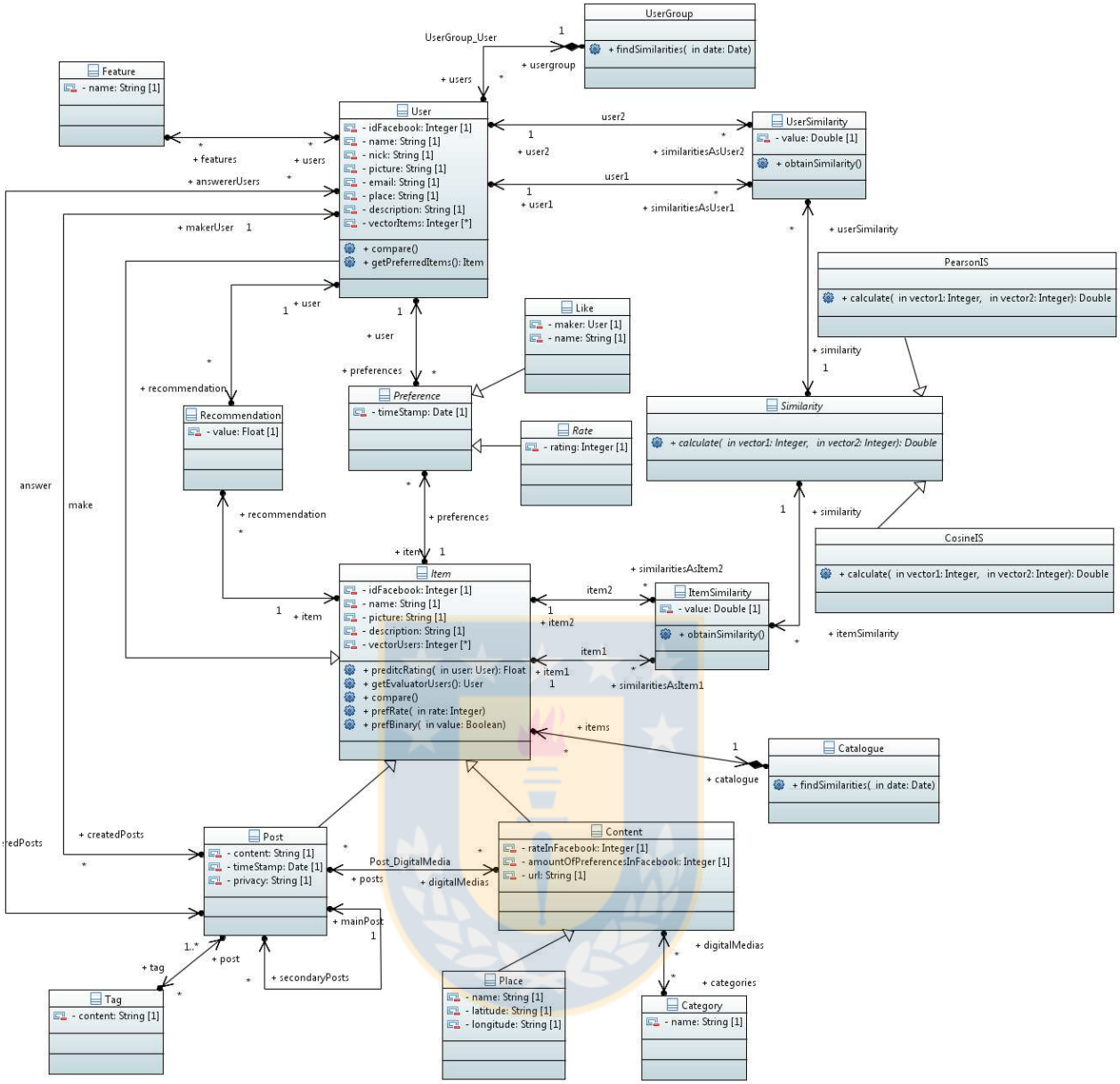

Figura 7.3: Ejemplo de modelo desarrollado desde cero.

utilizando el *plugin* Java Code Generator para generar el código del modelo propuesto en la Figura 6.1, complementándolo con las librerías de recomendación ofrecidas por Apache Mahout.

Para llevar a cabo este experimento se contabilizó el tiempo destinado a la implementación de cada prototipo de sistema de recomendación y se compararon las características presentes en cada uno de ellos.

7.3. Resultados

En cuanto a tiempo, se identificaron los resultados expuestos en la Tabla 7.1. Adicionalmente, se presenta la cantidad de líneas de código requeridas para el funcionamiento de cada prototipo (correspondiente a la Definición de ejecución descrita en la sección 6.2.1).

Prototipo	**Tiempo**	**Código**
Modelo propuesto + RS.	9 minutos.	103 líneas.
Modelo propuesto + JCG + Apache Mahout.	8 minutos.	117 líneas.
Sólo librerías de Apache Mahout.	3 minutos.	94 líneas.
Modelo desde cero.	Más de una hora.	103 líneas.

Cuadro 7.1: Tiempo requerido para implementar los diferentes prototipos.

- **Modelo propuesto + *plugin* RecommendationSupport:** Toma aproximadamente 9 minutos obtener un sistema de recomendación. Este tiempo se destina principalmente a importar los modelos y diagramas creados, generar su código y realizar las modificaciones pertinentes. Como resultado se obtienen 19 clases correspondientes al modelo abstracto, 9 clases correspondientes al modelo específico, una clase correspondiente a los algoritmos de recomendación y una clase correspondiente a la Definición de ejecución, la cual consta de 103 líneas de código.

- **Modelo propuesto + *plugin* JCG + Apache Mahout:** Toma aproximadamente 8 minutos obtener un sistema de recomendación. Este tiempo se destina principalmente a agregar las librerías requeridas por Apache Mahout, a generar la estructura interna del proyecto y a la manipulación del código generado por el *plugin* Java Code Generator. Como resultado se obtienen 15 clases correspondientes al modelo abstracto, 9 clases correspondientes al modelo específico y una clase correspondiente a la Definición de ejecución, la cual consta de 117 líneas de código.

- **Sólo librerías de Apache Mahout:** Toma aproximadamente 3 minutos obtener recomendaciones. Este tiempo se destina principalmente a transformar los datos de la red social al formato admitido por las librerías de Apache Mahout. Para esta implementación se requiere sólo una clase, la cual correspondiente a la Definición de ejecución y consta de 94 líneas de código.

- **Modelo desde cero:** Toma entre 30 minutos a una hora desarrollar un modelo que describa el funcionamiento de un sistema de recomendación y de una red social a la vez. A esto se debe agregar los 9 minutos que toma generar su código y llevar a cabo las modificaciones pertinentes. Como resultado se obtienen 19 clases correspondientes al modelo y una clase correspondiente a la Definición de ejecución, la cual consta de 103 líneas de código. Adicionalmente, se debe tener presente que si se desarrolla un modelo de este tipo, se debe dedicar tiempo al estudio de la red social seleccionada para identificar sus principales componentes y características, por lo cual el tiempo requerido aumenta considerablemente.

Dado los experimentos realizados y los resultados obtenidos en a través de su desarrollo, se identifican las siguientes características:

1. Experimento 1. Se observa que, pese a que utilizar únicamente las librerías de Apache Mahout requiere un tercio del tiempo que el requerido por implementar el modelo propuesto mediante el *plugin* RecommendationSupport, no resulta viable. Esto se debe a que al utilizar únicamente las librerías de Apache Mahout, aunque se obtienen las recomendaciones esperadas, no se genera un sistema que se haga cargo de administrar la información utilizada ni de los resultados obtenidos. Por otro lado, al utilizar el *plugin* RecommendationSupport y el modelo propuesto, se crea un sistema de recomendación completo que, además de generar las recomendaciones deseadas, facilita el manejo y almacenamiento de los datos.

2. Experimento 2. Se observa que, aunque desarrollar un modelo desde cero permitiría obtener un sistema de recomendación completo, requiere de demasiado tiempo para ser implementado. Adicionalmente, el problema del tiempo requerido se incrementa considerablemente si se considera que este modelo fue desarrollado exclusivamente para una red social, por lo que desarrollar un sistema de recomendación de la misma forma pero para otra red social requeriría el mismo tiempo descrito anteriormente, lo que crea un grave problema de portabilidad. En contraste, los sistemas de recomendación generados al utilizar el modelo propuesto y el *plugin* RecommendationSupport, además de ser completos, están diseñados para soportar cualquier tipo de red social, por lo que el tiempo empleado en su desarrollo no varía.

3. Experimento 3. Se observa que a través de los dos métodos se puede obtener un sistema de recomendación equivalente en términos de administración de la información y que su implementación toma aproximadamente la misma cantidad de tiempo.

En consecuencia, los dos métodos presentados en la sección 6.2 resultan igual de viables si se requiere generar un sistema de recomendación. Sin embargo, las necesidades particulares de un desarrollador determinarán cuál resulta una mejor alternativa. Esto se debe a que herramientas como las librerías de Apache Mahout no permiten modificaciones y se limitan al servicio ofrecido. De esta forma, un desarrollador se encuentra incapacitado de efectuar modificaciones a los algortimos en caso de necesitarlo. Por otro lado, el uso del *plugin* RecommendationSupport permitiría realizar modificaciones según los requerimiento que se presenten. Así, sería posible desarrollar nuevos módulos que apoyen el sistema de recomendación sin depender de las limitaciones establecidas por terceros.

7.4. Conclusión

Durante este capítulo se presentaron una serie de experimentos realizados con el objetivo de probar la viabilidad de la generación de sistemas de recomendación. A partir de ellos, se identificó que los prototipos propuestos presentan un buen desempeño en cuanto al tiempo que se requiere para implementarlos, pero que si lo que se busca es poder personalizar el comportamiento del sistema, es mejor implementar el modelo propuesto en la Figura 6.4 utiliando en *plugin* RecommendationSupport.

Capítulo 8

Conclusiones

En este capítulo se resume el todo el trabajo realizado y se explica cómo en base a este se consiguieron los objetivos presentados en la sección 1.3. Adicionalmente, se presentan las limitaciones identificadas en el trabajo propuesto y los posibles trabajos futuros.

8.1. Conclusiones finales

Debido al continuo y desenfrenado aumento de la información disponible en Internet, se hacen cada vez más necesarios los sistemas de recomendación que permitan encontrar fácilmente las necesidades de cada usuario sin un tedioso proceso de búsqueda. Actualmente se identifica una gran variedad de estos sistemas de recomendación y aplicados a múltiples dominios (tales como lugares de interés, libros, películas, etc.). Sin embargo, la mayoría genera sus sugerencias utilizando únicamente la información recopilada por ellos mismos, lo cual deja fuera una gran cantidad de potenciales fuentes de datos. Por otro lado, resulta evidente que las redes sociales se han posicionado como importantes pilares dentro de Internet y que su uso no parece declinar, ya que se presentan como una herramienta accesible para todas las personas, capaces de conectar a una gran cantidad de usuarios y de cubrir todo tipo de necesidades. En consecuencia, se propuso desarrollar una forma que facilitara el aprovechamiento de las redes sociales para mejorar algunos aspectos visibles en los sistemas de recomendación.

De esta forma, se cumplió con el objetivo de proponer formas de enriquecer los sistemas de recomendación con la información de las redes sociales, a través de un estudio de los métodos más utilizados por estos para generar sugerencias y de la información a la que ofrecen acceso las redes sociales a través de sus respectivas APIs. Adicionalmente, se pudo desarrollar un modelo UML que considera las características más importantes identificadas tanto en las redes sociales como en los sistemas de recomendación. Finalmente, se construyeron distintos prototipos de sistemas de recomendación que permitieron comparar su desempeño e identificar los beneficios de las distintas formas de desarrollo.

En resumen, el modelo propuesto en esta Memoria de Título ha resultado ser una exitosa herramienta a la hora de unir los dos elementos descritos previamente, pudiendo facilitar la creación de un sistema de recomendación capaz de integrar la información administrada por las redes sociales. No obstante, se identificaron las siguientes limitaciones:

1. Aunque hacer uso de la información disponible en las redes sociales permite generar recomendaciones instantáneamente, no es posible abandonar los procedimientos actuales destinados al ingreso de nuevos usuarios. Esto se debe a que, aunque las redes sociales tienen una profunda llegada a la población, no todos los usuarios hacen uso de estas, por lo que de todas formas se deben implementar los métodos tradicionales utilizados para solventar problemas como la partida en frío de los usuarios.

2. Se debe tener especial cuidado a la hora de seleccionar las redes sociales de las cuales se desea obtener información, debido a que algunas de ellas pueden ofrecer información que pueda malinterpretarse (por ejemplo, las relaciones entre usuarios de redes sociales destinadas a conocer personas nuevas en forma aleatoria podrían no representar la personalidad de sus usuarios). Adicionalmente, se debe considerar que no todas las redes sociales han desarrollado APIs.

3. Dado que la mayor parte de la información rescatada de las redes sociales está relacionada con contenido creado personalmente por los usuarios, muchas veces carecerán de una estructura estandarizada que permita sacar provecho de esta información. Ejemplos de esto son que algunos elementos registrados en *Facebook* (como libros o películas) carecen de un identificador que permita asegurar su correspondencia con ítems registrados en otro sistema o que las etiquetas presentes en redes sociales como *Twitter* no sigan un

protocolo y sean utilizadas de diversas formas por los distintos usuarios, consiguiendo distintos significados a las mismas etiquetas (por ejemplo, a expresiones sarcásticas).

4. En el caso de un sistema de recomendación que utilice Métodos de Filtrado Colaborativo, aunque un usuario cumpla con características como pertenecer a la red social seleccionada, que haya expresado sus preferencias y establecido relaciones con otros usuarios dentro de ella, si ninguna de sus amistades forma parte también del sistema de recomendación no será posible sacar provecho a las similitudes obtenidas.

Sin embargo, experimentar con el uso de redes sociales como complemento de los sistemas de recomendación ha permitido identificar una serie de beneficios, donde la facilidad de obtener similitudes entre usuarios abre las puertas a muchos otros. Entre ellos, se pudo apreciar que utilizar una fuente de datos como las redes sociales ofrece las herramientas para aliviar problemas típicos de los sistemas de recomendación como es la partida en frío. Este problema se puede abordar utilizando las similitudes entre usuarios para que cuando un usuario ingrese por primera vez al sistema, se le ofrezcan recomendaciones en base a las preferencias de sus amistades.

Adicionalmente, el desarrollo de un sistema de recomendación a partir del modelo UML propuesto permite administrar la información más relevante de cualquier red social, debido a que el estudio realizado permitió identificar las características comunes más importantes de estas. Por otro lado, el uso de este modelo también permite especificar el comportamiento más específico de una determinada red social, en caso de ser necesario. De esta forma, se puede especificar el nombre de las evaluaciones disponibles, sobre qué elementos son permitidas y el comportamiento de las relaciones entre usuarios. Finalmente, en el eventual caso de requerir integrar las características de nuevas o innovadoras herramientas que no fueron consideradas durante el modelado, basta con hacer pequeñas modificaciones y el modelo seguirá siendo funcional. En consecuencia, se obtuvo un método de desarrollo sumamente portable entre las diferentes redes sociales, lo cual facilita su aplicación en sistemas de recomendación ya existentes.

8.2. Trabajos futuros

Como trabajos futuros que complementen esta Memoria de Título, se sugieren los siguientes:

- **Estudio de otros algoritmos**. El trabajo propuesto ha resultado exitoso a la hora de aprovechar la información de las redes sociales para mejorar un sistema de recomendación. Sin embargo, no se sacó provecho a toda la información que podría usarse para generar recomendaciones. Esto se debe a que el estudio estaba enfocado a utilizar algoritmos y librerías de Filtrado Colaborativo, por lo que los modelos propuestos y los procedimientos de implementación se adaptaron a estos requerimientos. De esta forma, aunque el modelo también es capaz de administrar la información requerida por los demás métodos descritos en el capítulo 2, no ofrece el mismo grado de especialización. En consecuencia, llevar a cabo un estudio de los requerimientos específicos de otros algoritmos (como Métodos Basados en Contenido) permitiría identificar si algunas características del modelo deben modificarse para facilitar la generación de otro tipo de recomendaciones.

- **Complementar herramientas de desarrollo**. Durante el desarrollo de este trabajo se utilizaron dos herramientas para generar los sistemas de recomendación: el *plugin* RecommendationSupport y el *plugin* Java Code Generation con librerías externas de Filtrado Colaborativo. Si se aplicara esta misma metodología a los resultados esperados del punto anterior, se encontraría la limitante que tanto el *plugin* como las librerías de Apache Mahout sólo cuentan con algoritmos de Filtrado Colaborativo. En consecuencia, para generar recomendaciones utilizando otro tipo de información, resulta necesario efectuar un estudio con el fin de actualizar estas dos herramientas. Por un lado, se puede personalizar el *plugin* RecommendationSupport de tal forma que se facilite la implementación de los nuevos algoritmos. Mientras que por otro, se pueden investigar si existen librerías ofrecidas por terceros que faciliten la generación de recomendaciones a partir de la información seleccionada. El resultado serían herramientas de desarrollo mucho más completas, capaces aprovechar la totalidad de la información disponible en las redes sociales.

- **Nuevos experimentos**. En este trabajo, los experimentos se desarrollaron enfocándose en el tiempo que toma implementar un prototipo de sistema de recomendación sin considerar la calidad de las recomendaciones generadas. Esto se debe principalmente a que la precisión de las recomendaciones depende directamente del módulo que contiene los algoritmos y/o librerías de recomendación, el cual fue tratado como caja negra. De esta forma, llevar a cabo un estudio que se interiorice en este módulo e identifique qué métodos generan recomendaciones de mayor calidad (dependiendo de la información con que se cuente), permitiría implementar un mejor sistema de recomendación con la misma facilidad propuesta en capítulos anteriores.

- **Aprovechar características de contexto**. Actualmente, algunas redes sociales han comenzado a implementar un nuevo sistema de evaluación asociado a las emociones que generan los ítems. Por ejemplo, es posible evaluar un comentario como “me divierte”, “me alegra”, “me entristece”, “me gusta”, etc. Este nuevo tipo de evaluación es soportado por el modelo propuesto en esta Memoria de Título, debido a que se puede considerar como un tipo evaluación escalar. Sin embargo, también ofrece nuevas características de contexto a ser explotadas. De esta forma, efectuar un estudio sobre cómo varía la calidad de las recomendaciones si se consideran estas características de los usuarios (como su estado de ánimo, las emociones que les generan los diferentes ítems, la edad de los usuarios, la estación del año o su ubicación geográfica) podría ofrecer nuevas formas de enriquecimiento para los sistemas de recomendación.

Capítulo 9

Anexos

En este capítulo se profundiza en la documentación disponible sobre diversos tipos de información y procedimientos mencionados en capítulos anteriores, pero que por razones de continuidad y brevedad no fueron tan extensamente detallados. A continuación se puede encontrar información relacionada a los datos obtenidos de las redes sociales y a las herramientas, procedimientos y códigos[1] utilizados durante la implementación de los prototipos de sistemas de recomendación.

[1]Disponibles en la dirección adweb.inf.udec.cl/snrec/experiments/

9.1. Información disponible

En esta sección se detalla la información, disponible en las redes sociales a través de sus respectivas APIs, que fue descrita en forma más general en el capítulo 5. Cabe señalar que a través de las distintas APIs es posible obtener una gran cantidad de información, sin embargo no toda resulta útil para un sistema de recomendación. A continuación se detalla toda la información disponible que se consideró que puede resultar útil de alguna manera para implementar alguno de los métodos de recomendación descritos en el capítulo 2.

9.1.1. Facebook

Información personal

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de cuál usuario se está consultando.
Nombre.	Nombre completo, primer nombre, apellido, etc.
Edad.	Rango de edad al que pertenece el usuario.
Fecha de nacimiento.	
Correo electrónico.	
Descripción.	Descripción personal que ha declarado el usuario. Generalmente se trata de una pequeña autobiografía.
Educación.	Lista con todos los lugares en los que el usuario ha declarado haber estudiado.
Trabajo.	Lista con todos los lugares y cargos donde el usuario afirma haber trabajado.
Lugar de nacimiento.	
Lugar de residencia.	

Cuadro 9.1: Información personal del usuario.

Lugares

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué lugar se está consultando.
Fecha.	Fecha en la que se declaró la existencia del lugar dentro de *Facebook*.
Nombre.	Nombre del lugar.
Dirección.	País, ciudad y calle en la que se encuentra el lugar.
Coordenadas.	Latitud y longitud de la ubicación del lugar.

Cuadro 9.2: Información asociada a los lugares reconocidos por *Facebook*.

Relaciones con usuarios

Entidad	Descripción
Lista de amigos.	Cantidad de usuarios que son amigos del usuario autenticado.
Grupos de amigos.	*Facebook* permite crear grupos para reunir a cierta cantidad de usuarios (los cuales pueden o no ser amigos del usuario fundador del grupo). Se puede obtener el ID de todos los grupos donde el usuario autenticado es miembro. Adicionalmente, a través del ID de un grupo se puede consultar sobre su nombre, descripción, foto, nombre del fundador y tipo de privacidad.
Relaciones explícitas.	Además de las relaciones de amistad, dentro de *Facebook* es posible declarar si un usuario es, por ejemplo, familiar de otro. Así, a través de la API es posible obtener el nombre e ID del otro usuario, además del tipo de relación que poseen.
Mensajes privados.	Es posible acceder al ID de cada una de las conversaciones entabladas por el usuario autenticado, además del nombre e ID de cada uno de sus participantes y del contenido de los mensajes.

Cuadro 9.3: Información asociada a las relaciones establecidas por el usuario.

Publicaciones

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué publicación se está consultando.
Usuario creador.	Nombre e ID del usuario que realizó la publicación.
Contenido.	Texto contenido en la publicación. Puede incluir enlaces, imágenes u otros elementos.
Compartido.	Número de veces que ha sido compartido por otros usuarios.
Imagen.	Enlace directo a la imagen en caso de que la publicación contenga alguna.
Enlace.	Enlace en forma de texto en caso de que la publicación contenga alguno.
Descripción.	Descripción asociada al sitio web de destino, en caso de que la publicación incluya un enlace.
Privacidad.	Tipo de privacidad establecida para la publicación: pública, visible sólo por amigos, etc.
Me gusta.	Cantidad de *me gusta* recibidos por la publicación y el ID de los usuarios que los realizaron.
Comentarios.	Contenido de los comentarios o respuestas hechas a la publicación, además de los ID de los usuarios que los hicieron.
Lugares.	Lugar (información completa) donde fue elaborada la publicación, en caso de que el usuario lo haya declarado.

Cuadro 9.4: Información asociada a las publicaciones existentes en el muro del usuario.

Videos/Películas

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué elemento se está consultando.
Título.	Nombre registrado en *Facebook* para el video.
Tipo.	Puede ser de tipo *movie*, *tv show*, *video* o *tv episode*.
Nota.	Promedio de todas las evaluaciones asignadas por los usuarios de *Faceook*.
Categorías.	Categorías asociadas al video.
Trama.	Descripción general.
Premios.	Lista de premios obtenidos.
Director.	
Foto.	Portada.
Géneros.	Lista de géneros asociados.
Me gusta.	Cantidad de *me gusta* recibidos por parte de los usuarios.
Fecha.	Fecha de lanzamiento.
Actores.	Lista de actores y actrices participantes.
Escritor.	
Estudio.	Estudio a cargo de la producción.

Cuadro 9.5: Información de las películas registradas en *Facebook*.

Libros

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué libro se está consultando.
Título.	Nombre registrado en *Facebook* para el libro.
Nota.	Promedio de todas las evaluaciones asignadas por los usuarios de *Faceook*.
Categorías.	Categorías asociadas al libro.
Descripción.	Descripción general del libro.
Géneros.	Lista de géneros asociados.
Me gusta.	Cantidad de *me gusta* recibidos por parte de los usuarios.
Fecha.	Fecha de lanzamiento.
Escritor.	

Cuadro 9.6: Información de los libros registrados en *Facebook*.

Música

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué elemento se está consultando.
Nombre.	Nombre registrado en *Facebook*.
Miembros.	Lista de miembros del grupo musical.
Biografía.	Descripción general.
Géneros.	Lista de géneros asociados.
Me gusta.	Cantidad de *me gusta* recibidos por parte de los usuarios.
Ciudad de origen.	Ciudad donde se fundó la agrupación.
Disquera.	.

Cuadro 9.7: Información de los músicos y bandas registradas en *Facebook*.

9.1.2. Twitter

Información personal

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de cuál usuario se está consultando.
Nombre.	Nombre real declarado por el usuario dentro de la red social.
Cuenta.	Nombre de la cuenta (puede considerarse el apodo o *nick* del usuario).
Imagen.	Imagen que se muestra en el perfil del usuario.
Descripción.	Descripción breve destinada a mencionar las características más relevantes del usuario.
Seguidores.	Número de personas que *siguen* al usuario.
Seguidos.	Número de personas *seguidas* por el usuario.
Lugar de residencia.	

Cuadro 9.8: Información personal del usuario.

Información de cada *tweet*

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué *tweet* se está consultando.
En respuesta.	Se puede consultar si un determinado *tweet* se publicó en respuesta a otro y, en caso d ser así, obtener el ID de ese *tweet*.
Medio de publicación.	A través de qué aplicación y/o dispositivo se publicó el *tweet*.
Lugar.	En caso de estar asociado a un lugar, es posible obtener su nombre, ciudad, país y coordenadas.
Favoritos.	Número de veces que ha sido marcado como *favorito* por otros usuarios.
Retweets.	Número de veces que ha sido *retweeteado* por otros usuarios.

Cuadro 9.9: Información asociada a un *tweet*.

Relación con usuarios

Entidad	Descripción
Usuarios seguidos.	Lista con los ID de todos los usuarios *seguidos* por un determinado usuario.
Usuarios seguidores.	Lista con los ID de todos los *seguidores* de un determinado usuario.
Mensajes privados.	Si se trata de un usuario autenticado, es posible acceder a la información de emisor, del receptor y al contenido de los mensajes de todas las conversaciones entabladas.
Listas de usuarios.	Nombre de las listas de usuarios que han sido creadas por un determinado usuario.
Listas suscritas.	Nombre de todas las listas de usuarios a las cuales el usuario está suscrito.
Usuarios suscritos.	Nombre de todos los usuarios suscritos a las las listas de usuarios creadas por el usuario autenticado.
Búsquedas guardadas.	Luego de efectuar una búsqueda dentro de *Twitter*, se puede optar por guardarla. Es posible obtener todas las búsquedas guardadas por un determinado usuario.

Cuadro 9.10: Información asociada a las relaciones establecidas por el usuario.

Grupos de publicaciones

Entidad	Descripción
Menciones.	Lista con *tweets* más recientes en los que se ha mencionado a un usuario autenticado.
Tweets recientes.	Lista con los *tweets* más recientes publicados por un determinado usuario.
Muro.	Lista de todos los *tweets* y *retweets* mostrados en el muro de un usuario autenticado.
Retweets.	Lista de todos los *tweets* publicados por un usuario autenticado y que además han sido *retweeteados* por otros usuarios, incluyendo sus respectivos nombres e ID.
Favoritos.	Lista con todos los *tweets* que un determinado usuario ha marcado como *favoritos*.
Tweets de una lista.	Todos los *tweets* publicados por los usuarios pertenecientes a una lista de usuarios creada por un usuario autenticado.
Búsquedas guardadas.	Luego de efectuar una búsqueda dentro de *Twitter*, se puede optar por guardarla. Es posible obtener todas las búsquedas guardadas por un determinado usuario.

Cuadro 9.11: Información asociada a las listas de publicaciones.

9.1.3. Linkedin

Información personal

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de cuál usuario se está consultando.
Nombre.	Nombre completo declarado por el usuario.
Título profesional.	
Ubicación.	Lugar actual de residencia.
Conexiones.	Número de conexiones establecidas en la red social.
Industrias.	Lista de las industrias en las que el usuario declaró ser miembro.
Especialidades.	Lista de las especialidades declaras por el usuario.
Imagen.	Imagen utilizada en el perfil del usuario.
Correo electrónico.	
Asociaciones.	Lista con las asociaciones en las que el usuario forma parte.
Intereses.	Intereses personales declarados por el usuario.
Publicaciones.	Lista con todas las publicaciones realizadas.
Lenguajes.	Detalle del conocimiento del usuario sobre los distintos idiomas.
Habilidades.	Lista de las habilidades declaradas por el usuario.
Educación.	Lista con todas las instituciones donde el usuario ha declarado haber estudiado. Para cada institución se detalla su campo de estudio, la fecha de inicio, la fecha de termino y el grado académico obtenido.
Cursos.	Lista de todos los cursos tomados por el usuario.
Voluntariados.	Lista de todos los voluntariados realizados por el usuario.
Puesto actual.	Cargo laboral actual ejercido por el usuario.
Puestos anteriores.	Cargos laborales anteriores ejercidos por el usuario.
Usuarios recomendados.	Lista de todos los usuarios que han sido recomendados.
Entidades seguidas.	Lista con todas las personas, empresas e industrias seguidas por el usuario.
Entidades sugeridas.	Lista con todas las personas, empresas e industrias sugeridas para el usuario.
Trabajos seguidos.	Lista con los trabajos seguidos por el usuario (representan su interés sobre aquellos tipos de trabajo).
Fecha de nacimiento.	
Usuarios vistos.	Lista con todos los perfiles de otros usuarios que han sido vistos por el usuario.
Premios y honores.	Lista con todos los premios y honores recibidos por el usuario.

Cuadro 9.12: Información personal del usuario.

Información de compañías

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué compañía se está consultando.
Nombre.	Nombre de la compañía.
Tipo.	Tipo de compañía (dentro de un grupo de opciones predefinidas).
Sitio web.	
Industria.	Tipo de industria a la que pertenece.
Empleados.	Rango de la cantidad de empleados que posee.
Especialidades.	Especialidades declaradas por la compañía.
Descripción.	Descripción general sobre la compañía.
Fundación.	Fecha en que fue fundada la compañía.
Seguidores.	Número de usuarios que siguen a la compañía dentro de la red social.

Cuadro 9.13: Información de las compañía registradas dentro de *Linkedin*.

9.1.4. Instagram

Información personal

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de cuál usuario se está consultando.
Cuenta.	Nombre asociado a la cuenta del usuario.
Nombre.	Nombre completo real del usuario.
Imagen.	Foto usada en el perfil del usuario.
Biografía.	Descripción personal breve declarada por el usuario.
Medios publicados.	Número del total de medios publicados por el usuario .
Usuarios seguidos.	Número del total de usuarios *seguidos*.
Seguidores.	Número del total de personas que *siguen* al usuario.

Cuadro 9.14: Información personal del usuario.

Publicaciones

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué publicación se está consultando.
Tipo.	Puede tratarse de un video o de una imagen.
Lugar.	Lugar (ID, nombre del lugar, longitud y latitud) donde fue elaborada la publicación, en caso de que el usuario lo haya declarado.
Me gusta.	Cantidad de *me gusta* recibidos por la publicación y el ID de los usuarios que los realizaron.
Fecha.	Fecha y hora en la que se efectuó la publicación.
Usuarios etiquetados.	Lista de los usuarios mencionados en el medio.
Etiquetas.	Lista con todas as etiquetas utilizadas en la publicación.

Cuadro 9.15: Información asociada a las publicaciones dentro de *Instagram*.

Relaciones con usuarios

Entidad	Descripción
Usuarios seguidos.	Lista de todos los usuarios que son *seguidos* por el usuario autenticado.
Seguidores.	Lista de todos las personas que *siguen* al usuario.
Solicitudes.	Lista de todos los usuarios que han solicitado permiso para seguir al usuario autenticado. Esto ocurre según las restricciones de privacidad declaradas.

Cuadro 9.16: Información asociada a las relaciones establecidas por el usuario.

9.1.5. Foursquare

Información personal

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de cuál usuario se está consultando.
Nombre.	Nombre completo del usuario.
Amigos.	Lista de todos los amigos establecidos en la red social.
Residencia.	Ciudad donde vive.
Género.	Género del usuario.
Biografía.	Descripción personal breve declarada por el usuario.
Imagen.	Foto usada en el perfil del usuario.
Tips.	Lista de todos los *tips* publicados por el usuario.
Listas de usuarios.	Listas de usuarios creadas por el usuario autenticado.
Seguidores.	Lista de todos los usuarios que *siguen* al usuario autenticado.
Checkins.	Lista de todos los *checkin* asociados a la red social efectuados por el usuario.
Gustos.	Lista de todos los gustos personales declarados por el usuario.
Por hacer.	Lista de todos los lugares marcados por el usuario como *por hacer*.

Cuadro 9.17: Información personal del usuario.

Información de lugares

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué lugar se está consultando.
Nombre.	Nombre del lugar.
Ubicación.	Dirección del lugar.
Categorías.	Categorías en las que se puede clasificar el lugar.
Estadísticas.	Estadísticas generales como la cantidad de *checkins* realizados en el lugar, la cantidad de usuarios presentes o la cantidad de *tips* que los usuarios le han publicado.
Horarios.	Horario de atención del lugar y su horario más popular de la semana.
Menú.	Menú ofrecido en el lugar.
Precios.	Rango de precios entre los que se encuentran los servicios del lugar.
Calificación.	Calificación promedio asignada por los usuarios.
Descripción.	Descripción general del lugar.
Listas.	Número de listas de lugares en las que ha sido incluido.
Creación.	Fecha en la que se creó el lugar.
Etiquetas.	Lista con todas las etiquetas utilizadas para describir el lugar.
Fotos.	URL de las imágenes del lugar publicadas en la red social.
Me gusta.	Cantidad de usuarios que ha marcado el lugar con *me gusta*.
Atributos.	Lista de atributos booleanos como si ha publicado sus precios, si tiene estacionamiento, si toma reservaciones, etc.
Lugares recomendados.	Lista de lugares cercanos recomendados.
Lugares similares.	Lista de lugares similares.
Eventos.	Lista de eventos a llevarse a cabo en el lugar.
Lugar siguiente.	Lista de los lugares en los que se suele hacer *checkin* luego de haber estado en el lugar.
Tips.	Lista con todos los *tips*, publicados por los usuarios, asociados al lugar.

Cuadro 9.18: Información asociada a los lugares registrados dentro de *Foursquare*.

Checkin

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué *checkin* se está consultando.
Fecha.	Fecha en la que se efectuó el *checkin*.
Privacidad.	Restricción sobre quién puede ver el *checkin*.
Lugar.	Lugar al cual está asociado el *checkin*.
Aplicación.	Aplicación por la cual se realizó el *checkin*.
Comentarios.	Comentarios recibidos por el *checkin*.
Me gusta.	Cantidad de *me gusta* recibidos por el *checkin*.
Checkin de amigos.	Lista con todos los *checkin* realizados por los amigos de un determinado usuario en el mismo lugar.

Cuadro 9.19: Información asociada a los *checkin* publicados en la red social.

Tips

Entidad	Descripción
ID.	El ID dentro de la red social permite especificar a la API sobre la información de qué *tip* se está consultando.
Contenido.	Texto que contiene las impresiones de un usuario sobre un lugar.
Fecha.	Fecha en la que se efectuó el *tip*.
Usuario.	Usuario que creó el *tip*.
Lugar.	Lugar sobre el cual se hizo el *tip*.
Por hacer.	Cantidad de usuarios que marcaron el *tip* como *por hacer*.
Me gusta.	Cantidad de *me gusta* recibidos por el *tip*.
Listas.	Todas las listas en las que se incluye el *tip*.
Guardado.	Lista con todos los usuarios que han guardado el *tip*.

Cuadro 9.20: Información asociada a los *tips* publicados en la red social.

9.2. Prototipo a través del plugin Recommendation-Support

En esta sección se detallan los pasos necesarios para llevar a cabo la implementación de un prototipo de sistema de recomendación utilizando el *plugin* RecommendationSupport. Para esto, se considera que el *plugin* y el complemento Papyrus ya se encuentran instalados y que se posee información asociada a usuarios y algún tipo de preferencia. De esta forma, los archivos necesarios son:

- **Algoritmos:** El *plugin* RecommendartionSupport cuenta adicionalmente con algoritmos desarrollados para generar recomendaciones automáticamente a partir de los datos administrados, para lo cual se deben importar en el proyecto los algoritmos seleccionados.
- **Modelo:** El modelo propuesto en la Figura 6.4 debe ser importado en el proyecto. Este es el encargado de administrar toda la información del prototipo y es valido para cualquier red social.
- **Diagrama:** Pese a que el modelo propuesto soporta su uso en cualquier red social, también es posible que existan características únicas para una determinada red social que deban ser modeladas a parte para que sean consideradas en el prototipo. Es aquí cuando es posible modelar las características más específicas de cada red social. Por ejemplo, definiendo que existe un tipo de evaluación unaria sobre otros usuarios, que se llama amistad y que es bidireccional. Este diagrama sólo debe ser hecho una sola vez para describir el funcionamiento más específico de la red social en cuestión, no en cada implementación. A modo de ejemplo se presentan las Figuras 9.1 y 9.2, las cuales corresponden a diagramas utilizados para desarrollar prototipos sobre las redes sociales *Facebook* y *Twitter* respectivamente. En el caso de *Facebook*, se puede está definiendo que un usuario puede ser Amigo de otro y/o Seguirlo, que los ítems pueden estar en forma de Contenido o de Publicación, y que estos pueden ser evaluados a través de acciones como Compartir, Comentar, marcar con Me Gusta o asignar calificaciones según las alternativas disponibles para los tipos de ítems. Por otro lado, en el caso de *Twittee* se está definiendo que un usuario puede seguir a otro, que los ítems sólo son los *tweets* y que estos pueden ser evaluados a través de acciones como *Retweetear*, agregandolos a *Favoritos* y/o respondiéndolos.
- **Perfil:** El perfil UML propuesto en la Figura 9.3 permite estandarizar el comportamiento de las clases definidas en el diagrama. De esta forma, se puede especificar que las clases que representen personas deben ser de tipo *User*, que los ítems de las redes sociales pueden ser de tipo *DigitalMedia* o *Post* y que las evaluaciones pueden ser de tipo *Unary*, *Binary* o *Scalar*. Se puede apreciar su aplicación en forma de *stereotypes* en las Figuras 9.1 y 9.2 sobre el nombre de las clases.

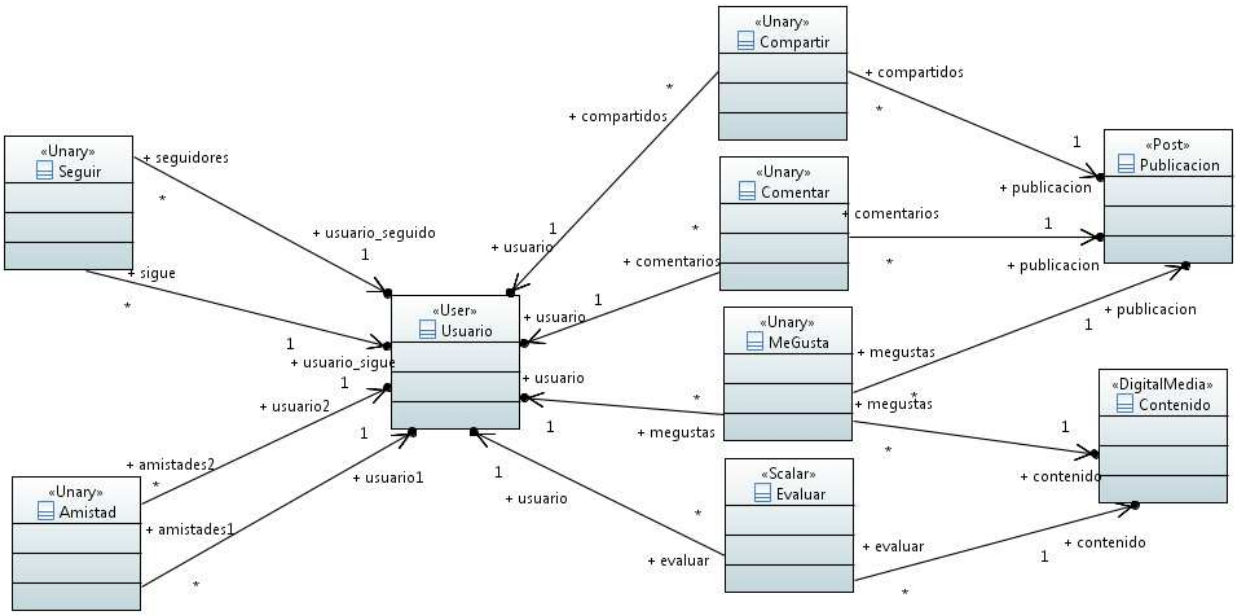

Figura 9.1: Diagrama de clases que describe las caracteríticas particulares de *Facebook*.

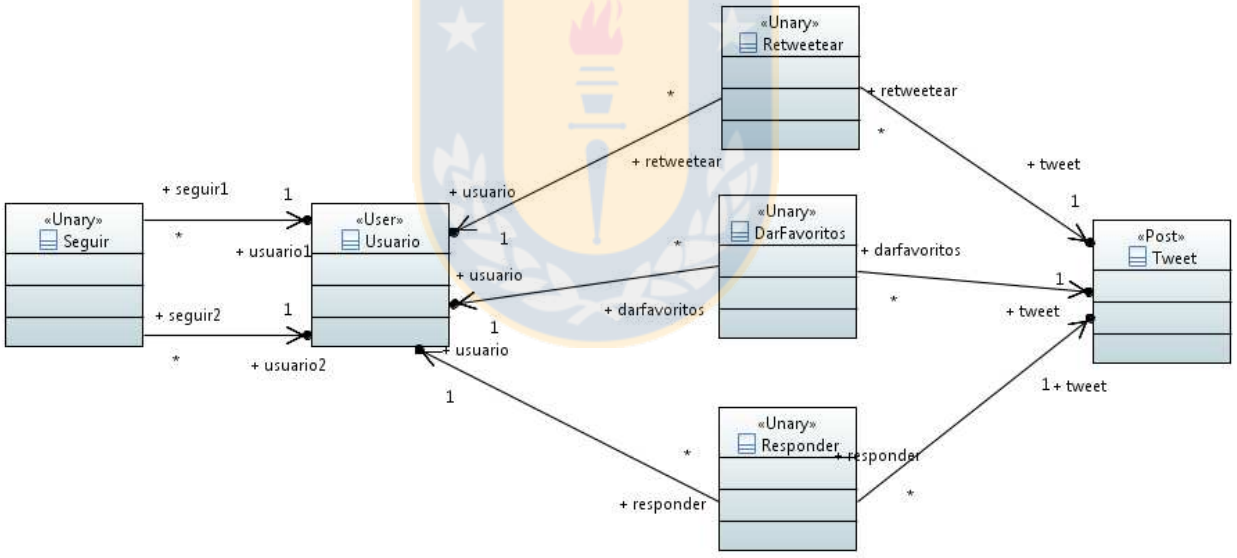

Figura 9.2: Diagrama de clases que describe las caracteríticas particulares de *Twitter*.

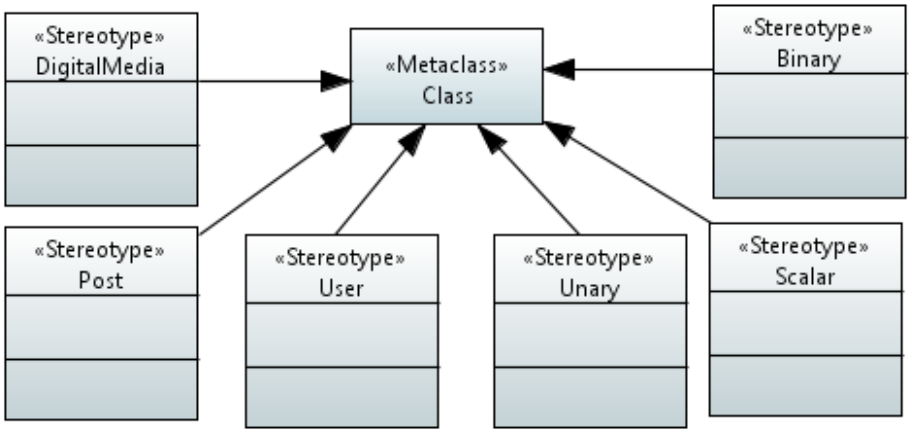

Figura 9.3: Perfil UML que permite estandarizar el comportamiento de las clases de una red social.

9.2.1. Generación del prototipo

Los pasos a seguir dentro del entorno de Eclipse para obtener el prototipo de sistema de recomendación son:

1. Crear un nuevo Java project. Importar los archivos Diagrama, Modelo, Algoritmos y Perfil mencionados anteriormente, cada uno de la siguiente forma:
   - Click derecho en el proyecto creado > Import > Archive File > buscar el archivo .zip.
2. Agregar el Perfil al Diagrama.
   - Seleccionar el Diagrama > Properties > pestaña Profile > Apply Profile > buscar el perfil importado.
   - Aplicar los *stereotypes* según correspondan dentro de la red social. Por ejemplo, el *stereotype* User a Usuario, Unary a Me Gusta, etc.
3. Generar el código.
   - Generar el código asociado al Diagrama de la red social de la siguiente forma: Click derecho en el archivo *.uml* del Diagrama > TransformationForRecommendation > From UML Class Diagram to Java Classes.
   - Generar el código asociado al Modelo desarrollado de la siguiente forma: Click derecho en el archivo *.uml* del Modelo > TransformationForRecommendation > From UML Class Diagram to Java Classes.
   - Generar el código asociado a los algoritmos de recomendación de la siguiente forma: Click derecho en el archivo *.uml* del algoritmo seleccionado > TransformationForRecommendation > From UML Communication Diagram to Java Algorithm.

4. Modificaciones.

   - Copiar el archivo "Recommender.java" generado por el *plugin* y pegarlo en el paquete generado del Modelo.
   - Modificar el nombre de los paquetes importados en las clases generadas del Diagrama y el algoritmo por el nombre del paquete generado del Modelo. Esto se debe a que el *plugin* genera el código de acuerdo al modelo presentado en la Figura 4.2 y ahora se está usando un modelo diferente.
   - En el paquete generado del Modelo, editar la clase User.java: buscar el método "getSimilarities()" y cambiarle el nombre por "getSimilaritiesU()". Este cambio debe efectuarse debido a que el modelo original, a diferencia del modelo propuesto, no considera a los usuarios como ítem lo que genera un conflicto al repetir el nombre de este método.
   - En el paquete generado del Modelo, editar la clase Recommender.java. Se debe buscar la llamada al método "user.getSimilarities()" y reemplazarla por "user.getSimilaritiesU()" (debido al punto anterior, se debe actualizar la llamada al método).
   - En el paquete del Diagrama se deben crear los constructores de todas aquellas clases que hayan sido marcadas con los *stereotypes Unary*, *Binary* o *Scalar*. A continuación se muestra un ejemplo de constructor para la clase MeGusta, donde se aprecia que debe tener dos constructores debido a que se trata de un tipo de evaluación unaria que puede ser asignada a elementos de tipo Publicación y de tipo Contenido:

```
1    public MeGusta(Integer id, Usuario u, Publicacion i){
2        super(u, i);
3        this.setId(id);
4    }
5    public MeGusta(Integer id, Usuario u, Contenido i){
6        super(u, i);
7        this.setId(id);
8    }
```

5. Ejecución del prototipo.

   - Crear un nuevo paquete de nombre "main" dentro del proyecto.
   - Dentro del paquete main, crear una clase "Main.java" que contenga el siguiente código.

```
1  package main;
2  import java.util.List;
3  import abstractModel.*;
4  import facebook.*;
5
6  public class Main {
7    public static void main(String[] args) {
8      /* En esta sección se deben hacer las consultas
9      *  respectivas a la base de datos para obtener
10     *  los datos requeridos por los algoritmos.
11     */
12
13     // Aplicando el algoritmo seleccionado
14     Recommender.findSimilarities();
15
16     /* Mostrar las recomendaciones obtenidas especificamente
17   *  para el usuario user1.
18       Estas recomendaciones pueden ser sobre usuarios o ítems.
19     Usuario user = user1;
20     List<Item> recommendations=Recommender.getRecommendations(user);
21
22     System.out.println("Social Recommendations");
23     System.out.println("----------------------");
24
25     System.out.println("A "+user.getName()+" se le podría recomendar
           :");
26     int j=1;
27     for(Item i:recommendations){
28       if(i instanceof Usuario){
29         System.out.println(j+". El usuario: "+((Usuario)i).getName())
               ;
30         j++;
31       }
32     }
33     System.out.println("----------------------");
34     for(Item i:recommendations){
35       if(i instanceof Publicacion){
36         System.out.println(j+". La publicación: "+((Publicacion)i).
               getContent());
37         j++;
38       }
39     }
40     System.out.println("----------------------");
41     for(Item i:recommendations){
42       if(i instanceof Contenido){
43         System.out.println(j+". El contenido: "+((Contenido)i).getUrl
               ());
44         j++;
45       }
46     }
47   }
48 }
```

9.3. Prototipo a través del plugin Java Code Generator

En esta sección se detallan los pasos necesarios para llevar a cabo la implementación de un prototipo de sistema de recomendación utilizando el *plugin* Java Code Generator y las librerías de recomendación de Apache Mahout. Para esto, se considera que tanto el *plugin* como Papyrus ya se encuentran instalados y que se posee información asociada a usuarios y algún tipo de preferencia. De esta forma, los archivos necesarios son:

- **Modelo:** El modelo propuesto en la Figura 6.1 debe ser importado en el proyecto. Este es el encargado de administrar toda la información del prototipo y es valido para cualquier red social.
- **Librerías:** El proyecto debe contar con las librerías de Apache Mahout necesarias para generar recomendaciones.
- **Diagrama:** En forma análoga a lo explicado para el caso del *plugin* Recommendation-Support, se debe importar un diagrama que describe el funcionamiento más detallado de la red social que se desee utilizar.
- **Perfil:** Similar a lo que ocurre al usar el *plugin* RecommendationSupport, el perfil UML presentado en la Figura 9.3 permite orientar al desarrollador sobre los tipos de clases reconocidas por el modelo y que permiten obtener recomendaciones.

9.3.1. Generación del prototipo

Los pasos a seguir dentro del entorno de Eclipse para obtener el prototipo de sistema de recomendación son:

1. Crear un nuevo Java project.
   - Crear dos carpetas dentro del proyecto, una de nombre “lib” y otra de nombre “data”.
   - Agregar todas las librería de Apache Mahout a la carpeta *lib* y luego:
     Click derecho en la carpeta *lib* > Build Path > Add to Build Path.
   - En la carpeta *src* del proyecto, crear tres nuevos paquetes: uno para el Modelo, otro para el Diagrama y un paquete“main”.
2. Importar archivos.
   - Para cada uno de los archivos (Modelo, Diagrama y Perfil):
     Click derecho en el proyecto > Import > Archive File > buscar el archivo .zip.
3. Generar código del Modelo.
   - Abrir el modelo > click derecho en alguna sección blanca > Java > Generate Java Code.
   - Se crea una carpeta con el nombre del Modelo en el *workspace*. Se deben copiar las clases que se encuentran dentro de su paquete “(default package)” en el paquete creado para el Modelo (punto 1).
4. Agregar el Perfil al Diagrama y generar su código.

- Seleccionar el Diagrama > Properties > pestaña Profile > Apply Profile > buscar el perfil importado.
- Aplicar los *stereotypes* según correspondan dentro de la red social. Por ejemplo, el *stereotype* User a Usuario, Unary a Me Gusta, etc.
- Click derecho en alguna sección blanca del diagrama > Java > Generate Java Code.
- Se crea una carpeta con el nombre del Diagrama en el *workspace*. Se deben copiar las clases que se encuentran dentro de su paquete "(default package)" en el paquete creado para el Diagrama (punto 1).

5. Ejecución del prototipo.

- Crear una clase "Main.java" dentro del paquete *main* que contenga el siguiente código:

```
1  package main;
2
3  import java.io.BufferedReader;
4  import java.io.BufferedWriter;
5  import java.io.File;
6  import java.io.FileReader;
7  import java.io.FileWriter;
8  import java.io.IOException;
9  import java.sql.Connection;
10 import java.sql.DriverManager;
11 import java.sql.ResultSet;
12 import java.sql.SQLException;
13 import java.sql.Statement;
14 import java.util.ArrayList;
15 import java.util.List;
16
17 import org.apache.mahout.cf.taste.*;
18
19 import abstractModel.*;
20 import facebook.*;
21
22 public class Main {
23
24   public static void main(String[] args) throws IOException {
25
26     /* En esta sección se deben hacer las consultas respectivas a la
27     *  base de datos para obtener los datos requeridos por las
           librerías
28     */
29
30     // Se escribe un archivo .csv con en forma de ''usuario, ítem,
           preferencia''.
31     // Preferencia se refiere a una nota escalar, pero en caso de
           dejarla en blanco se toma como una preferencia unaria de ese
           usuario a ese ítem.
32     BufferedWriter bw = new BufferedWriter(new FileWriter("data/datos
           .csv"));
33     for(Preference p: prefs){
34       bw.write(p.getUser().getIdSN() + "," + p.getItem().getIdSN() +
             "\n");
35     }
```

```
36        bw.close();
37
38        try{
39          // Se abren el archivo con las preferencias almacenadas.
40          DataModel dm = new FileDataModel(new File("data/datos.csv"));
41
42          // Se selecciona el algoritmo que se usará. En este caso,
43          // "LogLikelihoodSimilarity".
44          ItemSimilarity sim = new LogLikelihoodSimilarity(dm);
45
46          // Se generan las recomendaciones
47          GenericItemBasedRecommender recommender = new
                GenericItemBasedRecommender(dm, sim);
48
49          // Se reciben las recomendaciones para todos los usuarios y se
                almacenan.
50          for(User u: users){
51            List<RecommendedItem> recommendations = recommender.
                  mostSimilarItems(u.getIdSN(), 5);
52            for(RecommendedItem recommendation: recommendations){
53              Recommendation rmm = new Recommendation();
54              rmm.setUser(u);
55              rmm.setItem(getUser((int)recommendation.getItemID()));
56              rmm.setValue(recommendation.getValue());
57              recs.add(rmm);
58            }
59          }
60
61
62        } catch (IOException e){
63          System.out.println("There was an error");
64          e.printStackTrace();
65        } catch (TasteException e){
66          System.out.println("There was a Taste Exception");
67          e.printStackTrace();
68        }
69
70
71        for(User u: users){
72          int x = 1;
73          System.out.println("Al usuario "+u.getIdSN()+" se le podría
                recomendar:");
74          for(Recommendation r: recs){
75            if(r.getUser().getIdSN() == u.getIdSN()){
76              System.out.println(x+". "+r.getItem().getIdSN()+" con valor
                    : "+r.getValue());
77              x++;
78            }
79          }
80          System.out.println("-------------------------");
81        }
82
83      }
84
85      public static User getUser(int id){
86        User us = new User();
87        for (User u: users){
88          if (u.getIdSN() == id){
89            us = u;
```

```
90          }
91        }
92        return us;
93      }
94
95      public static void closeConnection(Connection c) {
96        try {
97          c.close();
98        }catch (Exception e) {
99          System.out.println("Error en el cierre de la connexion.");
100       }
101     }
102
103   }
```

9.4. Datos utilizados en los experimentos

Como se mencionó en el capítulo 7, los conjuntos de datos utilizados en los experimentos no incluyen información a cerca de ítems (como publicaciones). Es por esto que para mostrar su aplicación se obtuvo información de la red social *Facebook* y se probó en los prototipos. Sin embargo, esta información no fue rescatada a través de la API de *Facebook* debido a que para ello es necesario registrar una aplicación, tener la aprobación de la red social y conseguir un grupo de usuarios dispuestos a compartir sus datos. Debido a esto, la información se obtuvo manualmente de un pequeño grupo de diez usuarios, algunas de sus publicaciones realizadas y sus preferencias expresadas sobre las mismas, agregándolas al sistema de recomendación de la siguiente manera:

```
1   public class Main {
2
3     public static void main(String[] args) {
4
5   // Esta es la sección mencionada en los prototipos donde se
6   // deben hacer consultas a la base de datos.
7
8   /*=========================================================*/
9   /*========================= USUARIOS ======================*/
10  /*=========================================================*/
11
12    Usuario angela = new Usuario();
13    angela.setId(1);
14    angela.setIdSN(10001);
15    angela.setName("Angela Chavarria");
16    angela.setEmail("angelito@gmail.com");
17
18    Usuario amanda = new Usuario();
19    amanda.setId(2);
20    amanda.setIdSN(10002);
21    amanda.setName("Amanda Molina");
22    amanda.setEmail("amanda@gmail.com");
23
24    Usuario mauro = new Usuario();
25    mauro.setId(3);
26    mauro.setIdSN(10003);
27    mauro.setName("Mauricio Flores");
28    mauro.setEmail("mauro@hotmail.com");
29
30    Usuario victor = new Usuario();
31    victor.setId(4);
32    victor.setIdSN(10004);
33    victor.setName("Victor Cartes");
34    victor.setEmail("vitor@yahoo.com");
35
36    Usuario cata = new Usuario();
37    cata.setId(5);
38    cata.setIdSN(10005);
39    cata.setName("Catarina Von Bora");
40    cata.setEmail("juan@gmail.com");
41
42    Usuario javiera=new Usuario();
43    javiera.setId(6);
44    javiera.setIdSN(10006);
```

```
45      javiera.setName("Javiera Andrade");
46      javiera.setEmail("javi@facebook.com");
47
48      Usuario monse=new Usuario();
49      monse.setId(7);
50      monse.setIdSN(10007);
51      monse.setName("Montserrat Castello");
52      monse.setEmail("mon@terra.cl");
53
54      Usuario romi = new Usuario();
55      romi.setId(8);
56      romi.setIdSN(10008);
57      romi.setName("Romina Abaroa");
58      romi.setEmail("romina@gmail.com");
59
60      Usuario jose = new Usuario();
61      jose.setId(9);
62      jose.setIdSN(10009);
63      jose.setName("Jose Espinaza");
64      jose.setEmail("joseezpinaza@udec.cl");
65
66      Usuario david = new Usuario();
67      david.setId(10);
68      david.setIdSN(10010);
69      david.setName("David Gorkin");
70      david.setEmail("german@terra.cl");
71
72  /*=============================================================*/
73  /*=================== PUBLICACIONES ===========================*/
74  /*=============================================================*/
75
76      // ----- ANGELA -----
77      Publicacion p1 = new Publicacion();
78      p1.setId(11);
79      p1.setIdSN(20001);
80      p1.setMakerUser(angela);
81      p1.setContent("Tanto tanto los quiero :)");
82
83      Publicacion p2 = new Publicacion();
84      p2.setId(12);
85      p2.setIdSN(20002);
86      p2.setMakerUser(angela);
87      p2.setContent("mi tema favorito de alcest <3 https://www.youtube.com/
            watch?v=snCkaW6XShA");
88
89      // ----- AMANDA -----
90      Publicacion p3 = new Publicacion();
91      p3.setId(13);
92      p3.setIdSN(20003);
93      p3.setMakerUser(amanda);
94      p3.setContent("Una familia feliz! :D");
95
96      // ----- MAURICIO -----
97      Publicacion p4 = new Publicacion();
98      p4.setId(14);
99      p4.setIdSN(20004);
100     p4.setMakerUser(mauro);
101     p4.setContent("Ahora el merecido postre");
102
```

```
103         Publicacion p5 = new Publicacion();
104         p5.setId(15);
105         p5.setIdSN(20005);
106         p5.setMakerUser(mauro);
107         p5.setContent("Goloseria máxima con la gati, estos si ke son gyros");
108
109         // ----- VICTOR -----
110         Publicacion p6 = new Publicacion();
111         p6.setId(16);
112         p6.setIdSN(20006);
113         p6.setMakerUser(victor);
114         p6.setContent("Harto me duraron las regalonas, pero este viaje las
                terminaron de matar. Cumplieron su cometido a cada paso que dieron
                :)");
115
116         Publicacion p7 = new Publicacion();
117         p7.setId(17);
118         p7.setIdSN(20007);
119         p7.setMakerUser(victor);
120         p7.setContent("Este copete produce amnesia pero es bueno jajaja");
121
122         // ----- CATALINA -----
123         Publicacion p8 = new Publicacion();
124         p8.setId(18);
125         p8.setIdSN(20008);
126         p8.setMakerUser(cata);
127         p8.setContent("Estos son los detalles con los que, en el contexto
                actual, una se regocija.");
128
129         Publicacion p9 = new Publicacion();
130         p9.setId(19);
131         p9.setIdSN(20009);
132         p9.setMakerUser(cata);
133         p9.setContent("En la vida definitivamente hay que rodearse de buenas
                amistades, porque las malas (por tirar un nombre: Francisca) te
                obligan a recibir septiembre un lunes 31 de agosto.");
134
135         Publicacion p10 = new Publicacion();
136         p10.setId(20);
137         p10.setIdSN(20010);
138         p10.setMakerUser(cata);
139         p10.setContent("Interesante reflexión. Para la vida. Para compartir
                con los hijos como enseñanza. Enseñanza al nivel 'pan de de la
                palabra'");
140
141         // ----- JAVIERA -----
142         Publicacion p11 = new Publicacion();
143         p11.setId(21);
144         p11.setIdSN(20011);
145         p11.setMakerUser(javiera);
146         p11.setContent("cuando otros pajaritos???");
147
148         Publicacion p12 = new Publicacion();
149         p12.setId(22);
150         p12.setIdSN(20012);
151         p12.setMakerUser(javiera);
152         p12.setContent("Hoy me dio por ponerme más casera y patriota. Unos
                clasicos porotitos granados! #masterchef #ilovecook #porotos");
153
```

```
154         // ------ MONTSERRAT ------
155         Publicacion p13 = new Publicacion();
156         p13.setId(23);
157         p13.setIdSN(20013);
158         p13.setMakerUser(monse);
159         p13.setContent("Awww... cómo comermelo despues de esto? :P");
160 
161         Publicacion p14 = new Publicacion();
162         p14.setId(24);
163         p14.setIdSN(20014);
164         p14.setMakerUser(monse);
165         p14.setContent("Con mi hermano que le dio por tatuarse mas que yo ");
166 
167         Publicacion p15 = new Publicacion();
168         p15.setId(25);
169         p15.setIdSN(20015);
170         p15.setMakerUser(monse);
171         p15.setContent("Monchita Castelló Kock gracias por la atención y por
                el tatoo .... Ya subiré una foto .... Cuida esa mano .... Saludos..");
172 
173         Publicacion p16 = new Publicacion();
174         p16.setId(26);
175         p16.setIdSN(20016);
176         p16.setMakerUser(monse);
177         p16.setContent("Ok... no nos olvidemos de aplicar el ... en nuestros
                días!");
178 
179         // ------ ROMINA ------
180         Place l1 = new Place();
181         l1.setName("Aeropuerto Internacional Comodoro Arturo Merino Benítez");
182         Publicacion p17 = new Publicacion();
183         p17.setId(27);
184         p17.setIdSN(20017);
185         p17.setMakerUser(romi);
186         p17.setContent("Reencuentro <3");
187         p17.getDigitalMedias().add(l1);
188 
189         // ------ JOSE ------
190         Publicacion p18 = new Publicacion();
191         p18.setId(28);
192         p18.setIdSN(20018);
193         p18.setMakerUser(jose);
194         p18.setContent("Uno de nuestros clientes mas felices de los sillones
                BigSize Mira Tomas Ignacio Henning Benavente casi te supero! ...
                Soon! Mas info.");
195 
196         Publicacion p19 = new Publicacion();
197         p19.setId(29);
198         p19.setIdSN(20019);
199         p19.setMakerUser(jose);
200         p19.setContent("Oye bonita. Otra foto más, muchas risas que
                compartimos, bailes, lugares, sensaciones, personajes, adopciones,
                raptos, experiencias que"
201                 +"llenan y dejan. Me gusta caminar contigo, que hayas vuelto
                        no deja de sorprenderme gratamente. Gracias por ser mi
                        complice de fechorías"
202                 + " y estar. Te quiero mucho. Feliz cumpleaños atrasao y feliz
                        de compartir los dias contigo.");
203 
```

```
204         // ------ DAVID ------
205         Publicacion p20 = new Publicacion();
206         p20.setId(30);
207         p20.setIdSN(20020);
208         p20.setMakerUser(david);
209         p20.setContent("Mi hijo, el regalo más hermoso que me ha dado esta
                vida");
210
211
212     /*=============================================================*/
213     /*======================== AMISTADES ==========================*/
214     /*=============================================================*/
215
216         new Amistad(1, angela, amanda);
217         new Amistad(3, angela, victor);
218         new Amistad(4, angela, javiera);
219         new Amistad(5, angela, romi);
220
221         new Amistad(6, amanda, angela);
222
223         new Amistad(7, mauro, cata);
224         new Amistad(8, mauro, monse);
225         new Amistad(9, mauro, romi);
226         new Amistad(10, mauro, david);
227         new Amistad(11, mauro, jose);
228
229         new Amistad(12, victor, angela);
230         new Amistad(13, victor, monse);
231
232         new Amistad(14, cata, mauro);
233         new Amistad(15, cata, javiera);
234         new Amistad(16, cata, monse);
235         new Amistad(17, cata, jose);
236
237         new Amistad(18, javiera, angela);
238         new Amistad(19, javiera, cata);
239         new Amistad(20, javiera, monse);
240         new Amistad(21, javiera, jose);
241
242         new Amistad(22, monse, mauro);
243         new Amistad(23, monse, victor);
244         new Amistad(24, monse, cata);
245         new Amistad(25, monse, javiera);
246         new Amistad(26, monse, romi);
247         new Amistad(27, monse, david);
248
249         new Amistad(28, romi, angela);
250         new Amistad(29, romi, mauro);
251         new Amistad(30, romi, monse);
252
253         new Amistad(31, jose, mauro);
254         new Amistad(32, jose, cata);
255         new Amistad(33, jose, javiera);
256         new Amistad(34, jose, david);
257
258         new Amistad(35, david, mauro);
259         new Amistad(36, david, monse);
260         new Amistad(37, david, jose);
261
```

```
262
263
264 /*=============================================================*/
265 /*========================= ME GUSTA ==========================*/
266 /*=============================================================*/
267
268     // ------LIKES A LAS PUBLICACIONES DE ANGELA ------
269     new Compartir(38, angela, p1);
270     new Compartir(39, romi, p1);
271     new Compartir(40, victor, p1);
272
273     new Compartir(41, angela, p2);
274     new Compartir(42, romi, p2);
275
276     // ------ LIKES A LAS PUBLICACIONES DE AMANDA ------
277     new Compartir(43, angela, p3);
278     new Compartir(44, amanda, p3);
279
280     // ------ LIKES A LAS PUBLICACIONES DE MAURO ------
281     new Compartir(45, romi, p4);
282     new Compartir(46, cata, p4);
283     new Compartir(47, david, p4);
284     new Compartir(48, mauro, p4);
285
286     new Compartir(49, mauro, p5);
287     new Compartir(50, romi, p5);
288     new Compartir(51, david, p5);
289     new Compartir(52, monse, p5);
290
291     // ------ LIKES A LAS PUBLICACIONES DE VICTOR ------
292     new Compartir(53, victor, p6);
293     new Compartir(54, angela, p6);
294     new Compartir(55, monse, p6);
295
296     new Compartir(56, victor, p7);
297     new Compartir(57, angela, p7);
298     new Compartir(58, monse, p7);
299
300     // ------ LIKES A LAS PUBLICACIONES DE CATA ------
301     new Compartir(59, mauro, p8);
302     new Compartir(60, javiera, p8);
303     new Compartir(61, jose, p8);
304
305     new Compartir(62, cata, p9);
306     new Compartir(63, monse, p9);
307     new Compartir(64, jose, p9);
308     new Compartir(65, javiera, p9);
309
310     new Compartir(66, cata, p10);
311     new Compartir(67, monse, p10);
312     new Compartir(68, jose, p10);
313     new Compartir(69, javiera, p10);
314     new Compartir(70, mauro, p10);
315
316     // ------ LIKES A LAS PUBLICACIONES DE JAVIERA ------
317     new Compartir(71, javiera, p11);
318     new Compartir(72, jose, p11);
319     new Compartir(73, cata, p11);
320     new Compartir(74, monse, p11);
```

```
321
322         new Compartir(75, javiera, p12);
323         new Compartir(76, angela, p12);
324         new Compartir(77, cata, p12);
325         new Compartir(78, monse, p12);
326
327         // ------ LIKES A LAS PUBLICACIONES DE MONSE ------
328         new Compartir(79, monse, p13);
329         new Compartir(80, mauro, p13);
330         new Compartir(81, cata, p13);
331         new Compartir(82, romi, p13);
332
333         new Compartir(83, monse, p14);
334         new Compartir(84, mauro, p14);
335         new Compartir(85, cata, p14);
336         new Compartir(86, victor, p14);
337         new Compartir(87, romi, p14);
338
339         new Compartir(88, cata, p15);
340
341         new Compartir(89, monse, p16);
342         new Compartir(90, david, p16);
343         new Compartir(91, romi, p16);
344         new Compartir(92, javiera, p16);
345         new Compartir(93, mauro, p16);
346         new Compartir(94, victor, p16);
347
348         // ------ LIKES A LAS PUBLICACIONES DE ROMI ------
349         new Compartir(95, romi, p17);
350         new Compartir(96, angela, p17);
351         new Compartir(97, mauro, p17);
352         new Compartir(98, monse, p17);
353
354         // ------ LIKES A LAS PUBLICACIONES DE JOSE ------
355         new Compartir(99, cata, p18);
356         new Compartir(100, david, p18);
357
358         new Compartir(101, jose, p19);
359         new Compartir(102, mauro, p19);
360         new Compartir(103, cata, p19);
361         new Compartir(104, javiera, p19);
362         new Compartir(105, david, p19);
363
364         // ------ LIKES A LAS PUBLICACIONES DE DAVID ------
365         new Compartir(106, mauro, p20);
366         new Compartir(107, monse, p20);
367         new Compartir(108, david, p20);
368
369
370     }
371 }
```

Como resultado, a partir de esta información fue posible obtener recomendaciones para cada uno de los usuarios, las cuales que incluían tanto usuarios como publicaciones. Sin embargo, resulta necesario especificar cómo es posible obtener este tipo de información si se utilizara para ello la API de *Facebook*.

Como observación, no se profundizará en el código y los pasos necesarios para habilitar las conexiones requeridas por la API, por lo que se explicará únicamente las consultas necesarias y el manejo de la información de ejemplo.

9.4.1. API de Facebook

La API de *Facebook* se ha distribuido en una serie de lenguajes de programación, pero todas las pruebas realizadas en la etapa de obtención de información (ver capítulo 5) se efectuaron en PHP debido a que se trata de un lenguaje común con las APIs de las demás redes sociales.

Información personal

```php
<?php
// Se puede consultar por cualquier usuario autenticado
// cambiando 'me' por el ID del usuario.
$facebook_user = (new FacebookRequest($session, 'GET', '/me'))->execute()
    ->getGraphObject(GraphUser::className())->asArray();

$name = $facebook_user['name'];
$email = $facebook_user['email'];
$id = $facebook_user['id'];
?>
```

Publicaciones

```php
<?php
// Se puede consultar por cualquier usuario autenticado
// cambiando 'me' por el ID del usuario.
$getFeed = (new FacebookRequest($session, 'GET', '/me/feed'))->execute()->
    getGraphObject()->asArray();

//para cada publicación relacionada al usuario
foreach ($getFeed['data'] as $key) {
  $idPost = $key->id;
  $contentoPost = $key->message;
  $makerUser = $key->id;

  //todos los 'me gusta' que ha recibido
  foreach ($key->likes->data as $key2) {
    $likedUser = $key2->name;
    $likedUserId = $key2->id;
  }

  //todos los comentarios que ha recibido
  foreach ($key->comments->data as $key3) {
    $commenterUser = $key3->from->name;
    $commenterUserId = $key3->from->id;
    $commentedContent = $key3->message;
  }
}
?>
```

Relaciones

Con esta consulta es posible identificar las amistades de un usuario y utilizar esta información como una preferencia unaria.

```
<?php
// Se puede consultar por cualquier usuario autenticado
// cambiando 'me' por el ID del usuario.
$getFriends = (new FacebookRequest($session, 'GET', '/me/friends?limit
    =100000'))->execute()->getGraphObject()->asArray();

foreach ($getFriends['data'] as $key) {
        $nameFriend = $key->name;
        $idFriend = $key->id;
        echo $nameFriend." -> ".$idFriend."<br/>";
}
?>
```

9.5. Consultas a la base de datos

En esta sección se detalla el código utilizado por las dos herramientas de recomendación para realizar las consultas a la una base de datos para obtener la información de los *datasets* ofrecidos por la Universidad de Stanford y que fueron previamente almacenados. Los dos códigos a continuación deben incluirse en la sección mencionada en los respectivos prototipos como consultas a la base de datos.

9.5.1. Algoritmos del plugin RecommendationSupport

```
package main;

import java.util.ArrayList;
import java.util.List;
import abstractModel.*;
import facebook.*;
import java.sql.*;

public class Main {

  private static Connection con;

  public static void main(String[] args) {

    List<Usuario> users = new ArrayList<Usuario>();
    Statement stmt = null;

    String host = "127.0.0.1";
    String user = "root";
    String pass = "";
    String dtbs = "facebook";

    try{
      Class.forName("com.mysql.jdbc.Driver");
      String newConnectionURL = "jdbc:mysql://" + host + "/" + dtbs
          + "?" + "user=" + user + "&password=" + pass;
      System.out.println("Connecting to database...");
```

```
28        con = DriverManager.getConnection(newConnectionURL);
29
30          System.out.println("Creating statement...");
31          stmt = con.createStatement();
32
33        // consulta por todos los usuarios registrados
34        ResultSet res = stmt.executeQuery("SELECT * FROM user");
35        int i = 1;
36        while (res.next()) {
37          int id = res.getInt("id");
38          Usuario u = new Usuario();
39          u.setId(i);
40          u.setIdSN(id);
41          users.add(u);
42          i++;
43        }
44
45        //consulta por todas las relaciones de amistad
46        ResultSet res2 = stmt.executeQuery("SELECT * FROM friendship");
47        int j = 1;
48        while (res2.next()) {
49          int id1 = res2.getInt("id1");
50          int id2 = res2.getInt("id2");
51          new Amistad(j, getUser(users, id1), getUser(users, id2));
52          j++;
53        }
54
55        closeConnection(con);
56      }catch(SQLException se){
57          //Handle errors for JDBC
58          se.printStackTrace();
59       }catch(Exception e){
60          //Handle errors for Class.forName
61          e.printStackTrace();
62       }finally{
63          //finally block used to close resources
64          try{
65             if(stmt!=null)
66                stmt.close();
67          }catch(SQLException se2){
68          }// nothing we can do
69          try{
70             if(con!=null)
71                con.close();
72          }catch(SQLException se){
73             se.printStackTrace();
74          }//end finally try
75       }//end try
76
77    public static void closeConnection(Connection c) {
78      try {
79        c.close();
80      }catch (Exception e) {
81        System.out.println("Error en el cierre de la connexion.");
82      }
83    }
84
85    public static Usuario getUser(List<Usuario> users, int id){
86      for(Usuario u: users){
```

```
87          if(u.getIdSN() == id){
88            return u;
89          }
90        }
91        return null;
92      }
93    }
```

9.5.2. Apache Mahout

```
1   package main;
2
3   import java.io.BufferedReader;
4   import java.io.BufferedWriter;
5   import java.io.File;
6   import java.io.FileReader;
7   import java.io.FileWriter;
8   import java.io.IOException;
9   import java.sql.*;
10  import java.util.ArrayList;
11  import java.util.List;
12  import org.apache.mahout.cf.taste.*;
13  import abstractModel.*;
14  import facebook.*;
15
16  public class Main {
17
18    static Connection con;
19    static List<User> users = new ArrayList<User>();
20    static List<Recommendation> recs = new ArrayList<Recommendation>();
21
22    public static void main(String[] args) throws IOException {
23
24      Statement stmt = null;
25
26      String host = "127.0.0.1";
27      String user = "root";
28      String pass = "";
29      String dtbs = "facebook";
30
31      try{
32        Class.forName("com.mysql.jdbc.Driver");
33        String newConnectionURL = "jdbc:mysql://" + host + "/" + dtbs
34            + "?" + "user=" + user + "&password=" + pass;
35        con = DriverManager.getConnection(newConnectionURL);
36        stmt = con.createStatement();
37
38        // consulta por todos los usuarios registrados
39        ResultSet res = stmt.executeQuery("SELECT * FROM user");
40        while (res.next()) {
41            Usuario us = new Usuario();
42            us.setIdSN(res.getInt("id"));
43            users.add(us);
44        }
45
46        // consulta por todas las relaciones de los usuarios
47        // y las reescrbribe en el formato de entrada para
48        // los algortimos de Apache Mahout.
49        ResultSet res2 = stmt.executeQuery("SELECT * FROM friendship");
```

```
50          BufferedWriter bw = new BufferedWriter(new FileWriter("data/datos.
                csv"));
51          while (res2.next()) {
52            int id1 = res2.getInt("id1");
53            int id2 = res2.getInt("id2");
54            bw.write(id1 + "," + id2 + "\n");
55          }
56          bw.close();
57
58          closeConnection(con);
59        }catch(SQLException se){
60            //Handle errors for JDBC
61            se.printStackTrace();
62        }catch(Exception e){
63            //Handle errors for Class.forName
64            e.printStackTrace();
65        }finally{
66            //finally block used to close resources
67            try{
68                if(stmt!=null)
69                    stmt.close();
70            }catch(SQLException se2){
71            }// nothing we can do
72            try{
73                if(con!=null)
74                    con.close();
75            }catch(SQLException se){
76                se.printStackTrace();
77            }//end finally try
78        }//end try
79
80    }
81
82    public static User getUser(int id){
83      User us = new User();
84      for (User u: users){
85        if (u.getIdSN() == id){
86          us = u;
87        }
88      }
89      return us;
90    }
91
92    public static void closeConnection(Connection c) {
93      try {
94        c.close();
95      }catch (Exception e) {
96        System.out.println("Error en el cierre de la connexion.");
97      }
98    }
99
100 }
```

9.6. Experimentos

9.6.1. Experimento 1

```
1  import java.io.*;
2  import java.sql.*;
3  import java.util.List;
4  import org.apache.mahout.cf.taste.*;
5
6  public class facebookRecommend {
7
8    private static Connection con;
9
10   public static void main(String[] args){
11
12     Statement stmt = null;
13
14     String host = "127.0.0.1";
15     String user = "root";
16     String pass = "";
17     String dtbs = "facebook";
18
19     try{
20       Class.forName("com.mysql.jdbc.Driver");
21       String newConnectionURL = "jdbc:mysql://" + host + "/" + dtbs
22           + "?" + "user=" + user + "&password=" + pass;
23       con = DriverManager.getConnection(newConnectionURL);
24
25       stmt = con.createStatement();
26
27       // consultas sobre las relaciones de los usuarios y
28       // almacenamiento en el archivo .csv requerido por Apache Mahout
29       ResultSet res2 = stmt.executeQuery("SELECT * FROM friendship");
30       BufferedWriter bw = new BufferedWriter(new FileWriter("data/DB2T.csv
             "));
31
32       while (res2.next()) {
33         int id1 = res2.getInt("id1");
34         int id2 = res2.getInt("id2");
35         bw.write(id1 + "," + id2 + "\n");
36       }
37       bw.close();
38
39       closeConnection(con);
40     }catch(SQLException se){
41         //Handle errors for JDBC
42         se.printStackTrace();
43      }catch(Exception e){
44         //Handle errors for Class.forName
45         e.printStackTrace();
46      }finally{
47         //finally block used to close resources
48         try{
49            if(stmt!=null)
50               stmt.close();
51         }catch(SQLException se2){
52         }// nothing we can do
53         try{
```

```
54              if(con!=null)
55                  con.close();
56            }catch(SQLException se){
57              se.printStackTrace();
58            }//end finally try
59        }//end try
60
61
62
63        try{
64          // lectura de los datos
65          DataModel dm = new FileDataModel(new File("data/DB2T.csv"));
66
67          // selección del algoritmo
68          ItemSimilarity sim = new LogLikelihoodSimilarity(dm);
69
70          // cálculo de recomendaciones
71          GenericItemBasedRecommender recommender = new
                GenericItemBasedRecommender(dm, sim);
72
73          int x =1;
74          for(LongPrimitiveIterator items = dm.getItemIDs(); items.hasNext();)
                {
75            Long itemId = items.nextLong();
76            List<RecommendedItem> recommendations = recommender.
                  mostSimilarItems(itemId, 5);
77
78            for(RecommendedItem recommendation : recommendations){
79              System.out.println(itemId + "," + recommendation.getItemID() + "
                    ," + recommendation.getValue());
80            }
81            x++;
82            //if(x>10) System.exit(1);
83          }
84
85        } catch (IOException e){
86          System.out.println("There was an error");
87          e.printStackTrace();
88        } catch (TasteException e){
89          System.out.println("There was a Taste Exception");
90          e.printStackTrace();
91        }
92
93      }
94
95      public static void closeConnection(Connection c) {
96        try {
97          c.close();
98        }catch (Exception e) {
99          System.out.println("Error en el cierre de la connexion.");
100       }
101     }
102
103 }
```

Bibliografía

[1] J. He and W. W. Chu, *A social network-based recommender system (SNRS)*. Springer, 2010.

[2] M. Morita and Y. Shinoda, "Information filtering based on user behavior analysis and best match text retrieval," in *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 272–281, Springer-Verlag New York, Inc., 1994.

[3] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender systems: an introduction*. Cambridge University Press, 2010.

[4] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 253–260, ACM, 2002.

[5] P. Lops, M. De Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender systems handbook*, pp. 73–105, Springer, 2011.

[6] A. Umyarov and A. Tuzhilin, "Improving collaborative filtering recommendations using external data," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pp. 618–627, IEEE, 2008.

[7] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender systems handbook*, pp. 145–186, Springer, 2011.

[8] X. Amatriain, "The recommender problem revisited," in *Proceedings of the 8th ACM Conference on Recommender systems*, pp. 397–398, ACM, 2014.

[9] J. O'Donovan and B. Smyth, "Trust in recommender systems," in *Proceedings of the 10th international conference on Intelligent user interfaces*, pp. 167–174, ACM, 2005.

[10] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *Internet Computing, IEEE*, vol. 7, no. 1, pp. 76–80, 2003.

[11] X. Cai, M. Bain, A. Krzywicki, W. Wobcke, Y. S. Kim, P. Compton, and A. Mahidadia, "Collaborative filtering for people to people recommendation in social networks," in *AI 2010: Advances in Artificial Intelligence*, pp. 476–485, Springer, 2011.

[12] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 5–53, 2004.

[13] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of the 2007 ACM conference on Recommender systems*, pp. 17–24, ACM, 2007.

[14] M. Vanetti, E. Binaghi, B. Carminati, M. Carullo, and E. Ferrari, "Content-based filtering in on-line social networks," in *Privacy and Security Issues in Data Mining and Machine Learning*, pp. 127–140, Springer, 2011.

[15] I. Cantador, A. Bellogín, and D. Vallet, "Content-based recommendation in social tagging systems," in *Proceedings of the fourth ACM conference on Recommender systems*, pp. 237–240, ACM, 2010.

[16] G. Rojas and P. Valderas, "A model-driven development process of web-based recommender systems," tech. rep., Universidad de Concepción, 2014.