TESIS DOCTORAL

Determinación de tendencias en un portal web utilizando técnicas no supervisadas. Aplicación a sistemas de recomendaciones basados en filtrado colaborativo.

Realizada por
José David Martín Guerrero

Dirigida por
Dr. Emilio Soria Olivas y
Dr. Paulo Jorge Gomes Lisboa

Departament d'Enginyeria Electrònica
UNIVERSITAT DE VALÈNCIA – ESTUDI GENERAL
Valencia – Julio, 2004

Determinación de tendencias en un portal web utilizando técnicas no supervisadas. Aplicación a sistemas de recomendaciones basados en filtrado colaborativo.

José David Martín Guerrero, Julio 2004

Dpt. Enginyeria Electrònica.
Escola Tècnica Superior d'Enginyeria.

D. EMILIO SORIA OLIVAS, Doctor en Ingeniería Electrónica, Profesor Titular del Departamento de Ingeniería Electrónica de la Escola Tècnica Superior d'Enginyeria de la Universitat de València, y

D. PAULO JORGE GOMES LISBOA, Doctor en Ciencias, Professor of the School of Computing and Mathematical Sciences, Liverpool John Moores University.

HACEN CONSTAR QUE:

El Ingeniero en Electrónica D. José David Martín Guerrero ha realizado bajo nuestra dirección el trabajo titulado "Determinación de tendencias en un portal web utilizando técnicas no supervisadas. Aplicación a sistemas de recomendaciones basados en filtrado colaborativo", que se presenta en esta memoria para optar al grado de Doctor en Ingeniería Electrónica.

Y para que así conste a los efectos oportunos, firmamos el presente certificado, en Valencia, a 26 de Julio de 2004.

Emilio Soria Olivas	Paulo Jorge Gomes Lisboa	Javier Calpe Maravilla Director del Departamento

Tesis Doctoral:	DETERMINACIÓN DE TENDENCIAS EN UN PORTAL WEB UTILIZANDO TÉCNICAS NO SUPERVISADAS. APLICACIÓN A SISTEMAS DE RECOMENDACIONES BASADOS EN FILTRADO COLABORATIVO
Autor:	JOSÉ DAVID MARTÍN GUERRERO
Directores:	Dr. EMILIO SORIA OLIVAS Dr. PAULO JORGE GOMES LISBOA

El tribunal nombrado para juzgar la Tesis Doctoral arriba citada, compuesto por los señores:

Presidente: _____

Vocales: _____

_____

_____

Secretario: _____

Acuerda otorgarle la calificación de _____

Y para que así conste a los efectos oportunos, firmamos el presente certificado.

Valencia, a

Agradecimientos

Muchas son las personas a las que me gustaría agradecer de corazón su ayuda en el desarrollo de esta tesis, tanto desde el lado técnico o científico, como desde el lado humano. Seguramente me olvidaré de alguna de ellas, y por eso en primer lugar quisiera expresar mi agradecimiento a todas aquellas personas de las que injustamente me olvidé.

Quisera agradecer al Dr. Emilio Soria Olivas, primer codirector de mi tesis, tantas cosas que resulta difícil plasmarlas en unas pocas líneas. Él fue mi director de Proyecto Final de Carrera en Ingeniería Electrónica, el director de mi Trabajo de Investigación de Doctorado, y también el de la presente tesis. Si nos centramos en esta tesis, desde luego he de agradecerle sus fantásticas ideas (es fácil trabajar con directores tan inteligentes y trabajadores), sus correcciones y sugerencias, pero como ya he dicho, mi agradecimiento va mucho más allá. Te agradezco la confianza que has depositado en mí, y que intentaré no defraudar, tu preocupación por todas las situaciones laborales o personales que han podido afectarme, y sobre todo porque después de estos años, y de todas las experiencias que hemos tenido te considero un buen amigo.

I would like to express my gratitude to my second supervisor, Dr. Paulo Lisboa, leader of the Neural Computation Research Group, at the School of Computing and Mathematical Sciences, Liverpool John Moores University, where I carried out part of this thesis. It is a great honour for me to have the chance of collaborating with such an important and recognized researcher and professor. I knew about his research capabilities when I travelled to Liverpool, but the surprise was finding an extremely kind and helpful person. Many thanks Paulo for everything, your patient attitude, your suggestions and the extra-work of correcting this thesis and some paper or other. And,

especially, thanks for your good and fast ideas.

Quisiera agradecer la colaboración de la empresa tecnológica Tissat, S.A. (`http://www.tissat.es/`), que ha sido un activo colaborador de la investigación llevada a cabo en esta tesis, así como en cierta medida el germen de la misma a través del proyecto PROFIT "Evaluación de algoritmos de inteligencia artificial para la clasificación y predicción del comportamiento de los usuarios en un portal web (FIT-070000-2001-663)". El interés de Tissat, S.A. por los resultados alcanzados y la posibilidad de implementarlos en su software comercial han sido sin duda un aliciente y una motivación extra. Muy especial es mi agradecimiento a los doctores Emili Balaguer y Alberto Palomares, nuestras discusiones, en ocasiones bastante acaloradas, sirvieron para aumentar la calidad del trabajo.

Quisiera asimismo agradecer a la *Fundació OVSI (Oficina Valenciana per a la Societat de la Informació)* su interés por el trabajo y sobre todo, su permiso para poder disponer de los datos procedentes del portal web Infoville XXI (`http://www.infoville.es/`).

Mención especial merecen todos mis compañeros del Grupo de Procesado Digital de Señales de la Universitat de València, con los que he tenido la suerte de compartir trabajo y experiencias desde que me incorporé como becario en el año 99. No es en absoluto un tópico ni una obligación hacer esta dedicatoria, sino que es algo que considero totalmente necesario y justo. Gracias por permitirme formar parte de esta gran familia, de un ambiente tan agradable de trabajo y amistad, hay momentos y detalles que nunca olvidaré, y espero tener la fortuna de seguir disfrutando de vuestra compañía y amistad los próximos años, tanto en la universidad como en la vida social fuera de ella. Gracias también a aquellos otros compañeros del Departament d'Enginyeria Electrònica de la Universitat de València con los que también he compartido muy buenos momentos.

A mi amigo y compañero de despacho, Antonio, gracias por prestarme tu ayuda cuando te lo he pedido, pero sobre todo, cuando no lo he hecho.

Gustavo i Jordi, gràcies pel vostre recolzament, per la vostra disposició a ajudar-me quan he patit alguns entrebancs amb el LaTeX, i per eixos berenars de divendres per la vesprada que ajuden a tancar agradablement una setmana de treball; no m'oblide dels altres companys de divendres vesprada: Luis, Joan, Julia, ...

I also want to express my thanks to those people who made me feel good during my stay in Liverpool. Many thanks, Ian, for being so kind with me from the very first time we met each other, and of course, for the coffee-breaks! Thank you, Thoang for your help in the administrative tasks. And, what could I say about my dear housemates Rino and Anne Laure? You were living with me during all my stay, and we all became friends forever. I enjoyed every minute with you. Getting your friendship is a great honour and fortune for me.

Me gustaría acabar con toda la gente que está y ha estado conmigo más allá del desarrollo de esta tesis, empezando por mi familia, y muy en especial mis padres, siempre tan encantadores, ... També me'n recorde dels meus amics "eclèctics" i de la seua especial manera d'ajudar i de traure un somriure en els pitjors moments; aixina que gràcies a Kike, Juanra, Joanot, Xavi, Ferran, Garci, Arturo, i Alber.

Y Mª José, gracias por ser como eres, por cuidarme, por nuestros secretos y complicidades y por haber aguantado tan dulcemente mi egoismo, sobre todo durante el periodo que he estado escribiendo.

Gràcies a tots els que m'heu permés ser una persona feliç.

Torrent, 15 de juliol de 2004.

José D. Martín.

A M^a Carmen Martín

"When using a mathematical model, careful attention must be given to the *uncertainties* in the model"

(Richard P. Feynman, opinando sobre la fiabilidad del transbordador espacial Challenger).

Índice general

Resumen

La competencia que existe en la oferta de servicios web, así como la gran cantidad de usuarios que acceden usualmente a Internet, lo que supone una considerable cantidad de datos, hacen posible recomendar diferentes servicios web a usuarios individuales. Esto ha llevado al desarrollo de los sistemas de recomendaciones como uno de los campos de investigación más importantes dentro del modelado de usuarios. Los sistemas de recomendaciones sugieren a los usuarios aquellos servicios en los que probablemente estarán interesados. Existen recomendadores de diferentes tipos, siendo en ocasiones complicado escoger el más adecuado y asegurar su funcionamiento. Algunos recomendadores se basan en asociaciones entre diferentes productos, servicios o páginas web (por ejemplo, relaciones Bayesianas en recomendadores para buscar usuarios similares en función de los servicios que soliciten, como hace Amazon.com), mientras que otros sistemas se basan en la previa caracterización de grupos de usuarios similares.

La presente Tesis Doctoral propone el uso de técnicas no supervisadas para determinar los diferentes comportamientos que presentan los usuarios que se conectan a un portal web. Estas técnicas funcionan sin la necesidad de ningún tipo de conocimiento *a priori*, permitiendo establecer grupos que aglutinan a usuarios similares en su comportamiento. Esta fase de agrupamiento es utilizada para realizar recomendaciones a los usuarios del portal.

La justificación de este trabajo proviene del desarrollo de una metodología completa para llevar a cabo recomendaciones a usuarios individuales de portales web, así como de las diferentes aportaciones noveles que se han realizado en diversas partes de esta metodología.

Resum

La competència que existix en l'oferta de serveis web, així com la gran quantitat d'usuaris que accedixen a Internet, la qual cosa suposa una considerable quantitat de dades, fan possible recomanar diferents serveis web a usuaris individuals. Açò ha dut al desenvolupament dels sistemes de recomanacions com un dels camps de recerca més importants dins el modelat d'usuaris. Els sistemes de recomanacions sugerixen als usuaris aquells serveis en els que probablement estaran interessats. Existixen diferents tipus de sistemes de recomanacions, sent en ocasions complicat triar el més adient i assegurar el seu funcionament. Alguns recomanadors es basen en associacions entre diferents productes, serveis o pàgines web (per exemple, relacions Bayesianes en recomanadors pera buscar usuaris semblants en funció dels serveis que sol·liciten, com fa Amazon.com), mentre que uns altres sistemes es basen en la prèvia caracterització de grups d'usuaris semblants.

La present Tesi Doctoral proposa l'ús de tècniques no supervisades per tal de determinar els diferents comportaments que presenten els usuaris que es conecten a un portal web. Estes tècniques funcionen sense la necessitat de cap tipus de coneixement *a priori*, permetent establir grups que aglutinen a usuaris semblants en el seu comportament. Esta fase d'agrupament és utilitzada per tal de realitzar recomanacions als usuaris del portal.

La justificació d'este treball prové del desenvolupament d'una metodologia completa per a dur a terme recomanacions a usuaris individuals de portals web, així com de les diferents aportacions novels que s'han realitzat en diverses parts d'esta metodologia.

Abstract

The competition for new web-based services which, together with the amount of data routinely available, makes it possible in principle to target recommendations down to the level of individual users. This has dealt to the development of recommender systems as one of the more important fields of research in user modelling. Recommender systems suggest services to users who are likely interested in these services. There are different kinds of recommender systems, and it is often difficult to choose the most adequate system, and in addition, to guarantee its performance. Some of these recommender systems are based on associations between different products, services or view pages (e.g. Bayesian conditional frequencies in people-like-you recommendations from one product to another in Amazon.com), whereas other systems make use of the profiles of groups of users.

The present PhD Thesis proposes the use of unsupervised techniques in order to find out the different behaviours of logged-in users on a web portal. These techniques work without the need of any kind of *a priori* knowledge, thus allowing to find clusters of similar users. This clustering stage is used in order to suggest recommendations to the web portal users.

A new methodology to carry out recommendations to individual users of web portals, and the different novel contributions that have been developed in several stages of the methodology, all justify the present work.

Prólogo

Esta Tesis Doctoral se enmarca dentro del estudio de los sistemas de recomendaciones, que es uno de los campos más prolíficos de investigación y desarrollo actualmente dentro de la disciplina del modelado de usuarios.

Han sido muchos los trabajos publicados sobre esta materia durante los últimos años, centrándose en diferentes aspectos de los sistemas de recomendaciones, como su desarrollo con o sin la necesidad de introducción de información por parte del usuario, la comparación entre diferentes técnicas de recomendación en determinadas aplicaciones reales, o el desarrollo de una interfaz adecuada para asegurar el éxito de las recomendaciones. Como se verá a continuación, en esta tesis se propone una metodología global para el desarrollo de un recomendador web, en la que en ningún momento se presupone una colaboración por parte del usuario, es decir, no es necesario que éste introduzca ningún tipo de preferencias ni datos personales. La motivación para esta elección es que los usuarios pueden verse considerablemente retraídos de un sitio web que les solicita información, además de que la información introducida en este tipo de formularios es, a menudo, poco fiable. Esta metodología ha sido validada con datos reales procedentes del portal web Infoville XXI (`http://www.infoville.es/`); es éste un portal web financiado por la Generalitat Valenciana que ofrece una amplia variedad de servicios a ciudadanos de diferentes municipios de la Comunidad Valenciana, y que es un ejemplo bastante representativo de lo que son los portales web de servicio al ciudadano.

Las características de los portales web de servicios al ciudadano, así como las de las diferentes técnicas de recomendación hicieron que la técnica de recomendación escogida fuera de tipo colaborativo. La metodología propuesta para llevar a cabo las recomendaciones consta de cuatro diferentes etapas:

1. *Desarrollo de un modelo de usuario web.* En esta primera fase, se generan conjuntos de datos artificiales que sean paradigma de diferentes situaciones que puedan encontrarse en portales web reales.

2. *Determinación de grupos de usuarios.* En esta fase, se intenta encontrar grupos de usuarios en los conjuntos de datos artificiales generados en el paso anterior. Para ello se utilizan diferentes algoritmos de agrupamiento: C-Medias, C-Medias Difuso, Algoritmos de Agrupamiento Jerárquico, Algoritmo *Expectation-Maximization*, Mapas Autoorganizativos y Teoría de la Resonancia Adaptativa. Como los conjuntos de datos han sido generados controladamente, puede analizarse la bondad del agrupamiento en cada uno de los conjuntos de datos, pudiéndose de esta manera conocer al algoritmo más adecuado en caso. Por tanto, cuando aparece un conjunto de datos reales, se comparan sus características con las de los conjuntos de datos artificiales estudiados, para de esta manera aplicar a este conjunto real, el algoritmo de agrupamiento que funcionaba más adecuadamente con el conjunto de datos artificial más similar a éste.

3. *Estudio de viabilidad del recomendador.* Esta fase es novedosa dentro de lo que constituye habitualmente el desarrollo de un recomendador, que suele ser desarrollado y posteriormente analizado. En esta tesis se plantea llevar a cabo un estudio de viabilidad que permita determinar la utilidad que aportaría un recomendador colaborativo dentro de un determinado portal web. Este estudio de viabilidad se basa en la caracterización que se hace de los usuarios, separando el efecto de la interfaz del usuario.

4. *Estudio del efecto real de las recomendaciones.* Esta última fase de la metodología no ha sido realizada en esta tesis debido a la falta de datos. Una vez implementado el recomendador debe realizarse un seguimiento de la aceptación de las recomendaciones en el funcionamiento real del portal, ya que de esta manera puede incluirse algún tipo de adaptación que permita incorporar conocimiento al recomendador a partir de los accesos de nuevos usuarios, mejorándose de esta manera las recomendaciones. Además, puede estudiarse en qué medida la interfaz de la recomendación afecta al comportamiento del usuario.

En cuanto a las aportaciones más detacables de la tesis, cabría citar las siguientes:

- *Metodología propuesta globalmente.* Una primera, e importante aportación es la metodología propuesta en sí. Aunque algún otro autor también ha propuesto una serie de fases en el desarrollo de un recomendador (Geyer-Schulz y Hahsler, 2002), éstas no se corresponden con las propuestas en el presente trabajo.

- *Modelo de usuario.* El modelo de usuario basado en web es novedoso para el desarrollo del recomendador, ya que no se basa en el análisis de unos datos para caracterizar al usuario, sino que lo que hace es generar conjuntos de datos basándose en las restricciones y características que puedan darse en la web, constituyendo un modelo de usuario web de propósito general.

- *Técnica pseudo-supervisada de agrupamiento.* Los datos artificiales generados con el modelo de usuario son utilizados para analizar el comportamiento que con ellos tienen diferentes algoritmos de agrupamiento no supervisado. De esta manera, al comparar datos reales con artificiales se puede decidir la técnica más adecuada para los datos reales en función de los resultados alcanzados con los conjuntos de datos artificiales. Como la técnica de agrupamiento es no supervisada pero los conjuntos de datos artificiales son generados controladamente (pudiéndose, por tanto, comprobar el funcionamiento de los algoritmos con estos datos) podemos hablar de una metodología pseudo-supervisada (no supervisada en el algoritmo de agrupamiento y supervisada en la evaluación del agrupamiento de los datos artificiales y la comparación de los datos reales con los artificiales).

- *Utilización de descriptores.* Al trabajar con portales web que pueden ofrecer una enorme cantidad de servicios, es necesario reducir la dimensionalidad del espacio donde trabajan los algoritmos de agrupamiento. Por esta razón, se utilizan descriptores o etiquetas que aglutinan a servicios de similar contenido y que permiten reducir considerablemente la dimensión del espacio donde se agrupan los usuarios.

- *Medidas de evaluación de los algoritmos de agrupamiento con datos artificiales.* Se han propuesto medidas de evaluación del comportamiento

de los algoritmos de agrupamiento usando los conjuntos de datos artificiales. Estas medidas determinan si el número de grupos encontrados es correcto, lo bien que los grupos correctamente encontrados se ajustan a las distribuciones reales de los datos y además, como novedad, se realiza un análisis de normalidad que permite determinar si los algoritmos de agrupamiento han captado las distribuciones estadísticas subyacentes en los datos. Estas medidas de evaluación son, asimismo, utilizadas para determinar el número óptimo de grupos utilizando algoritmos de agrupamiento cuya principal limitación viene dada por los problemas que tienen para encontrar el número adecuado de grupos.

- *Extracción de grupos utilizando Mapas Autoorganizativos.* Dentro de los algoritmos de agrupamiento utilizados, son de destacar las aportaciones realizadas en el sentido de la extracción de grupos y prototipos cuando se utilizan Mapas Autoorganizativos.

- *Viabilidad del recomendador.* Si ya la metodología en sí es una aportación a destacar en el trabajo, mención especial merece el tercer paso de ésta, donde se estudia la viabilidad del recomendador. Este estudio de viabilidad consiste, realmente, en un análisis de predicción basado en una caracterización previa del usuario. De esta manera, se pretende conocer en qué medida las recomendaciones serán efectivas en base a una correcta caracterización de los usuarios y no debido a una determinada interfaz más o menos atractiva. Este estudio debe enfatizarse ya que constituye una importante novedad frente al desarrollo de otros recomendadores.

En resumen, se ha propuesto una metodología que aporta conocimiento en tres puntos fundamentalmente:

- El camino a seguir para un portal real desde los datos correspondientes a los accesos de usuarios hasta el desarrollo de un recomendador óptimo para ese portal.

- Contribución del agrupamiento de usuarios a las futuras recomendaciones; esta contribución se observa en el estudio de viabilidad del sistema de recomendaciones, que resulta fundamental en la metodología propuesta, así como en los análisis que se realizan respecto a la interpretabilidad de los *clusters* y la separación entre ellos, como un

paso crucial para determinar el número de grupos subyacente en la distribución.

- Cómo debe evolucionar el recomendador cuando se disponga del efecto real que las recomendaciones tienen sobre los usuarios.

Respecto a la estructura de la tesis, en el Capítulo 1 se realiza una introducción a la Minería Web focalizada en lo que será de interés para la presente tesis. El Capítulo 2 trata sobre algoritmos de agrupamiento; inicialmente, se explican conceptos básicos, pasándose posteriormente a presentar los algoritmos de agrupamiento utilizados en la presente tesis, así como las posibles implementaciones que éstos tienen y la forma en la que han sido desarrollados y utilizados. El Capítulo 3 versa sobre los sistemas de recomendaciones, haciéndose especial hincapié en la elección que se ha tomado para el recomendador en el presente trabajo. El Capítulo 4 concentra los resultados obtenidos con los conjuntos de datos artificiales; en primer lugar, se describe el modelo de usuario utilizado y los conjuntos de datos artificiales que con él se generan, detallándose a continuación los valores utilizados para los parámetros de diferentes algoritmos de agrupamiento y los resultados alcanzados por éstos, que permiten decidir el algoritmo más adecuado para cada tipo de conjunto de datos. En el Capítulo 5, se realiza un estudio sobre el portal web Infoville XXI, que consta tanto del agrupamiento de usuarios como del estudio de viabilidad que un recomendador colaborativo tendría en este portal, encontrándose que la mejora que aportaría un recomendador de este tipo sería considerable. Por último, en el Capítulo 6, se muestran los conclusiones del trabajo así como la proyección futura de éste. Es de destacar que al final del trabajo, tras la bibliografía, aparece un glosario de términos de uso habitual en la tesis que puede ayudar a su lectura.

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción a la minería de datos web

Resumen del capítulo

Desde un punto de vista muy general, podría decirse que el objetivo de esta tesis es el de extraer la información que puede encontrarse en los accesos de usuarios a sitios web, explotándola en forma de recomendaciones a estos usuarios. Este tipo de objetivos se encuadran dentro de la ***Minería Web (Web Mining)****, que a su vez es una parte de la disciplina conocida como* ***Minería de Datos (Data Mining)****. Estas dos disciplinas comprenden un amplio espectro del conocimiento, y no es la meta de este trabajo realizar un análisis a fondo de ellas. Sin embargo, es necesario al menos ofrecer una visión global, ya que son las grandes áreas donde se enmarca la presente tesis doctoral. Por tanto, en este primer capítulo se realizará un resumen sobre estas disciplinas, incidiendo sobre aquellos aspectos que directamente afectan a los temas tratados en este trabajo.*

1.1. Perspectiva general

La *World Wide Web (WWW)* o simplemente la *web* creció a un ritmo vertiginoso durante la década de los 90, y se espera que su crecimiento continúe durante los próximos lustros. Entre los factores que han contribuido a este crecimiento deben destacarse los avances informáticos y las oportunidades

que la web ofrece para la implantación de negocios a unos precios razonables. Una consecuencia directa de la popularidad de la web es la gran cantidad de información y datos disponibles a través de ella en un poco tiempo. Tal cantidad de información y el hecho de que los usuarios sean anónimos en la mayoría de ocasiones hacen que la extracción de conocimiento sea una labor tediosa. Es por ello que una gran cantidad de técnicas y herramientas han sido desarrolladas últimamente para llevar a cabo esta labor; en ocasiones, se trata de herramientas conocidas de otros campos y que ahora se aplican a éste, y en otras ocasiones se trata de técnicas específicamente diseñadas para la web. En cualquier caso, ambas aproximaciones se encuadran dentro de lo que se conoce como **Minería Web (MW)** o *Web Mining*.

A *grosso modo* puede decirse que la MW es la aplicación a sitios web de la **Minería de Datos (MD)** o *Data Mining*. Como el propio nombre indica, la MD ofrece relaciones deducidas de los datos que puedan aportar un cierto tipo de conocimiento. Las soluciones de la MD son de muchos tipos, como asociación, segmentación, agrupamiento o *clustering*, clasificación, predicción, visualización y optimización. Por ejemplo, la utilización de una herramienta de MD a una base de datos del sitio web puede segmentar la base de datos en grupos únicos de visitantes, cada uno de ellos con conductas individuales. Estas mismas herramientas realizan pruebas estadísticas sobre los datos y los pueden dividir en múltiples segmentos de mercado si, por ejemplo, se trata de datos pertenecientes a un sitio web de comercio electrónico. Estos tipos de herramientas de minería de datos arrojan generalmente sus resultados en forma de árboles de decisión gráficos, reglas *If-Then*, recomendaciones, etc.

Normalmente, los datos de un sitio web deberán ser depurados y preparados antes de empezar ningún tipo de análisis de MD. Por ejemplo, los archivos *log* pueden ser bastante redundantes, ya que un solo "click" genera un registro no sólo de ese HTML, sino también de todos los gráficos de esa página. Sin embargo, una vez desarrollada una plantilla, una macro, o un procedimiento para la generación de una sola visita, se pueden introducir los datos en un formato de base de datos a partir del cual se pueden llevar a cabo manipulaciones adicionales y filtrados.

1.2. Minería de Datos

1.2.1. Introducción

Los últimos avances en almacenamiento de datos, desarrollo tecnológico y de redes de ordenadores han posibilitado que los datos sean almacenados digitalmente en Bases de Datos (BD) de una manera eficiente, posibilitando su procesado, y la eventual extracción de información a partir de ellos. La MD se basa en el uso de técnicas que permitan descubrir y extraer conocimiento de estos datos. El conocimiento que se extraiga puede ser aprovechado para la toma de decisiones (Fayyad y Uthurusamy, 1996). La MD se ha convertido en un campo muy prolífico de investigación y publicación, debido fundamentalmente a la amplia disponibilidad de bases de datos de gran tamaño. La información y conocimiento obtenidos usando MD pueden aplicarse a un gran abanico de campos.

Desde un punto de vista cronológico, puede empezar a hablarse de MD desde los años 60, aunque realmente se trataría más de datos que de minería; de hecho, durante estos años lo que se desarrollaron fueron sistemas para el almacenamiento y recuperación de datos. No obstante, la ineficacia en los formatos de almacenamiento provocó que no se realizaran progresos significativos, ya que el concepto de base de datos no estaba aún desarrollado por lo que los datos se almacenaban en ficheros de manera individual. Fue ya en los años 70 cuando los avances en redes de ordenadores y bases de datos relacionales, supusieron un gran empuje para la MD. A mediados de los años 80, las bases de datos empezaron a dejar de limitarse únicamente al almacenamiento y comunicación de datos, para empezar a incorporar técnicas de procesado y modelado de datos, en lo que fueron los primeros pasos hacia la MD, tal y como la conocemos ahora. Es ya en los años 90, cuando la MW, una de las principales aplicaciones de la MD actualmente y objeto de interés de la presente tesis, empieza a desarrollarse, y donde la minería de datos se asienta como un área de conocimiento e investigación; aunque es difícil determinar un evento que marque el principio de la MD como área de conocimiento, un hecho a tener en cuenta es que en los años 90 es cuando empiezan a proliferar la existencia de conferencias, cursos y seminarios sobre este tema.

Dentro del proceso de MD, también llamado en ocasiones Descubrimiento de

Conocimiento en bases de datos (*Knowledge Discovery in Databases, KDD*) (Chang, Healey, McHugh y Wang, 2001), se suelen contemplar los siguientes pasos:

- *Limpieza de datos.* Este paso limpia los datos "sucios". Esto incluye datos incompletos, ruidosos o inconsistentes. Este paso es extremadamente importante, ya que los datos "sucios" implican un análisis inexacto y unos resultados, por tanto, incorrectos. Consideramos que los datos son "sucios" o "ruidosos" cuando existe una importante contribución aleatoria a los mismos, la cual no aporta conocimiento alguno.

- *Integración de datos.* En esta fase, se trata de combinar datos de diferentes fuentes incluyendo múltiples BDs, que pueden tener diferentes contenidos e, incluso, formatos.

- *Transformaciones.* Una vez limpios y filtrados los datos, éstos han de transformarse a un formato consistente y apropiado para las tareas que se desee hacer a continuación. Aquí se incluyen los usuales preprocesados de datos, como la normalización o el suavizado.

- *Reducción de datos.* Este paso reduce el tamaño de los datos intentando mantener la estructura e información del conjunto de datos original. Habituales estrategias en este sentido son la reducción de la dimensionalidad (eliminando entradas irrelevantes) o la compresión de datos (fundamentalmente, eliminando patrones redundantes).

- *Extracción y procesado de patrones.* Este punto recoge la complicada labor de convertir los datos en patrones, que son procesados con un determinado objetivo (agrupación de datos, predicción de acciones futuras, etc.). Los patrones son estructuras de datos que pueden ser procesadas directamente por los algoritmos de MD utilizados.

- *Evaluación.* En esta parte se evalúa el procesado realizado en el anterior apartado, y se lleva a cabo la extracción de conocimiento que sea posible.

Los cuatro primeros pasos se engloban dentro del preprocesado de datos y los dos últimos dentro de la extracción de conocimiento. Dependiendo de la

técnica elegida para la extracción de conocimiento, el tipo de preprocesado suele verse afectado. Como se verá en posteriores capítulos, esta tesis se centra en técnicas de *clustering* para la extracción de información, debido a los particulares objetivos que se persiguen. No obstante, la MD también se usa para realizar otro tipo de tareas, como la extracción de reglas (Agrawal, Imielinski y Swami, 1993; Agrawal y Srikant, 1994; Mannila y Toivonen, 1994), o la clasificación y predicción, usando herramientas tales como árboles de decisión (Murthy, 1998), clasificación Bayesiana (Heckerman, 1996), etc.

1.2.2. El agrupamiento o *clustering*

En este punto se va a realizar una corta introducción al *clustering*, ya que estas técnicas se desarrollarán más profundamente en posteriores capítulos. Se conoce como *clustering* al proceso por el cual un algoritmo realiza agrupaciones de los datos. Estas agrupaciones son conocidas como *clusters* (Kaufman y Rousseeuv, 1990; Theodoridis y Koutroumbas, 1999). El *clustering* tiene un amplio rango de aplicaciones prácticas, desde el procesado de imágenes, a las aplicaciones en bolsa, marketing o reconocimiento de patrones.

Contrariamente a la clasificación, donde la etiqueta de cada clase es conocida para cada objeto, en el *clustering* no existen estas etiquetas predefinidas, por lo que el tipo de aprendizaje que usa es *no supervisado*. El objetivo fundamental del *clustering* es el de descubrir las relaciones de similaridad y disimilaridad que existen en un determinado conjunto de datos. Para ello, básicamente, lo que hace es identificar regiones densas y dispersas en el espacio definido por el conjunto de datos. Dentro del *clustering*, destacan las siguientes técnicas:

- *Clustering por partición.* El método de partición separa n objetos en k grupos llamados particiones, donde cada partición representa un *cluster*. Este método asume que (1) $k \leq n$, (2) cada partición debe contener al menos un objeto, y (3) cada objeto debe ser miembro de una o más particiones (en principio, está permitido que un objeto pertenezca a más de una partición con diferentes grados de pertenencia, lo que suele hacerse utilizando técnicas basadas en **Lógica Difusa (LD)**) (Bezdek y Pal, 1992; Kaufman y Rousseeuv, 1990; Theodoridis y Koutroumbas, 1999). Estos métodos suelen crear una partición inicial, que es poste-

riormente mejorada usando técnicas iterativas. El proceso iterativo mueve los objetos de una partición a otra. Suelen utilizarse prototipos para definir cada partición con el fin de simplificar la notación, que podría ser bastante farragosa en el caso de arrastrar todos los objetos o patrones agrupados bajo una misma partición. Un ejemplo de técnica de este tipo es el algoritmo de las C-medias (Theodoridis y Koutroumbas, 1999), o el método E-M (*Expectation-Maximization*) (Lauritzen, 1995).

- *Clustering jerárquico.* Los métodos jerárquicos de *clustering* realizan una descomposición jerárquica del conjunto de datos, usando técnicas aglomerativas o divisivas (Guha, Rastogi y Shim, 1999; Karypis, Han y Kumar, 1999; Theodoridis y Koutroumbas, 1999; Kaufman y Rousseeuv, 1990).

- *Métodos basados en "densidad".* Los métodos basados en densidad emplean la noción física de densidad (Ester, Kriegel, Sander y Xu, 1996; Hinneburg y Keim, 1998). El funcionamiento se basa en encontrar *clusters* haciendo crecer un determinado *cluster* inicial siempre que la densidad en los patrones cercanos o vecinos sea mayor que un determinado umbral. Dado un *cluster* c, el umbral de densidad requiere que cada patrón en c contenga un mínimo número de otros patrones en un radio predeterminado. Dentro de estos métodos basados en densidad podríamos situar también los Mapas Topográficos Generativos (*Generative Topographic Mapping, GTM*) (Bishop, Svensén y Williams, 1998), que realmente son una reformulación probabilística de los Mapas Autoorganizativos (Kohonen, 1997), que por extensión podrían situarse también en este grupo de métodos, aunque se preferirán ubicar en el último punto, que comprende el resto de métodos no considerados anteriormente.

- *Otros métodos.* Otros métodos son por ejemplo los basados en cuantización, que discretizan el espacio donde se quiere hacer el *clustering* en un número finito de categorías formando una estructura de cuadrícula, y realizando el *clustering* sobre esa estructura "cuadriculada" (Sheikholeslami, Chatterjee y Zhang, 1998; Wang, Yang y Muntz, 1997). También son destacables los métodos basados en modelos (Shavlik y Dietterich, 1990), que relizan un modelado de cada *cluster* y en-

cuentran el mejor ajuste del modelo y los basados en redes neuronales, básicamente Mapas Autoorganizativos (Kohonen, 1997) y redes basadas en la Teoría de la Resonancia Adaptativa (Carpenter y Grossberg, 1987). Además, exiten algoritmos de *clustering* que se basan en combinar diferentes técnicas para mejorar el resultado final (Agrawal, Gehrke, Gunopulos y Raghavan, 1998).

1.3. Minería Web

1.3.1. Tipos de minería web

No es sencillo encontrar una definición para el término MW. De hecho, se suele usar la denominación MW para catalogar tres tipos de actividades considerablemente diferentes. Todas estas actividades se enmarcan dentro de la MD y, además, están relacionadas con la web, pero los datos que son objeto de la minería son diferentes (Linoff y Berry, 2001; Chang et al., 2001). Estas tres actividades son las siguientes:

- *Minería sobre la estructura de la web.* Se enmarcan aquí los procesos cuyo objeto es extraer información sobre la topología de la web, es decir, de los enlaces entre las páginas. Se trataría por tanto, de responder a preguntas del tipo *¿qué páginas son usualmente accedidas desde otras páginas?*, o *¿cuáles son las páginas origen que llevan a otras determinadas páginas?*.

- *Minería sobre los usuarios de los datos.* Este tipo de minería es el objeto de la presente tesis y será más ampliamente revisada en la sección 1.3.3. Básicamente, se trata de extraer información acerca del comportamiento de los usuarios en la web. En este caso, se responde a preguntas como *¿qué tipo de usuarios acceden a qué páginas?*, o *¿cuál será el próximo enlace al que accederán?*.

- *Minería sobre el contenido de los datos.* Aquí se engloban aquellos procesos cuya finalidad es extraer información a partir de la información contenida en un determinado portal. Esta información puede ser en forma de texto, imágenes, sonidos, o cualquier otro tipo de información que pueda estar contenida en el portal. La información que

puede aportar este tipo de MW puede responder a preguntas como *¿qué páginas están en francés?*, *¿cuáles están relacionadas con bailes tradicionales?*, etc.

Aunque esta clasificación entre diferentes tipos de MW pueda parecer algo arbitraria en principio, las diferencias entre ellas están basadas en consideraciones prácticas importantes, como la fuente o disponibilidad de los datos apropiados. Por ejemplo, el hecho de que un conjunto particular de páginas web pertenecientes a diferentes portales conectados mediante hipervínculos estuviera públicamente disponible, permitiría realizar minería sobre la estructura de los datos, sobre la particular topología de la red que explicita las conexiones entre las diferentes páginas. Sin embargo, puede que las empresas que explotan los portales web en cuestión manifiesten reticencias a hacer públicos los datos acerca de cómo los usuarios se mueven desde unos enlaces a otros, qué enlaces suelen implicar un final en la navegación del usuario y cuáles implican posteriores accesos del usuario. En un marco de trabajo como éste, podríamos por tanto, realizar minería de datos sobre la estructura de la red pero no sobre los usuarios. En general, la minería que implica datos de usuarios suele enfrentarse con el hándicap de que las empresas no suelen estar dispuestas a facilitar estos datos, porque justamente la información más útil, particularmente en el tema económico, reside en los usuarios de la web y en el uso que de ella hacen.

1.3.2. La web, vista desde la perspectiva de la minería de datos

La *World Wide Web* está formada básicamente por páginas unidas o conectadas mediante enlaces. Una determinada página web consiste de un determinado contenido (texto, imágenes, enlaces a otras páginas, etc.). Por otro lado, un servidor web se encargará de dar acceso a estos contenidos. Además, una página puede consistir de subunidades llamadas marcos, o *frames*, aunque desde el punto de vista utilizado en esta tesis, un marco será equivalente a una página web.

El material de partida para llevar a cabo la minería de estructura de la web es el conjunto de hiperenlaces que unen los contenidos de diferentes páginas web. Asimismo, el material de partida para hacer la minería de contenidos

web consiste básicamente en el texto[1] almacenado en los ficheros accessibles por web, cuyo número puede ser extremadamente elevado. Tanto la minería de estructura como la de contenidos funcionan con representaciones estáticas ideales de la web, es decir, utilizan los páginas y enlaces tal y como aparecen en un momento determinado. Por supuesto, tanto el contenido que aparece en las páginas web, como la conexión entre páginas cambia constantemente, por lo que esta aproximación estática puede no ser correcta, o al menos, puede no estar todo lo actualizada que sería deseable.

La representación más usual para la minería de estructura web se consigue mediante un diagrama de grafos que refleja el movimiento entre enlaces al navegar de una página a otra. Un diagrama ideal debería ser capaz de mapear todos los enlaces que conectan todos los documentos de la web unos con otros. Por otro lado, la representación más usual para la minería de contenidos web viene dada por un índice o tabla. El índice ideal debería relacionar todas las frases, palabras, expresiones, tonos e imágenes de la web con las páginas que contienen esta información. En su expresión más pura y formal, la minería de contenidos web no requiere conocimiento de los enlaces entre documentos, y la minería de estructura web tampoco requiere conocer la información contenida en los documentos (Linoff y Berry, 2001).

Es decir, que según se acaba de comentar, ni la minería de estructura ni la de contenidos requieren un conocimiento del comportamiento del usuario final de la web. La minería de estructura pone de manifiesto las páginas que se pueden acceder desde una determinada página, pero no la **cantidad de individuos** que realizan el recorrido de unas a otras páginas. Por otro lado, la minería de contenidos ofrece información sobre el tema de la página, pero no sobre **quién** la está leyendo. Por ejemplo, la minería de contenidos web puede servir para obtener aquellas páginas web que tratan sobre "casas rurales en la Comunidad Valenciana"; sobre este mismo ejemplo, la minería de estructura web puede ser capaz de organizar todas las páginas encontradas. Pero si lo que se quiere saber es, por ejemplo, quién lee estas páginas, cómo éstas afectan a lo que en el futuro pueda hacer un usuario en cuanto a alojarse en una de estas casas, lo que distingue a un usuario que prefiere las casas del interior de Castellón a las del interior de Valencia, u otra cuestión

[1]Realmente, no se trata de texto plano, ya que tiene embebidos los *tags* o etiquetas HTML y XML que se utilizan para darle al texto el aspecto y la utilidad deseada en la página web.

relacionada, entonces es necesario recurrir a otro tipo de MW. Este tercer tipo es, como se ha comentado anteriormente, el que está centrado en los usuarios web y el uso que de ésta hacen dichos usuarios.

Del mismo modo que existe un diagrama de flujos ideal para la minería de estructura y un índice ideal para la minería de contenidos, también existe una representación ideal para la minería de usuarios. Ésta podría ser una biblioteca de perfiles de usuario actualizada de todos los usuarios de la web. Cada uno de estos perfiles registraría el histórico de las interacciones de un usuario individual con la web, incluyéndose aquí, entre otras cosas, los servicios accedidos, la secuencialidad de la navegación, los documentos leídos o los objetos comprados.

Desgraciadamente, la representación ideal para la minería de usuarios en un contexto real resulta extremadamente complicada de obtener, y mucho más difícil en particular que los otros dos tipos de MW, ya que la información necesaria para construir un diagrama de grafos o un índice suele estar libremente accesible en la web (es de he hecho la finalidad de la WWW), mientras que los datos referentes a los usuarios son realmente difíciles de conseguir, tanto porque están repartidos en diferentes ficheros como porque en la mayoría de las ocasiones, las empresas o instituciones responsables de explotar las webs no suelen estar dispuestas a compartir los datos de sus usuarios. A esto último, habría que añadir las cada vez más restrictivas leyes de protección de datos, que exigen un total anonimato de los mismos, y que aún pueden hacer retractarse más a las empresas o instituciones responsables. El resultado de esto es que la minería de usuarios suele restringirse a modelar el comportamiento de los usuarios que visitan una web o una red particular. En esta tesis se intenta dar un paso más, ya que como se verá en posteriores capítulos se propone una estrategia que sea válida para diferentes sitios web, particularizándose posteriormente para un portal del que sí se dispone de datos reales.

1.3.3. Minería de usuarios web

Introducción

Analizar y entender el comportamiento de los usuarios que acceden a un sitio web son tareas de gran utilidad e interés. Aunque la estructura y con-

tenidos de la web cambia constantemente, para propósitos de MD se suele realizar la aproximación estática que se restringe a responder preguntas del tipo: *¿Cuáles son las páginas más parecidas a otra página en un instante determinado?*. No obstante, el tipo de análisis que se plantea en este apartado sí que tiene en cuenta el tiempo y la evolución de los comportamientos de los usuarios. El marco de tiempo considerado puede ser tan corto como una sesión, o tan largo como varios años, pero en cualquier caso incorpora la dinámica subyacente en la web de una manera natural. En esta tesis, se analizarán dos tipos de parecido: por un lado, los algoritmos de agrupamiento intentarán encontrar parecidos entre los usuarios para establecer grupos que determinen tipos diferentes de comportamiento y, por otro, parte de la metodología que se propone se basa en comparar la página web real sobre la que se desea llevar a cabo las recomendaciones con diferentes cojuntos de datos artificiales generados previamente con un modelo de usuario

Incluso cuando nuestro objetivo es la estructura o los contenidos, el estudio de los usuarios puede ser de gran ayuda. Por ejemplo, un atributo útil de un enlace o una página es su popularidad, lo que se mide a través de la gente que accede sobre el enlace o la página en un período de tiempo determinado. Y cuando el objetivo es entender a los usuarios de la web, es obvia la necesidad de utilizar este tipo de MW. Los patrones de usuarios pueden ser obtenidos a diferentes niveles, desde la secuencia de *clicks* realizada por los usuarios dentro de una sesión por un usuario individual hasta el conjunto de patrones que registran las compras de determinados productos por un conjunto de usuarios durante un período de tiempo. Esta información suele ser utilizada para obtener perfiles de los usuarios que, posteriormente, pueden ser utilizados para desarrollar productos personalizados o recomendaciones adecuadas para estos usuarios.

La minería de usuarios web tiene muchas aplicaciones, desde mejorar el diseño del sitio web hasta optimizar las relaciones entre clientes y responsables del sitio web. Conforme los objetivos se hacen más ambiciosos, los datos deben ser más numerosos y variados. Los siguientes apartados realizan una rápida revisión sobre los tipos de minería de usuarios web más utilizados. Algunos de ellos son objeto principal de esta tesis, y se analizarán con mayor profundidad en posteriores capítulos.

Análisis de la secuencia de navegación

El análisis de la secuencia de navegación empieza con los ficheros *log*. Los ficheros *log* de usuarios se almacenan como ficheros de texto en un directorio determinado por el servidor web. Estos ficheros se generan mediante el formato estándar especificado como parte del protolocolo HTTP que muchos servidores web utilizan y que está formado por los siguientes campos (`http://www.w3c.org/`):

- Número IP o nombre del *host* remoto que realiza el acceso.
- Nombre del usuario que accede remotamente.
- Nombre de usuario bajo el cual se ha autentificado.
- Fecha y hora en la que el usuario realiza la solicitud del servicio.
- La solicitud como se realizó exactamente por el cliente.
- El código de estado HTTP que se devolvió al cliente.
- La cantidad de información (en bytes) que se transfiere.

Una típica entrada del servidor web *Apache* (`http://www.apache.org/`) tiene un aspecto como:

```
 www.sample.org - - [15/Apr/2000:00:03:24 -0400]
‘‘GET /index.html HTTP/1.0’’ 200 11371
```

La cadena de navegación se define como el conjunto de ficheros explícitamente solicitados por un visitante del sitio web a través de su accesos sobre los enlaces correspondientes. Aunque se considera que un sitio web está formado por una serie de páginas web, los ficheros *log* realmente aportan una información ligeramente diferente, a saber; los servicios solicitados por el usuario dentro de una determinada página web. La razón para esta diferencia estriba en que lo que aparece en un navegador web como una página web simple es realmente un objeto complejo que incluye numerosos marcos, cada uno de los cuales muestra los contenidos de un fichero HTML diferente. Cada fichero HTML, a su vez, típicamente contiene referencias a múltiples ficheros de imagen. Para construir la página web que se ve en el navegador, cada uno de estos objetos ha de ser solicitado a un servidor web. A menudo

más de un servidor puede estar implicado, por ejemplo, un servidor remoto suministrando *banners*, una batería de servidores locales suministrando texto e imágenes, y una aplicación servidor que suministra contenido del tipo "carro de compra".

Como resultado, cuando un usuario "hace click" sobre un enlace para acceder a una determinada página web, esto se transforma en múltiples accesos al sitio web, uno para cada objeto constituyente de la página. Como estos objetos puede ser suministrados por diferentes servidores, los accesos pueden ser registrados a través de diferentes ficheros. Además, como los servidores están registrando accesos de numerosos navegadores diferentes, una sesión determinada no aparece registrada en el fichero de manera secuencial y continua. Por tanto, antes de que los datos del fichero *log* puedan ser usados para aprender algo acerca del comportamiento de los usuarios, es necesario realizar una gran labor de limpieza que incluye, básicamente, cinco pasos (Linoff y Berry, 2001): filtrado, eliminación de usuarios falsos, identificación de usuarios, determinación de sesiones y determinación de la secuencia.

Filtrado Los datos que aparecen en un fichero *log* son increíblemente voluminosos. Un sitio web con un tráfico elevado puede tener cientos o incluso miles de ordenadores sirviendo páginas simúltaneamente. De hecho, dos solicitudes, es decir, dos accesos a servicio realizadas por el mismo usuario pueden ser procesadas por diferentes servidores y registradas en ficheros diferentes. Estos *logs* deberán unirse antes de que cualquier secuencia de navegación con sentido pueda formarse.

Una vez que los datos han sido recogidos, el primer paso previo para realizar un análisis de la secuencia de navegación consiste en eliminar aquellos registros que no son necesarios. Muchos de los accesos registrados en el *log* son solicitudes de imágenes que, desde nuestro punto de vista, son solamente parte de la página HTML que las contiene. Por tanto, un primer paso muy sencillo consiste en eliminar aquella información que registra accesos a ficheros cuyas extensiones son '.gif', '.jpeg', '.png', ... Este filtrado da como resultado unos datos a nivel de página web, en una primera aproximación. Típicamente, este filtrado da como resultado una reducción en el tamaño del fichero *log* del 90 % respecto a los datos iniciales.

Eliminación de usuarios falsos Entendemos por usuarios falsos aquellos que no son útiles para nuestros propósitos, ya que realmente no son usuarios "reales". Por ejemplo, como veremos más adelante al estudiar el portal web *Infoville XXI*, existen registros correspondientes a los administradores y desarrolladores del portal, los cuales evidentemente no son los usuarios que nos interesa caracterizar para mejorar el portal mediante la individualización de servicios. Además, los motores usados por buscadores y alguna otras aplicaciones de minería de datos tienen como base de su funcionamiento el tener los contenidos de las páginas web indexados y actualizados. Esto hace que existan muchos usuarios cuyas sesiones están formadas por un único acceso a servicio, lo cual, dicho sea de paso, sería bastante negativo para el sitio web si representara un comportamiento real de usuarios. Por tanto, es interesante realizar una etapa de preprocesado basada en identificar a estos usuarios "ficticios", eliminándolos del fichero de datos.

Identificación de usuarios Antes de poder realizar cualquier minería de usuarios web es necesario disponer de datos a nivel de sesiones de usuario. Pero antes de poder reconocer o identificar sesiones, se necesita realizar una identificación de usuarios. Este paso se lleva a cabo en dos niveles: en un primer nivel se identifican las peticiones de páginas realizadas por el mismo usuario durante una visita para poder así crear lo que sería la sesión. El segundo nivel consistiría en reconocer a un usuario dentro de sus múltiples visitas a un determinado sitio web, ya que de esta manera puede analizarse el comportamiento del usuario a lo largo de días, meses, o años.

La mejor solución para la identificación es disponer de usuarios que se identifican al entrar al sitio web mediante un "nombre de usuario" y "contraseña". Sin embargo, la navegación web se lleva a cabo normalmente de forma anónima, por lo que hay que utilizar diferentes estrategias para averiguar que dos accesos distintos son realizados por un mismo ususario. Cada vez que se accede a una página, el fichero *log* registra tanto la dirección IP desde la que se accede como una cadena de caracteres que identifica el agente (navegador) utilizado. Desafortunadamente, esta información es a menudo insuficiente para identificar positivamente a un usuario ya que, por ejemplo, todos los empleados de una empresa o los miembros de una familia pueden acceder a través de la misma IP. Cuando se está en esta situación, existen algunos "trucos" que pueden ayudar a discernir a unos usuarios de otros; por ejem-

plo, páginas accedidas por una misma IP pero diferentes navegadores pueden corresponder a diferentes usuarios, o accesos a secuencias de páginas no vinculadas entre ellas sugieren que los accesos son realizados por diferentes usuarios.

En definitiva, resulta bastante complicado reconocer a un mismo usuario entre los diferentes servicios a los que accede dentro de una misma sesión, y mucho más compleja resulta, evidentemente, la identificación cuando se tiene en cuenta la evolución temporal. El único método realmente fiable consiste en que los usuarios lleven a cabo un registro en el sitio web, pero es algo en lo que no se puede confiar siempre ya que depende de la cooperación de los usuarios, que pueden verse retraídos a acceder al sitio cuando hay que rellenar algún tipo de formulario.

En ausencia de un formulario de registro, muchos sitios intentan reconocer a usuarios que previamente ya han accedido a ese sitio mediante el uso de *cookies*. De todos modos, para que las *cookies* cumplan esta función han de cumplirse varios requisitos: que el usuario esté utilizando el mismo navegador en el mismo ordenador, y que no haya eliminado o deshabilitado las *cookies*. Además, este sistema puede llevar a error en la identificación cuando, por ejemplo, son dos miembros diferentes de la misma familia los que acceden, y la *cookie* que se envió desde el servidor al primer usuario es reconocida cuando es el segundo usuario quien accede. Otros problemas adicionales son que un mismo usuario puede acceder a un determinado portal desde casa, el trabajo, etc. Esto no quiere decir que las *cookies* sean inútiles, pero su funcionamiento está optimizado cuando actúan identificando a usuarios previamente registrados, a los que se les personaliza la apariencia del portal o se les realiza algún tipo de recomendación.

Determinación de sesiones Aquí se considera el proceso por el cual se determina que una serie de servicios solicitados por un mismo usuario pertenecen a una única visita al sitio o portal web. Las sesiones son muy importantes ya que reflejan la percepción del usuario acerca de su visita al portal. De hecho, en ocasiones, las sesiones son también llamadas "visitas".

Para poder crear las sesiones, el primer paso será encontrar todos aquellos servicios solicitados por un mismo usuario y, a continuación, agruparlos en sesiones utilizando heurísticas tales como considerar la máxima longitud de

tiempo entre dos accesos dentro de una misma sesión. La mejor manera de realizar esta tarea es mediante una aplicación que cree un identificador de sesión la primera vez que un determinado visitante del portal accede al mismo. Esto puede hacerse mediante una *cookie* o alterando las URLs para incluir un identificador de sesión.

Evidentemente, cualquier método de determinación de sesiones basado únicamente en datos procedentes de ficheros *log* tiene el problema de que no existe ninguna manera de saber realmente cuánto tiempo está el usuario en una determinada página, particularmente la correspondiente al fin de la visita al portal. Esto es así porque puede que el navegador haya sido minimizado y el usuario no esté realmente consultando el servicio que el fichero está registrando. Por tanto, el algoritmo de determinación de sesiones debe decidir cuándo la visita finaliza independientemente de lo que esté registrando el fichero *log*. Evidentemente, esta decisión puede sesgar los datos que posteriormente se utilicen para la minería web.

Determinación de la secuencia Un problema que afecta tanto a la sesionización como a cualquier tipo de análisis de la secuencia de navegación basados en ficheros *log* viene del hecho de que muchos accesos a servicios no quedan registrados en el *log* del servidor. Esto es debido, principalmente, al uso que se hace de las cachés a diferentes niveles. Por un lado, el propio navegador mantiene una caché con las páginas recientemente visitadas. El usuario puede elegir la longitud de tiempo que se desea almacenar en la caché, típicamente unos días. Si un usuario solicita una página que está almacenada en la caché, el navegador no se molesta en solicitarla al servidor así que no se registra ninguna solicitud en el *log*. Además, en algunas redes locales, existe un nivel adicional de caché que viene dado por el servidor *proxy*; usualmente, todas las solicitudes de servicios que vengan de usuarios de la red local van en primer lugar al correspondiente servidor *proxy*, y si éste tiene una copia de la página porque algún usuario de esa red ha accedido a ella recientemente, el acceso se produce sin dejar rastro sobre el servidor real de la página en cuestión.

Los efectos de la caché sobre los *logs* del servidor son diversos:

1. El número de páginas solicitadas es bajo ya que algunas páginas que son realmente solicitadas no están en el *log*.

2. Las solicitudes de diferentes usuarios con un *proxy* en común no pueden distinguirse a través de la IP ya que todos ellos acceden a través de la IP del servidor *proxy*.

3. Las secuencias guardadas son incompletas. Para entenderlo mejor, consideremos un ejemplo. Supongamos que un usuario comienza su navegación en la página A, desde aquí accede a B y de nuevo vuelve a A a través del botón "atrás" del navegador, y desde aquí accede a la página C. El *log* registrará la solicitud de la página C a continuación de la de la página B, ya que el retorno a A no se registrará al usar el navegador una copia de la caché local. Si no existe ningún enlace entre B y C, puede inferirse al retorno que se produjo a A. Este tipo de procesos son los que se engloban en lo que llamamos "determinación de la secuencia", que básicamente se trata de completar el camino seguido por el usuario durante su visita al portal web en cuestión. Como prácticamente todas las técnicas de minería web, esta determinación de la secuencia no es, ni mucho menos, un proceso exacto. En el ejemplo considerado, podría interpretarse el registro del *log* como que el usuario accedió a B desde A y, a continuación, escribió la dirección correspondiente a C en el campo reservado a tal efecto en el navegador. Como esta segunda interpretación supone un conocimiento por parte del usuario de la URL de C, la explicación es algo menos plausible, pero igualmente válida, quedando, por tanto, la explicación a un juicio sin duda subjetivo.

Para acabar con este punto, debe decirse que el uso de páginas dinámicas (JSP[2] y ASP[3] básicamente) hace que éstas no se registren en la caché, ya que cada página creada dinámicamente es única. Actualmente, la mayoría de sitios web de comercio electrónico, por ejemplo, trabajan con páginas dinámicas, por lo que los accesos a ellos no quedan registrados en la caché.

Extracción de conocimiento

Como hemos estado viendo, los ficheros *log* registran las solicitudes de páginas realizadas por los diferentes usuarios de un sitio web, aunque la información que en ellos aparece es incompleta y confusa por lo que debe realizarse

[2] *Java Server Pages.*
[3] *Microsoft's Active Server Pages.*

un preprocesado que transforme los datos iniciales en datos organizados por usuarios y sesiones. Una vez que los datos están organizados y son manejables, se pasa a la extracción de conocimiento, que es el último paso de la minería web, la obtención de información útil y utilizable a partir de los datos de usuarios web. Esta información puede utilizarse posteriormente para realizar labores de personalización, predicción, recomendación, etc. En esta tesis nos centraremos en la obtención de información a partir de técnicas de agrupamiento no supervisado y su aplicación a sistemas de recomendaciones. Estos dos aspectos serán ampliamente explicados en los dos próximos capítulos, pero a continuación se realizará una rápida visión sobre estas y otras técnicas también utilizadas (Chang et al., 2001).

Análisis estadístico El análisis estadístico es la técnica más usada para extraer y ofrecer información acerca de los usuarios de un portal web. Muchas herramientas de análisis de ficheros *log* utilizan este tipo de técnicas para llevar a cabo sencillos análisis acerca del tráfico en el portal, incluyendo servicios más solicitados, tamaño medio de los ficheros transferidos, tráfico diario, número de visitantes del sitio web, informe de errores, etc. Esta información puede utilizarse para la monitorización de usuarios, comprobaciones de seguridad, ajuste del rendimiento y mejora del portal en general.

Extracción de reglas Con la extracción de reglas en la minería de usuarios web, nos referimos a la identificación de conjuntos de páginas que son accedidas en mayor o menor medida. Usualmente, suele asignarse un valor de importancia a cada página, normalmente dependiendo del número de accesos registrados; este valor debe superar un cierto umbral predefinido para que la página sea tenida en cuenta. Las reglas obtenidas representan conocimiento acerca del comportamiento del usuario, lo cual es de indudable utilidad.

Agrupamiento El agrupamiento, comentado ya en la sección 1.2.2 y que será ampliamente explicado en el siguiente capítulo, comprende un conjunto de técnicas no supervisadas. Por tanto, puede usarse en minería de usuarios web para obtener conocimiento acerca del comportamiento de los usuarios en el portal. Básicamente este conocimiento puede llevarse a cabo en dos sentidos: agrupar sesiones de usuarios (es decir, encontrar similitudes y difer-

encias entre las cadenas de navegación de los diferentes usuarios) y agrupar usuarios en función de los servicios accedidos. En esta tesis, nos centramos en el segundo de estos dos aspectos.

Clasificación Contrariamente al agrupamiento, la clasificación se usa cuando las categorías sí que están definidas previamente. Dado un patrón de usuario, éste puede clasificarse en diferentes categorías.

Extracción secuencial de patrones En este último apartado nos referimos a la minería de patrones que ocurren, frecuentemente, en episodios o sesiones de usuario. Ciertos usuarios pueden acceder a ciertas páginas con una determinada periodicidad (por ejemplo, un aficionado al fútbol puede consultar el servicio de "Resultados y clasificaciones" de un diario deportivo todos los lunes por la mañana). Patrones periódicos pueden, por tanto, encontrarse con este tipo de técnicas que pueden ser útiles para determinar y obtener tendencias en el comportamiento de los usuarios.

Capítulo 2

Algoritmos de agrupamiento

Resumen del capítulo

En este capítulo se presentan los algoritmos de agrupamiento que serán utilizados con posterioridad para encontrar parecidos y diferencias entre usuarios web. En primer lugar, se realiza una introducción a la noción de agrupamiento y a los conceptos relacionados con éste. En particular, se hace especial énfasis en las distancias que son utilizadas para medir las diferencias entre usuarios o entre grupos de usuarios. Como se verá en el desarrollo del capítulo, la utilización de una u otra distancia será crucial para el resultado final del agrupamiento. Una vez explicados los diferentes tipos de algoritmos de agrupamiento existentes, el capítulo se centra en describir con mayor detalle los seis que se han utilizado en esta tesis: C-Medias, C-Medias Difuso, Algoritmos de Agrupamiento Jerárquico, Algoritmo Expectation-Maximization *para el ajuste de una mezcla de distribuciones Gaussianas, Mapas Autoorganizativos de Kohonen y Teoría de la Resonancia Adaptativa. Las contribuciones fundamentales de la tesis en este capítulo son dos: los métodos de extracción de grupos en mapas de Kohonen y la estimación del número de grupos idóneo a través del estudio de la bondad y normalidad del agrupamiento obtenido.*

2.1. Introducción

Los algoritmos de agrupamiento se enmarcan dentro de los sistemas basados en aprendizaje no supervisado, es decir que no se conoce la clase a la que

pertenecen los patrones de entrenamiento, contrariamente al aprendizaje supervisado donde sí que se conoce la relación entre patrones y clases; de hecho, el aprendizaje supervisado se basa en minimizar los errores en la asignación de los patrones a las clases conocidas. Cuando no se dispone de esta información, la única solución es recurrir al aprendizaje no supervisado, que se basa en determinar la organización de los patrones en grupos o *clusters*, que permiten descubrir similaridades y diferencias entre los patrones, así como extraer conclusiones sobre el problema (Theodoridis y Koutroumbas, 1999).

Partiendo de un conjunto de patrones formado por vectores l-dimensionales, siendo l el número de características o variables discriminantes utilizadas para encontrar los *clusters*, una labor de agrupamiento usualmente involucra los siguientes pasos (Theodoridis y Koutroumbas, 1999):

- *Selección de características.* Las características deben seleccionarse de manera apropiada para codificar tanta información útil para nuestros objetivos como sea posible. Un objetivo vital en este sentido es el de minimizar la información redundante entre características, por lo que un análisis de correlaciones entre variables suele hacerse necesario para eliminar variables con información redundante. Además, en el caso de que los valores numéricos de las características sean muy diferentes, es interesante realizar una normalización que haga que todas las variables presenten, aproximadamente, los mismos valores ya que esto permite acelerar la convergencia de los algoritmos y mejorar su rendimiento[1].

- *Elección de la medida de proximidad.* Son medidas que cuantifican lo parecidos, o diferentes, que dos vectores de características son. Para asegurar que todas las características seleccionadas contribuyan de la misma manera a la medida de proximidad sin que unas características dominen a otras, es importante realizar una normalización de las entradas en el caso de que éstas presenten valores apreciablemente diferentes, como ya se ha comentado en el anterior punto.

- *Elección del criterio de agrupamiento.* Esto depende del tipo de grupos que se espera que subyazcan en el conjunto de datos. Por ejemplo, si se sabe que los grupos que se van a encontrar seguramente sean

[1] Habitualmente, la normalización hace que la distribución de datos presente un valor medio nulo y una varianza unidad.

alargados, el criterio de agrupamiento a utilizar será diferente de si se espera que los grupos sean compactos. Este criterio de agrupamiento puede consistir en una función de coste u otra, o en la formulación de unas determinadas reglas que permitan establecer la asignación de los patrones a los grupos.

- *Elección del algoritmo de agrupamiento.* Una vez elegida la medida de proximidad y el criterio de agrupamiento, el siguiente paso es la elección del esquema algorítmico que permite desenmarañar la estructura de *clusters* que subyace en el conjunto de datos del que se dispone.

- *Validación de resultados.* Una vez que los resultados del algoritmo de *clustering* se han obtenido, éstos se han de verificar. Dependiendo de la aplicación, esta validación se lleva a cabo de una manera u otra.

- *Interpretación de resultados.* En muchos casos, un experto en el campo en el cual se desarrolla la aplicación debe interpretar los resultados del *clustering* alcanzado para extraer conocimiento acerca del problema. Este conocimiento puede llegar a ser muy importante, ya que es lo que, al final, permite tomar decisiones reales.

Evidentemente, una elección diferente de características, medidas de proximidad, criterios de *clustering* y algoritmos de *clustering* puede llevar a resultados totalmente diferentes. Por ejemplo, consideremos la Figura 2.1. Si nos preguntamos cuántos *clusters* existen en esa agrupación de datos, la respuesta puede que fuera dos o cuatro. ¿Cuál es la elección correcta? No existe una respuesta definitiva, ya que ambos *clusterings* son válidos, el marcado por las líneas más gruesas que da lugar a dos *clusters* y el marcado por las más finas que da lugar a cuatro grupos. La mejor solución sería entregarle el resultado a un experto en el tema para que decidiera qué *clustering* es más idóneo. Por tanto, la respuesta final a estas cuestiones estará influenciada por el conocimiento del experto.

El *clustering* es una técnica ampliamente usada en gran cantidad de aplicaciones. En particular, podemos hablar de tres grandes categorías donde aplicar las técnicas de agrupamiento (Theodoridis y Koutroumbas, 1999):

1. *Extracción de datos informativos.* En muchas ocasiones, el número de datos disponibles (N) es muy grande y, como consecuencia, es necesario llevar a cabo un procesado para disponer de una cantidad de

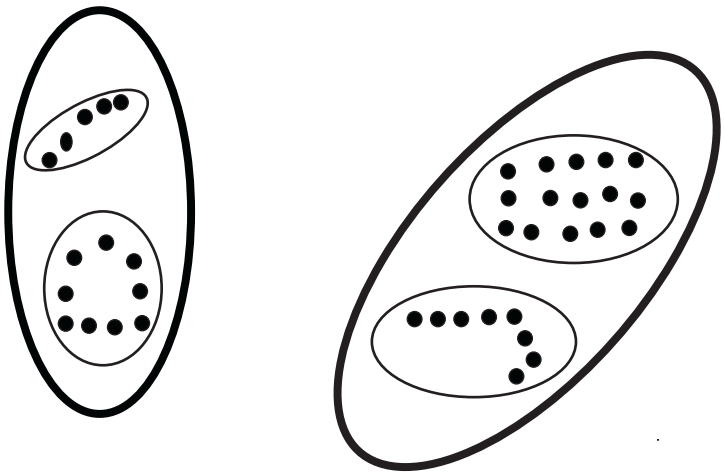


Figura 2.1: Ejemplo donde un *clustering* fino daría lugar a cuatro *clusters* y un ajuste más grueso a dos *clusters*.

datos menor pero, a la vez, suficientemente informativa. Por tanto, podemos utilizar el *clustering* para encontrar un número m de grupos ($m \ll N$), procesando cada grupo de manera individual. Una aplicación directa de esto es la compresión de datos; si se quieren transmitir datos, puede definirse un prototipo representativo de cada *cluster* y en lugar de transmitir el número correspondiente a cada dato, se transmite el correspondiente a cada prototipo, con lo que se tiene una cierta compresión.

2. *Extracción de conocimiento.* Las técnicas de agrupamiento también pueden utilizarse para extraer conocimiento, por ejemplo, en forma de reglas a partir de los datos de los que se dispone. Normalmente, en este tipo de aplicaciones debe realizarse también una fase en la que se valida el conocimiento extraído.

3. *Predicción basada en grupos.* En este caso, se utilizan las técnicas de agrupamiento sobre el conjunto de datos para determinar las características comunes de los patrones que forman cada grupo, y que a su vez los diferencian de los patrones que pertenecen a otros grupos. A continuación, si se dispone de un nuevo patrón desconocido, lo que se hace es determinar a qué grupo pertenecerá con mayor probabilidad, y en función del grupo al que se le asigne, se realizará una determinada acción sobre el patrón. Un ejemplo de este tipo de aplicación

es el desarrollado en esta tesis, donde una vez que se determina que un usuario web pertenece a un cierto grupo, se le recomiendan aquellos servicios a los que usualmente accede la gente de su grupo. Otra posible aplicación puede ser en medicina. Imaginemos que tenemos un determinado conjunto de pacientes que padecen una determinada enfermedad, y se realiza un agrupamiento de los pacientes en función de su reacción ante un determinado fármaco; de esta manera, si aparece un nuevo paciente podemos analizar a qué *cluster* pertenece, y en función de eso, decidir cuál será su medicación (Camps et al., 2002).

2.2. Concepto de agrupamiento o *clustering*

Para poder definir el concepto de agrupamiento o *clustering* es necesario definir en primer lugar qué significa *cluster*. Se han propuesto muchas definiciones para este concepto desde los años 60 (Johnson, 1967). La mayoría de estas definiciones están basadas en términos vagos o de naturaleza algo ambigua, apareciendo palabras como "similar" o "parecido", lo que pone de manifiesto la dificultad de encontrar una buena definición para este término. En (Everitt, 1981) se define *cluster* como *"aquella región continua del espacio que contiene una densidad relativamente alta de puntos, y que se encuentra a su vez separada de otras regiones de alta densidad por regiones cuya densidad de puntos es relativamente baja"*. Aquellos *clusters* que cumplen esta definición son llamados en ocasiones como *clusters naturales* (Theodoridis y Koutroumbas, 1999). Nótese que no es necesario que exista una separación entre los *clusters* sino que basta con que exista una diferencia considerable en la densidad de puntos; de hecho, pueden existir diferentes *clusters* uno a continuación del otro, lo cual puede ser una situación habitual en casos como el que ocupa a la presente tesis donde se trata de realizar una caracterización de usuarios, tal y como ocurre en (Lisboa y Patel, 2004).

Una vez definido el concepto de *cluster* podemos pasar a definir qué es el *clustering*. Sea X un conjunto de datos:

$$X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\} \tag{2.1}$$

Decimos que R es un *m-clustering* de X si realiza una partición de X en m conjuntos (*clusters*), $C_1, C_2, \ldots, C_m$, tales que se cumplen las siguientes tres condiciones:

1. $C_i \neq \emptyset$, $i = 1, \ldots, m$
2. $\bigcup_{i=1}^{m} C_i = X$
3. $C_i \bigcap C_j = \emptyset, i \neq j;\ i, j = 1, \ldots, m$

Además, debe cumplirse evidentemente que todos los patrones que pertenecen a C_i sean parecidos entre ellos y diferentes de los patrones que pertenecen a otros *clusters*. La cuantificación de este parecido depende en muchas ocasiones de la forma de estos *clusters* ya que una medida de similitud que funciona adecuadamente para *clusters* compactos puede que no lo haga para *clusters* alargados, donde determinadas combinaciones de características predominan sobre otras.

Si nos fijamos en la tercera condición que aparece en la definición de *clustering*, cada patrón pertenece únicamente a un grupo. Esto sería lo que llamaríamos un *clustering* "duro". Además de esta posibilidad, existe también lo que se conoce como *clustering* "difuso" (Zadeh, 1965). Un *clustering* difuso de X en m *clusters* viene caracterizado por m funciones $\{u_j\}_{j=1}^{m}$, que pueden presentar un valor entre 0 y 1, y que determinan en qué medida el patrón pertenece al *cluster*. La idea es que si un patrón presenta un valor cercano a la unidad, querrá decir que el *cluster* representa muy bien al patrón, mientras que si el valor es cercano a 0, quiere decir que el patrón no presenta las características del *cluster*. Por tanto, esta aproximación permite no solamente agrupar a los datos en diferentes grupos, sino también decir lo bien que el grupo representa al patrón en cuestión. Además, el *clustering* difuso permite que un determinado patrón pertenezca simultáneamente a más de un *cluster* con diferentes grados de pertenencia. Esto es útil para aquellas situaciones en las que se espera un solapamiento entre grupos o la presencia de patrones que difícilmente son asignables solamente a un determinado grupo. De manera formal, estas funciones podrían ser definidas por una función:

$$u_j :\ X \rightarrow [0, 1],\ j = 1, \ldots, m \tag{2.2}$$

que cumple

$$\sum_{j=1}^{m} u_j(\mathbf{x_i}) = 1,\ i = 1, 2, \ldots, N,\ 0 < \sum_{i=1}^{N} u_j(\mathbf{x_i}) < N,\ j = 1, 2, \ldots, m \tag{2.3}$$

Estas funciones $u_j(x_i)$ que miden en qué medida el patrón i-ésimo pertenece al cluster C_j son conocidas como funciones de pertenencia, y el valor que toman mide el grado en que el patrón se adecúa al grupo. Como puede comprobarse el *clustering* duro es un caso particular del difuso cuando las funciones de pertenencia difusas u_j sólo pueden tomar los valores $\{0, 1\}$ en lugar de cualquier valor entre 0 y 1 (en ocasiones a la función análoga a la función de pertenencia difusa, pero aplicada para el *clustering* duro, y que solamente puede tomar estos dos valores, se le llama *función característica*).

Hemos visto que los algoritmos de *clustering* funcionan encontrando grupos de patrones similares. El espacio en el que se intentan encontrar estos parecidos es lo que se conoce como espacio de representación y está formado por lo que podemos llamar características o variables discriminantes del problema. Estas características pueden tomar valores dentro de un rango continuo de valores o bien pueden tomar un conjunto finito de valores únicamente. En el caso de que tome solamente dos valores decimos que la variable es binaria o dicotómica.

Es muy importante realizar una adecuada codificación de variables, particularmente en el caso de variables que únicamente puedan tomar una serie de valores discretos, usualmente no descritos numéricamente. En este sentido, es importante observar cuál es el rango de valores del resto de variables, ya que suele ser positivo intentar mantener un mismo rango para que el *clustering* no esté sesgado debido a los diferentes valores de las variables.

Además, dependiendo del tipo de variables de las que se disponga, un algoritmo de *clustering* tendrá más posibilidades de funcionar adecuadamente que otro. Un caso extremo es, por ejemplo, el de las redes basadas en la teoría de la resonancia adaptativa (*Adaptive Resonance Theory*, ART) que tienen una versión para el caso de variables binarias (ART1) y otra para el caso de variables con un rango continuo de valores (ART2). Además, también existen versiones difusas para estas redes.

2.3. Medidas de proximidad

2.3.1. Definiciones

Dentro de las medidas de proximidad, podemos hablar fundamentalmente de las medidas de disimilitud y de las de similitud. Una medida de disimilitud (MDI) d en $X \in \Re^l$, siendo l la dimensión del espacio donde se miden las disimilitudes, se define como:

$$d : X \times X \rightarrow \Re \tag{2.4}$$

donde $\Re$ es el conjunto de números reales, cumpliéndose que:

$$\exists\ d_0 \in \Re : -\infty < d_0 \leq d(\mathbf{x}, \mathbf{y}) < +\infty, \forall\ \mathbf{x}, \mathbf{y} \in X \tag{2.5}$$

$$d(\mathbf{x}, \mathbf{x}) = d_0,\ \forall\ \mathbf{x} \in X \tag{2.6}$$

y

$$d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x}),\ \forall\ \mathbf{x}, \mathbf{y} \in X \tag{2.7}$$

Si además se cumple que

$$d(\mathbf{x}, \mathbf{y}) = d_0,\ sii\ \mathbf{x} = \mathbf{y} \tag{2.8}$$

y

$$d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z}),\ \forall\ \mathbf{x}, \mathbf{y}, \mathbf{z} \in X \tag{2.9}$$

se dice que d es una *métrica MDI*. La desigualdad (2.9) se conoce como *desigualdad triangular*. La condición (2.8) indica que el nivel mínimo de disimilitud d_0 entre dos vectores se alcanza cuando éstos son idénticos (en la mayoría de ocasiones, $d_0 = 0$). En muchas ocasiones, nos referimos al nivel de disimilitud como distancia, aunque el término no es estrictamente exacto desde el punto de vista matemático.

Por otro lado, una medida de similitud (MS) s en X se define como

$$s : X \times X \rightarrow \Re \tag{2.10}$$

tal que

$$\exists\ s_0 \in \Re : -\infty < s(\mathbf{x}, \mathbf{y}) \leq s_0 < +\infty, \forall\ \mathbf{x}, \mathbf{y} \in X \tag{2.11}$$

$$s(\mathbf{x}, \mathbf{x}) = s_0, \ \forall \ \mathbf{x} \in X \tag{2.12}$$

y

$$s(\mathbf{x}, \mathbf{y}) = s(\mathbf{y}, \mathbf{x}), \ \forall \ \mathbf{x}, \mathbf{y} \in X \tag{2.13}$$

Si además se cumple que

$$s(\mathbf{x}, \mathbf{y}) = s_0, \ sii \ \mathbf{x} = \mathbf{y} \tag{2.14}$$

y

$$s(\mathbf{x}, \mathbf{y})s(\mathbf{y}, \mathbf{z}) \leq [s(\mathbf{x}, \mathbf{y}) + s(\mathbf{y}, \mathbf{z})]s(\mathbf{x}, \mathbf{z}), \ \forall \ \mathbf{x}, \mathbf{y}, \mathbf{z} \in X \tag{2.15}$$

se dice que s es una *métrica SM*.

Existen muchas medidas de similitud y disimilitud que pueden ser usadas como medidas de proximidad entre dos puntos, entre dos distribuciones de puntos, o bien entre un punto y una distribución. Estas medidas están ampliamente referenciadas en la literatura (Theodoridis y Koutroumbas, 1999), siendo las más usadas la distancia Euclídea, la de de Manhattan, la de Mahalanobis y la de Bhattacharyya como medidas de disimilitud, y el producto escalar, y la medida de Tanimoto como medidas de similitud.

2.3.2. Distribuciones normales y distancias

Las distribuciones normales, también conocidas como Gaussianas, son la función de distribución de probabilidad más frecuentemente encontrada en la literatura. Puede demostrarse que la distribución normal es la que presenta una mayor entropía dados unos ciertos valores de media y varianza. Además, tal y como establece el Teorema del Límite Central, la suma de una cantidad grande de pequeñas perturbaciones aleatorias e independientes (proceso de muestras independientes e idénticamente distribuidas) conduce a una distribución Gaussiana, por lo que este tipo de distribuciones modelan adecuadamente una gran cantidad de casuística (Duda, Hart y Stork, 2000). Estas razones, junto con el hecho de que son fácilmente tratables desde el punto de vista computacional, hicieron que los conjuntos de datos sintéticos generados sigan este tipo de distribución.

Ya se ha comentado que existen muchas posibles medidas de similitud y disimilitud, que servirán para cuantificar la proximidad entre distribuciones

espaciales de puntos. Nos centraremos en aquellas distancias que se usan para la clasificación Bayesiana de distribuciones normales. Las funciones de verosimilitud de ω_i con respecto a $\mathbf{x}$ en un espacio de características l-dimensional siguen una distribución normal multivariante general cuando verifican la expresión[2]:

$$p(\mathbf{x}|\omega_i) = \frac{1}{(2\pi)^{1/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i)\right), \quad i = 1, \ldots, M \tag{2.16}$$

donde $\mu_i = E[\mathbf{x}]$ es el valor medio o prototipo de la clase ω_i y Σ_i es la matriz de covarianza, que es una matriz definida positiva, que tiene dimensiones $l \times l$ y que está definida como:

$$\Sigma_i = E\left[(\mathbf{x}-\mu_i)(\mathbf{x}-\mu_i)^T\right] \tag{2.17}$$

donde el superíndice T indica trasposición.

En el caso de tener clases equiprobables y con la misma matriz de covarianza, puede diseñarse un clasificador Bayesiano basado en mínima distancia. En particular, podemos considerar dos casos básicamente, que la matriz de covarianza sea diagonal o que no lo sea. En el caso de que la matriz de covarianza sea diagonal, es decir que pueda expresarse de la forma $\Sigma = \sigma^2 I$, el clasificador de máxima verosimilitud viene dado por el mínimo de la distancia Euclídea:

$$d_\epsilon = \|\mathbf{x} - \mu_i\| \tag{2.18}$$

Por tanto, los vectores se asignan a las clases correspondientes de acuerdo con la distancia Euclídea que separa a los vectores de los prototipos de las respectivas clases. Por tanto, las curvas que presentan el mismo valor de distancia $d_\epsilon = c$ respecto a los prototipos de cada clase, son circunferencias de radio c (hiperesferas en el caso general).

Para el caso de una matriz de covarianza no diagonal, el clasificador de máxima verosimilitud se corresponde con la minimización de la distancia de Mahalanobis:

$$d_m = \left((\mathbf{x}-\mu_i)^T \Sigma^{-1}(\mathbf{x}-\mu_i)\right)^{1/2} \tag{2.19}$$

En este caso, las curvas equidistantes correspondientes a una distancia $d_m = c$ son elipses (hiperelipses en el caso general). La matriz de covarianza es

[2]Nótese que las funciones de verosimilitud se corresponden con las funciones de probabilidad de $\mathbf{x}$ a la clase ω_i.

simétrica, y puede diagonalizarse:

$$\Sigma = \Phi \Lambda \Phi^T \tag{2.20}$$

donde $\Phi^T = \Phi^{-1}$ y Λ es la matriz diagonal cuyos elementos son los autovalores de Σ. Las columnas de la matriz Φ son los autovectores de Σ:

$$\Phi = [\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_l}] \tag{2.21}$$

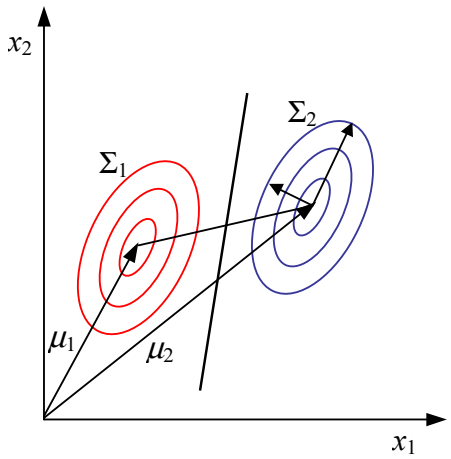


Figura 2.2: Curvas con igual distancia de Mahalanobis respecto a los prototipos de cada clase. La línea representa la máxima separación entre las clases, es decir, aquellos puntos cuya distancia de Mahalanobis a ambas distribuciones es máxima e idéntica

Combinando las ecuaciones (2.19) y ((2.20)) se tiene que:

$$(\mathbf{x} - \mu_i)^T \, \Phi \Lambda^{-1} \Phi^T \, (\mathbf{x} - \mu_i) = c^2 \tag{2.22}$$

Sea $\mathbf{x}' = \Phi^T \mathbf{x}$. Las coordenadas de $\mathbf{x}'$ son iguales a $\mathbf{v}_k^T \mathbf{x}$, $k = 1, 2, \ldots, l$, o lo que es lo mismo, a las proyecciones de $\mathbf{x}$ en los autovectores. Es decir, que son las coordenadas de $\mathbf{x}$ con respecto a las nuevas coordenadas del sistema cuyos ejes vienen determinados por $\mathbf{v}_k$, $k = 1, 2, \ldots, l$. Por tanto, la ecuación (2.22) puede reescribirse como:

$$\frac{(x'_1 - \mu'_{i1})^2}{\lambda_1} + \ldots + \frac{(x'_l - \mu'_{il})^2}{\lambda_l} = c^2 \tag{2.23}$$

que se corresponde con la ecuación de un hiperelipsoide en el nuevo sistema de coordenadas, y donde μ'_{ik} representa la k-ésima componente del prototipo

de la clase i-ésima, expresado en el nuevo sistema de coordenadas. La Figura 2.2 muestra el caso $l = 2$. El centro de masas de la elipse se encuentra en μ_i, encontrándose los ejes principales alineados con los correspondientes autovectores. De esta manera, todos los puntos que tienen la misma distancia de Mahalanobis respecto al prototipo de cada clase se encuentran situados sobre una misma elipse.

2.4. Clasificación de algoritmos de agrupamiento

Un algoritmo de agrupamiento intenta identificar las características específicas de los *clusters* subyacentes en el conjunto de datos. Dentro de los algoritmos de agrupamiento, podemos distinguir una serie de categorías principales (Theodoridis y Koutroumbas, 1999):

- *Algoritmos secuenciales.* Este tipo de algoritmos producen un único *clustering*. Se trata de métodos bastante rápidos y directos. Generalmente, los vectores de características son presentados una o varias veces (no más de cinco o seis) a los algoritmos, dependiendo el resultado final del orden en el cual se pasan estos vectores de características. Este tipo de técnicas suele producir *clusters* compactos e hiperesféricos o hiperelipsoidales, dependiendo de la distancia utilizada. Este tipo de algoritmos son muy sencillos pero presentan un rendimiento bastante pobre a no ser que el problema que se esté abordando sea muy sencillo.

- *Algoritmos jerárquicos.* Dentro de los algoritmos jerárquicos distinguimos entre los algoritmos acumulativos y los divisivos.

    - *Algoritmos acumulativos.* Estos algoritmos producen una secuencia de *clusterings* con un número decreciente de *clusters* en cada paso. El *clustering* producido en cada iteración es el resultado de la unión de dos de los *clusters* que existían en el paso anterior en un solo grupo. Aunque existen muchas variantes para los algoritmos acumulativos, las más utilizadas son las de *enlace simple* y *enlace completo*. Además, los algoritmos acumulativos pueden a su vez dividirse en algoritmos basados en teoría de las matrices y en algoritmos basados en la teoría de grafos. Este tipo de algoritmos es indicado para encontrar tanto *clusters* compactos (en el

caso del algoritmo de enlace completo) como *clusters* alargados (en el caso del algoritmo de enlace sencillo).

- *Algoritmos divisivos.* Estos algoritmos funcionan justamente al contrario que los acumulativos, es decir, producen una serie de *clusterings* con un número creciente de *clusters* a cada paso. El *clustering* producido en cada paso es el resultado de la separación de uno de los *clusters* presente en el paso anterior en dos *clusters* diferentes.

- *Algoritmos basados en la optimización de una función de coste.* En esta categoría se sitúan aquellos algoritmos cuyo *clustering* es evaluado en base a una función de coste J. Generalmente, el número de *clusters* m suele permanecer fijo. Estos algoritmos suelen utilizar conceptos de cálculo diferencial, produciendo sucesivos *clusterings* que intentan optimizar el valor de J, terminando el proceso cuando se encuentra un mínimo local para J. Dentro de esta categoría podemos encontrar los siguientes algoritmos:

  - *Algoritmos de* clustering *"duro".* En este tipo de algoritmos, los vectores que se desea agrupar pertenecen exclusivamente a un *cluster* determinado. La asignación de los vectores a los *clusters* individuales se lleva a cabo de acuerdo con el criterio de optimización que se haya adoptado. El algoritmo más famoso dentro de esta categoría es el algoritmo C-medias y la modificación de éste, el algoritmo Isodata, que serán explicados en la Sección 2.6, ya que es uno de los algoritmos aplicados en esta tesis.

  - *Algoritmos de* clustering *"probabilístico".* En este caso cada vector $\mathbf{x}$ es asignado a un cluster C_i siguiendo un esquema de clasificación Bayesiana; es decir, que esta asignación se producirá cuando la probabilidad *a posteriori* $P(C_i|\mathbf{x})$ sea máxima. Como ejemplo usado en esta tesis de este tipo de algoritmos tenemos el algoritmo *Expectation-Maximization* (E-M).

  - *Algoritmos de* clustering *"difuso".* En este caso, cada vector pertenece a un *cluster* en un determinado grado, pudiendo pertenecer a más de un grupo con diferentes grados de pertenencia. Como ejemplo de este tipo de algoritmos destacamos el algoritmo de las C-medias difuso.

- *Algoritmos de detección de fronteras.* En lugar de determinar los *clusters* utilizando los propios vectores de características, lo que se hace en este caso es ajustar las fronteras que delimitan los *clusters*. A pesar de que estos algoritmos también implican la optimización de una función de coste, su filosofía es algo diferente ya que las aproximaciones anteriores se basan en optimizar la localización de los prototipos de cada *cluster*, mientras que, en este caso, se trata de optimizar la localización de las fronteras entre *clusters*. No se ha utilizado ningún algoritmo de este tipo en esta tesis debido a que no parece ajustarse a las características del problema que nos planteamos resolver ya que puede esperarse un considerable solapamiento entre los grupos de usuarios, o al menos, la existencia de *clusters* contiguos al describir comportamientos de usuarios reales (Lisboa y Patel, 2004).

- *Otros algoritmos.* En esta categoría, situaremos aquellos algoritmos que no pueden asignarse a ninguna de las categorías anteriores. Podemos hablar de las siguientes categorías:

  - *Algoritmos de clustering branch and bound.* Este tipo de algoritmos ofrecen un *clustering* globalmente óptimo sin considerar todos los posibles *clusterings* para un número fijo de *clusters* m, y para un criterio especifíco. No obstante, la carga computacional de estos algoritmos es excesiva.

  - *Algoritmos genéticos.* Estos algoritmos utilizan un conjunto inicial de posibles *clusterings* y van generando iterativamente nuevos conjuntos, las cuales, en general, contienen *clusterings* cada vez mejores, de acuerdo con algún criterio que se haya especificado previamente.

  - *Algoritmos de búsqueda de valles.* Estos algoritmos tratan los vectores de características como ejemplos de una variable aleatoria multidimensional $\mathbf{x}$. Se basan en la presunción, comúnmente aceptada, de que aquellas regiones de $\mathbf{x}$ donde aparecen muchos vectores, corresponden a regiones con valores altos de la función de densidad de probabilidad (fdp) de $\mathbf{x}$. Por tanto, la estimación de la fdp puede resaltar aquellas regiones en que se forman los *clusters*.

- *Métodos de relajación estocástica.* Estos métodos garantizan, bajo ciertas condiciones, la convergencia desde el punto de vista de probabilidad, a un *clustering* óptimo global, respecto a un criterio previamente especificado, aunque es a expensas de una computación intensiva.
- *Algoritmos de aprendizaje competitivo.* Este tipo de esquemas no utilizan una función de coste sino que producen varios *clusterings* y convergen al más "sensible", entendiendo por más sensible aquel que mejor se ajusta a un determinado criterio de distancia especificado previamente.
- *Algoritmos basados en transformaciones morfológicas.* Estos algoritmos utilizan transformaciones morfológicas para alcanzar una mejor separación de los *clusters*.

Dos de los algoritmos usados en esta tesis, los mapas autorganizativos (*Self-Organizing Maps*, SOM) y las redes basadas en la teoría de la resonancia adaptativa (*Adaptive Resonance Theory*, ART) podrían situarse en esta gran categoría de otros algoritmos. En particular, se trata de sendas redes neuronales, que algunos autores las sitúan dentro de la subdivisión de algoritmos competitivos (Theodoridis y Koutroumbas, 1999), aunque realmente se trata de algoritmos y estructuras con la suficiente entidad como para ser estudiados individualmente, como se hará en posteriores apartados.

2.5. Validación del agrupamiento

Una característica común a la mayoría de algoritmos de *clustering* es que se supone conocido el número de grupos; es decir, que los algoritmos buscan optimizar una determinada estructura de *clusters* para un número determinado de ellos. Dar por conocido el número de *clusters* puede ser algo razonable en ciertas ocasiones, pero no tiene una clara justificación si se está realizando un *clustering* de un conjunto de datos cuyas propiedades se desconocen. Por tanto, un problema habitual en este tipo de análisis es, simplemente, decidir el número apropriado de grupos.

Cuando se utilizan técnicas de *clustering* basadas en la optimización de una determinada función, un método común para encontrar el número idóneo de

grupos es llevar a cabo el *clustering* aumentando el número de *clusters* (m), observando como la función cambia con dicho número. Por ejemplo, si esta función es la suma de los errores cuadráticos J_e, su valor decrecerá monóticamente con el valor de m. Si el conjunto de datos está formado por n muestras, y éstos pueden agruparse correctamente en $\hat{m}$ *clusters* compactos y bien separados, lo que se esperaría sería que J_e disminuyera rápidamente hasta llegar al valor $m = \hat{m}$, decreciendo después mucho más lentamente hasta llegar a $m = n$, por lo que un sencillo análisis gráfico de la dependencia entre J_e y m podría ser suficiente para determinar el número óptimo de grupos. Métodos similares pueden llevarse a cabo para los algoritmos jerárquicos, pudiendo ser la evolución del *clustering* suficientemente representativa como para determinar el número óptimo de grupos, ya que suele aceptarse que cuando la unión (o división) de dos *clusters* da lugar a una situación muy diferente de la que se tenía en el paso anterior, esto es indicativo de la presencia de un agrupamiento natural, de un *clustering* correcto (Duda et al., 2000).

Existen algunos métodos más formales basados en la medida de la bondad del ajuste realizado. En esta tesis se ha realizado un triple test para decidir el número correcto de *clusters*:

1. *Análisis de la bondad del clustering.* La metodología que se propone en esta tesis decide qué algoritmo de *clustering* es el más adecuado en una determinada situación real, basándose en comparar dicha situación con diversos conjuntos de datos artificiales que son conocidos, y con los cuales se han probado diferentes algoritmos de *clustering*, para así saber el algoritmo que resulta más adecuado y el número de grupos que puede ser más probable. Para conocer el algoritmo y el número de grupos más adecuado, se han propuesto medidas para evaluar la bondad del agrupamiento encontrado; en particular, una de estas medidas, que se explicará con detalle en el Capítulo 4, se basa en analizar si el número de *clusters* encontrado es correcto o no. Por tanto, la fase de evaluación del *clustering* con conjuntos de datos artificiales es utilizada como un primer paso para determinar el número de grupos que deben encontrarse.

2. *Estudio de la normalidad de los grupos encontrados.* Bastante relacionado con la evaluación del *clustering*, está otra fase donde se llevan a cabo pruebas de asimetría y curtosis del *clustering* encontrado. De-

bido a que los conjuntos de datos artificiales generados en esta tesis siguen una distribución normal, los valores de asimetría y curtosis deberían ser los correspondientes a una distribución de este tipo. Por tanto, este estudio sirve como una prueba de robustez del *clustering* encontrado, y de manera similar a lo que comentábamos respecto al análisis de la bondad del agrupamiento, cuando aparezca un conjunto de datos reales, puede ser comparado con los conjuntos artificiales, decidiendo como algoritmo y número de grupos más adecuados el que se haya obtenido de estas dos primeras fases.

3. *Índices de Dunn y Davies-Bouldin.* Además de las pruebas relacionadas con el número de *clusters* encontrados y con la robustez del agrupamiento, también se han utilizado dos índices que permiten determinar el número de *clusters* subyacente en una determinada distribución: el índice de Dunn y el índice de Davies-Bouldin. La manera de utilizar estos índices es medir su valor para diferentes situaciones, en las que el número de grupos es diferente, decidiendo finalmente como la situación idónea aquella que lleva a unos valores de los índices más deseables.

   - *Índice de Dunn.* Considerando la función de disimilitud entre dos *clusters* C_i y C_j como

   $$d(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(\mathbf{x}, \mathbf{y}) \tag{2.24}$$

   y definiendo el *diámetro del cluster* C como

   $$diam(C) = \max_{x,y \in C} d(\mathbf{x}, \mathbf{y}) \tag{2.25}$$

   Es decir, el diámetro del cluster C es la distancia entre sus dos vectores más distantes, y por tanto, puede verse como una medida de la dispersión de C. El índice de Dunn para un número m de *clusters* se define como:

   $$D_m = \min_{i=1,\ldots,m} \left\{ \min_{j=1,\ldots,m,\ j \neq i} \left(\frac{d(C_i, C_j)}{\max_{k=1,\ldots,m} diam(C_k)} \right) \right\} \tag{2.26}$$

   Por tanto, si el conjunto de datos X contiene *clusters* compactos y bien separados, el índice de Dunn presentará un valor elevado, ya que la distancia entre los *clusters* será grande y el diámetro de éstos pequeño. Por tanto, si se representa D_m frente a m podemos

elegir como número de *clusters* aquel valor de m que se corresponde con el máximo. Los inconvenientes más importantes del índice de Dunn son la carga computacional y que es bastante sensible a la presencia de ruido, ya que éste tenderá a aumentar el valor del denominador en la expresión (2.26).

- *Índice de Davies-Bouldin.* Sea s_i una medida de dispersión correspondiente al *cluster* C_i y $d(C_i, C_j) \equiv d_{ij}$ una medida de disimilitud entre dos *clusters*[3]. Entonces, puede definirse una medida de similitud R_{ij} entre C_i y C_j, que ha de cumplir las siguientes condiciones:

  a) $R_{ij} \geq 0$

  b) $R_{ij} = R_{ji}$

  c) *Si* $s_i = 0$ *y* $s_j = 0$ *entonces* $R_{ij} = 0$

  d) *Si* $s_j > s_k$ *y* $d_{ij} = d_{ik}$ *entonces* $R_{ij} > R_{ik}$

  e) *Si* $s_j = s_k$ *y* $d_{ij} < d_{ik}$ *entonces* $R_{ij} > R_{ik}$

  Estas condiciones establecen que R_{ij} es simétrica y definida postiva. Si ambos grupos, C_i y C_j colapsan a un único punto, entonces $R_{ij} = 0$. Un *cluster* C_i que está a la misma distancia de dos *clusters* diferentes, C_j y C_k, es más parecido a aquel *cluster* que presenta la mayor dispersión (como indica la cuarta condición). Para el caso de dispersiones iguales y niveles de disimilitud diferentes, el *cluster* C_i es más similar a aquel que se encuentre más cerca (por la quinta condición).

  Una elección sencilla para R_{ij} que cumple todas las condiciones anteriores es la siguiente (Theodoridis y Koutroumbas, 1999):

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{2.27}$$

  siendo d_{ij} simétrica. Además, se define R_i como:

$$R_i = \max_{j=1,\ldots,m, j\neq i} R_{ij}, \; i = 1, \ldots, m \tag{2.28}$$

  Entonces, el índice de Davies-Bouldin se define como:

$$DB_m = \frac{1}{m} \sum_{i=1}^{m} R_i \tag{2.29}$$

[3] Aunque otras elecciones también son posibles, puede tomarse $d_{ij} = \|\mathbf{w}_i - \mathbf{w}_j\|$ y $s_i = \frac{1}{n_i} \sum_{x \in C_i} \|\mathbf{x} - \mathbf{w}_i\|$, siendo $\mathbf{w}_i$ el prototipo del *cluster* C_i y n_i el número de vectores en C_i.

Esto es, DB_m representa la similitud promedio entre cada *cluster* C_i, $i = 1, \ldots, m$ y el grupo más similar a éste. Como lo que se desea es que estos *clusters* presenten el mínimo parecido disponible entre ellos, lo que se buscará será agrupamientos que minimicen DB. Valores pequeños de DB indican la presencia de grupos compactos y bien separados. Por tanto, si se representa gráficamente DB_m frente a m, aquel valor de m que se corresponde con el mínimo de esta función es el que se puede entender como correspondiente a un número correcto de grupos.

2.6. Algoritmo de las C-Medias

El algoritmo de las C-Medias (*C-Means (CM)*) es, sin duda, el algoritmo de *clustering* más conocido y utilizado. Este algoritmo se basa en encontrar la mejor representación posible de los grupos que pueda haber subyacentes en una distribución de datos; esta representación se obtiene a través de unos prototipos que definen al grupo en cuestión, utilizando la distancia Euclídea como medida de disimilitud entre los vectores x_i (que serán usuarios web en nuestro caso) y los prototipos de los grupos (Θ_j). El algoritmo busca la minimización de la siguiente función de coste (Theodoridis y Koutroumbas, 1999):

$$J(\Theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{M} u_{ij} \|x_i - \Theta_j\|^2 \tag{2.30}$$

$$u_{ij} = \left\{ \begin{array}{ll} 1, & d(x_i, \Theta_j) = \text{mín}_{k=1,\ldots,M}\, d(x_i, \Theta_k) \\ 0, & \textit{en otro caso} \end{array} \right\} \quad i = 1, \ldots, N \tag{2.31}$$

donde U es una matriz de dimensión $N \times M$ cuyo elemento (i, j) se corresponde con $u_j(x_i)$, $d(x_i, \Theta_j)$ es la distancia entre el usuario i-ésimo y el prototipo j-ésimo, N el número de patrones (usuarios) y M el número de grupos. Típicamente, Θ_j es el vector correspondiente al valor medio del grupo j-ésimo. Este algoritmo encuentra *clusters* compactos, convergiendo hacia un mínimo local de la función de coste. No obstante, esta convergencia a un mínimo local no está garantizada en el caso de que se utilicen distancias diferentes a la Euclídea.

Una modificación interesante de CM es el algoritmo *Isodata*, que es el acrónimo de la expresión inglesa *Iterative Self-Organizing Data Analysis Technique Algorithm*. Isodata realiza una serie de comprobaciones durante el proceso de agrupamiento para decidir automáticamente el número de grupos (Carman y Merickel, 1990). Básicamente, realiza tres pasos de forma iterativa, primero obtiene los agrupamientos igual que CM, después divide los agrupamientos cuyas muestras sean dispares y, por último, une los agrupamientos que estén muy próximos antes de volver al primer paso. El principal problema de este algoritmo reside en que debe elegirse previamente al valor de los parámetros que se utilizan para realizar las comprobaciones pertinentes (número mínimo de muestras por agrupamiento, número de agrupamientos aproximado, dispersión máxima para dividir un agrupamiento, distancia máxima para unir dos agrupamientos y número máximo de agrupamientos que se puede unir). Por tanto, el precio que se debe pagar por no elegir el número de grupos puede ser excesivamente elevado.

2.7. Algoritmo de las C-Medias Difuso

El algoritmo de las C-Medias Difuso (*Fuzzy c-Means (FCM)*) es la versión difusa del algoritmo CM (Theodoridis y Koutroumbas, 1999). Con este algoritmo, los patrones que desean agruparse no pertenecen exclusivamente a un grupo, sino que pueden presentar un grado (que puede ser diferente) de pertenencia a más de un grupo, y por tanto, un usuario puede pertenecer a dos o más grupos simultáneamente con sus correspondientes valores de pertenencia a ellos. Lo que mide este valor de pertenencia es el grado de similitud, que varía entre 0 y 1, entre el patrón y el grupo; cuanto más cercano a la unidad sea el valor de pertenencia, más parecido será el patrón al grupo, o lo que es lo mismo, el valor de pertenencia será menor cuanto mayor sea la distancia del patrón al grupo. Este procedimiento es particularmente interesante cuando aparece solapamiento entre grupos, ya que es capaz de ofrecer esta información dando el valor de pertenencia con el cual el patrón pertenece a los diferentes grupos.

El valor de pertenencia con el cual el patrón (usuario) x_i pertenece al grupo

representado por el prototipo Θ_j viene dado por (2.32):

$$\mu_{ij} = \frac{(\|x_i - \Theta_j\|_\Sigma)^{(-\frac{2}{m-1})}}{\sum_{j=1}^{M}(\|x_i - \Theta_j\|_\Sigma)^{(-\frac{2}{m-1})}} \tag{2.32}$$

donde m es un parámetro que controla el grado de borrosidad considerado; si $m = 1$, el *clustering* no es difuso, y se tiene el algoritmo CM. Si $m > 1$, el *clustering* sí que es difuso; en particular, un mayor grado de borrosidad se obtiene cuando m aumenta. Por otro lado, Σ hace referencia a la matriz de covarianza, que permite generalizar la norma Euclídea a la de Mahalanobis, permitiendo de este modo *clusterings* hiperelipsoidales, no solamente esféricos, como ocurría en el caso de la norma euclídea:

$$\|x\|_\Sigma = x^T \Sigma^{-1} x \tag{2.33}$$

En este caso, la función de coste pasa a valer:

$$J(\Theta, U) = \sum_{i=1}^{N}\sum_{j=1}^{M} \mu_{ij}^m \|x_i - \Theta_j\|_\Sigma^2 \tag{2.34}$$

2.8. Algoritmos jerárquicos

2.8.1. Introducción

Los algoritmos de *clustering* jerárquico (ACJs) tienen una filosofía diferente a los dos algoritmos anteriormente comentados. Producen una jerarquía de agrupamientos anidados a través de un proceso iterativo. Un agrupamiento R_1 que contiene k grupos, se dice que está anidado en el agrupamiento R_2, que contiene $r(< k)$ grupos, si cada grupo de R_1 es un subconjunto de R_2 y, al menos, un grupo de R_1 es un subconjunto propio de R_2. En este caso, se denota $R_1 \subset R_2$. En particular, estos algoritmos constan de N pasos, tantos como el número de patrones. En cada paso t se obtiene un nuevo agrupamiento que está basado en el agrupamiento que se tenía en el paso anterior $t - 1$.

Existen dos categorías principales dentro de los ACJs: los algoritmos divisivos y los acumulativos. Los algoritmos divisivos parten de un primer agrupamiento R_0 que está formado por un único grupo constituido por todos los patrones del conjunto, estando el último agrupamiento R_{N-1} formado

por tantos grupos N como patrones tenga el conjunto de datos. Por tanto, este tipo de algoritmos funcionan a partir de un agrupamiento amplio, con todos los patrones del conjunto de datos, que va subdividiéndose en conjuntos cada vez más pequeños, en base a criterios de medidas de proximidad. Evidentemente, los primeros agrupamientos del proceso iterativo incluyen a los últimos, cumpliéndose que $R_{N-1} \subset R_{N-2} \ldots \subset R_0$. En la Figura 2.3 se muestra el modo de funcionamiento de este tipo de algoritmos en un caso sencillo.

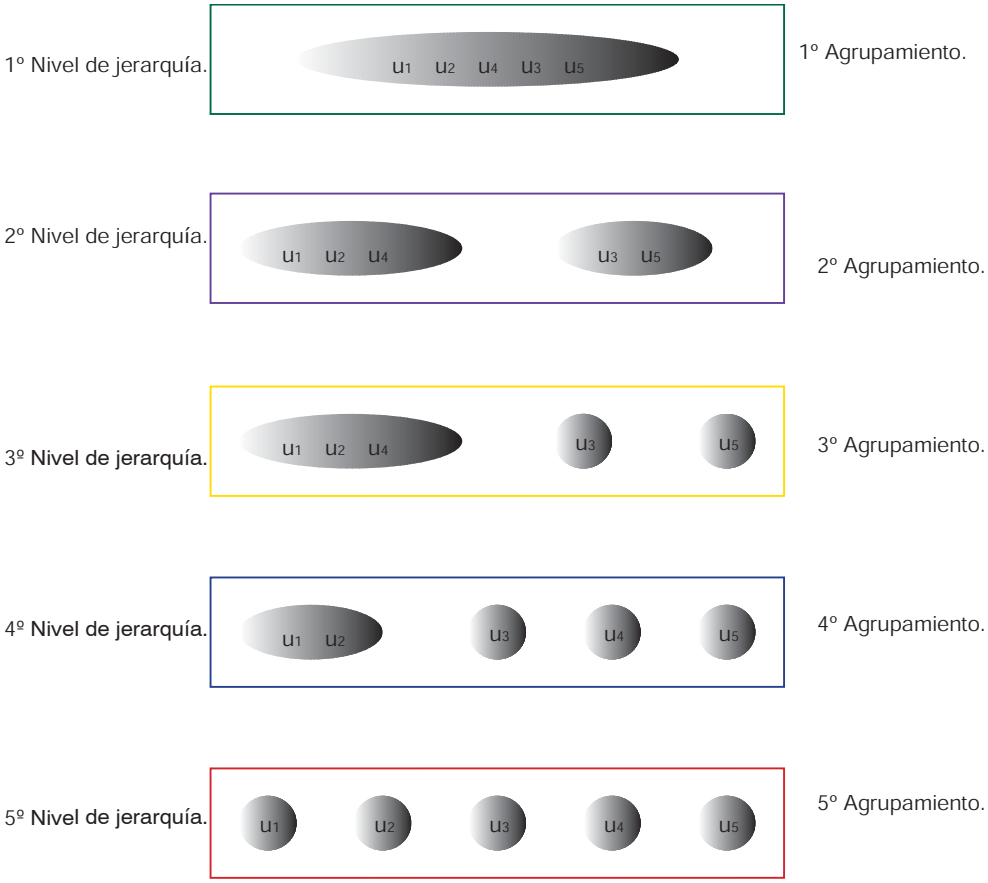


Figura 2.3: Diagrama que muestra el funcionamiento del agrupamiento jerárquico divisivo para un conjunto de datos formado por cinco patrones.

Los algoritmos acumulativos presentan el modo de funcionamiento inverso, es decir, se caracterizan porque el primer agrupamiento R_0 está formado por tantos grupos como patrones N, estando por tanto cada grupo formado por un único patrón. En el primer paso del algoritmo se producirá el agrupamien-

to R_1, que contiene $N-1$ grupos, de forma que se cumple que $R_0 \subset R_1$. A partir de aquí, los siguientes agrupamientos van formándose en un proceso iterativo mediante la unión de patrones en base a ciertas medidas de proximidad, de manera que el último agrupamiento R_{N-1} está formado por un único grupo que comprende todos los patrones del conjunto. Por tanto, se cumple que $R_0 \subset R_1 \ldots \subset R_{N-1}$. En la Figura 2.4 se muestra un ejemplo sencillo del modo de proceder de los algoritmos jerárquicos acumulativos.

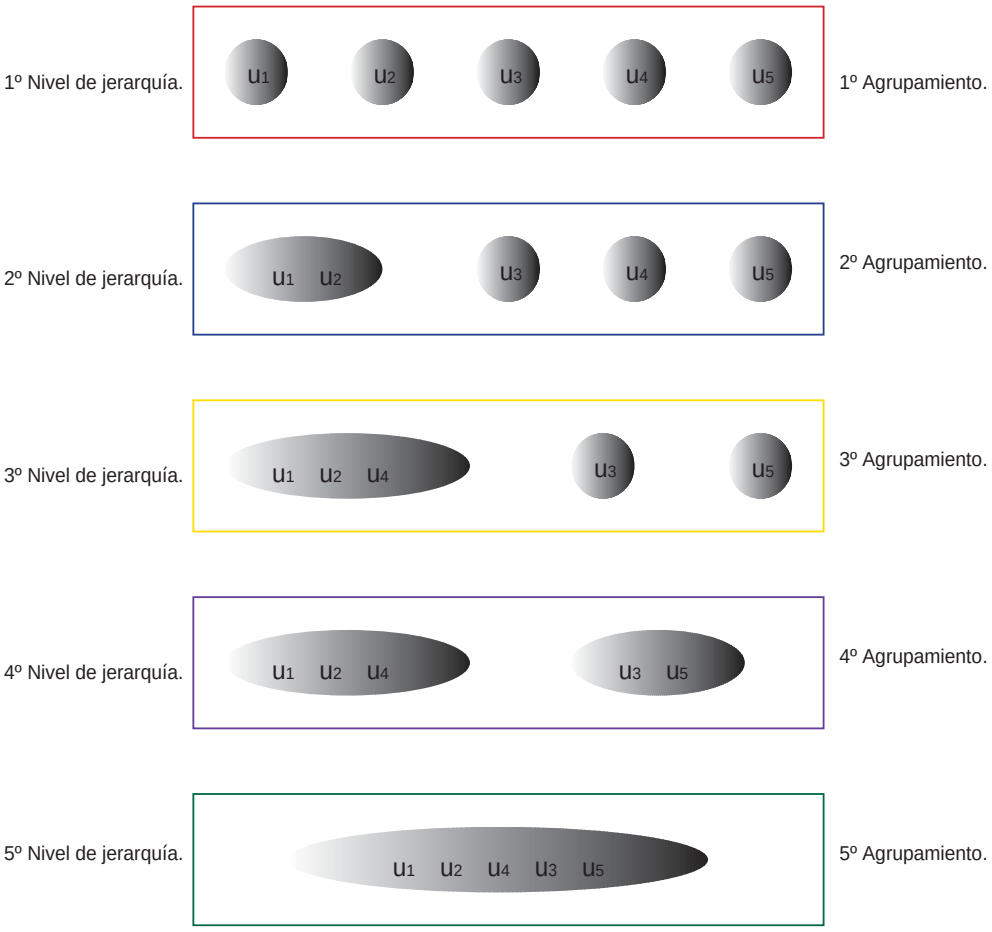


Figura 2.4: Diagrama que muestra el funcionamiento del agrupamiento jerárquico acumulativo para un conjunto de datos formado por cinco patrones.

Nos centraremos en los ACJs acumulativos ya que han sido ampliamente comparados con los divisivos, mostrando en general, un funcionamiento mejor, y sobre todo, una carga computacional mucho menor (Theodoridis y Koutroumbas, 1999).

El funcionamiento de los ACJs acumulativos necesita de la utilización de medidas de proximidad que determinarán el grado de similitud, o disimilitud, entre los elementos del conjunto. Para ello se tiene que definir la función $g(C_i, C_j)$; esta función medirá la proximidad entre los *clusters* C_i y C_j. El agrupamiento inicial vendrá dado por:

$$R_0 \equiv \{C_i = \{\mathbf{x_i}\},\, i = 1, \ldots, N\} \tag{2.35}$$

El proceso iterativo del algoritmo es el siguiente[4]:

- *Paso 1.* Avance de la variable de iteración t, $t = t + 1$ siempre que $t < N$, siendo N el número de patrones.
- *Paso 2.* Cálculo de las medidas de proximidad $g(C_r, C_s)$. Es decir, se calcula la proximidad entre todos los *clusters* posibles C_r, C_s del agrupamiento anterior R_{t-1}, seleccionando para la unión aquella pareja de *clusters* C_i, C_j que verifique:

  $$g(C_i, C_j) : \quad \begin{cases} min\ g_d(C_r, C_s) \\ max\ g_s(C_r, C_s) \end{cases} \tag{2.36}$$

  donde g_d y g_s son medidas de disimilitud y similitud respectivamente.
- *Paso 3.* Se define un nuevo *cluster* C_q, constituido por los *clusters* C_i, C_j seleccionados en la etapa anterior:

  $$C_q = C_i \cup C_j \tag{2.37}$$

  El nuevo agrupamiento R_t está formado por el nuevo *cluster* C_q, y el resto de *clusters* que constituían el agrupamiento R_{t-1}, exceptuando C_i y C_j, y que se denota como R^*_{t-1}. Se tiene entonces

  $$R_t \equiv \left\{R^*_{t-1} \cup C_q\right\} \tag{2.38}$$

- *Paso 4.* Vuelta al paso 1. El proceso termina cuando hay un único *cluster*, llegándose al agrupamiento R_{N-1}.

Existe un tipo de representación gráfica, que se conoce como dendograma y que permite ver cómo se van generando los agrupamientos en los diferentes

[4]En el caso de los algoritmos divisivos, el proceso es el inverso.

niveles de jerarquía. Un ejemplo de dendograma se muestra en la Figura 2.5, donde se selecciona el *clustering* formado por tres grupos. Este tipo de representación da una información más intuitiva del agrupamiento jerárquico, pudiendo ayudar a seleccionar de una manera más adecuada el número de grupos que se desea.

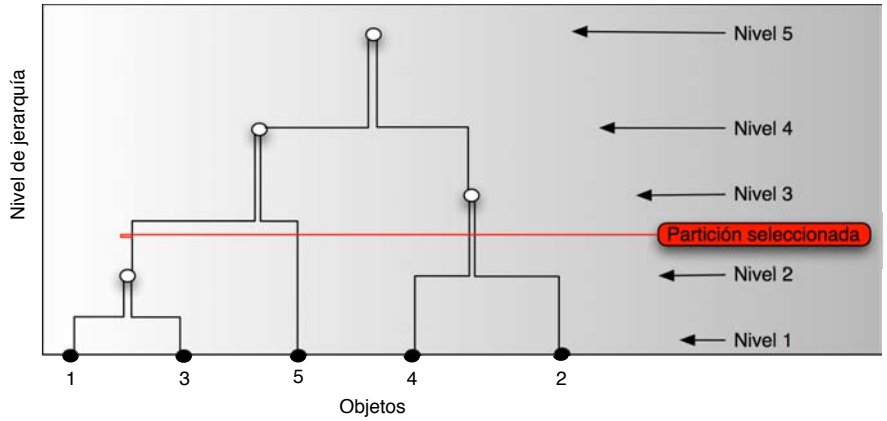


Figura 2.5: Ejemplo de dendograma para un algoritmo de *clustering* jerárquico que agrupa un conjunto de cinco patrones, y que por tanto presenta cinco niveles de jerarquía.

Este tipo de dendogramas, donde solamente se muestra información acerca de cómo van formándose los grupos es llamado *dendograma límite*. En algunas ocasiones, se ofrece además información sobre la distancia entre los grupos, llamándose a éstos *dendogramas de proximidad*. La utilidad de este último tipo de dendogramas es que permite seleccionar el nivel de jerarquía basándose en la distancia que se desea tener entre grupos, lo que en ocasiones puede ser útil.

2.8.2. Posibles implementaciones de algoritmos jerárquicos acumulativos

La mayoría de implementaciones de ACJs acumulativos están basadas en la actualización de una matriz de proximidad. Para ello, si se dispone del conjunto de datos $X = \{\mathbf{x_i},\ i = 1, \ldots, N\}$, se puede definir la *matriz de*

características $D(X)$ como:

$$D(X) = \begin{pmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \mathbf{x_3} \\ \vdots \\ \mathbf{x_N} \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1l} \\ x_{21} & x_{22} & \dots & x_2 \\ x_{31} & x_{32} & \dots & x_{3l} \\ \vdots & \vdots & & \vdots \\ x_{N1} & x_{N2} & \dots & x_{Nl} \end{pmatrix} \tag{2.39}$$

La dimensión de $D(X)$ es $N \times l$, donde N es el número de patrones y l el número de componentes de cada patrón. Una vez definida esta matriz, ya puede definirse la *matriz de proximidad* $P(X)$, que será una matriz de dimensión $N \times N$ y que estará constituida por todas las medidas de proximidad, entre los diferentes vectores fila de la matriz $D(X)$:

$$P(X) = \begin{pmatrix} g(\mathbf{x_1}, \mathbf{x_1}) & g(\mathbf{x_1}, \mathbf{x_2}) & \dots & g(\mathbf{x_1}, \mathbf{x_N}) \\ g(\mathbf{x_2}, \mathbf{x_1}) & g(\mathbf{x_2}, \mathbf{x_2}) & \dots & g(\mathbf{x_2}, \mathbf{x_N}) \\ g(\mathbf{x_3}, \mathbf{x_1}) & g(\mathbf{x_3}, \mathbf{x_2}) & \dots & g(\mathbf{x_3}, \mathbf{x_N}) \\ \vdots & \vdots & & \vdots \\ g(\mathbf{x_N}, \mathbf{x_1}) & g(\mathbf{x_N}, \mathbf{x_2}) & \dots & g(\mathbf{x_N}, \mathbf{x_N}) \end{pmatrix} \tag{2.40}$$

En la práctica se adoptarán medidas de disimilitud, es decir distancias, como medidas de proximidad. Aplicando dos propiedades elementales de las distancias, como que $d(\mathbf{x_i}, \mathbf{x_j}) = d(\mathbf{x_j}, \mathbf{x_i})$ y tomando que $d(\mathbf{x_i}, \mathbf{x_i}) = 0$, la matriz $P(X)$ es simétrica y, además, los elementos de la diagonal son nulos. Los algoritmos basados en actualizar la matriz de proximidad lo que hacen es partir de una matriz de proximidad correspondiente al agrupamiento inicial y, a partir de ella, actualizarla en cada nivel jerárquico mediante el proceso iterativo correspondiente a los ACJs acumulativos, y explicado anteriormente, teniendo en cuenta que la medida de proximidad será, en particular, una medida de disimilitud.

La utilización de la matriz de proximidad permitirá ir formando los sucesivos agrupamientos sin necesidad de recalcular todas las distancias posibles entre *clusters* . De hecho, el único cálculo adicional entre pasos sucesivos del proceso iterativo vendrá dado por las distancias que se tengan que calcular entre el nuevo *cluster* C_q, resultado de la unión de C_i y C_j, y el resto de *clusters*. Estas distancias pueden calcularse en función de las distancias de los *clusters* C_i y C_j a través de la fórmula de Lance y Williams (Theodoridis

y Koutroumbas, 1999):

$$d(C_q, C_s) = a_i \cdot d(C_i, C_s) + a_j \cdot d(C_j, C_s) + \\ + b \cdot d(C_i, C_j) + c \cdot |d(C_i, C_s) - d(C_j, C_s)| \tag{2.41}$$

Dependiendo del valor que tomen los parámetros a_i, a_j, b y c, se tendrán diferentes versiones de algoritmos acumulativos, entre las que destacan las siguientes:

- *Algoritmo de enlace simple.* Presenta una clara tendencia a encontrar grupos alargados, por tanto su uso está recomendado cuando se supone que pueda existir este tipo de grupos. La distancia entre el *cluster* recién formado C_q y el resto de *clusters* que ya existían en la iteración anterior del algoritmo viene dada por:

$$d(C_q, C_s) = \text{mín}\{d(C_i, C_s), d(C_j, C_s)\} \tag{2.42}$$

- *Algoritmo de enlace completo.* En este caso, la tendencia es a producir *clusters* compactos y pequeños. En este caso, la distancia entre el nuevo *cluster* y los anteriores vendrá dada por:

$$d(C_q, C_s) = \text{máx}\{d(C_i, C_s), d(C_j, C_s)\} \tag{2.43}$$

- *Algoritmo de promedio no pesado.* Se realiza un promediado simple entre las distancias de los dos grupos seleccionados al resto de grupos del siguiente agrupamiento:

$$d(C_q, C_s) = \frac{1}{2} \left[d(C_i, C_s) + d(C_j, C_s)\right] \tag{2.44}$$

- *Algoritmo de promedio pesado.* Este algoritmo es similar al anterior, aunque en este caso el promediado está pesado mediante el número de elementos que pertenecen a cada uno de los *clusters* que se unen.

$$d(C_q, C_s) = \frac{1}{n_i + n_j} \left[n_i \cdot d(C_i, C_s) + n_j \cdot d(C_j, C_s)\right] \tag{2.45}$$

  donde n_i y n_j son el número de elementos de C_i y C_j respectivamente. Este algoritmo es especialmente interesante cuando el número de elementos en los *clusters* sea dispar.

- *Algoritmo de centroide no pesado.* La característica principal de este algoritmo es que considera el centro del nuevo *cluster* como su centroide geométrico obteniendose éste sin realizar ningún tipo de pesado en la medida. La distancia entre el nuevo *cluster* y los anteriores depende explícitamente de la distancia entre los *clusters* seleccionados.

$$d(C_q, C_s) = \frac{1}{2}d(C_i, C_s) + \frac{1}{2}d(C_j, C_s) - \frac{1}{4}d(C_i, C_j) \tag{2.46}$$

- *Algoritmo de centroide pesado.* En este caso también se considera el nuevo *cluster* como su centroide geométrico, diferenciándose del anterior en que las distancias están ponderadas por el número de patrones que pertenecen a los dos grupos que se unen.

$$d(C_q, C_s) = \frac{n_i}{n_i + n_j}d(C_i, C_s) + \frac{n_j}{n_i + n_j}d(C_j, C_s) - \frac{n_i n_j}{(n_i + n_j)^2}d(C_i, C_j) \tag{2.47}$$

Además de todos estos algoritmos, basados en la actualización de la matriz de proximidad a través de la fórmula de Lance y Williams, se propone un ACJ acumulativo basado, simplemente, en la actualización de los centroides. Partiendo de un *clustering* inicial R_0, formado por tantos grupos como patrones, este algoritmo calcula todas las distancias entre todos los patrones y une los dos que están más cercanos, representándolos por el centroide correspondiente a la unión de los dos patrones, formándose así el *clustering* R_1 que comprende el nuevo *cluster* y el resto de los anteriores. A partir de este momento, en cada iteración, el algoritmo calculará las distancias entre todos los grupos, uniendo los dos más cercanos, y calculando el centroide correspondiente a la unión de los dos grupos. Este centroide estará pesado por el número de patrones que formen cada uno de los *clusters* que se unen. En este caso, la distancia entre el nuevo *cluster* y los que ya estaban presentes en la anterior iteración se calcula explícitamente y no, como en el caso anterior, utilizando la fórmula de Lance y Williams para expresar esta distancia en función de las distancias que ya estaban calculadas de los *clusters* que se unen para formar el nuevo. Esta aproximación ofrece una visión algo diferente, ya que las distancias entre grupos se calculan para cada iteración; por tanto, es de esperar una mayor carga computacional a expensas de un agrupamiento más directo, ya que en cada iteración se tiene la certeza de que se unen los dos grupos más cercanos. Una ventaja de esta aproximación

es que en principio no existirá un sesgo o tendencia a encontrar grupos con una determinada forma, como sí ocurría al utilizar variantes de la fórmula de Lance y Williams, que tendían a encontrar grupos alargados o compactos, por ejemplo; en este caso, la tendencia a encontrar determinadas estructuras vendrá, en todo caso, dada por la distancia escogida para medir la cercanía entre grupos, por lo que si esta elección es adecuada, el funcionamiento ha de ser correcto. Además, esta aproximación debe dar lugar a dendogramas más sencillos y fáciles de entender.

2.9. Algoritmo *Expectation-Maximization*

El algoritmo *Expectation-Maximization (E-M)* maximiza la esperanza (*expectation*) de la función de verosimilitud de un determinado vector de parámetros Φ sobre un conjunto de muestras, siendo este vector inicialmente desconocido (Theodoridis y Koutroumbas, 1999). Este algoritmo ha sido muy utilizado para la estimación de mezclas de distribuciones Gaussianas, donde los parámetros a estimar son la media μ y la varianza σ^2 (matriz de covarianza Σ). A continuación veremos como se realiza el proceso de estimación de mezcla de distribuciones con este algoritmo, centrándonos después en el caso de distribuciones Gaussianas. Supongamos que tenemos un conjunto de datos (x_k, j_k), $k = 1, 2, \ldots, N$, donde j_k toma valores enteros en el intervalo $[1, M]$, informando de la distribución (mezcla) a partir de la cual se genera x_k, siendo los patrones de entrada l-dimensionales. Suponiendo que las muestras del conjunto de datos son independientes, la función de verosimilutud viene dada por la siguiente expresión:

$$L(\Phi) = \sum_{k=1}^{N} \ln(p(x_k|j_k; \Phi)P_{jk}) \tag{2.48}$$

donde p es la fdp y P_{jk} la probabilidad de que el usuario k-ésimo pertenezca a la mezcla (*cluster*) j-ésima. En esta situación, el vector de parámetros desconocido es $\Psi^T = [\Phi^T, P^T]^T$, con $P = [P_1, P_2, \ldots, P_M]^T$. Considerando que la esperanza depende de las muestras de entrenamiento y de las estimaciones actuales del vector de parámetros $\Psi(t)$, aparecen los dos pasos que le dan nombre al algoritmo:

- *Paso E*: En el instante $(t+1)$ de la iteración, donde se conoce $\Psi(t)$, se calcula el valor esperado de:

$$Q(\Psi;\Psi(t)) = E\left[\sum_{k=1}^{N}\ln\left(p(x_k|j_k;\Phi)P_{jk}\right)\right]$$
$$= \sum_{k=1}^{N}\sum_{j_k=1}^{M} P(j_k|x_k;\Psi(t))\ln(p(x_k|j_k;\Phi)P_{jk}) \tag{2.49}$$

  Centrándonos en el caso de mezcla de Gaussianas, que será lo que resultará más interesante para nuestra aplicación como veremos en posteriores capítulos, se tiene que los parámetros desconocidos son la media y la covarianza, por lo que (2.49) puede expresarse como[5]:

$$Q(\Psi;\Psi(t)) = \sum_{k=1}^{N}\sum_{j=1}^{M} P(j|x_k;\Psi(t))(-\frac{l}{2}\ln\sigma_j^2 - \\ -\frac{1}{2\sigma_j^2}\|x_k-\mu_j\|^2 + \ln P_j) \tag{2.50}$$

  donde μ_j y σ_j representan la media y la desviación estándar del *cluster* j-ésimo, respectivamente.

- *Paso M*: La estimación de Ψ en el instante $(t+1)$ se calcula maximizando $Q(\Psi;\Psi(t))$:

$$\Psi(t+1) : \frac{\partial Q(\Psi;\Psi(t))}{\partial\Psi} = 0 \tag{2.51}$$

  Para el caso de mezcla de Gaussianas, esta maximización puede expresarse como sigue:

$$\mu_j(t+1) = \frac{\sum_{k=1}^{N} P(j|x_k;\Psi(t))x_k}{\sum_{k=1}^{N} P(j|x_k;\Psi(t))} \tag{2.52}$$

$$\sigma_j^2(t+1) = \frac{\sum_{k=1}^{N} P(j|x_k;\Psi(t))\|x_k-\mu_j(t+1)\|^2}{l\sum_{k=1}^{N} P(j|x_k;\Psi(t))} \tag{2.53}$$

$$Pj(t+1) = \frac{1}{N}\sum_{k=1}^{N} P(j|x_k;\Psi(t)) \tag{2.54}$$

[5]La notación se ha simplificado eliminando el índice k de j_k, debido a que para cada k, se suma sobre los M posibles valores de j_k, y éstos son los mismos $\forall$ k.

Para tener la iteración completada solamente se necesita calcular $P(j|x_k;\Psi(t))$, que puede obtenerse de la siguiente manera (Teorema de Bayes):

$$P(j|x_k;\Psi(t)) = \frac{p(x_k|j;\Phi(t))P_j(t)}{p(x_k;\Psi(t))} \tag{2.55}$$

$$p(x_k;\Psi(t)) = \sum_{j=1}^{M} p(x_k|j;\Phi(t))P_j(t) \tag{2.56}$$

Las ecuaciones (2.52)–(2.56) constituyen el algoritmo E-M para la estimación de los parámetros desconocidos de una mezcla de Gaussianas.

Para aplicar el algoritmo E-M, se empieza de una estimación inicial $\Psi(0)$, concluyéndose las iteraciones cuando $\|\Psi(t+1)-\Psi(t)\| \leq \epsilon$ para una elección apropiada tanto del vector de parámetros como del umbral de convergencia ϵ. Puede demostrarse que las estimaciones sucesivas de $\Psi(t)$ nunca hacen decrecer la verosimilitud. De hecho, la función de verosimilitud va aumentando su valor hasta que se alcanza un máximo (local o global) y el algoritmo converge por debajo del umbral señalado ϵ (Theodoridis y Koutroumbas, 1999). Evidentemente, dependiendo del número de mezclas que se haya considerado inicialmente, podrá alcanzarse el umbral de convergencia o no.

2.10. Mapas autoorganizativos

2.10.1. Introducción

Los mapas autoorganizativos (*Self-Organizing Maps, SOMs*) fueron propuestos por Teuvo Kohonen en 1984 (Kohonen, 1984). Se trata de una red neuronal que intenta plasmar la característica del cerebro humano de que las neuronas de una misma zona del cerebro se especializan en una misma tarea o en tareas similares.

El SOM consigue que patrones de entrada similares queden asociados a una misma neurona o a neuronas vecinas. Aunque no es un algoritmo excesivamente complejo desde el punto de vista matemático, involucra un elevado conjunto de variables que hay que ajustar.

En el SOM original introducido por Kohonen la estructura y el número de neuronas del mapa eran fijos desde el principio. El número de neuronas se toma tan grande como sea posible; contrariamente a otro tipo de modelos

neuronales que tienden al sobreajuste, para el SOM tener más neuronas que patrones de entrada no afecta al resultado final, siempre que se escoja de manera adecuada el radio de vecindad, concepto que se explicará posteriormente. El problema de aumentar el número de neuronas radica en que el coste computacional se incrementa, y puede darse el caso de tener un entrenamiento demasiado costoso, computacionalmente hablando. Las neuronas vienen definidas a través de los pesos; los pesos representan la pertenencia de la neurona a cada una de las componentes del espacio definido por las variables de entrada, y que se conoce como espacio de representación. Aunque el SOM es bastante robusto ante diferentes inicializaciones, si ésta es adecuada puede conseguirse una convergencia más rápida. Básicamente, existen tres tipos posibles de inicialización de los pesos:

- *Inicialización aleatoria.* Los pesos se escogen de manera aleatoria, de manera que cubran todo el espacio de representación.
- *Inicialización por muestreo.* Los pesos iniciales se asignan de manera arbitaria a algunos patrones de entrada.
- *Inicialización monótona.* Los pesos iniciales se asignan de acuerdo con una función monótona creciente, generalmente lineal, que cubra todo el espacio de representación. Como el mapa de características suele ser bidimensional, generalmente se realiza un análisis de componentes principales (*Principal Component Analysis, PCA*) para determinar las dos primeras direcciones principales del conjunto de entrada. Seguidamente se incrementan los pesos siguiendo las direcciones principales obtenidas con la PCA a lo largo de cada una de las dimensiones del mapa de características.

2.10.2. Arquitectura

El SOM plantea un modelo de dos capas, como se muestra en la Figura 2.6. La primera capa se conoce como capa de entrada o sensorial y contiene tantas neuronas como características forman el patrón. Todo el procesado se realiza en la segunda capa, que es la que forma el mapa de características.

El resultado que proporciona el algoritmo es altamente sensible a la forma en la que se disponen las neuronas en la segunda capa. Generalmente,

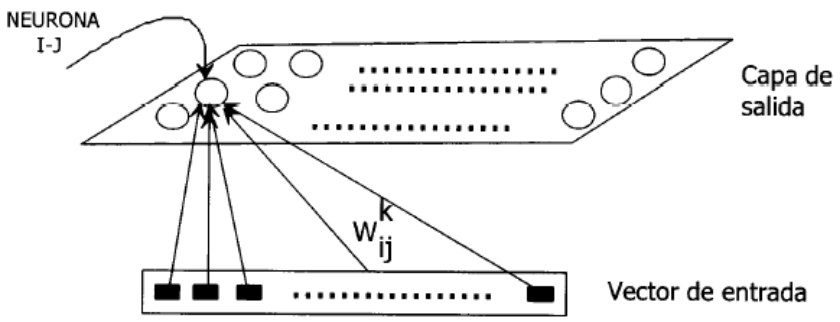


Figura 2.6: Estructura de un mapa de Kohonen con una capa de neuronas de salida bidimensional.

las neuronas se ordenan formando una malla bidimensional, aunque en algunos casos basta con una capa unidimensional. Solamente en determinadas ocasiones se justifica el uso de mapas tridimensionales, ya que se complica bastante la implementación del mapa y se aumenta de manera considerable la carga computacional. Además, la visualización y el análisis de resultados resultan bastante más complicados.

A cada una de las neuronas que forman el mapa de características se le asigna un vector de pesos, un prototipo w_{ij}, cuya dimensión es igual a la del patrón de entrada, y que da información sobre la neurona en el espacio de representación:

$$\mathbf{w_{ij}} = \left(w_{ij}^1, w_{ij}^2, \ldots, w_{ij}^l\right) \tag{2.57}$$

En (2.57), los subíndices i, j hacen referencia a la posición que ocupa la neurona en la red. Además, las neuronas estarán relacionadas con sus vecinas mediante la función de vecindad, como veremos posteriormente. Se dice que las neuronas adyacentes a una determinada neurona forman su *uno-vecindad*, las siguientes forman su *dos-vecindad*, y así sucesivamente. En el caso unidimensional, no queda más remedio que disponer las neuronas formando una retícula lineal. En cambio, en los mapas bidimensionales se pueden disponer las neuronas formando una retícula rectangular o hexagonal, por ejemplo. El tipo de retícula escogida afectará directamente al número de neuronas vecinas.

2.10.3. Aprendizaje

El ritmo del aprendizaje, es decir, la velocidad con la que cambian los pesos, viene determinada por un parámetro $\alpha(t)$, que se conoce como constante de adaptación o de aprendizaje. Dependiendo de la aplicación puede resultar interesante que esta velocidad sea variable y dependiente del número de iteraciones. En particular, lo deseable suele ser que la constante de aprendizaje al final del entrenamiento tenga un valor menor que al principio, de modo que inicialmente el aprendizaje es rápido y, a medida que pasan las iteraciones, y el mapa va aprendiendo la información subyacente en los patrones, el ritmo del aprendizaje va ralentizándose para evitar que la red se haga inestable. Algunas de las expresiones más comunes para determinar este cambio en la constante de aprendizaje son:

- *Exponencial.* La expresión del parámetro que controla la velocidad de aprendizaje viene dada por:

$$\alpha(t) = \alpha_{in}\left(\frac{\alpha_{fin}}{\alpha_{in}}\right)^{\left(\frac{t}{n_{it}}\right)} \tag{2.58}$$

  donde t es la iteración actual y n_{it} el número total de iteraciones, mientras que α_{in} y α_{fin} hacen referencia a los valores inicial y final de la velocidad de aprendizaje, respectivamente.

- *Inversamente proporcional.* En este caso, se tiene la expresión:

$$\alpha(t) = \frac{A}{t+B} \tag{2.59}$$

  siendo A y B constantes que determinan los valores inicial y final del aprendizaje.

En cualquier caso, el ritmo de aprendizaje suele tomar valores comprendidos entre 0 y 1. La actualización de los pesos viene dada por (2.60). Una representación gráfica de este proceso se muestra en la Figura 2.7.

$$\mathbf{w_{ij}}(t+1) = \mathbf{w_{ij}}(t) + \alpha(t)h_t(\mathbf{w_{ij}})\left(\mathbf{x}(t) - \mathbf{w_{ij}}(t)\right) \tag{2.60}$$

donde $\mathbf{x}(t)$ es el patrón de entrada correspondiente.

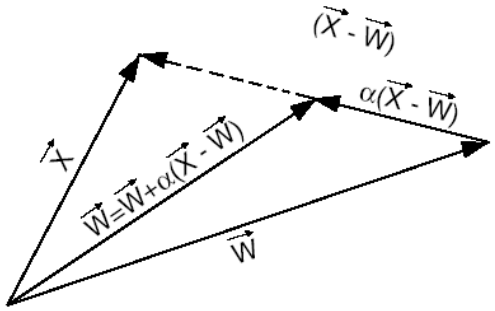


Figura 2.7: Representación gráfica de la actualización de los pesos en SOM.

2.10.4. Función vecindad

La función de vecindad establece qué neuronas se van a actualizar. Generalmente, esta función decrece con la distancia a la neurona vencedora, siendo los dos tipos de función vecindad más utilizados los que se muestran en la Figura 2.8.

- *Función rectangular.* Esta función establece un umbral para la distancia, que llamamos radio de vecindad, de manera que si la distancia de una neurona a la ganadora[6] es menor que el radio de vecindad la neurona se actualiza, y en caso contrario no se actualiza:

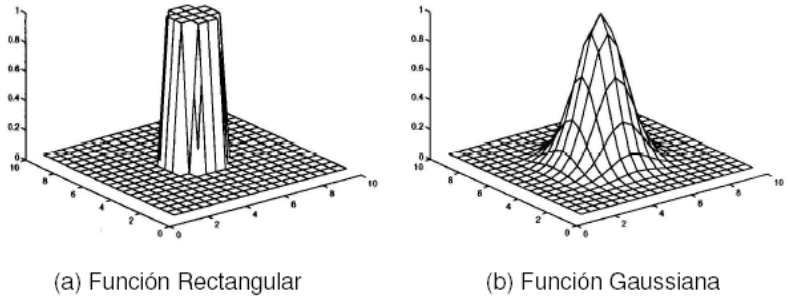
$$h_t(\mathbf{w_j}) = \left\{ \begin{array}{ll} 1, & \|\mathbf{w_{ij}} - \mathbf{w_{gan}}\| \leq r(t) \\ 0, & \|\mathbf{w_{ij}} - \mathbf{w_{gan}}\| > r(t) \end{array} \right\} \tag{2.61}$$

  donde $\mathbf{w_{gan}}$ es el peso de la neurona ganadora y $r(t)$ es el radio de vecindad.

- *Función Gaussiana.* En este caso, la función vecindad tiene la forma de una Gaussiana centrada en la neurona ganadora. El radio de vecindad controla la anchura de la Gaussiana, es decir, determina su desviación estándar:

$$h_t(\mathbf{w_j}) = e^{-\left(\frac{\left\|\mathbf{w_{ij}}-\mathbf{w_{gan}}\right\|^2}{r(t)^2}\right)} \tag{2.62}$$

[6]Para un determinado patrón de entrada, la neurona ganadora es aquella que mejor representa al patrón, o lo que es lo mismo, la que se encuentra a la mínima distancia de éste.



(a) Función Rectangular (b) Función Gaussiana

Figura 2.8: Representaciones de una función de vecindad (a) rectangular y (b) Gaussiana.

Por tanto, el radio de vecindad es el parámetro que controla qué vecindad de la neurona se actualiza. Cuanto mayor sea el radio de vecindad, más neuronas se actualizarán junto con la ganadora. Como sucedía cuando se analizó el ritmo de aprendizaje, suele interesar que el radio de vecindad decrezca a medida que avance el entrenamiento. Al principio del entrenamiento interesan radios de vecindad relativamente grandes para hacer un mapeado general de todo el espacio de representación. En cambio, al final del entrenamiento interesan radios de vecindad menores para mapear los detalles del conjunto de datos.

2.10.5. Algoritmo de aprendizaje

El objetivo del entrenamiento de un SOM es mapear el espacio de representación en el mapa de características, definido en la capa de salida del SOM. Para ello se tiene un proceso iterativo, en el que se escoge aleatoriamente un patrón del conjunto de datos cada iteración. En función del patrón de entrada, se van actualizando los pesos asociados a cada una de las neuronas del mapa de características. El algoritmo básico puede describirse en los siguientes pasos:

- *Paso 1.* Inicialización de los pesos.
- *Paso 2.* Se introduce un patrón del espacio de representación.

- *Paso 3.* Se calcula la similitud entre el patrón y cada uno de los diferentes pesos correspondientes a las neuronas de la capa de salida.
- *Paso 4.* Se determina la neurona ganadora, que es la más cercana al patrón.
- *Paso 5.* Se actualizan los pesos de las neuronas vecinas de acuerdo con la ecuación (2.60).
- *Paso 6.* Se comprueba si se ha alcanzado la condición de parada (típicamente esta condición viene dada por un cambio pequeño en los pesos, por debajo de un umbral determinado). Si se ha llegado a la condición de parada, entonces acaba el algoritmo; si no se cumple la condición, entonces se vuelve al paso 2.

Una cuestión importante tiene que ver con el hecho de que la vecindad de las neuronas de los bordes de la red es menor que la de las neuronas que están en el centro del mapa. Por tanto, la probabilidad de actualización de estas neuronas es estadísticamente menor. Para solventar este problema se proponen estructuras toroidales en el caso bidimensional, de manera que la última neurona de una fila está junto a la primera de la misma fila, y lo mismo ocurre con las columnas. En el caso de un mapa unidimensional, esto conduce a un mapa con forma de anillo.

Respecto al algoritmo clásico del SOM, se han planteado algunas variantes. Una de las más utilizadas es la variante de *Neural Gas*. Como se ha visto, en el SOM se actualiza la neurona vencedora y su vecindad, con lo que puede ocurrir que el peso asociado a una neurona alejada de la vencedora sea muy similar al patrón de entrada, pero menos similar que la neurona vencedora, pero no se actualiza por el hecho de estar alejada de la vencedora en el mapa de características. Una posible solución a esta cuestión la aporta el algoritmo *Neural Gas* (Martinetz y Schulten, 1991). En él se plantea ordenar las neuronas en función de la similitud con el patrón de entrada, siendo actualizadas en función de la ordenación establecida en cada iteración. Por tanto, el algoritmo de actualización de los pesos es parecido al del SOM pero en este caso la función de vecindad no afecta a las neuronas vecinas en la capa de salida sino a aquellas que están próximas unas a otras en el orden de parecido respecto al patrón. Una ventaja adicional de esta variante es que permite acelerar la velocidad de convergencia del mapa. Con esta variante

se consigue un mejor mapeado del conjunto de entrenamiento porque no se penaliza tanto aquellas neuronas que están alejadas de la vencedora. Evidentemente, el problema es que se pierde la relación topológica del SOM con el conjunto de datos, que es una de las principales ventajas del SOM.

Otra importante variante del SOM es la que se conoce como *algoritmo de consciencia*. La motivación de esta variante viene del hecho de que, en ocasiones, el SOM tiende a actualizar de manera repetida las mismas neuronas, no actualizando las otras. Para evitar este problema, este algoritmo cuenta las veces que se ha actualizado cada neurona, penalizando aquellas neuronas que más veces se han actualizado, y por tanto, favoreciendo las que menos veces lo han hecho. Este algoritmo permite eliminar aquellas neuronas que no se han actualizado durante un número considerable de iteraciones.

2.10.6. Extracción de grupos

El mapa de Kohonen permite realizar una proyección bidimensional de los datos en el espacio de características, y de hecho ha sido ampliamente utilizado con esa finalidad. Esta proyección permite ver las relaciones existentes entre los datos, ya que además se guarda la relación topológica entre los patrones de entrada.

No obstante, si se desea utilizar el mapa de Kohonen como herramienta de *clustering* típica, es decir, aquella que permite agrupar los patrones de entrada en un determinado conjunto, se ha de realizar un cierto procesado. El objetivo es al final disponer de una serie de *clusters* que agrupan a patrones en función de su similitud y que están descritos por un cierto prototipo que representa al grupo. En esta tesis se proponen dos métodos para llevar a cabo la extracción de grupos en un SOM.

Método basado en tratamiento digital de imágenes

El primer paso a desarrollar en este método es, una vez obtenido el mapa de Kohonen, generar una matriz de proximidad, similar a la que se vio en el apartado dedicado a los ACJs. Esta matriz estará formada por medidas de disimilitud, siendo el elemento (i, j) de esa matriz, la distancia entre las neuronas i y j. Una vez que se tiene esta matriz de disimilitud se realiza una binarización. Ésta consiste en asignar un valor unidad a todos aquellos

elementos de la matriz que presentan un valor inferior al umbral especificado para la binarización, y en asignar un valor nulo a aquellos elementos cuyo valor se sitúa por encima del umbral. Tras la binarización se realiza un procesado digital de imágenes bastante sencillo que consiste en una erosión seguida de una dilatación (González y Woods, 2002). La intención de este procesado es limpiar la imagen de puntos aislados que incrementan de forma sustancial el coste computacional y que no aportan información. En la Figura 2.9 se muestra un ejemplo de los tres pasos del procesado de imágenes que se realiza.

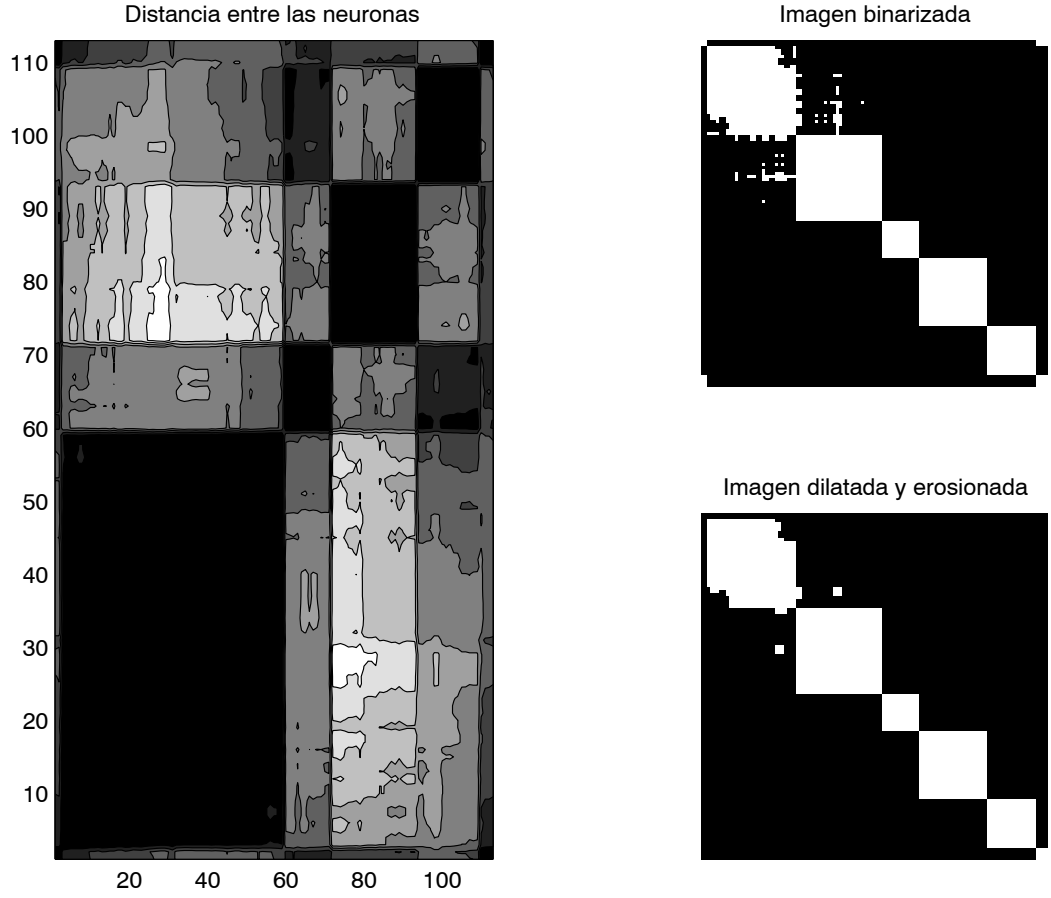


Figura 2.9: Ejemplo de los diferentes pasos del procesado de imágenes que se realiza para la extracción de grupos con el SOM. En la imagen de la izquierda se representa la distancia entre las neuronas en escala de grises, mientras que en la imagen de la derecha superior se muestra la imagen binarizada y en la de la derecha inferior la imagen tras la dilatación y la erosión.

Como la matriz de disimilaridad es simétrica, solamente será necesario tener en cuenta la diagonal principal y la parte, o bien superior, o bien inferior a la diagonal. Los conjuntos conexos que se encuentran en la diagonal principal se corresponden con los *clusters*, teniendo en cuenta que si se implementa una topología de anillo o toroidal, el *cluster* del principio y del final de la diagonal son realmente el mismo, es decir, que en el ejemplo de la Figura 2.9 habría cinco grupos. Si aparecen conjuntos conexos fuera de la diagonal principal, lo que esto indica es que dos *clusters* diferentes están solapados, ya que neuronas en principio alejadas en el mapa aparecen como pertenecientes a un mismo grupo. Cuando aparece esta situación se ha de analizar la cantidad de neuronas que forman el solape. Si el número es pequeño, estas neuronas pueden tratarse como neuronas aisladas, mientras que si el número de neuronas solapadas es elevado, se considera que los grupos a los que pertenecen las neuronas están unidos. En el caso de nuestro ejemplo, solamente aparece un grupo de neuronas solapadas, además en pequeña cantidad por lo que no se considerará solape de grupos, y se seguirán teniendo cinco *clusters* para definir el conjunto de datos.

Tras el tratamiento digital de imágenes, puede ocurrir que determinadas neuronas no quedan asociadas a ningún *cluster*, aunque sí que tengan patrones asociados; es lo que llamamos "neuronas aisladas". Generalmente, estas neuronas representan a patrones alejados de cualquiera de los grupos encontrados. Debe decidirse si realmente esas neuronas pertenecen o no a alguno de los grupos encontrados. Para ello se evalúa la distancia de la neurona a los centroides de los grupos más próximos. Si esta distancia es inferior a la distancia máxima de una neurona del grupo al centroide, la neurona aislada queda asignada a ese grupo.

Para el cálculo del centroide de cada grupo, lo que se hace es introducir de nuevo los patrones al mapa ya obtenido, viendo a qué grupo de neuronas se asigna cada patrón de entrada, y realizando posteriormente la media para cada grupo de los patrones que a él han sido asignados.

Método basado en agrupamiento jerárquico

Este método de extracción de grupos consta de dos pasos. El primer paso es, una vez obtenido el mapa de Kohonen, calcular la distancia entre neuronas vecinas con la intención de unir aquellas que están más próximas,

aprovechando el hecho de que el SOM mantiene las relaciones topológicas del conjunto de entrada. Este proceso puede verse como una especie de ACJ de enlace simple ya que primeramente se generan grupos uniendo los patrones cercanos en neuronas, y posteriormente se unen las neuronas más próximas. Este proceso de unión de neuronas vecinas continúa hasta que se llega a un número determinado de grupos (que podríamos llamar "neuronas representativas" y que son los centroides resultantes de la unión de neuronas cercanas) especificado previamente por el usuario. La idea es que este número de grupos sea algo superior al número de grupos que se desee encontrar en el conjunto de datos. Por ejemplo, si se espera encontrar un número de grupos de 10, entonces un número adecuado para esta primera fase de la extracción puede ser 20.

El segundo paso consiste en utilizar un ACJ de enlace completo para pasar de la primera aproximación de *clustering* que contiene un número de grupos por exceso al agrupamiento final. En esta segunda parte se utiliza un algoritmo de enlace completo para intentar compensar el efecto de encontrar grupos alargados que produce el algoritmo de enlace simple utilizado en el primer paso. El número final de grupos se decide utilizando las técnicas citadas en la Sección 2.5.

2.11. Teoría de la resonancia adaptativa

2.11.1. Introducción

La Teoría de la Resonancia Adaptativa (*Adaptive Resonance Theory, ART*) fue originalmente propuesta por Carpenter y Grossberg (Carpenter y Grossberg, 1987) para modelizar el aprendizaje que se realiza en las fases iniciales del procesado visual humano. Al igual que sucede con el SOM, se trata de una RNA.

La primera red de este tipo que se propuso fue la ART1, que realiza *clustering* de patrones binarios. Posteriormente, el modelo se extendió a patrones que podían presentar un rango continuo de valores, en lo que se conoce como red ART2 (Carpenter y Grossberg, 1991). A grandes rasgos, el funcionamiento de esta red se basa en que cada vez que un patrón se presenta a la red, se elige el *cluster* más apropriado para ese patrón, estando cada

grupo representado por un prototipo del mismo. Además, este prototipo se actualiza para permitir que el *cluster* al que representa aprenda este patrón, o lo que es lo mismo, para que la información aportada por este patrón se incluya en el grupo correspondiente. Si no se encuentra ningún *cluster* que represente lo suficientemente bien al patrón, entonces se crea un nuevo grupo que está formado por ese patrón; a partir de ese momento, si un nuevo patrón se presenta a la red, la comparación para determinar el *cluster* que mejor representa al patrón, incluirá también el último grupo formado.

Las redes ART permiten al usuario controlar el grado de similitud que se establece para considerar que los patrones se asignen a un mismo grupo; una vez que esta elección se lleve a cabo, no es necesario elegir el número de *clusters* por adelantado, sino que la red encuentra el número correspondiente al grado de similitud escogido. Esto es una de las ventajas más importantes para aplicar este tipo de redes en la aplicación de encontrar grupos de usuarios web, donde no se conoce, *a priori*, el número correcto de grupos para describir adecuadamente el conjunto de datos, ya que lo que se hace es escoger el grado de similitud deseado entre los usuarios para considerar que pertenecen a un mismo grupo, y a continuación, la red es capaz de encontrar el número de *clusters* "naturales" subyacentes en la distribución de datos. Durante el entrenamiento de la red cada patrón se presenta varias iteraciones. Un determinado patrón puede ser asignado a un determinado prototipo la primera iteración que es presentado y después ser asignado a un prototipo diferente (debido a cambios en los valores del prototipo si éste ha aprendido otros patrones mientras tanto). Cuando ocurra que un patrón esté oscilando entre diferentes prototipos en sucesivas iteraciones, estaremos ante una red inestable.

Algunas RNAs autoorganizadas, como se vio para el SOM, pueden alcanzar la estabilidad a base de reducir gradualmente la velocidad del aprendizaje a medida que los patrones son presentados a la red en sucesivas iteraciones, es decir, dándole cada vez menos peso a los posibles cambios que un patrón puede introducir en la red (Kohonen, 1997). Sin embargo, es evidente que esta metodología no permitirá a la red aprender rápidamente un nuevo patrón que se presenta por primera vez después de que algunas iteraciones ya hayan tenido lugar. La capacidad de una red para responder a un nuevo patrón igual de bien en cualquier fase del aprendizaje se conoce como *plasticidad*. Las redes ART fueron diseñadas para resolver el dilema de la

estabilidad-plasticidad, es decir, que se trata de modelos estables que, no obstante, son lo suficientemente flexibles para aprender rápidamente nuevos patrones en cualquier fase de su aprendizaje.

La arquitectura de estas redes está formada básicamente por tres grupos de neuronas: un primer grupo de neuronas, llamado capa F_1, que define el procesado a la entrada de la red, un segundo grupo de neuronas llamado capa F_2 que define los diferentes *clusters* y un tercer grupo de neuronas que implementa el mecanismo de control de similitud de aquellos patrones que pertenecen a un mismo grupo, utilizando una especie de mecanismo de *reset* (Fausett, 1994).

2.11.2. Red ART2

Como la aplicación práctica que nos planteamos implica el *clustering* de vectores que presentan un rango continuo de valores, ya que como veremos son vectores de probabilidades de acceso a distintos servicios web, nos centramos en la red ART2. Una arquitectura típica para esta red se muestra en la Figura 2.10, donde las tres capas de neuronas anteriormente comentadas pueden observarse. La entrada se denota por $\mathbf{s} = (s_1, \ldots, s_i, \ldots, s_l)$[7]. La capa F_1 consiste de seis tipos de neuronas (W, X, U, V, P y Q). Habrá l neuronas de cada uno de estos tipos (donde l es la dimensión de un patrón de entrada). Además, una neurona suplementaria entre las unidades W y X recibe señales de todas las neuronas W, calcula la norma del vector $\mathbf{w}$, y envía una señal (inhibitoria) a las correspondientes neuronas X, como se muestra en la Figura 2.11. Además, las unidades X reciben también una señal excitatoria de la unidad W correspondiente. Existe una unidad suplementaria similar entre las neuronas P y Q, y otra entre las neuronas U y V. Cada neurona X está conectada con la neurona V correspondiente, y cada unidad Q está también conectada con la neurona V correspondiente.

La señal de entrada va transformándose a medida que pasa por los diferentes tipos de neuronas de la capa F_1. Las conexiones entre las neuronas P_i (de la capa F_1) y las Y_j (de la capa F_2) representan los pesos que multiplican a la señal transmitida. La activación de la neurona ganadora de F_2, ϕ presenta un

[7]Para el caso de la ART se ha cambiado la notación seguida hasta ahora que representaba las entradas por x para evitar posibles confusiones con el grupo de neuronas X que aparecen en esta red.

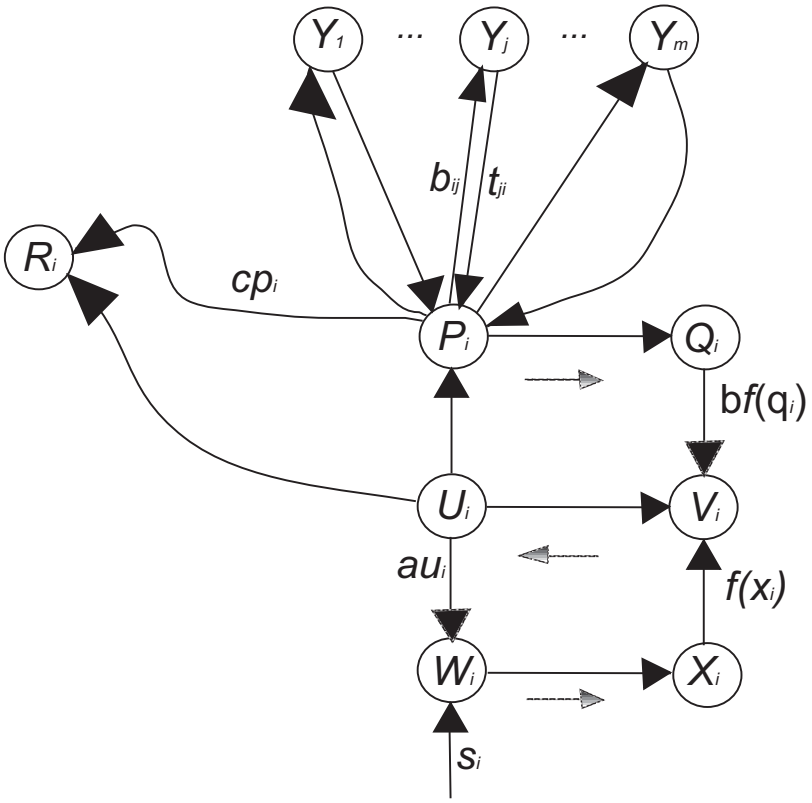


Figura 2.10: Arquitectura típica de una red ART2, tal y como fue propuesta por Carpenter y Grossberg.

valor en el intervalo $]0, 1[$. Esta activación se aplica posteriormente a $\mathbf{x}$ y $\mathbf{q}$, que son los vectores normalizados de las unidades W y P, respectivamente.

Podemos considerar que la capa F_1 está formada por dos partes: la entrada y el interfaz. Las unidades U forman la fase de entrada, incluyendo una combinación de normalización y cancelación de ruido procedente de los datos de entrada. Las unidades P representan la parte de interfaz, combinando señales de la parte de entrada de F_1 y también de la capa F_2 para usarse en la comparación de la similitud entre el patrón de entrada y el prototipo del *cluster* que ha sido seleccionado.

Las unidades de la capa F_2 (Y_j) implementan una competición del tipo *winner-take-all* para decidir cuál es la que aprenderá el patrón de entrada. Una vez que está seleccionada la neurona, la parte del interfaz de F_1 combinará información de la entrada y de las neuronas de F_2. El hecho de que

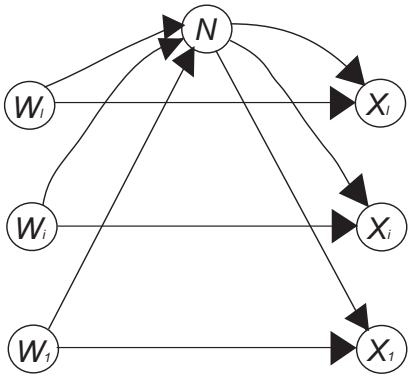


Figura 2.11: Conexiones entra las unidades W y X en una red ART2. N es la unidad suplementaria que se utiliza para normalizar.

a la neurona ganadora se le permita o no aprender el patrón de entrada dependerá de lo similar que su vector de pesos sea al patrón de entrada, es decir de lo adecuadamente que el prototipo del *cluster* represente al patrón de entrada. La decisión es tomada por la unidad de *reset* R, basándose en las señales que recibe de la entrada y de la parte de interfaz de la capa F_1. Si no se le permite al *cluster* que aprenda el patrón, la unidad correspondiente de F_2 recibe una señal inhibitoria y una nueva unidad de *cluster* es creada.

Las unidades X_i y Q_i aplican una función de activación que suprime todas las componentes de vectores cuyas activaciones están por debajo de valor θ seleccionado por el usuario. Las conexiones desde U a W y desde Q hasta V tienen unos valores fijos a and b, respectivamente.

El grado de similitud requerido para que los patrones sean asignados a un mismo *cluster* se controla a través de un parámetro que puede especificar el usuario, y que se conoce como *parámetro de vigilancia* ρ.

2.11.3. Implementación práctica de ART2

La implementación práctica de la red ART2 resulta bastante sencilla e intuitiva, resumiéndose en los siguientes pasos:

- *Paso 1.* Se lleva a cabo una normalización a la unidad de las entradas ($\mathbf{s}$).

- *Paso 2.* Para cada entrada $\mathbf{s_i}$, y dado un prototipo P, el primer test es comprobar si $\mathbf{s_i} \cdot P > \alpha \cdot \sum_i \mathbf{s_i}$, siendo α la constante de aprendizaje (un valor pequeño ralentiza el aprendizaje pero asegurará que los pesos, así como la asignación de los patrones a los *clusters* alcanzará, finalmente, el equilibrio).

- *Paso 3.* Si se pasa el test indicado en el anterior paso, entonces se lleva a cabo el test de vigilancia. $\mathbf{s_i} \cdot P > \rho$; si este test se pasa entonces el prototipo P se actualiza, y si cualquiera de los dos tests no se pasa, entonces se inicializa un nuevo prototipo. El parámetro ρ es el parámetro de vigilancia, cuyo valor determina cuántos *clusters* acabarán formándose; aunque teóricamente se permite que este parámetro tome valores entre 0 y 1, sólo valores entre aproximadamente $0,7$ y 1 son útiles para controlar el número de grupos; de hecho, asignar a este parámetro cualquier valor por debajo de 0,7 tendrá el mismo efecto que hacer que ρ sea igual a cero.

- *Paso 4.* Finalmente, los nuevos prototipos también se normalizan a la unidad.

Los valores de los parámetros α y ρ se normalizan automáticamente a partir del número total de patrones de entrada utilizados para el entrenamiento. De esta manera, se evita que la actualización iterativa del algoritmo lleve a la red a una situación de inestabilidad.

Capítulo 3

Sistemas de recomendaciones

Resumen del capítulo

Una de las partes más importantes de la metodología presentada en esta tesis es la que se refiere al sistema de recomendaciones. Este capítulo comienza realizando una introducción a las técnicas más utilizadas en los sistemas de recomendación: sistemas colaborativos, basados en contenido, técnicas demográficas, sistemas basados en utilidad y basados en conocimiento. A continuación, se realiza una comparación entre todos estos métodos, mostrando sus ventajas e inconvenientes, y ofreciendo ejemplos que permiten vislumbrar el campo de aplicación más adecuado para cada una de estas técnicas. Posteriormente, se describe con más detalle el recomendador colaborativo propuesto, así como un estudio de viabilidad del mismo que permite evaluar por adelantado el rendimiento que éste puede ofrecer en el conjunto real de datos utilizado, y que corresponde al portal web de servicios al ciudadano Infoville XXI. Por último, se propone un recomendador adaptativo que mejora las recomendaciones al ir recibiendo información sobre la aceptación, o no, de las recomendaciones ofrecidas al usuario.

3.1. Introducción

Los recomendadores fueron originalmente definidos como sistemas en los cuales los usuarios explicitaban aquellas recomendaciones que serían útiles para ellos en un formulario, y el sistema almacenaba esta información para

utilizarla en el momento adecuado y para los usuarios adecuados (Resnick y Varian, 1997). No obstante, el término tiene ahora una connotación mucho más amplia, describiendo cualquier sistema que ofrece recomendaciones individualizadas o bien que tiene el efecto de guiar al usuario de manera personalizada, escogiendo para él los servicios más útiles entre una colección que puede llegar a ser bastante grande. Estos sistemas tienen un evidente atractivo en un marco donde el usuario tiene muchas más opciones de las que realmente puede examinar para comprobar si le interesan o no. De hecho, los sistemas de recomendación son parte importante en algunos sitios que proveen servicios de comercio electrónico, como por ejemplo *Amazon.com* (`http://www.amazon.com/`) o 'CDNow' (`http://www.cdnow.com/`), que realmente es la parte destinada a CDs dentro de *Amazon*. En la Figura 3.1 se muestra un ejemplo de recomendación en este portal. El usuario ha solicitado la película "Pulp Fiction", y el sistema le recomienda también "Reservoir Dogs"; este ejemplo puede servir como una pequeña muestra del buen funcionamiento de estos sistemas, ya que en efecto ambas películas son del mismo director, y a pesar de que el argumento no es similar, sí que se pueden considerar como películas de temática parecida.

La diferencia entre un sistema de recomendaciones y un motor de búsqueda o un sistema de recuperación de información estriba en que la información que ofrece un sistema de recomendaciones se intenta que sea "individualizada" y además, "interesante y útil".

Existen múltiples clasificaciones para catalogar los sistemas de recomendaciones (Resnick y Varian, 1997; Schafer, Konstan y Riedl, 1994; Terveen y Hill, 2001). A rasgos muy generales, podemos considerar que un recomendador está formado por las siguientes partes (Burke, 2002):

- *Datos previos.* Aquí se incluye la información de la que dispone el sistema antes de que el proceso de recomendación comience.
- *Datos de entrada.* Ésta es la información que el usuario debe suministrar para que entre en funcionamiento el proceso de recomendación.
- *Algoritmo de recomendación.* El algoritmo de recomendación se encarga de combinar los datos previos y los datos de entrada para suministrar las correspondientes recomendaciones.

Teniendo en cuenta estas tres partes, podemos hablar de cinco diferentes

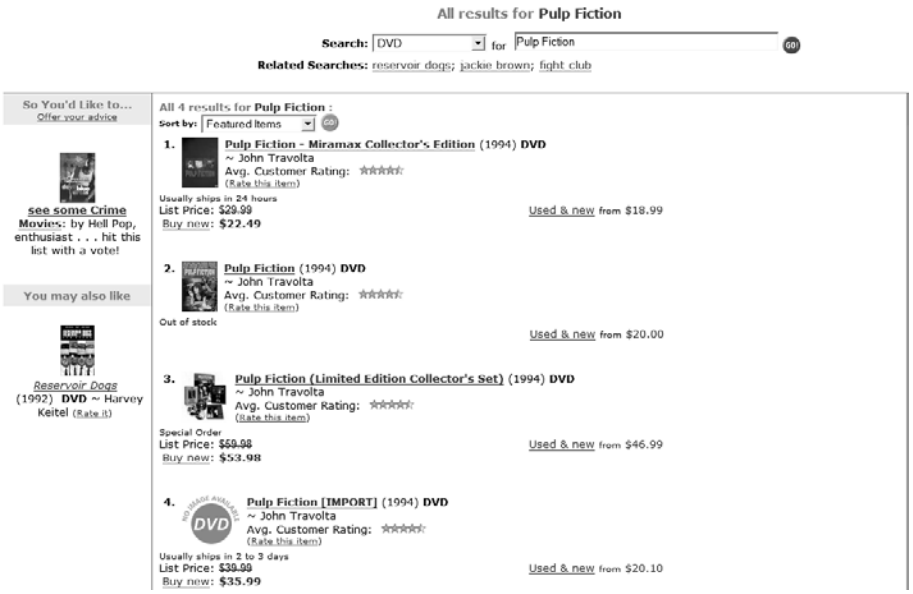


Figura 3.1: Recomendaciones ofrecidas por el portal CDNow, donde se muestran las recomendaciones para un usuario que está interesado en la película "Pulp Fiction". En particular, el recomendador ofrece como posible película en la cual puede estar interesado "Reservoir Dogs".

técnicas de recomendación: recomendación colaborativa, basada en contenido, demográfica, basada en la utilidad y basada en el conocimiento. Para explicar cada una de estas técnicas, vamos a suponer que S es el conjunto de servicios que se pueden recomendar, U el conjunto de usuarios cuyas preferencias son conocidas, u el usuario para el cual tiene que generarse la recomendación, y por último, s es un servicio para el cual se desea saber el grado de preferencia del usuario u. En la Tabla 3.1 se resume el funcionamiento de las cinco técnicas de recomendación consideradas. En particular, en *"background"* se muestra la información almacenada por la técnica en cuestión, y que es lo que posteriormente utilizará para hacer la recomendación a un usuario determinado tras un proceso de extracción de conocimiento, comparación, etc. En "Dato entrada" se muestra el dato del usuario que va a ser recomendado que necesita la técnica para poder ofrecer la recomendación, y en "Procesado" el procesado llevado a cabo por la técnica para finalmente ofrecer la recomendación

La recomendación colaborativa es la más usada y desarrollada en cuanto a su tecnología. Un recomendador de este tipo se basa en almacenar accesos

Tabla 3.1: Comparativa de cinco diferentes técnicas de recomendación en cuanto a la información que almacenan (*background*), la información de entrada que necesitan para ofrecer una recomendación (Dato entrada) y el procesado llevado a cabo por la técnica en cuestión.

Técnica	***Background***	**Dato entrada**	**Procesado**
Colaborativa	Selecciones realizadas por U en S.	Selecciones realizadas por u en S.	Identificar aquellos usuarios en U que son similares a u y extrapolar a partir de sus selecciones en S.
Basada en contenido	Características de los servicios en S.	Selecciones realizadas por u en S.	Generar un clasificador que relaciona el comportamiento de u y lo usa en S.
Demográfica	Inform. demográfica sobre U y sus selecciones en S.	Inform. demográfica de u.	Identificar usuarios demográficamente parecidos a u, y extrapolar a partir de sus selecciones en S.
Basada en utilidad	Características de los servicios en S.	Función de utilidad sobre los servicios en S que describen las preferencias de u.	Aplicar la función a los servicios determinando la significancia de s.
Basada en conocimiento	Características de los servicios en S, y conocimiento de cómo los servicios se corresponden con las necesidades del usuario.	Descripción de las necesidades o intereses de u.	Inferir una correspondencia entre S y las necesidades de u.

de usuarios anteriores y reconocer parecidos entre los usuarios del portal basándose en los servicios que son accedidos por éstos. Estas comparaciones inter-usuario son utilizadas para generar las recomendaciones; la idea es recomendar aquellos servicios que son habitualmente accedidos por aquellos usuarios que son parecidos a u. Un perfil de usuario típico en un sistema colaborativo está formado por un vector cuya longitud es la del número de servicios que pueden ser recomendados, y cuyas componentes toman como valores una evaluación de los accesos que registra el usuario u a cada servicio; es decir, que este vector mide lo que a un usuario u le gusta cada uno de los posibles servicios que pueden recomendarse. Este vector es modificado de manera continua conforme el usuario interactúa con el sistema. Algunos sistemas tienen en cuenta la información temporal para pesar en mayor grado aquellos servicios que han sido accedidos más recientemente (Billsus y Pazzani, 2000; Schwab, Kobsa y Koychev, 2001). En algunas ocasiones, los gustos que definen al usuario u respecto al servicio s pueden ser binarios, es decir, solamente se tiene información de si el servicio es del agrado del usuario o no. En otras ocasiones, el grado de preferencia que el servicio tiene para el usuario se especifica por un número real. Algunos de los sistemas colab-

orativos más importantes son *GroupLens/NetPerceptions* (Resnick, Iacovou, Suchak, Bergstrom y Riedl, 1994), *Ringo/Firefly* (Shardanand y Maes, 1995), y *Recommender* (Hill, Stead, Rosenstein y Furnas, 1995). Estos sistemas pueden estar basados en memoria, que funcionan comparando unos usuarios con otros directamente empleando correlaciones u otro tipo de medidas, o basados en modelos, en los cuales un modelo se obtiene a partir del histórico de datos disponible y se usa para llevar a cabo predicciones (Breese, Keckerman y Kadie, 1998). Los sistemas de recomendaciones basados en modelos utilizan diferentes técnicas para realizar el modelado, desde redes neuronales (Jennings y Higuchi, 1993), o indexado semántico latente (Foltz, 1990), hasta redes Bayesianas (Condliff, Lewis, Madigan y Posse, 1999), por nombrar unas cuantas.

La ventaja más significativa de las técnicas colaborativas es que son completamente independientes de cómo se representen los servicios que van a ser recomendados, funcionando de manera correcta en situaciones complejas como puedan ser la recomendación de música o películas, donde la variación de los gustos de los usuarios es la principal razón para la variación en sus preferencias. Es decir, que realmente las técnicas colaborativas pasan por alto la influencia de los servicios que deben ser recomendados para centrarse en los usuarios objeto de esas recomendaciones; esto es lo que se conoce como "correlación usuario a usuario" (Schafer et al., 1994).

Los sistemas de recomendación demográficos tienen como objetivo clasificar al usuario en función de sus características demográficas, realizando a continuación las recomendaciones basándose en clases demográficas. Un primer ejemplo de este tipo de recomendación lo constituía *Grundy* (Rich, 1979), que era un sistema que recomendaba libros basándose en la información personal que se almacenaba en el sistema a través de un diálogo interactivo. Se buscaba la correspondencia entre las respuestas de los usuarios en este diálogo y una biblioteca de estereotipos de usuario, que había sido compilada de manera manual. Otros sistemas de recomendación más recientes también hacen uso de este tipo de técnicas; por ejemplo, en (Krulwich, 1997), se usan grupos demográficos para llevar a cabo una investigación de marketing que permite sugerir una serie de productos y servicios; la clasificación del usuario en un determinado grupo demográfico se realiza mediante una pequeña encuesta. En otros sistemas, se utilizan métodos de aprendizaje en máquinas para clasificar a los usuarios basándose en sus datos demográfi-

cos (Pazzani, 1999). La representación de la información demográfica en un modelo de usuario puede variar considerablemente; así, *Grundy* (Rich, 1979) usaba características de los usuarios que se anotaban manualmente con unos determinados intervalos de confianza, y ahora sin embargo existen técnicas demográficas que realizan "correlaciones persona a persona", de manera similar a cómo lo hace el filtrado colaborativo pero con distintos datos. El beneficio de la aproximación demográfica radica en que puede no necesitar un histórico de datos de usuario, contrariamente al caso del filtrado colaborativo, y como veremos, a las técnicas basadas en contenido.

En los sistemas de recomendaciones basados en contenido, los objetos de interés se definen en función de las características asociadas a éstos. Por ejemplo, los sistemas de recomendación de texto como el sistema implementado en el grupo de noticias *NewsWeeder* (Lang, 1995), utilizan las palabras de sus textos como características. Un recomendador basado en contenido intenta aprender los perfiles de usuario basándose en las características de los objetos que, previamente, han sido seleccionados por el usuario. Esto es lo que se conoce como "correlación servicio a servicio" (Schafer et al., 1994). El tipo de perfil de usuario obtenido por un recomendador de este tipo depende del método de aprendizaje utilizado. Al igual que ocurre para el caso de las técnicas colaborativas, los perfiles de usuario basados en contenido se pueden considerar como modelos a largo plazo que son actualizados cuando se observan nuevas preferencias en los usuarios que hacen que los modelos anteriores pierdan validez.

Por otro lado, los recomendadores basados en utilidad y en conocimiento no intentan realizar modelos a largo plazo, sino que basan su recomendación en analizar la correspondencia que existe entre las necesidades del usuario y el conjunto de opciones disponibles. Los recomendadores basados en utilidad recomiendan utilizando el cálculo de la utilidad de cada uno de los servicios para el usuario. Evidentemente, el problema clave a resolver aquí es cómo crear una función que defina la utilidad para cada usuario y que después pueda ser empleada de manera adecuada para la recomendación (Burke, 2002). El beneficio de las recomendaciones basadas en utilidad viene del hecho de que puede tener en cuenta para el cálculo de la utilidad algunas características que no están estrictamente relacionadas con los servicios ofrecidos, como por ejemplo, la confianza en el vendedor o la disponibilidad del producto, siendo posible llegar a soluciones de compromiso, por ejemplo entre precio

y plazo de entrega para un usuario que tiene una necesidad inmediata.

Los sistemas de recomendación basados en conocimiento sugieren servicios basándose en inferencias al respecto de las necesidades del usuario y de sus preferencias. En cierta manera, todas las técnicas de recomendación descritas hasta ahora incorporan alguna clase de inferencia. Sin embargo, los sistemas basados en conocimiento se distinguen del resto en que poseen conocimiento funcional, es decir, estos sistemas tienen conocimiento acerca de cómo un servicio en particular se corresponde con las necesidades individuales de un usuario y, por tanto, tiene capacidad de "pensar" sobre la relación entre una necesidad y una posible recomendación. El perfil de usuario en este caso viene dado por cualquier tipo de estructura de conocimiento que pueda ayudar a inferir la mejor recomendación para las necesidades de cada usuario. Un caso muy simple puede ser el del buscador *Google* (`http://www.google.com/`), que devuelve los resultados correspondientes a la búsqueda formulada por el usuario. En casos más complejos, entrarían de una manera más detallada las necesidades del usuario (Towle y Quinn, 2000).

El conocimiento utilizado por un recomendador basado en conocimiento puede tomar múltiples formas. Por ejemplo, *Google* utiliza información de los enlaces entre páginas web para inferir la popularidad de éstas (Brin y Page, 1998). Otro sistema, como *Entree* (Burke, 2002), utiliza conocimientos de gastronomía para inferir similitudes entre restaurantes. Las aproximaciones basadas en utilidad calculan un valor de utilidad para los objetos que son recomendados, y en principio, estos cálculos están basados en el conocimiento almacenado en la función de utilidad (conocimiento funcional). Sin embargo, los sistemas existentes no utilizan tales inferencias, sino que le solicitan a los propios usuarios que faciliten tanto sus necesidades como las características de los productos a través de cuestionarios.

Un ejemplo de recomendador colaborativo es *Moonranker*[1], que se trata de un recomendador gratuito que ofrece sugerencias acerca de bandas de música, películas y libros que pueden ser del agrado del usuario (Zhou, Weston, Gretton, O. y Schölkopf, 2003). El recomendador es del tipo colaborativo, y el funcionamiento es el siguiente: un usuario especifica una

[1] *Moonranker* (`http://www.moonranker.com/`) es el resultado de un proyecto de investigación del Departamento de Inferencia Empírica del Instituto Max Plack para Cibernética Biológica, en Alemania.

pequeña lista de sus favoritos (ya sea en cuanto a grupos de música, libros y películas), y el sistema ofrece automáticamente una lista con aquellos otros grupos, películas o libros que pueden ser del interés del usuario. En la Figura 3.2 se muestra la primera pantalla de resultados para un usuario que ha indicado como grupos de música de su agrado "David Bowie", "The Flaming Lips" y "Pulp". Estos resultados se obtienen de comparar a este usuario con los usuarios previos almacenados en la base de datos, y que mostraban gustos parecidos. Como puede observarse, junto a cada sugerencia de grupo de música, aparece un icono que permite al usuario determinar el grado de aceptación de la sugerencia. Una vez escogidas estas preferencias, el usuario puede refinar la búsqueda, y además la información de éste se almacena en la base de datos, con lo cual la información para llevar a cabo la recomendación colaborativa en un futuro se ve mejorada.

Por tanto, *Moonranker* es un recomendador que incorpora un sistema de realimentación iterativa. Su mayor inconveniente es que necesita de la colaboración de los usuarios, que han de proporcionar información de qué cosas les gustan y disgustan para el buen funcionamiento del sistema.

3.2. Comparación entre técnicas de recomendación

Todas las técnicas de recomendación tienen sus puntos fuertes y débiles. De entre todos los problemas, podemos destacar quizás como el más importante el problema de la falta de datos cuando se empieza a utilizar el sistema, conocido en la literatura como el problema del *ramp-up* (Konstan, Riedl, Borchers y Herlocker, 1998). Este término realmente se refiere a dos problemas diferentes, aunque relacionados:

- *Nuevos usuarios.* Debido a que las recomendaciones son el resultado de la comparación entre el usuario objetivo y otros usuarios, basándonos solamente en anteriores interacciones de los usuarios con el sistema, un usuario del que se disponen pocos datos resulta difícil de clasificar, y por tanto, de recomendar.

- *Nuevos servicios.* Análogamente, un nuevo servicio del que no se disponen todavía suficientes datos de accesos de los usuarios a éste, puede ser complicado de recomendar. Este problema se manifiesta en mayor

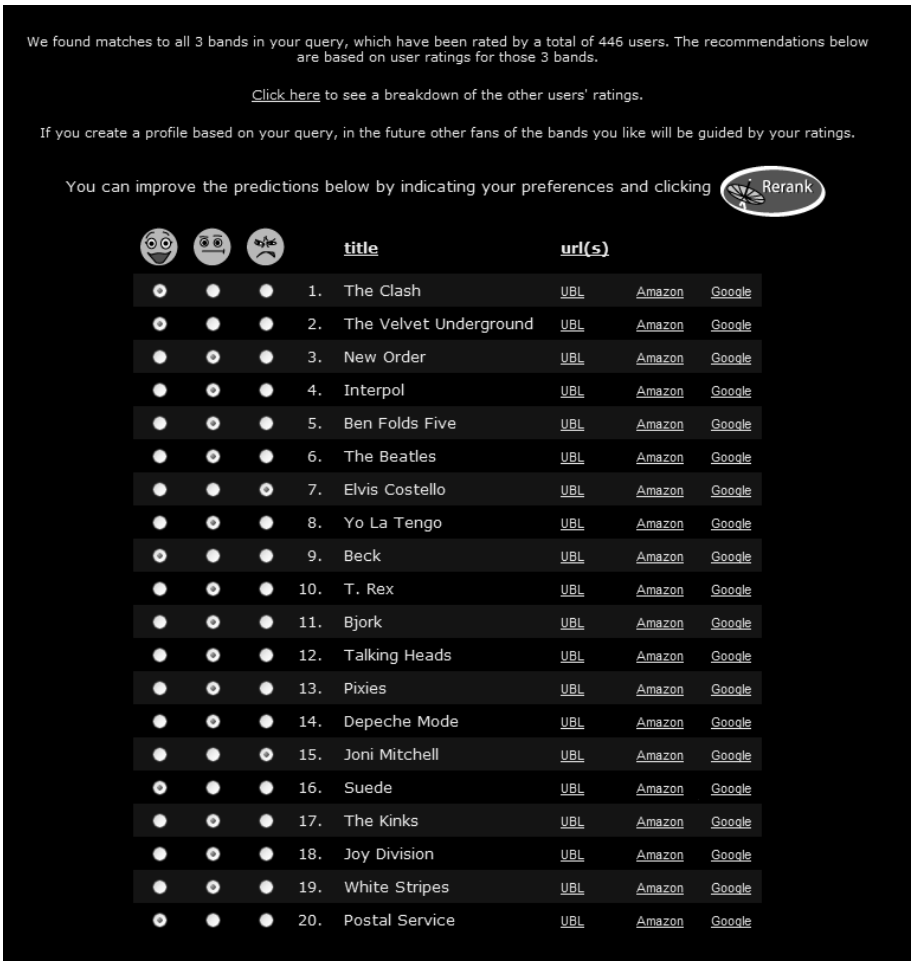


Figura 3.2: Primera pantalla de recomendaciones ofrecida por *Moonranker* para un usuario que ha introducido como intérpretes de música de su gusto: "David Bowie", "The Flaming Lips" y "Pulp". El recomendador puede refinar la búsqueda y a la vez realimentar el sistema indicando su grado de afinidad con las bandas de música recomendadas.

> medida, por ejemplo, en un campo como el de los periódicos digitales donde constantemente aparecen nuevas noticias pero solamente unas pocas son accedidas.

Particularizando un poco en las diferentes técnicas de recomendación, los sistemas colaborativos dependen del solapamiento que exista entre los servicios seleccionados por los diferentes usuarios. Tienen problemas cuando no existe tal solapamiento sino que el espacio que recoge la selección de

los diferentes servicios es, más bien, disperso. Este problema se ve reducido cuando se utilizan sistemas de recomendación que están basados en modelos, y que pueden utilizar descomposición en valores singulares (Strang, 1988), lo cual puede reducir la dimensionalidad del espacio en el que tiene lugar la comparación entre los usuarios (Foltz, 1990; Rosenstein y Lochbaum, 2000). En esta tesis, nos centramos en el filtrado colaborativo como técnica de recomendación, y en particular las similitudes entre los usuarios se llevan a cabo en un espacio de reducida dimensionalidad. De todos modos, la dispersión de los datos representa un importante inconveniente en algunos problemas, como el de la recomendación de noticias, donde al haber tantos servicios (noticias) disponibles, a no ser que se tenga un histórico muy grande de datos del usuario, la probabilidad de que dos usuarios distintos compartan muchos accesos a noticias es bastante pequeño.

Por tanto, el filtrado colaborativo puro es más indicado para aquellos problemas donde los intereses del usuario están focalizados en un conjunto de servicios pequeño y estático. Si el conjunto de servicios cambia muy rápidamente, las selecciones anteriores de usuarios pueden no ser útiles para un nuevo usuario, por lo que no es posible establecer ninguna clase de comparación. Tampoco son adecuadas estas técnicas cuando el conjunto de servicios es muy grande.

Sin embargo, los recomendadores basados en filtrado colaborativo funcionan muy apropiadamente para un usuario que encaja dentro de un perfil donde existen muchos otros usuarios con gustos parecidos. Las recomendaciones pueden no ser adecuadas en los bordes entre clases diferentes de usuarios. Este problema aparece también para los recomendadores demográficos que tratan de clasificar a los usuarios en función de sus características personales. Por otro lado, las técnicas demográficas no sufren del problema de "nuevos usuarios", ya que no necesitan una lista de selecciones realizadas por el usuario. No obstante, tienen el problema de necesitar la información demográfica. Otro problema con el que se encuentran las técnicas demográficas proviene de las cada vez más restrictivas leyes de protección de datos y la gran sensibilidad que existe en cuanto a mantener la privacidad de las transacciones electrónicas, particularmente en el caso del comercio electrónico. Además, la información más útil es, normalmente, aquella que los usuarios son reacios a revelar.

El problema fundamental de las técnicas basadas en contenido viene de la necesidad de acumular una cantidad considerable de accesos para construir un clasificador fiable. También tienen el problema de que están limitadas por el número de servicios disponibles para la recomendación, igual que sucedía para el caso del filtrado colaborativo. Por ejemplo, un recomendador de películas basado en contenido únicamente podrá utilizar información relacionada con la película como nombres de los actores, el director o argumento de la misma, ya que lo que es la película en sí resulta totalmente opaca para el sistema. Esto hace que estas técnicas estén a merced de los datos disponibles para describir el producto en cuestión. Esto es un inconveniente de estas técnicas frente a las colaborativas, que solamente dependen de las selecciones realizadas por los usuarios y que pueden recomendar servicios sin ningún tipo de datos descriptivos de los mismos. Incluso existiendo datos descriptivos suficientes, algunos experimentos han mostrado que los sistemas colaborativos son más exactos que los basados en contenido (Alspector, Koicz y Karunanithi, 1997).

La gran ventaja de los sistemas colaborativos frente a los basados en contenido es su capacidad para recomendar servicios de diferente tipo. En otras palabras, un recomendador basado en contenido entrenado, por ejemplo, con las preferencias de un usuario en música *blues* no podría recomendar servicios relacionados con la música *techno* debido a que ninguna de las características de ambos tipos de música (intérpretes, instrumentos, repertorio, ...) se comparten. Sin embargo, como un recomendador colaborativo se basa en los parecidos entre usuarios, sí que sería capaz de sugerir servicios de categorías distintas si piensa que el usuario puede estar interesado en ellas. Esto tiene un gran interés tanto desde el punto de vista del usuario final como desde el punto de vista comercial de la empresa que está explotando el sistema, ya que un usuario puede ser dirigido hacia servicios que, en principio, no están relacionados directamente con aquellos a los que ha accedido hasta ahora y a los que, probablemente, no accedería por él mismo sin recomendaciones (a pesar de que sí sean de su agrado).

Evidentemente, para que un recomendador sea útil debe ofrecer aquellos servicios que no han sido previamente accedidos o vistos. De hecho, en la metodología que se presentará en la Sección 3.3 se contempla este aspecto, de manera que la primera premisa para recomendar un determinado servicio es que éste no haya sido previamente accedido por el usuario. En el caso

real que se analiza en la presente tesis esto no supone un gran problema; sin embargo, en otros campos, como por ejemplo en un periódico digital, la cosa cambia ya que noticias que puedan parecer similares a algunas ya leídas, pueden presentar nuevas perspectivas o enfoques que pueden ser interesantes para el usuario. Y por otro lado, diferentes presentaciones de una misma noticia pueden no ser útiles, ya que la información que hay en ellas es la misma y, sin embargo, son consideradas como noticias diferentes. Cuando estamos en campos tan complicados para llevar a cabo recomendaciones útiles, es importante hacer algo adicional; en este sentido, el sistema *DailyLearner* (Billsus y Pazzani, 2000) por ejemplo, utiliza un umbral superior de similitud en su recomendador basado en contenido, que permite filtrar aquellas noticias que son similares a otras ya vistas por el usuario. En ocasiones, se prefiere implementar recomendadores que intentan captar la confianza del usuario en lugar de buscar como primer objetivo la utilidad (Burke, 2002); para ello, lo que se hace es recomendar servicios que se sabe con bastante seguridad que serán accedidos por el usuario, típicamente se trata de servicios que ya han sido previamente accedidos por éste; lo que se pretende así es captar la confianza del usuario, que comprueba como el recomendador le sugiere servicios que, en efecto, son de su agrado.

Los sistemas basados en utilidad y conocimiento no presentan el problema del *ramp-up* ni tampoco problemas cuando existen datos dispersos, ya que las recomendaciones no están basadas en ningún tipo de procesado estadístico. Las técnicas basadas en utilidad se basan en que el sistema crea una función de utilidad sobre todas las características de servicios disponibles por el usuario. Un evidente beneficio de estas técnicas reside en que pueden incorporar muchos factores diferentes que contribuyen al valor de un producto, desde el plazo de entrega del producto hasta la garantía del mismo. Además, estas características que no son estrictamente del producto pueden ser extremadamente útiles para determinado tipo de usuarios. Por tanto, un sistema basado en utilidad permite al usuario expresar todos los aspectos que necesita para obtener la recomendación de un servicio, más allá de lo que son sus propias características o las del servicio (Burke, 2002).

Desde el punto de vista negativo, la flexibilidad que ofrecen los sistemas basados en utilidad es también un inconveniente, ya que el usuario debe construir totalmente una función que determine sus preferencias, lo cual implica pesar la significancia de cada posible característica. Esto puede ser

factible para servicios que presentan pocas características (precio, calidad y fecha de entrega, por ejemplo) pero no para otros campos mucho más complejos y donde los gustos son extremadamente subjetivos, como por ejemplo cuando se trata con música o películas.

Por otro lado, los sistemas basados en conocimiento presentan el obvio problema de la necesidad de adquirir conocimiento. Podemos hablar de tres tipos de conocimiento en este tipo de sistemas:

- *Conocimiento del producto.* Se trata del conocimiento de los servicios que se recomiendan y de sus características. Por ejemplo, dentro de un portal web gastronómico, un recomendador de vinos internacionales debe saber que el vino de "Burdeos" es un vino francés.

- *Funcionalidad.* El sistema debe ser capaz de relacionar las necesidades del usuario con el servicio que puede satisfacerlas. En el mismo ejemplo del recomendador de vinos, el sistema debe saber que el vino más adecuado para un pescado fresco es el "vino blanco".

- *Conocimiento del usuario.* Para poder ofrecer buenas recomendaciones, el sistema debe poseer conocimiento acerca del usuario. Este conocimiento puede ser desde información demográfica hasta conocimiento más específico acerca de la necesidad del usuario para la cual se está buscando la recomendación. Por ejemplo, un conocimiento útil en el caso del recomendador de vinos sería saber si el usuario prefiere "vino blanco" español o el similar *"vinho verde"* portugués.

De todas maneras, a pesar del inconveniente que supone la necesidad de disponer de esta información, los sistemas basados en conocimiento también presentan ciertas ventajas. Por ejemplo, estos sistemas son muy apropiados para usuarios esporádicos, ya que se pide menos información al usuario que en el caso de los sistemas basados en utilidad.

En la Tabla 3.2 se resumen las ventajas e inconvenientes de cada una de las cinco técnicas de recomendación explicadas. La notación seguida en esta tabla es la siguiente:

- A: Capacidad para identificar usuarios similares que acceden a servicios de diferente tipo; en otras palabras, capacidad para identificar

usuarios similares a pesar de que puedan parecer heterogéneos analizando estrictamente los accesos a servicios que realizan.

- B: No necesita conocimiento del campo que tratan los servicios.
- C: Adaptibilidad, es decir, capacidad de mejorar con el tiempo.
- D: El sistema es capaz de autorealimentarse.
- E: No presenta problemas de *ramp-up*.
- F: Sensibilidad a cambios en las preferencias.
- G: Puede incluir características que no tienen que ver, estrictamente, con el servicio a recomendar.
- H: Capacidad de relacionar las necesidades de los usuarios con los servicios a recomendar.
- I: Problema de *ramp-up* para nuevos usuarios.
- J: Problema de *ramp-up* para nuevos servicios.
- K: No funciona bien para usuarios cuyo comportamiento pueda considerarse como difuso, en el sentido de que pertenece a dos grupos de usuarios distintos, cada uno de los cuales define un claro comportamiento.
- L: Depende de disponer de un histórico de datos grande.
- M: Problema de la "estabilidad-plasticidad".
- N: Debe obtener información demográfica.
- O: El usuario debe introducir una función de utilidad.
- P: No posee capacidad de aprendizaje.
- Q: Se necesita conocimiento.

Como se puede observar en la Tabla 3.2, las técnicas colaborativas y demográficas son las únicas que tienen la capacidad para identificar y, por tanto, recomendar con éxito a usuarios heterogéneos; en otras palabras, son capaces de encontrar similitudes entre usuarios más allá de que accedan, o

Tabla 3.2: Comparativa de las ventajas e inconvenientes de las diferentes técnicas de recomendación.

Técnica	**Ventajas**	**Inconvenientes**
Colaborativa	A, B, C, D	I, J, K, L, M
Basada en contenido	B, C, D	I, L, M
Demográfica	A, B, C	I, K, L, M, N
Basada en utilidad	E, F, G	O, P
Basada en conocimiento	E, F, G, H	P, Q

no, a los mismos servicios. Las técnicas basadas en conocimiento también pueden tener esta capacidad, aunque es a expensas de poseer el conocimiento adecuado para ello.

Todas las técnicas basadas en aprendizaje (colaborativas, basadas en contenido y demográficas) padecen el problema del *ramp-up* de una forma u otra. Otro de los problemas considerados es el que ya se comentó en el Capítulo 2 al hablar de la ART: el dilema de la "estabilidad-plasticidad", que en este caso se particulariza en que una vez que las preferencias de un usuario han sido establecidas en el sistema, resulta difícil cambiar sus preferencias. Pensemos de nuevo en el recomendador de vinos del portal gastronómico; si un consumidor habitual de vino se hace abstemio, seguirá recibiendo recomendaciones de vinos por parte de recomendadores colaborativos o basados en contenido durante algún tiempo, hasta que nuevas selecciones de servicios hagan cambiar las recomendaciones del sistema. Muchos sistemas adaptativos incorporan algún tipo de peso que hace decrecer la influencia de previas selecciones conforme pasa el tiempo, es decir, que selecciones recientes pesan más en las recomendaciones a ofrecer que selecciones pasadas. El inconveniente de este tipo de procesado es que pierde información sobre servicios que le interesan al usuario a largo plazo, y a los que solamente accede esporádicamente (Billsus y Pazzani, 2000; Schwab et al., 2001). Por otro lado, los sistemas basados en conocimiento y utilidad responden a las necesidades del usuario de manera inmediata, y no necesitan por tanto ningún tipo de reeentrenamiento al cambiar las preferencias del usuario.

El problema del *ramp-up* tiene además el efecto colateral de excluir a usuarios no habituales de recibir todos los beneficios de las técnicas basadas en

contenido o colaborativas. Es posible realizar recomendaciones en un portal de comercio electrónico, del tipo: "La gente que compra el artículo X también compra el artículo Y", que es la manera de funcionar de *Amazon.com*. No obstante, esta metodología tiene pocas de las ventajas comúnmente asociadas con el concepto de filtrado colaborativo. Los sistemas basados en aprendizaje funcionan mejor para aquellos usuarios que están dispuestos a introducir sus preferencias en el sistema, aunque normalmente los usuarios son reacios a ofrecer esta información, que no tiene por qué ser fiable, además de que implica, en ocasiones, una gran pérdida de tiempo. Los sistemas basados en conocimiento y utilidad tienen algunos problemas en este sentido debido a que no funcionan en base a ningún histórico de preferencias del usuario. Por otro lado, los sistemas basados en utilidad tienen problemas para aquellos usuarios no habituales que pueden no estar dispuestos a desarrollar un función de utilidad simplemente para consultar un catálogo, por ejemplo.

3.3. Metodología de recomendación propuesta

3.3.1. ¿Por qué filtrado colaborativo?

El sistema de recomendaciones propuesto en nuestra metodología será explicado con más detalle en el Capítulo 5 de esta tesis. El sistema empleado es de tipo colaborativo, constando de una primera etapa donde se establecen grupos de usuarios mediante algoritmos de *clustering*, y de una segunda etapa donde los nuevos usuarios se comparan con los distintos grupos. Una vez determinado el grupo al que pertenece el nuevo usuario, se le recomienda el servicio más probable de su grupo, siempre y cuando no haya accedido a él previamente.

Las razones por las cuales se ha optado por un sistema colaborativo en lugar de alguna otra de las otras técnicas anteriormente explicadas son las siguientes:

- Para un portal web como el analizado en la presente tesis, *Infoville XXI* (`http://www.infoville.es/`), que es un portal que ofrece servicios a los ciudadanos de la Comunidad Valenciana, una técnica colaborativa parece la más adecuada ya que este tipo de portales están pensados

para recibir multitud de visitas. Además, los usuarios pueden ser muy diferentes entre sí; de hecho, cuanto más diferentes sean los usuarios, mayor será el éxito del portal ya que habrá llegado a un mayor espectro de la población. Si además se mira la metodología propuesta como algo más general fuera del ámbito de esta aplicación particular, las técnicas colaborativas se han mostrado como más exactas que otros sistemas de recomendación (Alspector et al., 1997), además de ser las que menos requisitos requieren, en cuanto a información que deban ofrecer los usuarios, o a conocimiento de los contenidos y características del portal.

- Cuando existe una gran cantidad de servicios disponibles en el portal (como es el caso de *Infoville XXI*), los cuales son de diferente naturaleza, no parece razonable usar un sistema basado en contenido, que lo que hace es buscar comportamientos repetidos entre usuarios, ya que será difícil encontrar este tipo de comportamientos.

- Un recomendador demográfico tampoco parece el más adecuado ya que necesita información demográfica, y nuestra metodología tiene como premisa no solicitar ninguna información al usuario. Además, en muchas ocasiones, la información puede no ser fiable, siendo los usuarios bastante reacios a ofrecerla. Esta misma razón nos sirve para descartar los sistemas basados en utilidad que necesitan del usuario la inclusión de una función de utilidad.

- Por último, los sistemas basados en conocimiento, además de necesitar conocimiento, no poseen capacidad de aprendizaje por lo cual no pueden ajustarse ni tratar adecuadamente a los usuarios en función de su comportamiento.

- Además, intentaremos minimizar los inconvenientes que la Tabla 3.2 refleja respecto al filtrado colaborativo:

    - Para evitar el problema del *ramp-up* de usuarios, lo que se hará es recomendar inicialmente al nuevo usuario el servicio más probable del portal sin utilizar información del *clustering* y, por tanto, sin utilizar filtrado colaborativo hasta que haya suficiente información para que éste funcione correctamente. Esto permite salvar el problema del *ramp-up* para nuevos usuarios, funcionando

aceptablemente bien para estos primeros accesos que suelen ser un difícil escollo para los sistemas colaborativos.

- Para aliviar el problema del *ramp-up* de servicios, lo que se hace es trabajar con etiquetas informativas, llamadas *descriptores* o *page categories* que aglutinan a servicios de naturaleza similar (Cadez, D., Meek, Smyth y White, 2001; Martín, 2003). Como veremos con mayor profundidad en los siguientes capítulos, los parecidos entre usuarios se miden en un espacio definido por las probabilidades de descriptores, y solamente a la hora de recomendar se salta al espacio de servicios. Por tanto, nuevos servicios pueden ser "escondidos" en el descriptor correspondiente, y los accesos de usuarios a ellos serán los que hagan que ese servicio sea considerado para recomendaciones o no. Esto permite que la incorporación de un nuevo servicio no suponga ningún cambio en el sistema, sino que se trate de un proceso natural llevado a cabo desde los descriptores.

- Algo más complicado es el problema de los usuarios difusos, es decir, aquellos individuos que presentan un comportamiento que podría considerarse como un híbrido entre varios comportamientos de grupos de usuarios bien definidos. Aun así, el agrupamiento de usuarios que consituye la primera etapa del sistema está definido para que haga un buen procesado de los usuarios, encontrando grupos suficientemente representativos, para lo que se utilizan diferentes algoritmos de *clustering* de considerable potencia. Además, como se verá en la Sección 3.4, también se propone como proyección futura el desarrollo de algoritmos (alguno de ellos utilizando conceptos de Lógica Difusa) para regir cómo el sistema se adapta a los nuevos usuarios en función de cómo éstos reaccionen ante las recomendaciones.

- A pesar de que sí que es necesario disponer de un histórico grande de datos, debe pensarse que para que un portal web se plantee realizar un sistema de recomendaciones, normalmente es porque el portal ya lleva un desarrollo previo y en función de él es cuando se plantea mejorarlo mediante un recomendador. Esta fase previa que constituye el inicio del portal puede servir a su vez como recogida de datos para empezar a desarrollar el recomendador,

que es lo que ocurre, por ejemplo, en el caso que nos ocupa.

- El problema de la estabilidad-plasticidad recibe un tratamiento adecuado ya que dentro de las técnicas de *clustering* que constituyen el primer paso del recomendador, una de ellas es la ART, como se ha visto en el Capítulo 2, que es una técnica específicamente diseñada para solventar este problema. Además, otras técnicas de agrupamiento utilizadas, como por ejemplo SOM, presentan resultados similares a la ART, con lo que en principio este problema no es demasiado acusado en nuestra metodología. Además, se proponen estrategias para adaptar las recomendaciones a las nuevas preferencias de los usuarios y, cuando el número de usuarios nuevos (no incluidos en la fase de agrupamiento) es grande, puede realizarse una fase de re-agrupamiento, que permite volver a solventar el problema en cuestión.

3.3.2. Fases en el desarrollo del recomendador

Como ya se comentó en el prólogo, la metodología presentada consta de cuatro grandes fases. La primera de ellas es el desarrollo de un modelo de usuario web, mientras que las otras tres fases afectan al desarrollo del recomendador:

1. *Agrupamiento de usuarios.* La segunda fase es la que hace referencia a la utilización de algoritmos de *clustering* para establecer grupos de usuarios de comportamiento similar dentro del portal web. Los dos algoritmos que han sido seleccionados como los más idóneos dentro de todos los que se han utilizado han sido ART y SOM.

2. *Viabilidad de la implementación del recomendador.* Esta fase es clave en nuestra metodología a pesar de que, habitualmente, suele ser ignorada. Lo que se hace es analizar la mejora que supone utilizar un recomendador colaborativo frente a un recomendador trivial que simplemente ofrece el servicio más probable no accedido anteriormente. Si la mejora es apreciable, se deduce que la información aportada por el *clustering* es relevante y, por tanto, resulta adecuada la utilización de este tipo de sistemas. Esto es especialmente interesante ya que permite analizar la viabilidad y el eventual beneficio que se puede obtener de

un recomendador. Como el desarrollo de un recomendador puede llegar a ser bastante costoso, no solamente en términos económicos, esta fase es vital ya que resultados pobres pueden indicar que no sea de interés la implementación de dicho recomendador. Además, realmente lo que se hace no es ver cómo reacciona el usuario ante las recomendaciones, ya que hay que recordar que el sistema todavía no ha sido instalado, sino que se predicen los servicios a los que el usuario accederá por sí mismo. Esto puede verse como un inconveniente ya que no se ve el efecto real de la recomendación, pero es asimismo una ventaja ya que se supone que recomendaciones atractivas permitirán captar mejor el comportamiento del usuario con lo que, realmente, la tasa de éxito calculada puede ser un umbral inferior de lo que sería el éxito real al presentar las recomendaciones. Además, se separa la influencia de la interfaz de la recomendación al usuario de los efectos del conocimiento extraído mediante nuestra aproximación, que es lo que realmente se está evaluando.

3. *Estudio del efecto real de las recomendaciones.* Una vez analizada la viabilidad del recomendador e implementado éste en el sistema, es importante realizar un seguimiento del éxito real de las recomendaciones sobre los usuarios. Esto permitirá mejorar el recomendador; de hecho, en la Sección 3.4 se proponen algunos algoritmos para llevar a cabo esta adaptación conforme se van disponiendo de nuevos datos de los usuarios. De todos modos, cuando el número de usuarios nuevos y distintos a los utilizados en la fase de *clustering* es elevado, la mejor solución es realizar un nuevo agrupamiento sobre todos los usuarios porque puede que hayan aparecido nuevos comportamientos, quizás incluso inducidos por las propias recomendaciones realizadas por el sistema.

3.4. Efecto de las recomendaciones sobre el usuario. Recomendadores adaptativos

Como se ha comentado, la última parte del desarrollo de un recomendador es, necesariamente, el estudio del efecto real que producen las recomendaciones sobre el usuario. A pesar de que esta fase no ha sido realizada en la presente

tesis debido a que no se dispone de este tipo de datos, sí que se han realizado algunas pruebas preliminares con datos simulados. El problema es que los datos sintéticos no son de utilidad cuando se quiere analizar el efecto de las recomendaciones. Esto es por un doble motivo:

1. No existe un protocolo fiable que permita simular si una determinada recomendación es aceptada o no, ya que esto es una decisión que adopta el usuario de manera subjetiva, y por tanto, cualquier tipo de datos sintéticos sobre este particular pueden considerarse como poco menos que aleatorios.

2. El efecto real de las recomendaciones puede depender en gran medida de su interfaz. De hecho, en algunas ocasiones, las recomendaciones esconden detrás de una determinada interfaz atractiva un servicio que difícilmente sería accedido sino fuera por el diseño que incorpora. Justamente la tercera fase de nuestra metodología, que analiza la viabilidad de implementar un sistema de recomendaciones, tiene como una de sus principales ventajas el que separa el efecto de la interfaz de la recomendación del modelado del perfil de los usuarios. Por esta razón, las medidas de rendimiento que utilizamos para analizar si las recomendaciones serán exitosas o no, hay que considerarlas con precaución como un umbral inferior del éxito que tendrán las recomendaciones reales, ya que se entiende que la presentación de unas recomendaciones atractivas ha de afectar de manera positiva la actitud del usuario frente a la recomendación.

El análisis del éxito de las recomendaciones debe valer para mejorar el sistema recomendador. Formas evidentes de llevar a cabo esta mejora es, por ejemplo, analizando el éxito de cada servicio recomendado y promoviendo más los servicios con más éxito, o bien adoptando para aquellos servicios en los que estamos más interesados la misma interfaz que los que tienen más éxito.

Sin embargo, la manera más adecuada de mejorar un recomendador es, probablemente, a través de un sistema adaptativo *on-line*. La idea es que la reacción de cada usuario ante las recomendaciones sirva para realimentar el propio sistema, de tal manera que el recomendador pueda "aprender" a partir del éxito o fracaso de las recomendaciones, refinando por tanto los

prototipos que definen los grupos de usuarios. Este tipo de sistemas puede permitir incorporar comportamientos de usuarios que no están contemplados en una primera instancia y que, no obstante, puedan aparecer con el tiempo y con el uso del portal. Realmente, más que una actualización *online*, lo que suele hacerse para no ralentizar la navegación en el portal es realizar estas actualizaciones en momentos de poco tránsito del portal.

El sistema de recomendaciones propuesto se explica más detalladamente en el Capítulo 5, aunque a *grosso* modo, funciona una vez realizado un *clustering* del conjunto de datos, y establecidos unos prototipos representativos de cada grupo de usuarios, cuando llega un nuevo usuario al portal, primeramente se determina a qué grupo pertenece (grupo o *cluster* ganador), y posteriormente, se recomienda el servicio más probable para ese grupo y al que el usuario en cuestión todavía no ha accedido.

Además, utilizando un recomendador adaptativo podemos, no solamente recomendar aquellos servicios que puedan resultar interesantes sino también llevar a cabo una mejora del ajuste de los grupos encontrados a los datos reales. Lo que se propone es llevar a cabo esta adaptación *post-recomendación* utilizando esquemas basados en *Learning Vector Quantization (LVQ)*: LVQ1, LVQ1 óptimo, LVQ2.1, LVQ3 y LVQ difuso (Ripley, 1996; Alpaydin, 1998).

Estos algoritmos funcionarían acercando al usuario al prototipo ganador si el usuario acepta la recomendación, y alejándolo si no la acepta. No obstante, esto que sería el funcionamiento normal de un LVQ debe adaptarse al marco de las recomendaciones. Pensemos en nuestro comportamiento cuando estamos navegando en Internet, ¿cuántas veces aceptamos una recomendación o un determinado *banner*? Muy pocas, de hecho, las referencias que existen en la literatura sobre el éxito de recomendaciones en portales reales es muy bajo (Geyer-Schulz y Hahsler, 2002), casi siempre por debajo del 10 %. Por tanto, nuestra propuesta es actualizar los prototipos únicamente cuando el usuario acepta la recomendación. Cuando no lo hace los prototipos permanecen inalterados o, en todo caso, se modifican alejando al usuario en una medida mucho menor que el acercamiento que se produce cuando el usuario acepta la recomendación. Debe pensarse que el hecho de que un usuario no acepte una recomendación no quiere decir que ésta no sea de su agrado sino que puede que no disponga de tiempo para consultarla o que haya cerrado

involuntariamente el *banner* donde se le ofrecía la recomendación. Incluso en el caso de que realmente no acepte la recomendación porque no le guste, esto no es un indicador de que no pertenezca al grupo de usuarios donde se le ha clasificado.

Otra posible aproximación para adaptar los grupos es considerar que el *cluster* ganador se va modificando incluyendo a aquellos usuarios que aceptan las recomendaciones propuestas, permaneciendo inalterado si no las acepta. De todos modos, cuando el número de nuevos usuarios es elevado, típicamente igual al usado para obtener los primeros grupos, los algoritmos de *clustering* deben volver a buscar similitudes sobre todo el conjunto de datos porque nuevos comportamientos de usuario pueden haber aparecido.

Capítulo 4

Resultados experimentales en portales web ideales

Resumen del capítulo

Los resultados experimentales obtenidos en esta tesis se muestran en dos capítulos. En este primero, nos centramos en los resultados obtenidos con datos sintéticos. Utilizar datos sintéticos o artificiales es de gran utilidad pues permite realizar un análisis de resultados mucho más completo; por un lado, pueden considerarse diferentes condiciones controladas, observándose el efecto de éstas sobre las herramientas utilizadas; y además, en el caso de técnicas no supervisadas, como la mayoría de las que nos ocupan en la presente tesis, donde no existe una salida deseada con la que comparar, utilizar datos sintéticos permite evaluar la bondad de los modelos obtenidos de una manera eficaz. Los datos sintéticos utilizados han sido generados con un modelo de usuario basado en un simulador de accesos de usuarios a un sitio web. Este simulador ha sido diseñado teniendo en cuenta tanto la información que ha podido extraerse de comportamientos reales de usuarios web como las características de otros conjuntos de datos previamente utilizados en la literatura. El desarrollo de este capítulo será el siguiente: en primer lugar, se describirá el simulador utilizado para generar accesos de usuarios, así como los conjuntos de datos sintéticos seleccionados para la evaluación de los algoritmos; a continuación, se comparará el rendimiento ofrecido por los diferentes algoritmos con estos conjuntos.

4.1. Justificación del uso de datos sintéticos

Cualquier herramienta de MW debe ser aplicable a conjuntos de datos reales si quiere observarse su utilidad práctica. Sin embargo, antes de este paso, es importante estudiar su rendimiento. Además de que resulta más sencillo trabajar con conjuntos de datos artificiales, un análisis exacto y riguroso del rendimiento de cada algoritmo solamente es posible con datos ausentes de ruido, es decir con conjuntos de datos artificiales. Pueden llevarse a cabo evaluaciones similares con datos reales pero además de que su distribución de probabilidad no es conocida, puede existir un nivel considerable de ruido en ellos. Los datos sintéticos generados fueron utilizados para:

- *Obtención de representaciones de sitios Web con diversas características.* La aplicación a diferentes sitios web es un punto crucial de cualquier herramienta de MW, si se desea que estas técnicas sean capaces de funcionar adecuadamente con una gran cantidad de sitios web cubriendo, por tanto, comportamientos y situaciones heterogéneas. No hay disponibles muchos datos reales que registren accesos de usuarios a sitios web, debido a las cada vez más restrictivas leyes de protección de datos y a la confidencialidad que mantienen la mayoría de empresas sobre los datos de accesos a sus portales web. De todos modos, aunque se disponga de un determinado conjunto de datos, cualquier resultado obtenido sobre ese conjunto, será solamente válido para dicho conjunto o para aquellos que presenten unas características muy similares.

- *Evaluación del rendimiento de un determinado algoritmo.* Antes de la aplicación real de una determinada técnica o algoritmo, resulta absolutamente necesario llevar a cabo un riguroso análisis sobre el rendimiento que puede ofrecer. En particular, si estamos trabajando con algoritmos de *clustering*, que como ya se ha comentado en anteriores capítulos, funcionan con aprendizaje no supervisado, y se desea realizar un agrupamiento en un conjunto de datos reales, surge el problema de averiguar si los grupos encontrados realmente son correctos o no. Por contra, cuando un conjunto artificial de datos es generado bajo una situación controlada, los grupos que deben ser encontrados por los algoritmos se definen *a priori* y, por tanto, se puede realizar una evaluación del rendimiento de los algoritmos utilizados de una manera

eficiente.

Por tanto, se desarrolló un simulador de accesos de usuario a un portal web con el objetivo de poder crear conjuntos de datos sintéticos "a la carta", que sean paradigma de diferentes situaciones que puedan darse en un portal web.

4.2. Simulador de accesos de usuario

El simulador de accessos desarrollado pretende emular accesos de usuario a un portal web sintético y universal, por lo que realmente se trata de un modelo de usuario. A pesar de que se han introducido ciertas restricciones en su implementación con el fin de respetar los habituales comportamientos que se observan en la práctica en un portal web, el simulador es fácilmente modificable para, o bien dejar de tener en cuenta estas restricciones, o bien incorporar otras más que puedan considerarse interesantes para un caso concreto. Aun así, en su estado actual, el simulador es capaz de crear comportamientos de usuarios (determinados a través de sus accesos al portal web) de una manera considerablemente general y realista. Por tanto, será capaz de generar conjuntos de datos sintéticos que sean paradigma de situaciones reales en portales web, los cuales podrán ser utilizados, posteriormente, para comparar el rendimiento de los diferentes algoritmos de agrupamiento y, de este modo, evaluar qué algoritmos son los más adecuados para cada conjunto de datos.

4.2.1. Restricciones del modelo de usuario

Para el desarrollo del simulador se tuvieron en cuenta determinadas características y restricciones que pueden observarse en accessos a sitios web reales, y que han sido puestas de manifiesto por diversos autores (Balaguer y Palomares, 2003; Breslau, Cao, Fan, Phillips y Shenker, 1999; Andersen et al., 2000; Su, Ye-Lu y Zhang, 2000). En particular, las dos restricciones que se han tenido en cuenta para la implementación del simulador son las siguientes:

1. El número de usuarios que abren una nueva sesión en un portal web decrece al aumentar el número de sesiones. Esta restricción es evidente

ya que todos los usuarios que acceden al portal lo hacen, al menos, una vez, siendo menos los usuarios que vuelvan a acceder al portal en otra sesión posterior, y así sucesivamente.

2. Dentro de cada sesión, el número de usuarios que acceden a un servicio es menor cuando el número de servicios previamente consultados es mayor. Esta restricción también tiene su lógica, ya que todos los usuarios[1] solicitarán al menos un servicio, y dependiendo de la profundidad de la sesión que realice el usuario, el número de servicios accedidos será mayor o menor, pero como las sesiones más profundas serán las menos, esta restricción surge de manera natural. Además, esta segunda restricción se manifiesta, en mayor grado, al ir aumentando el número de sesiones previamente accedidas, ya que conforme un usuario abre nuevas sesiones conoce más el portal y, por tanto, va directamente hacia aquellos servicios que realmente le interesan. Por contra, en las primeras sesiones la navegación es más caótica y se puede acceder a algunos servicios que realmente no interesan como resultado del proceso de prueba y error que requiere tener un conocimiento del portal.

Estas dos restricciones pueden ser descritas de manera adecuada a través de una distribución exponencial, donde el número de usuarios N que acceden a un determinado número x de servicios en la sesión y-ésima vendría dado por:

$$N(x, y) = N_M \cdot e^{-(\alpha \cdot x + \beta \cdot y)} \tag{4.1}$$

donde N_M es el máximo número de usuarios del portal, es decir, aquellos que acceden al menos a un servicio en al menos una sesión; siendo α y β constantes cuyo valor determina la pendiente con que decae el número de usuarios. La Figura 4.1 representa estas restricciones para un caso particular generado por el simulador. Como se ha comentado anteriormente, estas restricciones han sido contempladas por otros autores y observadas en portales reales. A modo de ejemplo, en la Figura 4.2, se representa de manera separada la evolución del número de usuarios respecto al de servicios y sesiones para un caso real, el del portal web *Infoville XXI*, que será objeto de

[1] A efectos prácticos consideraremos como usuarios solamente a los usuarios "útiles", entendiendo por útiles aquellos que al menos solicitan un servicio de los ofrecidos por el portal.

un estudio más profundo en el siguiente capítulo, comprobándose como la distribución exponencial describe de manera adecuada estas evoluciones.

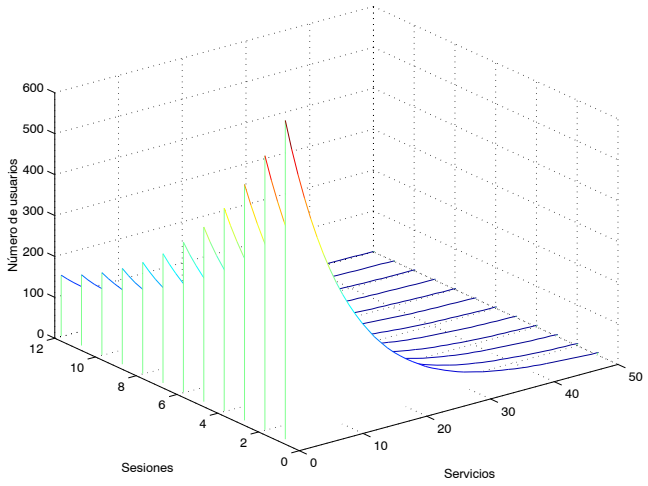


Figura 4.1: Restricciones contempladas por el simulador de accesos de usuarios para un portal web donde se admite una profundidad de 50 servicios por sesión y el máximo de sesiones abiertas por los usuarios del portal es de 12. En esta figura se supone un decrecimiento exponencial del número de usuarios respecto al de servicios y sesiones.

A pesar de que la distribución exponencial es bastante adecuada y fiel a los propósitos de estas restricciones, existe la posibilidad de cambiar la dependencia entre usuarios, servicios o sesiones utilizando otro tipo de distribución. En particular, algunas distribuciones que podrían representar las restricciones de manera adecuada serían las siguientes (Yates y Goodman, 1999):

- *Distribución aleatoria geométrica.* Considerando que x sea, o bien el número de servicios, o bien el de sesiones, la dependencia del número de usuarios N respecto a estas variables vendría dado por:

$$N(x) = \left\{ \begin{array}{ll} N_M \cdot p \cdot (1-p)^{x-1}, & x = 1, 2, \ldots \\ 0, & \textit{en otro caso} \end{array} \right\} \tag{4.2}$$

  siendo p un parámetro tal que $p \in [0, 1]$.

- *Distribución aleatoria binomial.*

$$N(x) = \left\{ \begin{array}{ll} \binom{n}{x} N_M \cdot p^x \cdot (1-p)^{n-x}, & x = 0, 1, \ldots, n \\ 0, & \textit{en otro caso} \end{array} \right\} \tag{4.3}$$

  siendo p un parámetro tal que $p \in [0, 1]$ y n un entero, tal que $n \geq 1$.

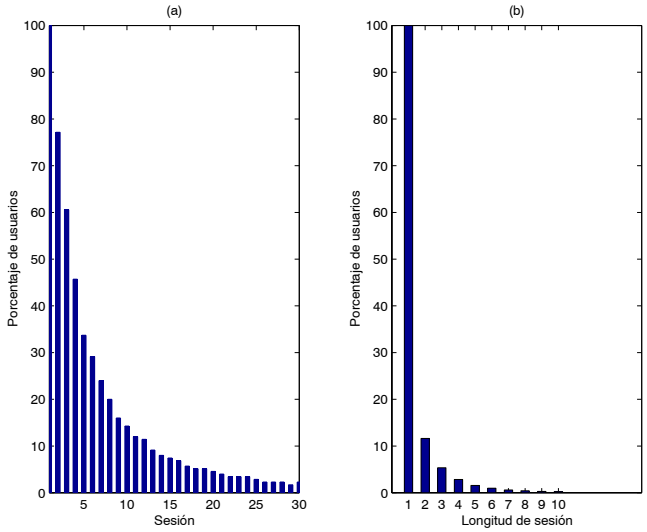


Figura 4.2: Histogramas (normalizados a porcentajes) referidos a accesos al portal web *Infoville XXI* (`http://www.infoville.es/`). (a) representa el porcentaje de usuarios frente al número de sesiones abiertas; y (b) representa la cantidad de usuarios en función de la longitud de la sesión, es decir, del número de servicios accedidos dentro de una misma sesión.

- *Distribución aleatoria de Pascal.*

$$N(x) = \left\{ \begin{array}{ll} \binom{x-1}{k-1} N_M \cdot p^k \cdot (1-p)^{x-k}, & x = k, k+1, \ldots \\ 0, & \textit{en otro caso} \end{array} \right\} \quad (4.4)$$

  siendo p un parámetro tal que $p \in [0,1]$ y k un entero tal que $k \geq 1$. Como puede observarse, La distribución geométrica es equivalente a la de Pascal para $k = 1$.

- *Distribución aleatoria de Poisson.*

$$N(x) = \left\{ \begin{array}{ll} N_M \cdot \alpha^x \cdot \left(\frac{e^{-\alpha}}{x!} \right), & x = 0, 1, 2, \ldots \\ 0, & \textit{en otro caso} \end{array} \right\} \quad (4.5)$$

  siendo α un parámetro cuyo valor estará relacionado con el valor medio de la distribución.

Es inmediato incorporar cualquiera de estas distribuciones en lugar de la exponencial para representar las restricciones del modelo de usuario, aunque las simulaciones realizadas parecen indicar que la exponencial es adecuada. La distribución geométrica y la de Poisson pueden ser también bastante indicadas, si bien la de Poisson es extremadamente sensible al valor de α.

4.2.2. Simulación en el espacio de descriptores

El simulador planteado es un modelo de usuario que intenta emular accesos de usuarios a los diferentes servicios de un portal web. Una vez desarrollado el simulador, el siguiente paso será hacer uso de los algoritmos de *clustering* para encontrar parecidos en el comportamiento que presentan los diferentes usuarios del portal. Utilizaremos los conjuntos de datos artificiales para llevar a cabo una segmentación de usuarios dependiendo de sus gustos, emplazando en el mismo grupo a aquellos usuarios que presentan parecidos intereses. No obstante, este proceso es muy difícil de llevar a cabo con datos de entrada de una elevada dimensión por dos razones fundamentalmente; por un lado, es complicado encontrar similitudes y diferencias entre usuarios en un espacio de elevada dimensionalidad, especialmente si el número de componentes que definen el espacio donde llevar a cabo el *clustering* (número de servicios del portal en nuestro caso) es del mismo orden o incluso mayor que el número de patrones disponibles (usuarios, en nuestro caso); por otro lado, aunque pudiesen encontrarse unos determinados grupos de usuarios, sería complicado realizar un proceso de extracción de conocimiento sobre los grupos encontrados, ya que sería difícil encontrar unas reglas o pautas de comportamiento comunes entre los usuarios de un mismo grupo.

Usualmente, los portales web comerciales están formados por una cantidad elevada de servicios (en ocasiones, millares de ellos), por lo que llevar a cabo un proceso de agrupamiento de usuarios en un espacio definido por servicios resulta inviable en la práctica en una gran cantidad de ocasiones. Se propone trabajar con un espacio de dimensionalidad menor, donde cada dimensión, cada componente del espacio considerado, tiene un significado en sí mismo (Cadez et al., 2001; Martín, 2003). Las componentes de este nuevo espacio se llaman "descriptores" o "etiquetas", y cada una de ellas está formada por un conjunto de servicios similares. Por ejemplo, en un periódico deportivo digital, todas las noticias relacionadas con "fútbol" pueden incluirse en el mismo descriptor, y lo mismo puede hacerse para aquellas noticias que están relacionadas con "baloncesto", "tenis", "ciclismo" o cualesquiera categorías consideradas por el *webmaster*. En el espacio definido por descriptores, los grupos de usuarios que puedan ser encontrados son más informativos y fáciles de entender (Balaguer y Palomares, 2003; Martín, 2003). Dependiendo de las características del portal web considerado, un determinado servicio puede

asignarse a más de un descriptor; por ejemplo, en el portal web de una universidad, el servicio "calendario de exámenes" puede pertenecer a los descriptores "Información para profesores" y también a "Información para estudiantes".

Usualmente, los descriptores están predeterminados por el *webmaster* del portal. Es más, en algunos tipos de portal, los descriptores estándar están ya predefinidos; sería el caso de las secciones de un periódico ("nacional", "internacional", "deportes", etc.) o de las posibles etiquetas que pueden aparecer en el portal de un banco ("cuentas", "préstamos", "tarjetas", etc.). Esto es importante ya que habitualmente los descriptores no son improvisados por el administrador del portal, sino que vienen dados por el portal con el que se está trabajando. Esto sucede para la mayoría de portales grandes, en los cuales puede ser interesante llevar a cabo una labor de agrupamiento y recomendación como la propuesta en la presente tesis. Lo más habitual es que aquellos portales que no tienen una estructura de descriptores predefinida no estén tampoco interesados en utilizar estrategias como las aquí presentadas. Aun así, para aquellos portales que únicamente constan de servicios y no tienen descriptores predefinidos, puede obtenerse un conjunto significativo de descriptores sobre el cual llevar a cabo el agrupamiento mediante diferentes técnicas. Se han realizado algunas simulaciones con aproximaciones basadas en Análisis de Componentes Principales (Jolliffe, 1986), minería de texto (*text mining*) (Lagus, 2000) o agrupación de servicios por frecuencia de accesos (Martín et al., 2004), siendo esta última la que mostró unos resultados más prometedores, si bien tiene el inconveniente de que puede perderse el significado de los descriptores, para lo cual la técnica más apropiada estaría basada en *text mining*. De todos modos, la obtención automática de descriptores no es un objetivo principal de la tesis, debido a que como se ha comentado, las etiquetas suelen venir dadas por el propio tipo de portal web con el que se esté trabajando y, en todo caso, pueden ser también elegidas por el *webmaster* que, a menudo, preferirá elegir los descriptores para llevar a cabo un desarrollo jerárquico del portal en lugar de que se los elija automáticamente una herramienta que desconoce.

4.2.3. Funcionamiento del simulador

El objetivo principal del modelo de usuario considerado es el de generar conjuntos de datos sintéticos que representan accesos de usuarios a un portal web, para que después herramientas de *clustering* encuentren parecidos inter-usuario. Como ya se ha comentado, es preferible realizar los agrupamientos de usuarios en un espacio de dimensión "manejable", razón por la cual la información que mayormente utilizarán los algoritmos de *clustering* serán los accesos de los usuarios en el espacio definido por descriptores. En particular, las coordenadas del espacio en el que trabajarán los algoritmos de agrupamiento representan la frecuencia normalizada de los accesos de un usuario a un determinado descriptor. Por ejemplo, si consideramos un portal web de servicios al ciudadano que solamente consta de dos descriptores, y un usuario u_1 que registra 45 accesos a este portal, 30 de ellos relacionados con el descriptor D_1 "Ocio" y 15 con el descriptor D_2 "Servicios sanitarios", este usuario vendrá descrito por un vector $(0{,}67, 0{,}33)$ que representa la frecuencia normalizada, es decir, la probabilidad *a priori* de acceso a los descriptores. Estos vectores serán, por tanto, usados por los algoritmos de *clustering* para buscar similitudes y diferencias entre los usuarios, encontrando de este modo aquellos grupos que representen los diferentes comportamientos observados en los accesos registrados de usuarios.

La Figura 4.3 representa con detalle el funcionamiento del simulador. Se empezará generando los grupos de usuarios en un espacio definido por las probabilidades *a priori* de los descriptores, de manera que se obtendrá para cada usuario un vector como el del ejemplo anteriormente citado. Posteriormente, los accesos de los usuarios a los servicios se obtendrán a partir de la relación entre servicios y descriptores y finalmente, tanto la información de los accesos a descriptores como a servicios serán almacenadas en sendos tensores, que recogen la información de los accesos de cada usuario en una sesión determinada a cada descriptor o servicio del portal. Cabe decir que a pesar de que el agrupamiento se llevará a cabo en el espacio definido por las frecuencias normalizadas de descriptores, la información en el espacio de servicios también se ofrece por los siguientes motivos:

1. En portales pequeños, donde se tiene o se considera, por interés, un reducido número de servicios, el agrupamiento sí que puede llevarse a cabo en el espacio de servicios.

2. Cuando llega la hora de aplicar la metodología propuesta en esta tesis a portales reales, debemos comparar los portales reales con los datos artificiales que hemos generado con el fin de conocer qué algoritmo es potencialmente el más adecuado para ese portal real. En este sentido, conocer los accesos a servicios puede ser un factor más a valorar para comparar si realmente el portal real se ve bien representado por el correspondiente portal simulado.

3. Por completitud, debido a que el sistema propuesto está pensado que se incorpore a la herramienta comercial de diseño de portales web iSUM[2] (`http://www.isum.com/`), y esta herramienta almacena los accesos a servicios.

Profundizando un poco más en la manera de trabajar del simulador, como se muestra en la Figura 4.3, el primer paso para la generación de los grupos de usuarios, será cargar los datos que utilizará el modelo. Parámetros obligatorios son los siguientes: número de servicios del portal, número de descriptores, máximo número de sesiones, máxima profundidad, número de grupos de usuario, número de usuarios en cada grupo y las constantes α y β que definen las caídas exponenciales referidas a las dos restricciones del modelo de usuario. Además, dependiendo del tipo de generación que se haga, como veremos a continuación, puede ser necesario conocer también otros dos parámetros como son la probabilidad *a prori* de que sea accedido cada descriptor (frecuencias de acceso normalizadas) y la correspondencia existente entre los descriptores y los servicios del portal. Por último, dependiendo del tipo de simulación que se haga, las estadísticas de los grupos de usuarios requerirán un tipo de información u otro.

El siguiente bloque que aparece en el diagrama de la Figura 4.3 pregunta acerca de si la correspondencia entre descriptores y servicios está indicada en el fichero de datos previamente cargado, es decir si la relación entre descriptores y servicios es conocida o no. Como ya se ha comentado, esta relación es habitualmente conocida, viniendo de hecho definida por defecto con el tipo de portal muy a menudo. En el caso de la generación de datos sintéticos que nos planteamos aquí, puede resultar algo complejo establecer esta relación cuando el número de servicios y/o descriptores sea elevado, ya

[2]iSUM es un producto desarrollado y distribuido por la empresa tecnológica Tissat, S.A. (`http://www.tissat.es/`)

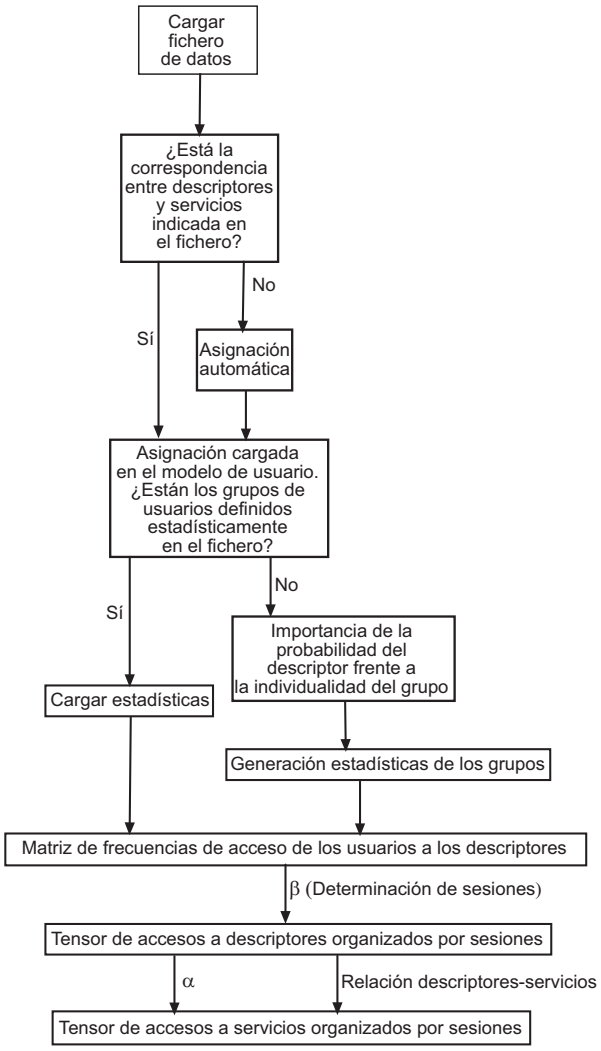


Figura 4.3: Diagrama de bloques del procesado que realiza el modelo de usuario desde la carga del fichero con los parámetros del simulador hasta la obtención de sendos tensores registrando los accesos de los usuarios, organizados por sesiones, a servicios y descriptores.

que no estamos con portales reales sino con datos artificiales y, por tanto, los descriptores no tienen un significado en sí mismos. Si nos encontramos en este caso, puede realizarse una asignación automática por parte del simulador, sujeta a las restricciones que se consideren oportunas (número máximo de servicios por descriptor, posibilidad de solapamiento, ...). Si, por contra, esta correspondencia está explicitada, este paso de asignación automática no sería llevado a cabo.

Una vez cargada esta relación entre servicios y descriptores, se vuelve a plantear otra pregunta acerca del fichero de datos. La pregunta es si los grupos de usuarios se encuentran estadísticamente definidos en el fichero. Suponiendo grupos que sigan una distribución normal[3], esta pregunta se refiere a las correspondientes matrices de centroides y covarianzas (o bien, desviaciones estándar) necesarias para definir donde están situados los centros de cada grupo y cómo se reparten en cada una de las direcciones. Si estas matrices existen en el fichero de datos, basta con cargarlas, y a continuación generar una secuencia pseudo-aleatoria que siga una distribución normal usando los parámetros cargados en estas matrices para generar una matriz de probabilidades de acceso de los usuarios a los descriptores, que será la que se utilice para buscar semejanzas por los algoritmos de *clustering* en el espacio de descriptores. En el caso de que no se disponga de esta información en el fichero, se puede escoger qué porcentaje de importancia se le otorga a la frecuencia de accesos a los descriptores para la generación de los grupos, de tal forma que la matriz de probabilidades vendrá determinada en el porcentaje escogido por las propias probabilidades *a priori* de los descriptores. Por tanto, las frecuencias normalizadas de los descriptores se usarán como centroides de los grupos y en torno a ellos se generará una distribución normal; cuanto mayor sea la importancia de la frecuencia de acceso a los descriptores menos disperso estará el grupo, y por contra si esta importancia es menor, el grupo, aunque centrado en las probabilidades *a priori* de los descriptores, será mucho más disperso. Además, una vez que se tiene esta información, puede realizarse una corrección de la desviación estándar en alguna de las direcciones (componentes correspondientes a los descriptores), generando de esta manera la matriz de probabilidad que usarán los algoritmos de *clustering*.

Una vez que se tiene la matriz que define las frecuencias de acceso de los usuarios a los descriptores, que tendrá dimensiones $N \times N_D$, siendo N el número de usuarios y N_D el número de descriptores, el último paso es el de obtener la información de los accesos pero en el espacio de servicios. Esto se lleva cabo en dos pasos, en el primero de ellos, se lleva a cabo la determi-

[3]Se ha utilizado una distribución normal para generar los grupos de usuarios debido a que es una distribución ampliamente estudiada y controlable, que permitirá realizar posteriores tests de normalidad a los grupos encontrados por los algoritmos de *clustering* como una prueba de robustez del agrupamiento encontrado, y también debido a la gran cantidad de casuística que puede modelar, como establece el Teorema del Límite Central.

nación de sesiones a partir de los accesos en el espacio de descriptores. Por tanto, se hará uso del parámetro β que define la dependencia entre sesiones y número de usuarios para obtener la frecuencia de accesos de cada usuario a cada descriptor (ordenados por sesiones). Estos accesos serán almacenados en un tensor T_1 cuya dimensión será $N \times N_D \times N_{S_{max}}$, siendo $N_{S_{max}}$ el máximo número de sesiones que se pueden abrir por un mismo usuario. Para entender cómo almacena los datos este tensor, consideremos un ejemplo. Supongamos que estamos simulando un portal con $N_D = 3$, y vemos el vector de accesos correspondientes al usuario 7 durante su 4ª sesión. Esta información estará en las componentes $(7, :, 4)$ del tensor T_1. Si tenemos que, por ejemplo, $T_1(7, :, 4) = [0,42,\ 0,29,\ 0,29]$, lo que quiere decir esto es que el usuario 7 durante su 4ª sesión en el portal, realiza el 42 % de los accesos al descriptor D_1, el 29 % a D_2, y el restante 29 % a D_3.

Una vez finalizado el proceso de determinación de sesiones, el último paso consiste en obtener un tensor que proporcione información sobre los servicios accedidos. Para ello tendremos en cuenta el parámetro α que describe la relación entre el número de usuarios y el de servicios accedidos, así como la relación existente entre descriptores y servicios, para extraer un número de servicios correspondiente a cada descriptor, tal y como indique T_1. El resultado será un tensor T_2, que tendrá dimensiones $N \times L_{max} \times N_{S_{max}}$ siendo L_{max} la longitud o profundidad máxima de una sesión, es decir, el máximo número de servicios que pueden ser accedidos. Si, análogamente al ejemplo anterior, quisiéramos saber los servicios accedidos por el usuario 7 durante su 4ª sesión, el resultado sería un vector del tipo $T_2(7, :, 4) = [43, 27, 2, 6, 22, 19, 5, 0, \ldots, 0]$, siendo la longitud de este vector igual a L_{max}. La información de este vector sería que el usuario 7 en su 4ª sesión accede en primer lugar al servicio 43 y, a continuación, al 27, 2, 6, 22, 19 y 5; por tanto, el último servicio accedido sería el servicio 5, y en él acabaría la navegación por el portal para este usuario en esta sesión. El vector se completa con ceros para poder almacenar de manera eficiente en el tensor aquellos usuarios con secuencias más o menos largas dentro del portal.

Resumiendo, podemos decir que el modelo de usuario que implementa el simulador tiene cuatro posibles modos de funcionamiento, dependiendo del camino que se escoja dentro del diagrama de bloques que aparece en la Figura 4.3:

1. *Desarrollo absolutamente controlado.* En este caso todos los parámetros del modelo son escogidos en el propio fichero de datos: número de servicios; número de descriptores, sus probabilidades *a priori* y los servicios que forman cada uno de los descriptores; máximo número de sesiones que pueden ser abiertas por un mismo usuario; profundidad de la sesión, es decir, máximo número de servicios que pueden ser solicitados dentro de una misma sesión; número de grupos de usuario, usuarios en cada uno de ellos y matrices con la estadística de los grupos; y finalmente, los parámetros que controlan la dependencia del número de usuarios con el de sesiones y servicios.

2. *Desarrollo pseudo-aleatorio "Nivel 1".* Estamos en este caso cuando los servicios que forman cada descriptor son asignados pseudo-aleatoriamente. Tanto en este caso como en el anterior, se permite que un servicio pueda pertenecer a más de un descriptor.

3. *Desarrollo pseudo-aleatorio "Nivel 2".* En este caso, no se dispone de las matrices de centroides y covarianzas de los grupos de usuarios. La generación se hace escogiendo los centros de los grupos a partir de las probabilidades *a priori* de los descriptores, y generando una distribución normal pseudo-aleatoria en torno a ellos.

4. *Desarollo absolutamente pseudo-aleatorio.* Este tipo de desarrollo se lleva cabo cuando las dos circunstancias comentadas en los dos anteriores puntos se cumplen simultáneamente.

4.3. Descripción de los conjuntos de datos sintéticos

El modelo de usuario descrito en el punto anterior, permite obtener multitud de conjuntos sintéticos con diferentes características. Evidentemente, ha de seleccionarse una colección representativa de estos conjuntos para poder realizar una comparación eficiente entre el rendimiento de los diferentes algoritmos de agrupamiento. Se seleccionarán conjuntos con características diferentes viendo qué algoritmo funciona mejor con unas condiciones u otras, en función de la dimensionalidad del espacio donde se agrupa, del número de grupos que existen y del mayor o menor grado de solapamiento que haya entre ellos. Además de representar diferentes condiciones para los algoritmos

de agrupamiento, los conjuntos generados deben ser paradigmas de situaciones que puedan darse en portales reales para que, cuando se desee realizar la labor de agrupamiento con datos reales, se disponga de un banco de conjuntos artificiales "realistas" con los que comparar. Por tanto, además de utilizarse las restricciones anteriormente comentadas durante el desarrollo del modelo de usuario, los conjuntos presentan características similares a las que se han observado en conjuntos reales de datos, y en otros conjuntos de datos sintéticos previamente utilizados en la literatura (Balaguer y Palomares, 2003; Banerjee y Ghosh, 2002; Ghosh, Strehl y Meregu, 2002).

Se han seleccionado seis conjuntos de datos diferentes en el espacio de descriptores para comparar el rendimiento obtenido con los diferentes algoritmos de agrupamiento. No obstante, cada uno de estos conjuntos en el espacio de descriptores representa, a su vez, diferentes conjuntos en el espacio de servicios, dependiendo del número de servicios, de su relación con los descriptores considerados, de la profundidad de la sesión, de los parámetros que describen las restricciones del simulador, etc. A continuación, se pasará a describir los conjuntos utilizados.

4.3.1. Conjunto nº 1

Este conjunto, contrariamente al resto, no está inspirado en las características de portales reales, sino que se trata de un conjunto de datos extremadamente sencillo. Sirve únicamente como referencia para comprobar el funcionamiento de algoritmos de *clustering* en una situación muy sencilla. Las características básicas de este conjunto son las siguientes:

- *Dos descriptores.* Es decir, que el *clustering* se lleva a cabo en un espacio bidimensional.
- *Dos grupos de usuarios.* Solamente se consideran dos grupos que deben encontrar los algoritmos; uno de los grupos está formado por 72 patrones (usuarios) y el otro por 28.

Las matrices[4] que describen los centroides y desviaciones estándar para estos conjuntos de datos se muestran en (4.6) y (4.7), donde cada fila representa un grupo diferente.

[4]Todos los valores se corresponden con frecuencias de acceso en el espacio de descriptores.

$$C_1 = \begin{pmatrix} 0,25 & 0,25 \\ 0,75 & 0,75 \end{pmatrix} \tag{4.6}$$

$$\sigma_1 = \begin{pmatrix} 0,1 & 0,1 \\ 0,1 & 0,1 \end{pmatrix} \tag{4.7}$$

Se trata de dos grupos esféricos (misma varianza en cada dirección), y que además no presentan solapamiento entre ellos, como se observa en la Figura 4.4, por lo que se trata de un conjunto fácilmente resoluble para los algoritmos de agrupamiento.

Este conjunto puede representar diferentes situaciones cuando se tiene en cuenta la información de servicios además de la de descriptores. En particular, los diferentes subconjuntos que fueron considerados son los que se muestran en la Tabla 4.1.

Tabla 4.1: Características de los diferentes situaciones que aparecen para el conjunto de datos sintéticos nº 1 cuando se tiene en cuenta la información de servicios, siendo N_{ser} el número de servicios del portal, $N_{S_{max}}$ el máximo número de sesiones que pueden abrise, L_{max} la longitud máxima de una sesión, y α y β las constantes que controlan la dependencia entre el número de usuarios y el de servicios y sesiones, respectivamente.

	N_{ser}	$N_{S_{max}}$	L_{max}	α	β
Subconjunto 1.1	8	2	8	0,15	0,20
Subconjunto 1.2	100	2	8	0,15	0,20
Subconjunto 1.3	100	2	40	0,15	0,20
Subconjunto 1.4	100	2	40	0,10	0,10

4.3.2. Conjunto de datos nº 2

Este conjunto de datos es mucho más complicado que el anterior y refleja una situación más realista, correspondiente a un portal real pequeño en cuanto a descriptores pero con una importante complejidad para encontrar grupos, ya que como veremos en la Sección 4.4, los algoritmos de *clustering* encuentran grandes dificultades para agrupar correctamente los patrones de

este conjunto de datos. Las características básicas de este conjunto son las siguientes:

- *Tres descriptores.* El *clustering* se lleva a cabo en un espacio de dimensionalidad tres.
- *Cuatro grupos de usuarios.* Se consideran cuatro grupos formados respectivamente por 10, 20, 30 y 40 patrones (usuarios).

Las matrices que describen los centroides y desviaciones estándar para estos conjuntos de datos se muestran en (4.8) y (4.9).

$$C_2 = \begin{pmatrix} 0,1 & 0,25 & 0,5 \\ 0,75 & 0,75 & 0,1 \\ 0,3 & 0,5 & 0,3 \\ 0,5 & 0,1 & 0,9 \end{pmatrix} \tag{4.8}$$

$$\sigma_2 = \begin{pmatrix} 0,1 & 0,08 & 0,04 \\ 0,12 & 0,08 & 0,08 \\ 0,15 & 0,15 & 0,15 \\ 0,2 & 0,1 & 0,05 \end{pmatrix} \tag{4.9}$$

La matriz de desviaciones estándar mostrada en (4.9) hace que cada uno de los grupos presente una forma elipsoidal distinta; además, como se observa en la Figura 4.4, los grupos están muy cercanos unos a otros, lo que dificulta el agrupamiento. De hecho, resulta complicado distinguir en la proyección mostrada en la Figura 4.4 dónde acaba un *cluster* y dónde empieza el siguiente. En otra proyección, como la mostrada en la Figura 4.5, se resaltan los diferentes grupos, que como se puede ver son bastante diferentes unos de otros (en tamaño y forma).

Se han considerado diferentes situaciones dentro de este conjunto teniendo en cuenta la información de servicios que ha dado lugar a los subconjuntos que se muestran en la Tabla 4.2.

4.3.3. Conjuntos de datos nº 3 y nº 4

Estos dos conjuntos de datos están formados por ocho grupos de usuarios en un espacio de cinco descriptores. Cada uno de los grupos está formado

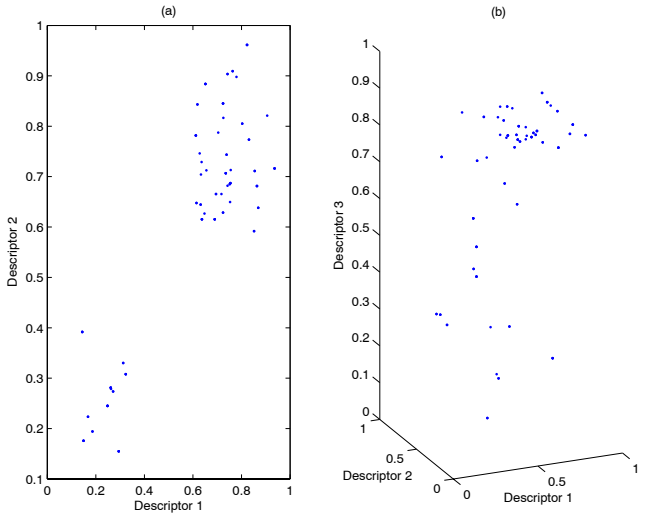


Figura 4.4: (a) Conjunto de datos nº 1: *clusters* esféricos entre los cuales no existe solapamiento; (b) Conjunto de datos nº 2: *clusters* elipsoidales muy cercanos unos a otros.

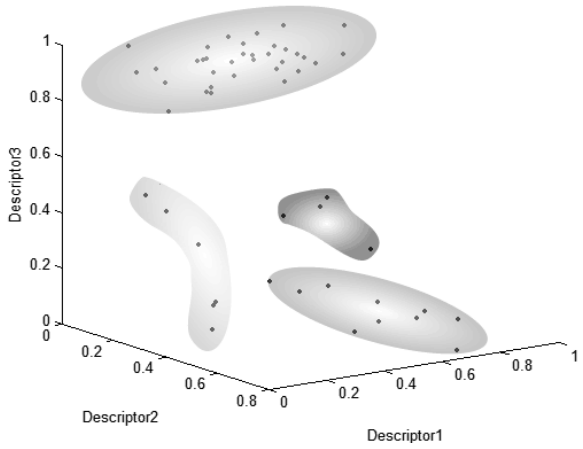


Figura 4.5: Conjunto de datos nº 2, resaltándose los diferentes grupos del conjunto.

por un número diferente de patrones (desde 10 hasta 80). Las matrices de centroides y desviaciones estándar fueron generadas pseudo-aleatoriamente con el simulador. La diferencia entre un conjunto de datos y el otro estriba en lo dispersos que son los agrupamientos, o lo que es lo mismo, en lo cerca que están unos grupos de otros (cuanto más cerca están los grupos, mayor es el grado de solapamiento entre ellos). En particular, el conjunto nº 3, cuyas

Tabla 4.2: Características de los diferentes situaciones que aparecen para el conjunto de datos sintéticos nº 2 cuando se tiene en cuenta la información de servicios.

	N_{ser}	$N_{S_{max}}$	L_{max}	α	β
Subconjunto 2.1	8	5	8	0,15	0,20
Subconjunto 2.2	8	5	8	0,30	0,30
Subconjunto 2.3	100	5	100	0,30	0,30
Subconjunto 2.4	100	5	40	0,30	0,30
Subconjunto 2.5	100	5	40	0,10	0,30
Subconjunto 2.6	1000	5	8	0,30	0,30
Subconjunto 2.7	1000	5	40	0,08	0,30

matrices de centroides y desviaciones estándar se muestran en (4.10) y (4.11) resultará más facil para los algoritmos de *clustering* ya que presenta un menor grado de solapamiento, resultando el valor de la desviación estándar igual a 0,05 en promedio (siendo los valores mínimo y máximo 0,0022 y 0,1118).

$$C_3 = \begin{pmatrix} 0,6261 & 0,6241 & 0,1093 & 0,5059 & 0,7112 \\ 0,8410 & 0,4090 & 0,3643 & 0,6500 & 0,8084 \\ 0,3452 & 0,6312 & 0,3098 & 0,7035 & 0,2811 \\ 0,5583 & 0,5536 & 0,8291 & 0,5596 & 0,2461 \\ 0,5761 & 0,2902 & 0,7392 & 0,2312 & 0,4736 \\ 0,0558 & 0,7629 & 0,2461 & 0,1403 & 0,1564 \\ 0,8877 & 0,1306 & 0,0629 & 0,5715 & 0,1970 \\ 0,8175 & 0,2831 & 0,0315 & 0,8218 & 0,2019 \end{pmatrix} \tag{4.10}$$

$$\sigma_3 = \begin{pmatrix} 0,0722 & 0,0811 & 0,0876 & 0,0432 & 0,0498 \\ 0,0306 & 0,0313 & 0,0377 & 0,0340 & 0,0608 \\ 0,0662 & 0,0046 & 0,0032 & 0,0277 & 0,0271 \\ 0,0331 & 0,0404 & 0,0146 & 0,0501 & 0,0456 \\ 0,0073 & 0,0231 & 0,0041 & 0,0630 & 0,0086 \\ 0,0124 & 0,0703 & 0,0383 & 0,0022 & 0,0168 \\ 0,0038 & 0,0187 & 0,1118 & 0,0157 & 0,0271 \\ 0,0869 & 0,0235 & 0,0163 & 0,0113 & 0,0466 \end{pmatrix} \tag{4.11}$$

El conjunto de datos nº 4, cuyas matrices de centroides y desviaciones estándar se muestran en (4.12) and (4.13) resultará más complicado para los algoritmos de *clustering* ya que la desviación estándar es mayor, por lo que los grupos estarán más dispersos y existirá un mayor solapamiento. La desviación estándar presenta un valor promedio de 0,1, resultando los valores mínimo y máximo iguales a 0,0031 y 0,2370.

$$C_4 = \begin{pmatrix} 0,6762 & 0,1969 & 0,8356 & 0,4200 & 0,2204 \\ 0,5819 & 0,5449 & 0,1701 & 0,5984 & 0,0312 \\ 0,8943 & 0,2837 & 0,7291 & 0,1310 & 0,3447 \\ 0,1825 & 0,1129 & 0,5461 & 0,4121 & 0,6614 \\ 0,2258 & 0,4660 & 0,7467 & 0,0782 & 0,5065 \\ 0,7494 & 0,7868 & 0,1235 & 0,5117 & 0,3645 \\ 0,1026 & 0,5585 & 0,5783 & 0,4987 & 0,1840 \\ 0,5107 & 0,0747 & 0,1241 & 0,2346 & 0,4530 \end{pmatrix} \tag{4.12}$$

$$\sigma_4 = \begin{pmatrix} 0,0308 & 0,0057 & 0,0316 & 0,0966 & 0,0541 \\ 0,0486 & 0,0423 & 0,1985 & 0,0040 & 0,0067 \\ 0,0331 & 0,0075 & 0,0645 & 0,2122 & 0,2256 \\ 0,0777 & 0,0787 & 0,0313 & 0,1836 & 0,0212 \\ 0,0333 & 0,0137 & 0,1592 & 0,1542 & 0,0188 \\ 0,2096 & 0,0636 & 0,0973 & 0,0235 & 0,0285 \\ 0,0389 & 0,1578 & 0,1148 & 0,1199 & 0,1530 \\ 0,0653 & 0,0031 & 0,1484 & 0,0033 & 0,2370 \end{pmatrix} \tag{4.13}$$

El mayor solapamiento para el conjunto de datos nº 4 se muestra con claridad en la Figura 4.6, donde se representan gráficamente las proyecciones tridimensionales sobre los descriptores 1, 3 y 5 correspondientes a ambos conjuntos de datos. Algunos grupos son distinguibles en el caso del conjunto nº 3, aunque no desde luego los ocho grupos que forman el conjunto de datos, mientras que para el conjunto nº 4, es muy difícil encontrar fronteras de separación entre los diferentes *clusters*.

Respecto a los subconjuntos que aparecen cuando se tiene en cuenta la información de servicios, éstos aparecen definidos en la Tabla 4.3.

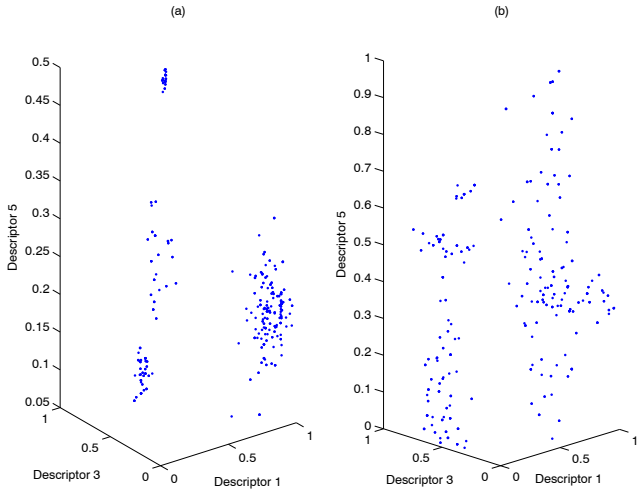


Figura 4.6: Proyecciones tridimensionales basadas en los descriptores 1, 3 y 5 para (a) el conjunto de datos nº 3 y (b) nº 4.

Tabla 4.3: Características de las diferentes situaciones que aparecen para los conjuntos de datos sintéticos 3 y 4 cuando se tiene en cuenta la información de servicios. Como se consideró la misma casuística para ambos conjuntos, 'x' representa tanto al conjunto nº 3 como al 4.

	N_{ser}	$N_{S_{max}}$	L_{max}	α	β
Subconjunto x.1	20	10	8	0,20	0,20
Subconjunto x.2	100	10	8	0,20	0,20
Subconjunto x.3	100	10	40	0,10	0,20
Subconjunto x.4	1000	10	8	0,20	0,20
Subconjunto x.5	1000	10	40	0,10	0,20

4.3.4. Conjuntos de datos nº 5 y nº 6

Análogamente al caso anterior, se presentan dos conjuntos con similares características (ocho descriptores y doce grupos de usuarios, cada uno de los cuales formado por un número diferente de patrones, desde 10 hasta 120) pero que presentan diferentes características en lo que hace referencia al solapamiento entre grupos. De hecho, el conjunto de datos nº 5 presenta una desviación estándar igual a 0,05 en promedio (los valores mínimo y máximo son 0,0006 y 0,1142), mientras que el nº 6 tiene una desviación estándar 0,09 (siendo los valores extremos 0,0001 y 0,2048). Por tanto, ha de resultar más

complicado determinar los grupos en el conjunto de datos nº 6 que en el nº 5. Al tratarse de espacios de dimensionalidad relativamente alta, la inspección visual mediante proyecciones tridimensionales puede resultar engañosa, como lo indica la Figura 4.7, donde se muestran dos diferentes proyecciones para el conjunto nº 5. Una de ellas permite distinguir algunos grupos visualmente, mientras que en la otra los *clusters* aparecen considerablemente apiñados.

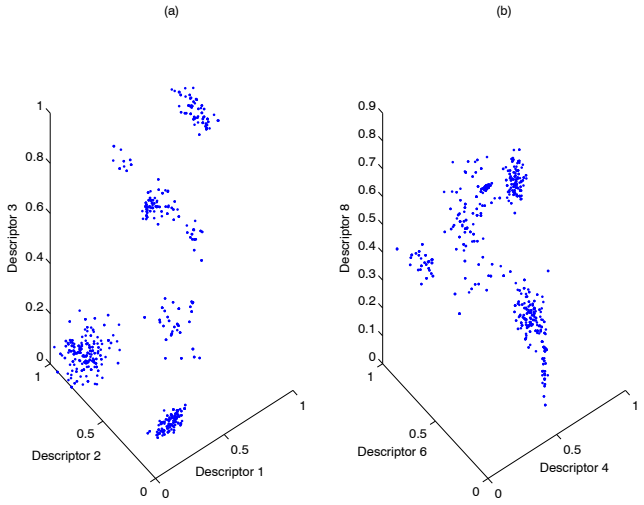


Figura 4.7: Dos diferentes proyecciones para el conjunto de datos nº 5. En (a), puede observarse que usando las frecuencias de acceso a los descriptores 1, 2 y 3, algunos grupos pueden distinguirse; en (b), por otro lado, se observa que una proyección sobre las frecuencias de los descriptores 4, 6 y 8, da lugar a un solapamiento visual bastante importante.

Para acabar con la descripción de estos dos últimos conjuntos de datos, en la Tabla 4.4 se muestran las diferentes situaciones consideradas teniendo en cuenta la información de servicios.

4.4. Funcionamiento de los algoritmos de agrupamiento con los conjuntos de datos sintéticos

4.4.1. Introducción

El objetivo de la presente tesis es intentar modelar a los usuarios de un portal web, de manera que se pueda implementar un sistema de recomendaciones

Tabla 4.4: Características de las diferentes situaciones que aparecen para los conjuntos de datos sintéticos 5 y 6 cuando se tiene en cuenta la información de servicios. Como se consideró la misma casuística para ambos conjuntos, 'x' representa tanto al conjunto nº 5 como al nº 6.

	N_{ser}	$N_{S_{max}}$	L_{max}	α	β
Subconjunto x.1	30	12	25	0,10	0,15
Subconjunto x.2	100	12	25	0,10	0,15
Subconjunto x.3	100	12	50	0,06	0,15
Subconjunto x.4	1000	12	25	0,10	0,15
Subconjunto x.5	1000	12	50	0,06	0,20

que resulte útil y eficaz, es decir, que aumente la proporción de las recomendaciones aceptadas por el usuario. Nuestra metodología se basa en realizar un modelo de usuario que sea capaz de generar datos sintéticos con los que comparar el rendimiento ofrecido por los diferentes algoritmos de *clustering* considerados. Una vez estudiado el comportamiento de los diferentes algoritmos, cuando se disponga de un conjunto real de datos, podrá decidirse el algoritmo que resulta más adecuado. En este apartado, nos centraremos en el rendimiento que ofrecen los diferentes algoritmos presentados en el Capítulo 2 con los conjuntos de datos sintéticos descritos en el apartado 4.3.

4.4.2. Medidas de evaluación usadas

Para evaluar el rendimiento de los algoritmos de *clustering*, hemos usado tres medidas que ofrecen tres tipos de información diferente sobre el *clustering* alcanzado. En primer lugar, se analizará si el número de grupos que encuentra el algoritmo se corresponde, o no, con el número real de grupos que se ha generado con el modelo de usuario. En segundo lugar, se estudiará la bondad con la cual los grupos encontrados se ajustan a los grupos reales. Por último, se verá en qué medida los grupos encontrados siguen una distribución normal; esta última medida es importante ya que como los conjuntos de datos sintéticos han sido generados siguiendo una distribución normal, se podrá conocer si los algoritmos de agrupamiento son capaces de determinar la distribución estadística subyacente en los datos.

Evaluación del número de grupos

El primer paso para comprobar el funcionamiento de los algoritmos de *clustering* es comprobar en qué medida el número de grupos encontrado se corresponde con el verdadero número de grupos, que evidentemente es conocido, ya que ha sido generado por el modelo de usuario, tal y como se ha venido explicando en el desarrollo de este capítulo.

Al analizar la diferencia existente entre el número de prototipos encontrados por los algoritmos y el valor deseado, se tendrá en cuenta el valor absoluto de esta diferencia, es decir, un número incorrecto de *clusters* se considerará igualmente un error tanto si es por exceso como si lo es por defecto. Consideramos que un grupo ha sido correctamente encontrado cuando la distancia[5] de Mahalanobis entre el grupo encontrado por el algoritmo y el centro (prototipo de grupo real) más cercano se sitúa por debajo de la unidad[6]. Como ya se explicó en el Capítulo 2, la ventaja de utilizar esta distancia en lugar de la euclídea es que presenta una dependencia explícita en la covarianza del *cluster*, lo que la hace robusta frente a la forma que éste presente. Una distancia de Mahalanobis igual a la unidad es un umbral exigente, ya que suponiendo una distribución normal, solamente el 68 % de los datos se encuentran dentro del umbral de una desviación estándar que es justamente lo que viene a expresar esta distancia; por tanto, con este umbral se consideran puntos de la distribución que se encuentran relativamente cerca del centroide de la misma. Teóricamente, para medir distancias entre dos distribuciones la distancia ideal es la de Bhattacharyya; no obstante, se ha utilizado la de Mahalanobis por dos razones:

- Es más intuitiva su interpretación en el sentido de que puede entenderse su valor como las desviaciones estándar que separan el punto que se está considerando del centroide de la distribución.
- Es de esperar que si los algoritmos de agrupamiento funcionan correctamente, el grupo encontrado y el real sean parecidos, y por tanto, las matrices de covarianzas de las distribuciones de ambos grupos sean

[5] La distancia se mide en el espacio definido por las frecuencias de acceso a los descriptores.

[6] Cuando el número de patrones que forman un grupo es muy pequeño, debe usarse la distancia euclídea en lugar de la de Mahalanobis, estableciéndose el umbral de esta distancia para considerar que el grupo ha sido correctamente encontrado en 0,1.

similares, por lo que en este caso las distancias de Bhattacharyya y Mahalanobis proporcionan una información similar.

Finalmente, la Tasa de Éxito (TE [%]) que mide el porcentaje de grupos corretamente encontrados, utilizada para comparar el rendimiento de los diferentes algoritmos viene dada por (4.14):

$$TE[\,\%] = 100 \cdot \left(\frac{N_r - |N_r - N_e|}{N_r} \right) \tag{4.14}$$

donde N_r es el número real de grupos del conjunto y N_e el número de grupos correctamente encontrados por el algoritmo.

Evaluación de la bondad del agrupamiento

Una vez determinado el porcentaje de grupos que hemos considerado que se han encontrado correctamente, evaluamos hasta qué punto los grupos encontrados se corresponden con los reales. Para ello, se utiliza la distancia de Mahalanobis, que se ha calculado anteriormente para decidir si los grupos habían sido encontrados correctamente. Obviamente, cuanto menor sea la distancia entre los grupos encontrados y los reales, mayor será la calidad del *clustering* alcanzado. La medida que utilizaremos para comparar la bondad del agrupamiento que ofrecen los diferentes algoritmos viene dada por (4.15), y se basa en medir la distancia de Mahalanobis, únicamente para los *clusters* correctamente encontrados, entre el *cluster* encontrado y el centro correspondiente al grupo real más cercano. Además, cada distancia irá pesada por el número de patrones que forman cada uno de los grupos correctamente encontrados, de manera que en el valor final que se utiliza para la comparación, tendrán más peso aquellos grupos que poseen un mayor número de patrones.

$$D = \frac{1}{N_c} \sum_{i=1}^{N_e} N_i d_i \tag{4.15}$$

En (4.15), D ofrece información sobre la distancia entre los *clusters* encontrados y los centros reales que corresponden a ellos. Cuanto menor sea el valor de D, mayor será la correspondencia entre los grupos encontrados y los reales. N_c es el número total de patrones (considerando solamente grupos correctamente encontrados), N_i el número de patrones que pertenecen al

i-ésimo *cluster* encontrado, y d_i la distancia de Mahalanobis entre el i-ésimo *cluster* y el centro del correspondiente grupo real más cercano.

Prueba de normalidad

Teniendo en cuenta que los grupos se han generado siguiendo una distribución normal, se realizará una prueba que dé idea de cuán normales (no normales, realmente) son los grupos encontrados. Por tanto, se medirá la capacidad de los algoritmos para captar la distribución estadística subyacente en los datos. Aunque existen bastantes pruebas de normalidad, muchas de ellas no son aplicables al presente problema por diferentes motivos, y otras que sí lo son, como la prueba de Shapiro-Wilks, presenta algunos inconvenientes como que es necesario disponer de una distribución normal con la que comparar los datos que se desea analizar. Por todo ello, se ha decidido realizar un análisis de la normalidad de los grupos encontrados basado en los momentos de orden superior a dos, en particular, el momento de orden tres (asimetría o *skewness*) y el de orden cuatro (curtosis).

La asimetría as, como el propio nombre indica, es una medida de simetría, o más exactamente, de la falta de simetría (Hair, Anderson, L. y Black, 1999). La asimetría del *cluster* i-ésimo vendría dada por:

$$as_i = \frac{\sum_{i=1}^{N_i}(x_i - \overline{X_i})^3}{(N_i - 1)\sigma_i^3} \tag{4.16}$$

donde $\overline{X_i}$ es la media (prototipo) del *cluster*, σ_i la desviación estándar, y N_i el número de patrones del cluster en cuestión. El valor de asimetría para una distribución normal es igual a cero, y cualquier conjunto de datos simétrico presenta un valor cercano a cero. Valores negativos indican que los datos están sesgados hacia la izquierda mientras que valores positivos indican datos sesgados hacia la derecha.

La curtosis ct es una medida de si la distribución de datos es más bien picuda o más bien plana respecto a una distribución normal. Aquellos conjuntos de datos que presentan un alto valor de curtosis tienden a tener un pico cerca del promedio de la distribución, y a partir de él descienden rápidamente. Por contra, aquellos conjuntos de datos que presentan un valor bajo de curtosis, tienen un pico más bien plano que agudo, llegándose a una distribución uniforme en el caso más extremo (Hair et al., 1999). La curtosis del *cluster*

i-ésimo vendría dada por:

$$ct_i = \frac{\sum_{i=1}^{N_i}(x_i - \overline{X_i})^4}{(N_i - 1)\sigma_i^4} \tag{4.17}$$

El valor de curtosis para una distribución normal es igual a tres. Por esta razón, se define lo que se conoce como curtosis en exceso como:

$$ct_{i_{exc}} = \frac{\sum_{i=1}^{N_i}(x_i - \overline{X_i})^4}{(N_i - 1)\sigma_i^4} - 3 \tag{4.18}$$

de manera que una distribución normal presentará un valor de curtosis en exceso nulo, valores positivos para la curtosis indicarán distribuciones picudas y valores negativos, distribuciones planas.

Para calcular la posible no normalidad de la distribución, pueden realizarse pruebas basadas en la asimetría y la curtosis. En particular, el parámetro estadístico (z) para la asimetría y la curtosis del *cluster* i-ésimo se define como:

$$z_{as_i} = \frac{as_i}{\sqrt{\frac{6}{N_i}}} \tag{4.19}$$

$$z_{ct_i} = \frac{ct_{i_{exc}}}{\sqrt{\frac{24}{N_i}}} \tag{4.20}$$

Si el valor de z (ya sea de asimetría o de curtosis) excede un determinado valor crítico, entonces la distribución es no normal en cuanto a la característica o grupo que esté estudiándose (Hair et al., 1999). El valor crítico está basado en el nivel de confianza que se desee para afirmar que la distribución es no normal. Por ejemplo, una cifra mayor en valor absoluto que $\pm 2,58$ indica que podemos rechazar la asunción sobre la normalidad de la distribución con una probabilidad de error de $0,01$ o, lo que es lo mismo, con una confianza del $99\,\%$. Otro valor crítico típicamente usado es el de $\pm 1,96$, que corresponde a una confianza del $95\,\%$.

El valor z de asimetría y curtosis se analizó para las distribuciones encontradas por los algoritmos de agrupamiento con los conjuntos de datos artificiales. En particular, utilizamos el valor crítico de $\pm 1,96$, considerando la distribución como no normal cuando se obtenía un valor mayor de $1,96$ en valor absoluto.

4.4.3. Ajuste de los modelos y comparativa de algoritmos

Antes de pasar a mostrar la comparativa entre los mejores resultados para cada algoritmo, se describirá brevemente el proceso seguido para la selección de los modelos, especificando los parámetros correspondientes al mejor ajuste para cada grupo (si procede).

C-medias

Para el caso del algoritmo de las C-medias (CM), los modelos fueron desarrollados eligiendo C igual al número de grupos que debía encontrarse, y realizando diversas inicializaciones de modo aleatorio. La distancia euclídea fue usada para decidir el centro que se encontraba más cerca de cada patrón, y se consideró un número de iteraciones t para el algoritmo, tal que el cambio entre las posiciones de los centros de los grupos entre las iteraciones $t-1$ y t se encontrara por debajo de un umbral 10^{-10} (distancia euclídea entre los centros de los grupos, medida en el espacio definido por las frecuencias de acceso a los descriptores).

C-medias difuso

En cuanto al algoritmo de las C-medias difuso (FCM), el desarrollo fue bastante similar en lo que hace referencia a la elección de C, a la inicialización aleatoria de los centros, y al número de iteraciones respetando el mismo umbral para la convergencia que en el caso anterior. La diferencia reside en el parámetro que controla la borrosidad del agrupamiento, m, cuyo valor fue típicamente variado en el intervalo $[1,\ 2]$, aunque los mejores resultados se han encontrado en todos los conjuntos de datos sintéticos para $m = 1,25$.

Expectation-Maximization

Respecto al algoritmo *Expectation-Maximization* (E-M), los centros de los grupos fueron inicializados mediante el agrupamiento realizado por el CM, con lo que podríamos ver el efecto de este algoritmo como un ajuste fino sobre el primer ajuste llevado a cabo por las CM.

Algoritmos jerárquicos

En lo que hace referencia a los algoritmos de *clustering* jerárquicos (ACJ), se han utilizado las diferentes versiones de *clustering* jerárquico acumulativo explicadas en el capítulo 2, cortando las iteraciones en el nivel de jerarquía correspondiente al número de grupos que se desea encontrar. No se han contemplado las versiones divisivas de estos algoritmos ya que su mayor carga computacional no se ve reflejada en una mejora del agrupamiento realizado, según queda reflejado en la literatura (Theodoridis y Koutroumbas, 1999). Dependiendo del conjunto de datos, la variante que mejor comportamiento ofrecía era distinta, como se muestra en la Tabla 4.5.

Tabla 4.5: Variante de *clustering* jerárquico acumulativo que mejor ha funcionado para cada uno de los conjuntos de datos considerados. 'Indiferente' indica que todos los algoritmos han presentado el mismo rendimiento.

	Mejor método
Conjunto nº 1	Indiferente
Conjunto nº 2	Centroide no pesado
Conjunto nº 3	Centroide no pesado, enlace sencillo y enlace completo
Conjunto nº 4	Centroide no pesado
Conjunto nº 5	Promedio no pesado, centroide no pesado, actualización centroides
Conjunto nº 6	Promedio pesado

Mapas autoorganizativos

Los valores óptimos de los parámetros para la primera parte del proceso llevado a cabo con el SOM, que consiste en el mapeo del espacio de representación en el espacio de salida, se muestran en la Tabla 4.6. Los mapas utilizados han sido unidimensionales y bidimensionales, y la función de vecindad que mejor ha funcionado ha sido la rectangular. Se ha escogido un total de 20 grupos de neuronas representativas al final de esta primera parte del proceso (mapeado y unión de neuronas cercanas), ya que el número de *clusters* máximo para los conjuntos considerados era de 12. Además, los patrones han sido presentados varias veces, y en distinto orden, para asegurar la robustez del mapa obtenido. Para la segunda parte, donde se extraen los grupos mediante un algoritmo jerárquico de enlace completo, se ha cortado en aquel nivel de jerarquía que se corresponde con el número de grupos que

se desea obtener.

Tabla 4.6: Parámetros óptimos del SOM para cada uno de los conjuntos de datos considerados. La constante de adaptación inicial se denota por α_{in}, la constante de adaptación final por α_{fin} y el número de neuronas que constituyen el radio de vecindad inicial por R_{in}.

	Nº neuronas	α_{in}	α_{fin}	R_{in}
Conjunto nº 1	30	0,5	0,1	8
Conjunto nº 2	60	0,7	0,7	10
Conjunto nº 3	250	0,2	0,16	25
Conjunto nº 4	285	0,85	0,8	16
Conjunto nº 5	300	0,08	0,04	20
Conjunto nº 6	300	0,6	0,6	40

Teoría de la resonancia adaptativa (ART)

Cabe decir que para todos los algoritmos anteriores los resultados han de considerarse algo sobreoptimistas, ya que se introduce como un parámetro de los algoritmos el número de grupos que deben encontrar. En un caso real esta información será evidentemente desconocida, por lo que se deberá hacer uso de determinados índices que permitan evaluar el número óptimo de grupos, aunque su funcionamiento no sea óptimo en todos los casos. La red ART2, por contra, no presenta este problema ya que no se le introduce el número de grupos que debe encontrar como un parámetro de entrada, sino que en función del parecido que se establezca para determinar si un patrón pertenece a un grupo o no, el número final de *clusters* surgirá como algo natural. En particular, para la aplicación de la ART2 a los conjuntos de datos sintéticos, el parámetro de vigilancia se varió entre 0,9 y 0,99, mientras que la activación de la unidad vencedora en la capa $F2$ y la constante que rige el ritmo del aprendizaje se varió entre 0,9 y 1. Los valores óptimos para los diferentes conjuntos se muestran en la Tabla 4.7. Además, el número de iteraciones fue elegido de manera que la red fuera estable, es decir, que no hubiera actualizaciones en las últimas iteraciones (en particular, se consideró que no hubiera actualizaciones durante las últimas 5 iteraciones).

Tabla 4.7: Parámetros óptimos de la red ART2 para cada uno de los conjuntos de datos considerados. La constante de aprendizaje se denota por α, la activación de la neurona ganadora en la capa $F2$ por β y el parámetro de vigilancia por ρ.

	α	β	ρ
Conjunto nº 1	$0,9$	$0,9$	$0,9$
Conjunto nº 2	$0,9$	$0,9$	$0,95$
Conjunto nº 3	1	1	$0,99$
Conjunto nº 4	1	1	$0,94$
Conjunto nº 5	1	1	$0,98$
Conjunto nº 6	1	1	$0,95$

Comparativa de TE

La Figura 4.8 compara el rendimiento de los diferentes algoritmos de *clustering* en relación al número de grupos correctamente encontrados en los diferentes conjuntos de datos artificiales. Como puede observarse, ninguno de los algoritmos tiene dificultad en encontrar dos grupos para el conjunto de datos nº 1. Fuera de este conjunto trivial, el comportamiento de los algoritmos es considerablemente diferente.

Respecto al algoritmo CM, los resultados obtenidos son bastante pobres, excepto para el conjunto de datos nº 1, que como hemos visto, es un conjunto trivial, que se utiliza como referencia que cualquier algoritmo ha de resolver.

El algoritmo FCM presenta un comportamiento algo mejor que CM, aunque aun así los resultados son bastante pobres, presentando porcentajes por debajo del 50 % en casi todos los conjuntos de datos con algo de complejidad.

El método basado en la optimización de mezcla de Gaussianas usando el algoritmo E-M muestra un porcentaje de TE considerablemente mejor. Su comportamiento es mejor o igual que el de los mejores algoritmos para los conjuntos de datos nº 2, 4 y 5. Es de destacar además que este algoritmo es bastante independiente de las características del conjunto de datos en cuestión, ya que muestra unos valores de TE bastante similares para los seis conjuntos de datos.

Curiosamente, ACJ y SOM obtienen exactamente los mismos valores de TE para todos los conjuntos de datos. Esta tasa de éxito es más que aceptable,

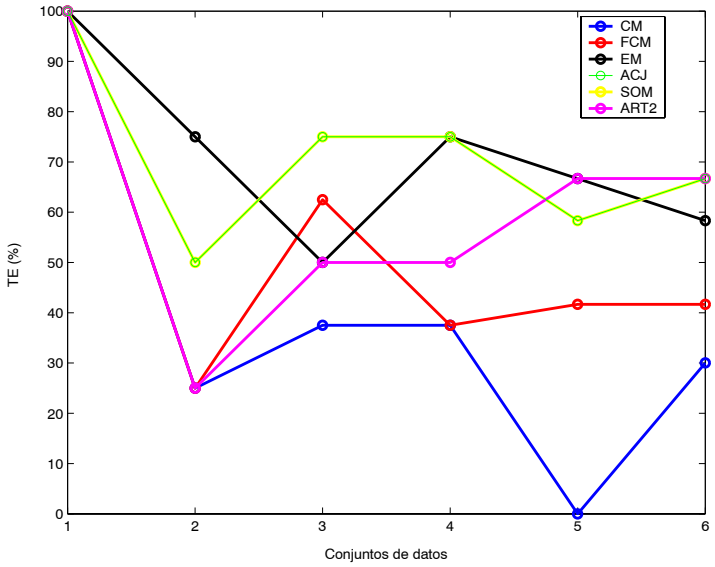


Figura 4.8: Porcentaje de *clusters* correctamente encontrados utilizando C-medias (CM), C-medias difuso (FCM), mezcla de Gaussianas ajustada por *Expectation-Maximization* (E-M), *clustering* jerárquico (ACJ), mapa autoorganizativo (SOM), y red basada en la teoría de la resonancia adaptativa (ART2).

similar para todos los conjuntos de datos, y desde un punto de vista global, la más alta de todos los algoritmos, si bien es bastante similar a la obtenida por E-M. El hecho de que SOM y ACJ presenten exactamente el mismo comportamiento parecería indicar que realizan un mismo agrupamiento de datos; no obstante, como se verá en los siguientes análisis esto no es así ya que al analizar otros aspectos del agrupamiento los resultados sí que son diferentes, lo que indica que estos dos algoritmos presentan el mismo comportamiento solamente en lo que corresponde al número de grupos que subyace en la distribución.

Por último, la red ART2 presenta un comportamiento no demasiado bueno para conjuntos de dimensionalidad baja o media, pero sin embargo, TE presenta valores extraordinariamente altos en conjuntos de dimensionalidad alta, mostrándose como el mejor algoritmo para los conjuntos de datos nº 5 y 6, que son los que presentan unas características más complicadas. Es muy importante destacar aquí que mientras el resto de algoritmos tienen la información del número de grupos que han de encontrar y aun así, no todos los encuentran correctamente, la ART2 no dispone de esta información por

lo que la comparación en el valor de TE ha de tener en cuenta este hecho. De hecho, puede encontrarse aquí la razón de por qué la ART2 presenta un comportamiento bajo en promedio comparado con los otros algoritmos en conjuntos de dimensionalidad reducida, donde saber el número de grupos a encontrar es fundamental para conseguir un valor alto de TE, y sin embargo, destaca como el que más para conjuntos de dimensionalidad alta, donde a pesar de saber el número de grupos que se han de encontrar, es más probable realizar una asignación de grupos incorrecta. Teniendo en cuenta esto, los resultados obtenidos por la ART2 han de entenderse como mejores de lo que la simple observación de TE pueda sugerir.

Comparativa de D

Una vez determinado el porcentaje de grupos correctamente encontrados por los diferentes algoritmos, el siguiente paso es determinar en qué medida estos grupos, que consideramos correctos, se corresponden con los reales. Para ello utilizamos la expresión (4.15) que nos dará un valor para la distancia entre los grupos reales y los correctamente encontrados por los algoritmos, normalizado por la cantidad de patrones que integran el grupo. El valor de esta distancia para los diferentes algoritmos en los conjuntos de datos artificiales se muestra en la Tabla 4.8. Evidentemente estos valores han de interpretarse conjuntamente a los valores de TE, puesto que de poco sirve que los grupos correctamente encontrados se correspondan feacientemente con las reales, si solamente se ha encontrado correctamente un grupo de seis, por ejemplo.

Tabla 4.8: Distancia de Mahalanobis normalizada (D) entre los centros correspondientes a los grupos reales y los *clusters* correctamente encontrados por los diferentes algoritmos para los conjuntos de datos artificiales. Las distancias están medidas en el espacio definido por las frecuencias de acceso a los descriptores.

	CM	**FCM**	**E-M**	**ACJ**	**SOM**	**ART2**
Cjto. nº 1	0,0330	0,0330	0,0330	0,1125	0,0165	0,0330
Cjto. nº 2	0,6818	0,0262	0,9097	0,0682	0,0819	0,2492
Cjto. nº 3	0,1498	0,2154	0,3091	0,1880	0,0797	0,2314
Cjto. nº 4	0,2227	0,2051	0,2861	0,2600	0,3101	0,2747
Cjto. nº 5	···	0,2134	0,5892	0,2016	0,0418	0,2411
Cjto. nº 6	0,2858	0,3583	0,2738	0,2615	0,2403	0,3143

Por tanto, aunque los valores de D que se obtienen con los algoritmos CM y FCM son bastante aceptables, no debemos interpretar que esto quiere decir que el *clustering* alcanzado sea especialmente correcto, ya que hay que recordar que el porcentaje de *clusters* correctamente encontrados era bastante bajo para estos algoritmos.

De entre los otros algoritmos, podemos extraer varias conclusiones. Por un lado, podemos decidir qué algoritmo es más idóneo entre SOM y ACJ, que ofrecían exactamente los mismos valores de TE. Globalmente, SOM presenta valores menores para D, lo que implica una mayor calidad en el *clustering*, aunque en algún conjunto de datos, muestran valores similares, o incluso más favorables para ACJ (conjunto nº 4).

Por otro lado, el algoritmo E-M, que presentaba unos resultados bastante buenos en cuanto a TE, muestra algunas carencias en el valor de D, siendo las distancias considerablemente grandes, excepto para el conjunto nº 4, donde junto al ACJ es el que muestra un mejor funcionamiento global.

Por último, los valores obtenidos por ART2 son satisfactorios, ya que para todos los conjuntos el valor de D es aproximadamente el mismo, siendo éste relativamente bajo (en torno a $0,2$). Por ello, el comportamiento global de este algoritmo puede entenderse como muy correcto, teniendo en cuenta la robustez de esta red que no tiene información acerca del número de grupos que debe obtener.

Normalidad del *clustering*

Una vez comparada la bondad del *clustering* encontrado por los diferentes algoritmos con los conjuntos de datos artificiales, se realiza una prueba de normalidad de los grupos que se consideran correctamente encontrados, utilizando los valores de z_{as} y z_{ct}. En particular, prestaremos atención al porcentaje de grupos que tienen componentes no normales con una probabilidad del 95 %, que corresponde a unos valores umbral para las z's de $\pm 1,96$. Debido a que los datos fueron generados siguiendo distribuciones Gaussianas, si los grupos hubieran sido correctamente encontrados, éstos deberían ofrecer valores bajos para ambas z's. Por lo tanto, esto constituye una prueba final para conocer si los algoritmos de *clustering* han sido capaces de captar la estadística subyacente en los datos.

Los resultados obtenidos por los diferentes algoritmos son bastante similares. El primer resultado que se obtiene a este respecto es que ninguno de los algoritmos utilizados produce grupos no normales (con la confianza explicitada anteriormente) en ninguno de los agrupamientos realizados a partir de los conjuntos nº 1 y nº 2.

A partir de aquí, ya empiezan a aparecer agrupaciones que resultan ser no normales. Con el conjunto de datos nº 3, no existen tampoco diferencias apreciables entre los algoritmos, de hecho, todos ellos presentan o uno o dos grupos con componentes no normales (sobre el total de ocho). La no normalidad de los conjuntos provenía tanto del análisis de asimetría como del de curtosis.

Mayores diferencias aparecen cuando se analiza el conjunto de datos nº 4 y, además, un mayor porcentaje de grupos no normales. Esto último es lógico, ya que el mayor solapamiento que existe en este conjunto hace que pueda haber una mayor confusión en el establecimiento de los grupos y, por tanto, que no se pueda captar en la misma medida la estadística de los *clusters*. Para este conjunto, el peor algoritmo en términos de normalidad es CM, ya que el 50 % de los grupos encontrados por este algoritmo son no normales en cuanto a que son asimétricos. La no normalidad por curtosis tiene menos incidencia, afectando solamente a uno de los grupos. Por contra, los algoritmos que presentan un menor porcentaje de grupos no normales son la ART2 y el SOM, ya que únicamente un grupo es no normal por asimetría y otros dos por curtosis. El resto de los algoritmos presentan un $37,5$ % de grupos asimétricos (tres grupos) y un $12,5$ % de grupos (otro grupo) con valores de curtosis correspondientes a distribuciones no normales.

Respecto al conjunto de datos nº 5, de nuevo casi todos los algoritmos presentan un comportamiento muy similar, en torno al 25 % de grupos asimétricos, mientras que no existe ningún grupo no normal analizando el valor de curtosis. El único comportamiento algo diferente lo presenta la ART2, que ofrece un $8,33$ % de grupos no normales por asimetría, y el mismo porcentaje de grupos no normales por curtosis.

Por último, en cuanto al conjunto nº 6, el mejor comportamiento viene dado por el E-M que no tiene ningún grupo no normal; esta situación no debe extrañarnos ya que este algoritmo intenta optimizar una mezcla de distribuciones Gaussianas. El segundo mejor comportamiento viene ofrecido

por el SOM, que no presenta ningún grupo no normal debido a curtosis y solamente un $8,33\,\%$ de grupos asimétricos. El resto de algoritmos tampoco presentan no normalidad debido a curtosis, pero el porcentaje de grupos no normales por asimetría es tres veces mayor ($25\,\%$).

Por tanto, desde un punto de vista general, puede considerarse que el porcentaje de grupos no normales es relativamente bajo, por lo que los algoritmos de *clustering* sí que parecen captar la estadística que subyace en los datos. Más específicamente, y aunque no existen diferencias demasiado grandes entre los algoritmos, SOM y ART2 son los dos algoritmos que presentan un mejor comportamiento en este sentido. Por contra, los mayores porcentajes de grupos no normales corresponden a los algoritmos CM, FCM y ACJ.

Comentarios finales

Hemos comparado el comportamiento de los diferentes algoritmos de *clustering* en los conjuntos de datos sintéticos generados por el modelo de usuario en cuanto a cantidad de grupos correctamente encontrados, calidad de estos grupos respecto a los que se debería encontrar y análisis de la normalidad de los grupos. Sobre todo, en lo que concierne a los dos primeros puntos, el análisis debe hacerse conjunto, ya que la calidad de los grupos afecta solamente a aquellos que se han encontrado correctamente.

Dejando aparte el conjunto de datos nº 1, los algoritmos ACJ, E-M y SOM presentan un comportamiento bastante adecuado, aunque puestos a elegir uno de ellos, la elección recaería sobre el SOM, porque globalmente es el que presenta un mejor comportamiento en el agrupamiento, tanto en calidad como en cantidad y en normalidad de los grupos, aunque se puede observar en la Tabla 4.8 que el solapamiento entre grupos afecta a su rendimiento ostensiblemente, como es lógico por otra parte. En conjuntos de mayor dimensionalidad (conjuntos nº 5 y 6), el mejor comportamiento global parece ser el de la ART, si bien SOM, y E-M en menor medida, presentan un resultado bastante aceptable.

Es de destacar que clásicos algoritmos de *clustering* ampliamente usados y conocidos, como CM y FCM, presentan unos resultados bastante pobres en comparación con los otros algoritmos utilizados, por lo que su uso en conjun-

tos de datos reales no se aconseja en principio. Debe destacarse también el buen comportamiento que suele ofrecer E-M, que sin llegar a ser el que mejor resultado presenta, sí que suele comportarse de manera bastante aceptable. De todos modos, tampoco se aconseja en principio su utilización en datos reales por dos motivos principalmente; por un lado, porque es relativamente complejo de implementar con lo cual su rendimiento no compensa el esfuerzo de su implementación en un sitio web real, y por otra parte, como estos conjuntos de datos han sido generados siguiendo una distribución Gaussiana, el resultado está algo sesgado, ya que el algoritmo E-M está basado en mezcla de Gaussianas, y por tanto, tiene "ventaja" frente al resto de algoritmos, que no parten de esa hipótesis.

Por tanto, como conclusión los dos algoritmos que presentan un funcionamiento más adecuado son SOM y ART2, que serán analizados con mayor profundidad al tratar con datos reales, destacando una vez más ART2, que presenta excelentes resultados sin utilizar como dato de entrada el número de conjuntos que debe encontrar.

Capítulo 5

Estudio del portal web *Infoville XXI*

Resumen del capítulo

El objetivo de este capítulo es el de aplicar la metodología propuesta (recomendaciones basadas en agrupamiento previo) a un conjunto de datos real. En particular, el método se aplicará al portal web Infoville XXI, que es un portal que ofrece servicios de diferente tipo a los ciudadanos de la Comunidad Valenciana. Una vez presentado el portal y el conjunto de datos del que se dispone, se describe la metodología general que se propone y que básicamente consiste en utilizar los resultados obtenidos con los conjuntos de datos artificiales para decidir qué algoritmo debe utilizarse cuando se tiene un conjunto de datos real. Para asegurar aun más el éxito de la decisión del algoritmo a utilizar, se realiza primeramente un agrupamiento con una versión reducida del conjunto de datos, intentando analizar la información que aportan los agrupamientos obtenidos con los diferentes algoritmos. Posteriormente, se realiza un agrupamiento utilizando los algoritmos seleccionados (SOM y ART2) con el conjunto de datos real y completo, utilizando la información aportada por el agrupamiento para desarrollar un recomendador colaborativo, realizando un análisis de predicción que permita determinar la viabilidad de un recomendador de este tipo. Las aportaciones más destacables de este capítulo son el análisis de interpretabilidad que se realiza con el agrupamiento preliminar de Infoville, y el estudio de la viabilidad del recomendador.

5.1. Datos reales de un portal web: Infoville XXI

5.1.1. Introducción

La utilización de conjuntos de datos artificiales resulta muy útil para analizar el funcionamiento de los algoritmos de agrupamiento en diferentes escenarios; no obstante, el uso de datos reales se hace absolutamente necesario como prueba final. De hecho, ésta es la piedra angular de la metodología que se propone en esta tesis, y que se basa en comparar el portal web real en el que se desea implementar el sistema de recomendaciones con los diferentes conjuntos de datos artificiales, aplicando aquel algoritmo que mejor ha funcionado con el conjunto de datos artificial más parecido.

En particular, en esta tesis nos centramos en datos que proceden del portal web Infoville XXI (`http://www.infoville.es`). Se trata de un portal web regional que ofrece servicios a los ciudadanos de la Comunidad Valenciana. Este tipo de portales constituyen una vía interactiva para el intercambio de información entre los ciudadanos y la Administración. Además, involucran a los ciudadanos en la sociedad de la información, ofreciendo un número cada vez mayor de servicios en Internet, creando de esta manera un nueva forma de ofrecer servicios al público como resultado de la interacción entre servicios básicos que ofrecen el Gobierno e, incluso, entidades privadas, y que acaban en el ciudadano que solicitó el servicio.

El éxito y la aceptación generalizada de estos portales depende en gran medida de conseguir atraer la atención de los ciudadanos, así como de la Administración y de las entidades públicas y privadas de la región en cuestión. En este sentido, la personalización del portal es una manera evidente de hacer más atractivo el portal para los usuarios. En particular, se propone llevar a cabo parte de esta personalización con recomendaciones sobre aquellos objetos del portal en los que el usuario probablemente estará interesado. Como se verá posteriormente, el análisis realizado permite concluir que el agrupamiento de usuarios es importante de cara a disponer de un recomendador que funcione adecuadamente o, al menos, bastante mejor a como lo haría un recomendador que no tuviera en cuenta esta información.

El portal web utilizado, Infoville XXI, es un sitio web oficial financiado por la Generalitat Valenciana, que ofrece a los ciudadanos de la Comunidad Valenciana más de 2.000 servicios, agrupados en 22 descriptores: Administración

Pública, agenda/eventos, área infantil, ayuntamientos, callejero, canales[1], compras, comunidad Infoville[2], diario Infoville, educación y formación, finanzas, información al ciudadano, Interno, Registro[3], Lanetro (información local, básicamente de ocio), envío de mensajes SMS, ocio, prensa digital, turismo en la Comunidad Valenciana, turismo nacional e internacional, buscador, y utilidades del usuario[4]. Actualmente, más de 50.000 hogares valencianos pertenecientes a unos 125 diferentes ayuntamientos de la Comunidad Valenciana se encuentran conectados a Infoville XXI, habiendo registrado más de dos millones de accesos hasta la fecha.

Es de remarcar que Infoville, que comenzó como un proyecto abanderado por la Generalitat Valenciana, forma actualmente parte de un proyecto europeo que usa este término para portales web de servicios al ciudadano en diferentes países de la Unión Europea (Reino Unido, Dinamarca, Alemania, Italia, Suecia y Francia).

5.1.2. Preprocesado de los datos

Los datos utilizados en la presente tesis corresponden a 34.580 accesos de 4.800 usuarios al portal web Infoville XXI, entre junio de 2002 y febrero de 2003. Los datos registrados una vez procesado el fichero *log*[5], constan de un número que identifica al usuario (que debido al anterior preprocesado, se trata simplemente de un número aleatorio que no ofrece ninguna información sobre la identidad del usuario), un identificador de sesión y un identificador de servicio, junto a la fecha y hora correspondiente a cada acceso. Además, se dispone de la relación existente entre servicios y descriptores, que per-

[1] Actualmente, este descriptor está formado por información sobre cuatro materias específicas: educación, búsqueda de empleo, fundación de empresas y vivienda.

[2] Este descriptor permite la comunicación de los usuarios que acceden al portal a través de correo electrónico, foros, envío de postales, tablones de anuncios, etc.

[3] Interno y Registro son descriptores utilizados por los administradores del portal para la gestión del mismo.

[4] Estas utilidades comprenden agenda personal, personalización del portal, página web personal, asistente de ayuda, etc.

[5] Tissat, S.A. (`http://www.tissat.es/`), que es la compañía designada por la Generalitat Valenciana y la *Fundació OVSI, Oficina Valenciana per a la Societat de la Informació* realizó un primer procesado de los datos procedentes de los ficheros *log* de acceso de usuario para eliminar redundancias y otro tipo de información no útil, como se comentó en el primer capítulo, así como para camuflar las direcciones IP de los usuarios con el fin de conservar su privacidad.

mite, por un parte, disponer de los accesos a descriptores para llevar a cabo el agrupamiento y, por otra, de los accesos a servicios para realizar el estudio sobre la recomendación real de servicios. Partiendo de esta información, se llevó a cabo un preprocesado para eliminar aquellos datos que no ofrecían información útil para nuestros objetivos, así como para construir conjuntos de datos que pudieran ser utilizados en las fases de agrupamiento y predicción de recomendaciones. Este preprocesado involucra los siguientes pasos:

1. *Eliminación de los administradores del portal.* Los administradores del portal crean una gran cantidad de usuarios ficticios con fines de test. Estos usuarios no son útiles en términos de extracción de conocimiento y por tanto, fueron eliminados del conjunto de datos. En particular, aquellos usuarios que tenían registradas más de 500 sesiones abiertas fueron eliminados.

2. *Eliminación de usuarios "anómalos".* Aquellos usuarios que accedieron una única vez al portal durante todo el período de estudio pueden considerarse como usuarios perdidos y, por consiguiente, fueron eliminados del conjunto de datos. Además, más del 95 % de los usuarios accedieron al portal menos de 30 sesiones, por lo que aquellos usuarios que registraron más de 30 sesiones abiertas también fueron eliminados del conjunto de datos.

3. *Eliminación de descriptores que registran un número extremo de accesos.* Como el agrupamiento se lleva a cabo en el espacio de descriptores, es importante analizar la información aportada por éstos. Aquellos descriptores que registran un número muy bajo de accesos no deben ser tenidos en cuenta para la fase de agrupamiento de usuarios ya que no contendrán una gran cantidad de información útil. Por otro lado, aquellos descriptores que registran un número muy elevado de accesos deben ser asimismo eliminados ya que pueden sesgar el comportamiento del algoritmo de agrupamiento considerablemente. En este sentido, fueron eliminados 6 descriptores (agenda/eventos, área infantil, finanzas e Interno por los pocos accesos registrados a ellos y, al contrario, envío de mensajes SMS y Comunidad Infoville por su elevado número de accesos), permaneciendo los 16 restantes en el conjunto de datos. Este fase del preprocesado es similar a realizar un modelado de los descriptores de acuerdo con la Ley de Zipf (Breslau et al., 1999). Debe

señalarse que estos descriptores fueron eliminados únicamente en lo que hace referencia a las tareas de agrupamiento, pero los servicios que pertenecen a cada descriptor fueron tenidos en cuenta para la fase de predicción de recomendaciones, ya que desde un punto de vista real, todos los servicios del portal pueden ser accedidos y, por tanto, recomendados.

4. *Eliminación de usuarios que acceden al portal menos de tres sesiones.* Aquellos usuarios que acceden al portal menos de tres sesiones también fueron eliminados del conjunto de datos, ya que resultaría bastante complicado para los algoritmos de agrupamiento encontrar similitudes entre los usuarios utilizando una información tan escasa.

5. *Preparación final para el agrupamiento.* Los accesos al descriptor "Registro" no fueron tenidos en cuenta, ya que todos ellos corresponden a los administradores del portal y, por tanto, no resultan útiles en lo que hace referencia a recomendaciones sobre usuarios del portal. Además, los accesos fueron codificados siguiendo una notación probabilística para facilitar su procesado por los algoritmos de agrupamiento, de manera análoga a cómo se trabajaba con los conjuntos de datos artificiales[6]. Los datos fueron separados en dos conjuntos: un primer conjunto formado por 17.404 accesos correspondientes a los primeros meses del estudio fue utilizado para llevar a cabo la tarea de agrupamiento de usuarios; el segundo conjunto formado por 14.079 accesos correspondientes a la segunda mitad de los meses contemplados en el estudio fue utilizado para el análisis de las recomendaciones. Como ya se ha venido comentando, el agrupamiento se lleva a cabo en el espacio de 15 descriptores seleccionados, mientras que las recomendaciones tienen lugar en el espacio de servicios. Este segundo conjunto utilizado para el estudio de las recomendaciones servirá evidentemente también para comprobar la robustez del agrupamiento encontrado, ya que si las recomendaciones basadas en el agrupamiento funcionan bien con un conjunto de datos distinto, esto demuestra la robustez del agrupamiento encontrado.

[6]Esta codificación fue llevada a cabo observando para cada usuario los accesos que éste registraba a los distintos descriptores, normalizando a continuación el número de accesos a la unidad para cada usuario.

5.2. Recomendaciones

5.2.1. Descripción general del proceso de recomendaciones

El método de recomendaciones que se propone consta de cuatro etapas. Otros autores han propuesto diferentes etapas para describir el proceso de recomendaciones (Geyer-Schulz y Hahsler, 2002). No obstante, y a pesar de que algunas de las etapas que proponen otros autores pueden ser más o menos similares a las que se proponen en esta tesis, no se han encontrado referencias de metodologías análogas a la propuesta. Los cuatro pasos de la metodología propuesta son: desarrollo de un modelo de usuario, agrupamiento de usuarios, estudio de la viabilidad de un recomendador colaborativo y estudio del efecto real de las recomendaciones.

En la Figura 5.1 se representan con más detalle los diferentes pasos de la metodología. El primer paso es sin duda una de las partes más relevantes y noveles dentro de lo que es la metodología en general; se trata de un modelo de usuario web. Este modelo de usuario, explicado en el Capítulo 4, se utiliza para generar conjuntos de datos artificiales que puedan ser paradigma de diferentes situaciones que puedan encontrarse en portales web reales.

Seguidamente, se realiza una comparación de los diferentes algoritmos de agrupamiento utilizados para estos conjuntos de datos. Al tratarse de conjuntos de datos artificiales que han sido generados controladamente, pueden establecerse, como se vio en el Capítulo 4, medidas para evaluar la bondad de los agrupamientos encontrados en cada uno de los escenarios. De esta manera, la información sobre el funcionamiento de cada algoritmo en cada conjunto de datos puede ser almacenada en una tabla de almacenamiento y búsqueda (*Look-Up Table, LUT*).

Por tanto, cuando se vaya a trabajar con un conjunto de datos reales, podrán compararse sus características con las de los diferentes conjuntos de datos artificiales que han sido generados utilizando el modelo de usuario. Una vez que se determine cuál es el conjunto de datos artificial más parecido, lo que se hará será consultar la LUT, aplicando al conjunto de datos real, aquel algoritmo de agrupamiento que mejor funcionaba con el conjunto de datos artificial que se ha encontrado como más similar al real. Para realizar esta comparación entre el conjunto de datos real y los artificiales se ha de valorar qué aspectos objetivos pueden ser conocidos de ambos. En este sentido, los

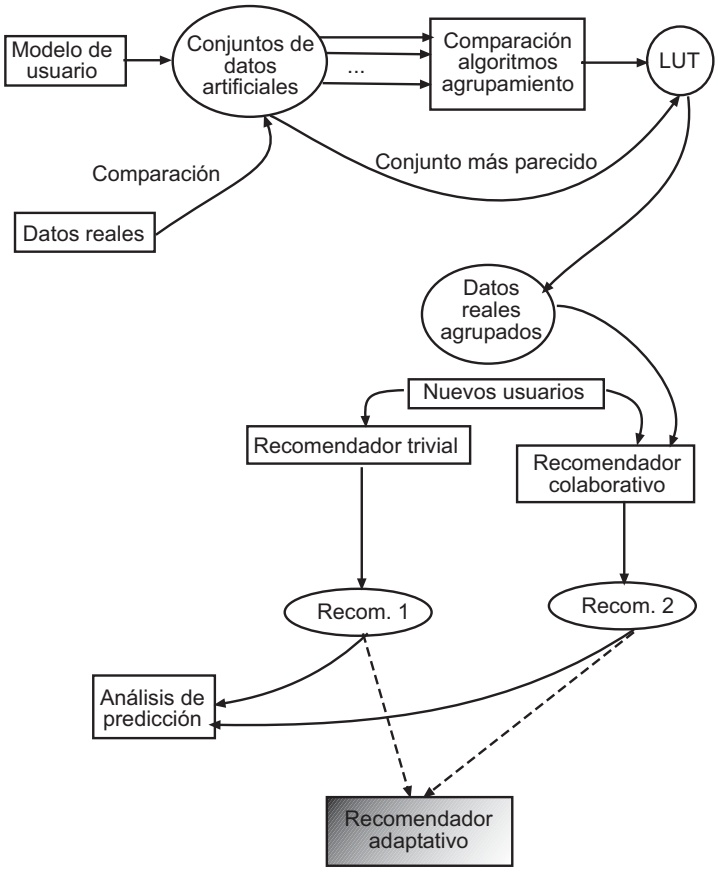


Figura 5.1: Esquema general de la metodología de recomendaciones propuesta. Se representan mediante elipses aquellos pasos de la metodología que son el resultado de un paso anterior. Las flechas discontinuas y el recuadro en gris indican la parte de la metodología que será desarrollada en el futuro.

dos puntos más importantes a valorar serían los siguientes:

1. *Dimensionalidad del portal.* Se trata de tener en cuenta tanto el número de descriptores como el número de servicios. En nuestro caso se trataría preferentemente de número de descriptores ya que el agrupamiento se realiza en el espacio definido por éstos.

2. *Número de usuarios y número de accesos.* En el caso del portal real este dato puede ser algo incierto dependiendo del estado de desarrollo del mismo. Por ejemplo, si está en un claro crecimiento puede que el número de usuarios y accesos actual sea mucho menor que el potencial que pueda tener. Aun así, los responsables del desarrollo del portal

suelen disponer de unas estimaciones en este sentido, más o menos fiables.

Entre estos dos puntos, la dimensionalidad aparece como el más fundamental. Sería deseable poder disponer de una estimación del número de grupos que deben encontrarse, ya que eso facilitaría tanto la comparación con los conjuntos de datos artificiales como la tarea del propio algoritmo de agrupamiento. Sin embargo, esta estimación no va a poder ser posible en la mayoría de los casos. En cuanto al número de usuarios del portal, además de la variabilidad del mismo en determinados momentos de desarrollo del portal, realmente la información más importante es la de conocer la proporción de usuarios en los diferentes grupos, con lo cual se va otra vez a la información de grupos, que es un dato con el que no se puede contar *a priori*. Por tanto, básicamente lo que se hará es comparar la dimensionalidad del conjunto de datos real con la de los artificiales para decidir qué algoritmo, o algoritmos, son los más adecuados para agrupar a estos usuarios reales. Aunque como se explicó en el Capítulo 2, se utilizan técnicas para decidir el número correcto de *clusters* en cada caso, el número de grupos del conjunto de datos artificial más similar al conjunto real puede utilizarse como primera referencia.

Una vez que se tienen los datos pertenecientes al portal web real agrupados, se entra en el tercer paso de la metodología, que se explicará con más detalle en la Sección 5.2.4. Se trata de aplicar la información que aporta este *clustering* para ofrecer recomendaciones a los usuarios a través de un filtrado colaborativo. Para estudiar la viabilidad del recomendador propuesto se compara el éxito de las recomendaciones ofrecidas por éste con el que tendría un recomendador que ofrece simplemente los servicios más probables del portal, siempre que no hayan sido accedidos previamente. Lo que se hace es realizar un análisis de predicción que no mide el efecto real de las recomendaciones sobre el usuario sino los servicios a los que por él mismo va a acceder. Esto permite analizar la información aportada por el *clustering*, separando la caracterización de usuarios que se haya hecho del posible efecto de la interfaz real de la recomendación. Si la mejora que se observa utilizando la información del *clustering* es apreciable, se deduce que la información aportada por éste es relevante y, por tanto, resulta interesante la implementación real de un sistema de este tipo.

Si el sistema se ha considerado interesante y ha sido implementado, puede disponerse de datos acerca de la aceptación real de las recomendaciones que, en principio, se prevé que ha de ser mayor que la obtenida a través del análisis de predicción. Este estudio final es interesante y debe realizarse, ya que además puede permitir incluir algún tipo de adaptación en el recomendador para que se adecúe mejor a los usuarios y, eventualmente, a nuevos comportamientos que puedan aparecer.

5.2.2. Agrupamiento preliminar de usuarios de Infoville XXI

Si se compararan las características del conjunto de datos de Infoville XXI con las de los conjuntos de datos artificiales, y se utilizara esta comparación, como se vio en el anterior apartado, para decidir el algoritmo que debe utilizarse para agrupar a los usuarios de este portal web, los algoritmos más idóneos serían, o bien SOM, o bien ART2, según puede deducirse de los resultados mostrados en el Capítulo 4 para conjuntos con elevado número de descriptores. No obstante, como prueba adicional para decidir el algoritmo más idoneo, se llevó a cabo un agrupamiento preliminar de los usuarios de este portal web, simplemente para conocer las capacidades de los algoritmos para encontrar *clusters* útiles y que aporten información relevante. En este sentido, se consideró una versión reducida del conjunto de datos (desde noviembre de 2002 hasta enero de 2003) y se seleccionó un pequeño conjunto de descriptores. En primer lugar, las frecuencias de acceso a los descriptores durante este período de tiempo fueron analizadas siguiendo la Ley de Zipf para eliminar aquellos descriptores menos discriminantes. De entre los restantes descriptores, cinco fueron seleccionados por Tissat, S.A. y la Fundació OVSI como los más significativos, a saber: Administración Pública, ayuntamientos, canales, compras y ocio. Esta selección llevó a disponer de un total de 1.676 usuarios para este estudio preliminar. En particular, se utilizaron 1.000 usuarios para obtener los *clusters*, reservando los 676 restantes para comprobar la robustez del agrupamiento, es decir, que los *clusters* encontrados describían correctamente a estos 676 usuarios no utilizados por los algoritmos de agrupamiento.

Como en este caso no se conocen los grupos que se deben encontrar *a priori*, los resultados fueron analizados en función de la interpretabilidad de los agrupamientos encontrados. Esto fue posible porque el agrupamiento fue rea-

lizado en un espacio de baja dimensionalidad, definido únicamente por cinco *clusters*, donde además el significado de cada una de las componentes del espacio era conocido. En este sentido, los agrupamientos encontrados por los algoritmos CM, FCM y E-M resultaban bastante complicados de interpretar atendiendo a lo que se podrían considerar como comportamientos normales de usuarios. Por otro lado, ACJ, SOM y ART2 ofrecieron agrupamientos claramente interpretables, ya que agruparon los datos en siete conjuntos diferentes; cinco de estos conjuntos se encontraban claramente centrados en cada uno de los cinco descriptores considerados, mientras que los otros dos *clusters* contenían a usuarios que estaban interesados, o bien en los servicios de ocio ofrecidos por el portal, o bien en los servicios administrativos. En particular, uno de estos dos *clusters* se encontraba centrado entre los descriptores "compras" y "ocio"; por tanto, este *cluster* contenía a usuarios que principalmente acceden al portal buscando los servicios de entretenimiento o tiempo libre que éste ofrece. El otro *cluster* se encontraba centrado entre los descriptores "Administración Pública" y "ayuntamientos", presentando asimismo una pequeña pertenencia al descriptor "canales"; es decir, que este *cluster* contenía a gente que accedía al portal buscando básicamente los servicios administrativos que éste ofrecía. Estos siete *clusters* sirven para demostrar dos hechos importantes: por un lado, la capacidad de los algoritmos ACJ, SOM y ART2 para encontrar similitudes entre los usuarios de este portal y por otro, la propia utilidad del portal, que queda claramente demostrada ya que éste fue básicamente diseñado para cumplir dos objetivos: acelerar las tareas administrativas mediante la utilización de las nuevas tecnologías, y ofrecer una rápida solución a los ciudadanos en lo que se refiere a los servicios de entretenimiento (de esta manera pueden familiarizarse de un modo mucho más "divertido" y, por tanto, natural con Internet, lo que en niños o ciudadanos de edad avanzada es de especial importancia).

Llegado el momento de decidir qué algoritmo debería usarse para el agrupamiento final, es decir, aquel que tiene en cuenta todos los descriptores y usuarios, los algoritmos CM, FCM y E-M deberían rechazarse, ya que si no funcionan con una versión sencilla del portal, difícilmente lo harán con la versión completa, y por consiguiente más complicada. De entre los tres algoritmos restantes, podríamos descartar ACJ, ya que aunque presenta un comportamiento similar a SOM y ART2 con la versión reducida del portal, en el Capítulo 4 se vio que SOM y ART2 presentaban un comportamiento

más adecuado con conjuntos de datos artificiales de alta dimensionalidad. Por tanto, se estudiará el efecto que produce el *clustering* sobre el recomendador colaborativo, considerando como algoritmos de agrupamiento SOM y ART2.

5.2.3. Agrupamiento final de usuarios

Como ya se ha indicado, el resultado con los conjuntos de datos artificiales y el agrupamiento preliminar con una versión reducida del conjunto de datos de Infoville, hizo que se considerara idóneo para agrupar a los usuarios de este portal la utilización tanto de mapas autoorganizativos como de la teoría de la resonancia adaptativa.

La selección de los modelos finales se llevó a cabo atendiendo, en primer lugar, al número de grupos encontrado y a las características de éstos. De entre una primera selección de modelos realizada de esta manera, los modelos finales que serán utilizados para la recomendación fueron escogidos en función de lo exitoso que sería un recomendador colaborativo basado en este *clustering* (este grado de éxito se mide en el tercer paso de la metodología propuesta, al estudiar la viabilidad del recomendador).

Respecto al SOM, aunque se hicieron pruebas con mapas unidimensionales y bidimensionales, se ha utilizado finalmente un mapa bidimensional con topología toroidal y formado por 460 neuronas. Una vez obtenido el mapa, se siguió el procedimiento de enlace simple para unir neuronas cercanas llegándose a una situación donde existían 20 neuronas representativas del mapa. Partiendo de esta situación, se utilizó un ACJ de enlace completo, evaluando mediante comparación con los conjuntos de datos artificiales y los índices de Dunn y Davies-Bouldin el número de grupos más conveniente. En la Figura 5.2 se representan estos índices en función del número de grupos. Como puede observarse, el índice de Davies-Bouldin experimenta un cambio abrupto para ocho grupos, permaneciendo prácticamente con un valor constante tanto para un número de grupos mayor como menor. Este hecho indica un cambio significativo en lo que se refiere a los parecidos *inter-cluster* para esa situación. Por otro lado, el índice de Dunn muestra su valor máximo para el caso de dos grupos, lo cual quiere decir que es ésa la situación en la que los *clusters* aparecen como más compactos y separados, aunque eso no es suficiente para considerar esa situación como la más idónea del agrupamiento;

otro factor que también se observa en el índice de Dunn es que a partir de ocho grupos este índice empieza a incrementar su valor por lo que teniendo el cuenta los resultados del agrupamiento preliminar y la comparación con los conjuntos de datos artificiales de mayor dimensionalidad, se consideró al final el agrupamiento formado por ocho *clusters*.

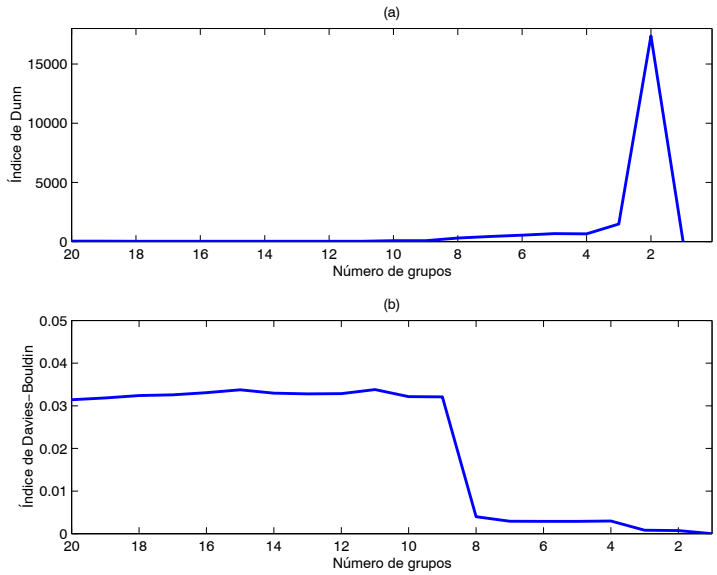


Figura 5.2: (a) Representación del índice de Dunn frente al número de grupos para un agrupamiento realizado por un SOM encadenado con un ACJ; (b) representa el índice de Davies-Bouldin frente al número de grupos.

En el caso de la ART se encontraron 12 grupos de usuarios con una selección del parámetro de vigilancia y de la activación de la neurona ganadora de la capa F2 iguales a 0.8. La constante de aprendizaje también tomó este valor, y el número de iteraciones considerado fue 50, observándose que no se producían actualizaciones durante las últimas iteraciones. La situación parece corresponderse con un agrupamiento bastante "natural" ya que un barrido en los parámetros de esta red mostró que prácticamente era el único agrupamiento formado por un un número lógico de grupos. Si se tomaban umbrales de parecido mayores se llegaba a un número desproporcionado de grupos, y umbrales de parecido menores conducían a que prácticamente hubiera un único grupo representativo de toda la población.

5.2.4. Metodología

Una vez realizado el agrupamiento de usuarios, se pasa a la fase de recomendación colaborativa propiamente dicha. La metodología se basa en medir el parecido que los usuarios que van a ser recomendados tienen con los grupos que han sido encontrados en la parte del conjunto de datos reservada para *clustering*. Para medir ese parecido hace falta una cierta información que describa el comportamiento del usuario, y que será obtenida a partir de los servicios que vaya demandando el usuario; esta información no estará obviamente disponible en los primeros accesos del usuario, ya que no se dispone del suficiente conocimiento para haberlo caracterizado. Para que el sistema recomiende objetos desde el primer acceso del usuario, lo que se hace es recomendar al usuario en un primer momento, los servicios más probables del portal, a los cuales él todavía no ha accedido. Es decir, si un nuevo usuario accede al portal por primera vez lo que se hará es recomendarle el servicio del portal que más veces es accedido por los usuarios en general; caso de que acepte la recomendación, siguientemente se le ofrecerá el segundo servicio más probable del portal, y así sucesivamente. Si por contra el usuario no accede a esa primera recomendación, ésta se le seguirá ofreciendo. Lo que es importante es ofrecer siempre los servicios más probables pero a los cuales él no ha accedido todavía para que las recomendaciones sean útiles en el sentido de que muestren al usuario servicios que él todavía no conoce (Burke, 2002). Esta estrategia de recomendar el servicio más probable del portal no previamente accedido por el usuario se llevará a cabo durante los P primeros accesos del usuario, representando por tanto P la profundidad necesaria para poder caracterizar al usuario en función de los servicios a los que ha accedido.

A partir del acceso $(P+1)$-ésimo se considera que el usuario ya está caracterizado, siguiéndose una metodología de recomendaciones como la mostrada en la Figura 5.3. En primer lugar, se utilizan los accesos registrados del usuario para medir su parecido con los diferentes grupos de usuarios que han encontrado los algoritmos de agrupamiento. Para medir este parecido habrá que codificar los accesos a servicios del usuario con una notación probabilística (frecuencias normalizadas de acceso a los descriptores) y encontrar la mínima distancia entre el vector que caracteriza al usuario y los diferentes grupos de usuarios. Una vez que se ha encontrado el grupo más cercano al

usuario, lo que se hace es recurrir a una LUT donde aparecen ordenados los servicios del portal en probabilidad decreciente para ese grupo. De esta manera, lo que se hace es recomendar el servicio más probable del grupo al que se considera que pertenece el usuario, y que todavía no haya sido accedido por el usuario, para conservar la idea de utilidad de la recomendación. Como se explicó en el Capítulo 3, existe otra estrategia de recomendación basada en atraer la confianza del usuario, que consiste en recomendarle al usuario servicios a los que con gran certeza se sabe que va acceder por él mismo, típicamente se le pueden ofrecer servicios a los que ya se ha accedido; lo que se busca con esta estrategia es conseguir que el usuario confíe en el recomendador, ya que comprueba que en efecto las recomendaciones coinciden con sus preferencias. No obstante, para el análisis que nos planteamos en este trabajo, basado en caracterizar a los usuarios, resulta mucho más interesante el análisis de recomendaciones basadas en la utilidad de éstas.

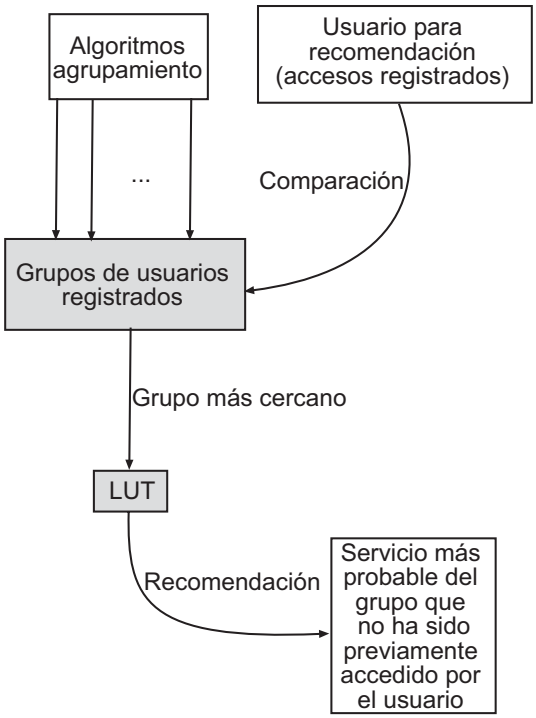


Figura 5.3: Diagrama de bloques que muestra el funcionamiento del recomendador colaborativo.

La recomendación en el acceso Q-ésimo del usuario ($\forall Q \geq P + 1$) tiene siempre en cuenta los $Q - 1$ accesos anteriores para caracterizar al usuario.

De esta forma, se aprovecha toda la información posible del usuario para, por un lado caracterizarlo mejor, y por otro, tener claro en todo momento los servicios a los que ya ha accedido para no volverlos a recomendar.

5.2.5. Viabilidad del recomendador

Se ha de hacer especial énfasis en el tercer paso de la metodología, es decir en el estudio de la viabilidad de implementación del recomendador. Es éste un paso no habitualmente contemplado en el desarrollo de un recomendador, ya que normalmente éste es implementado y a continuación evaluado. No obstante, este sencillo estudio de viabilidad puede aportar información valiosa sobre la mejora que puede suponer un recomendador que contemple la información procedente del agrupamiento. El hecho de que esta mejora sea mayor o menor puede ayudar a decidir con mayor seguridad si es interesante, o no, implementar un recomendador de este tipo para el portal sometido a estudio.

Para llevar a cabo este estudio de viabilidad lo que se hace no es ver si realmente las recomendaciones han sido aceptadas o no, ya que hay que recordar que este paso es anterior a la implementación del recomendador y, por tanto, no se dispone de esta información. Lo que se hace es comprobar si las recomendaciones que hubiera ofrecido el sistema coinciden con los servicios que de hecho son accedidos por el usuario (que no ha recibido recomendaciones). Se trata más bien de un análisis de predicción de los servicios que serán accedidos por el usuario. En particular, si el usuario accede al servicio que se hubiera recomendado, se considera como un éxito. La tasa de éxito (TE) así calculada será previsiblemente inferior a la que se obtendrá cuando las recomendaciones se presenten en el portal, ya que es de suponer que una interfaz atractiva y el hecho de mostrarle a un usuario un servicio en el que probablemente estará interesado deben afectar positivamente a su comportamiento. Por tanto, la TE calculada en la predicción debe entenderse, con precaución, como un umbral inferior de la TE que se obtendría en realidad con el recomendador. Relacionado con este último aspecto, hay que tener en cuenta que este estudio de viabilidad permite conocer con mayor exactitud la información aportada por el agrupamiento, ya que en el estudio de las recomendaciones, la interfaz también tiene una considerable importancia (Geyer-Schulz y Hahsler, 2002), que en este caso se ve sin embargo anulada;

es decir, que la TE únicamente obedece a la mejor o peor caracterización que los algoritmos de *clustering* hayan hecho de los usuarios y, por supuesto al atractivo que los servicios del portal puedan tener para los usuarios.

Para medir la información aportada por el *clustering* al recomendador lo que se hace es comparar la TE del recomendador propuesto con la TE de un recomendador trivial, que recomienda el servicio más probable del portal, siempre que no haya sido previamente accedido por el usuario. Por tanto este recomendador sería mucho más simple que el mostrado en la Figura 5.3, constando únicamente de una LUT que tiene ordenados los servicios del portal en probabilidad decreciente y que recomienda al usuario en cuestión el servicio más probable de todo el portal que no haya sido previamente accedido por él.

Tabla 5.1: TE [%] en la predicción de servicios accedidos como estudio preliminar de viabilidad en el desarrollo de un recomendador. Se compara la predicción cuando se utiliza la información del agrupamiento obtenido por una red ART2 (recomendador colaborativo) y cuando no se utiliza tal información (recomendador trivial). Esta comparación se lleva a cabo para diferentes valores de P y Q.

P	**Q**	**Trivial**	**Colaborativo**
2	**4**	6.91	12.84
2	**5**	10.12	14.57
2	**6**	13.07	16.48
2	**7**	16.13	18.74
3	**4**	3.47	13.73
3	**5**	7.04	15.11
3	**6**	10.31	16.81
3	**7**	13.70	18.97
4	**6**	7.56	18.06
4	**7**	11.32	19.94
5	**7**	8.16	20.94

En la Tabla 5.1 se muestra la TE obtenida para el recomendador colaborativo y el recomendador trivial en los 14.076 accesos que se reservaron para este fin, y que no fueron utilizados para obtener los grupos de usuarios. En particular, los resultados se muestran para el caso de utilizar la ART2 como herramienta de agrupamiento. Los resultados utilizando el SOM bidimensional anteriormente citado son muy similares por lo que no se muestran

en la tabla. Se muestra la TE para diferentes valores de P y Q de manera que pueda observarse el efecto que tiene el utilizar más o menos información para caracterizar al usuario, así como el efecto de recomendar más o menos servicios (profundidad de predicción mayor o menor). Como puede observarse, el recomendador colaborativo ofrece una TE considerablemente mayor que el recomendador trivial. Además, cuantos más accesos se utilizan para caracterizar al usuario, mayor es esta diferencia; esto es lógico ya que la información para caracterizar al usuario es mayor. De forma adicional, debe señalarse que la importancia del *clustering* parece ser más relevante en los primeros accesos comenzando desde el $(P + 1)$-ésimo; a medida que el número de accesos aumenta, es decir que el valor de Q va haciéndose mayor, la diferencia entre ambos recomendadores disminuye. Esto es interesante en el sentido que el recomendador puede aportar una importante mejora en las primeras recomendaciones que se le sigieran al usuario, con lo que se puede aumentar la confianza de éste en el portal en sus primeros accesos, donde es clave la satisfacción del usuario para conseguir y mantener su fidelidad al portal a medio y largo plazo. Esta última conclusión indica que aunque existen servicios representativos de los diferentes grupos, también hay servicios que podríamos llamar de interés general, que agradan a la práctica totalidad de los usuarios y que aparecen en orden de importancia justamente después de los servicios representativos del grupo, y que es lo que hace que las TEs de ambos recomendadores se acerquen para valores altos de Q.

Finalmente, la TE obtenida puede considerarse como muy satisfactoria ya que típicamente los recomendadores suelen presentar porcentajes de aceptación bastante menores (Geyer-Schulz y Hahsler, 2002), y además debe recordarse que la TE calculada debe entenderse como un umbral inferior de la que se obtendría con recomendaciones reales ya que una buena interfaz debería ayudar a mejorar el éxito de las recomendaciones.

Capítulo 6

Conclusiones y líneas futuras.

6.1. Conclusiones generales

Desde un punto de vista general, las conclusiones más importantes que pueden extraerse del presente trabajo son las siguientes:

- Se ha propuesto una metodología global para llevar a cabo recomendaciones útiles en un portal web. La metodología está principalmente basada en la generación de portales web artificiales (utilizando un modelo de usuario), en su comparación con el portal web real donde se desea realizar las recomendaciones, y en un estudio preliminar sobre la viabilidad del recomendador.

- Se ha llevado a cabo un agrupamiento pseudo-supervisado de usuarios, ya que aunque los algoritmos de agrupamiento tienen un tipo de aprendizaje no supervisado, la generación de conjuntos de datos artificiales de manera controlada mediante un modelo de usuario permite la evaluación del funcionamiento de los algoritmos con estos conjuntos.

- La metodología se ha aplicado sobre un conjunto de datos real consistente en accesos al portal web Infoville XXI. El estudio de este portal muestra la idoneidad de implementar un recomendador colaborativo.

En las siguientes secciones se mostrarán conclusiones más específicas sobre determinados aspectos del trabajo, así como la proyección futura del mismo.

6.2. Conclusiones sobre el modelo de usuario

Respecto al modelo de usuario web, cabría destacar las siguientes conclusiones:

- Se ha propuesto un modelo de usuario web de propósito general que tiene ciertas restricciones observadas en accesos de usuarios a portales web reales.
- El modelo de usuario genera vectores de frecuencia de acceso tanto en el espacio formado por todos los servicios web presentes en el portal, como en un espacio de descriptores, que consisten en etiquetas informativas que aglutinan diferentes servicios de características similares. Se permite entonces encontrar grupos de usuarios en un espacio donde tanto las componentes del espacio de representación como los propios grupos son altamente informativos.
- El modelo de usuario ha sido desarrollado de una manera modular por lo que puede ser fácilmente modificable o ampliable. Por ejemplo, es sencillo introducir nuevas restricciones al modelo o generar conjuntos con diferentes características a los que ya se tienen.

6.3. Conclusiones sobre los algoritmos de agrupamiento

Respecto a los algoritmos de agrupamiento, las conclusiones más importantes serían las siguientes:

- Se han utilizado seis diferentes algoritmos de agrupamiento para encontrar similitudes entre usuarios web, ya sean éstos reales o artificiales. Estos algoritmos han sido C-Medias, C-Medias Difuso, algoritmos de agrupamiento jerárquico, algoritmo *Expectation-Maximization*, Mapas Autoorganizativos y Teoría de la Resonancia Adaptativa.

- Se han realizado algunas aportaciones para la extracción de grupos utilizando Mapas Autoorganizativos. Éstas están basadas en tratamiento digital de imágenes y en la encadenación de una fase de mapeo entre el espacio de representación y el de salida, seguida de otra fase de agrupamiento jerárquico.

- Se han elaborado medidas para decidir el algoritmo más adecuado cuando se utilizaban conjuntos de datos artificiales. Estas medidas ofrecían información sobre el número correcto de grupos, el ajuste de los grupos correctamente encontrados con los grupos reales, y cómo los algoritmos eran capaces de captar la distribución estadística subyacente en los datos.

- Las medidas comentadas en el anterior punto fueron asimismo utilizadas para decidir el número correcto de grupos en aquellos algoritmos que no son capaces de encontrar por ellos mismos la cantidad de grupos que mejor describe los datos.

- Los algoritmos que mejor funcionamiento global han presentado con los conjuntos de datos artificiales han sido los Mapas Autoorganizativos y la Teoría de la Resonancia Adaptativa. Para el caso de conjuntos de datos con alta dimensionalidad, los mejores agrupamientos han sido encontrados por la Teoría de la Resonancia Adaptativa, y para conjuntos de dimensionalidad media, por los Mapas Autoorganizativos. Para conjuntos de dimensionalidad baja, más sencillos, prácticamente cualquiera de los algoritmos utilizados presenta un resultado satisfactorio. Los algoritmos de agrupamiento jeráquico y también el algoritmo *Expectation-Maximization* presentan, en general, unos resultados bastante aceptables, aunque los análisis estadísticos de los grupos encontrados muestran que presentan problemas para captar la estadística de datos subyacente en las distribuciones.

- Respecto al agrupamiento en conjuntos de datos reales, hay que destacar que se realiza un *clustering* previo con un conjunto de reducida dimensionalidad, que se basa en la interpretabilidad de los grupos encontrados. Este análisis de interpretabilidad permite reforzar la decisión sobre el algoritmo idóneo que debe utilizarse, arrojando a su vez dos importantes conclusiones; en primer lugar, que análogamente a lo que

sucedía con los conjuntos de datos artificiales, los Mapas Autoorganizativos y la Teoría de la Resonancia Adaptativa son los dos algoritmos que presentan un funcionamiento idóneo, o al menos que parece ser el más correcto. La segunda conclusión es que los grupos de usuarios encontrados se corresponden justamente con los tipos de comportamiento esperados, y para los que fue diseñado el portal, por lo que la utilidad de éste queda demostrada.

6.4. Conclusiones sobre los sistemas de recomendaciones

Los aspectos más relevantes que se concluyen sobre los sistemas de recomendaciones son los siguientes:

- Se ha establecido que el tipo de recomendación más idóneo para el portal web Infoville XXI es colaborativo.

- Las recomendaciones que se plantean están basadas fundamentalmente en su utilidad. En este sentido, los servicios recomendados son siempre los que se consideran más probables de acceder, siempre y cuando no hayan sido previamente accedidos por el usuario que está siendo recomendado.

- Se ha realizado un estudio de viabilidad para determinar si la implementación del recomendador en Infoville XXI sería relevante o no. Este estudio está basado en un análisis de predicción de los servicios accedidos basado en la caracterización previa de los usuarios utilizando algoritmos de agrupamiento. De esta manera, se elimina la influencia que sobre la aceptación de la recomendación pueda tener la interfaz de la misma. Caracterizando a los usuarios mediante Mapas Autoorganizativos y la Teoría de la Resonancia Adaptativa, y llevando a cabo una predicción basada en recomendación colaborativa, se ha obtenido que el éxito de la predicción dobla, aproximadamente, al que se obtendría con una predicción trivial. El éxito alcanzado debe entenderse, con precaución, como un umbral inferior del éxito que se obtendría con recomendaciones reales, por lo que teniendo en cuenta los valores

alcanzados, puede establecerse como interesante y relevante la implementación de un recomendador colaborativo en Infoville XXI.

6.5. Proyección futura

En cuanto a la proyección futura existen muchas posibles líneas de investigación que pueden completar y mejorar el presente trabajo. No obstante, se van a citar solamente aquellos aspectos que serán desarrollados en un futuro cercano y que aparecen como evidentes líneas futuras a tener en cuenta:

- La primera línea de trabajo futuro ha de ser necesariamente el seguimiento del éxito que tengan las recomendaciones reales sobre los usuarios de Infoville, una vez que se disponga de un volumen de datos considerable en este sentido. Esto permitirá además establecer por separado la importancia de la caracterización de los usuarios y de la propia interfaz de la recomendación.

- Utilizar este seguimiento para desarrollar un recomendador adaptativo que permita ir añadiendo al sistema el conocimiento adquirido de los nuevos usuarios. En este sentido, la propuesta es desarrollar un modelo basado en *Learning Vector Quantization* (LVQ) (Ripley, 1996), (Alpaydin, 1998) para ir ajustando los prototipos de los grupos dependiendo de la aceptación, o no, por parte de los usuarios. En este sentido, se trataría de un LVQ modificado que se actualizaría solamente si el usuario acepta la recomendación acercando el consiguiente prototipo y alejando el resto, y que no se actualizaría o lo haría en muy pequeña medida, si no acepta la recomendación. La justificación para esto es que puede que el usuario no acepte la recomendación porque no dispone de tiempo, porque la haya cerrado involuntariamente, e incluso puede que el usuario no esté realmente interesado en esa recomendación, pero no se puede concluir de aquí que este usuario no pertenezca al grupo al que se le ha asignado, ya que los grupos buscan similutudes entre usuarios a grandes rasgos y las recomendaciones de servicios son altamente específicas. En cualquier caso, si se dispone de una gran cantidad de nuevos datos, lo más conveniente será volver a realizar un agrupamiento de los usuarios, ya que pueden haber aparecido nuevos comportamientos en el portal que el algoritmo LVQ no

haya sido capaz de encontrar, ya que básicamente lo que hace no es esto sino ajustar los primeros prototipos de grupos que ya se tenían.

- Conectado con el punto anterior, puede ser interesante el análisis de aquellos nuevos usuarios que no puedan asignarse con claridad a ninguno de los grupos de usuarios ya existentes, sino que se encuentren en la frontera entre dos grupos, por ejemplo. Cuando esto suceda, una aproximación interesante puede ser determinar la pertenencia del usuario a los diferentes grupos. Para ello, puede considerarse que la pertenencia del usuario i al grupo k viene dada por:

$$\mu_{ik} = \frac{1}{1 + \left(\frac{d_{ik}}{A}\right)^B} \tag{6.1}$$

  donde d_{ik} es la distancia de Mahalanobis entre el usuario i y el grupo k, mientras que A y B son parámetros que deben ser ajustados (las primeras simulaciones llevadas a cabo muestran que valores adecuados para A se sitúan entre 0,4 y 0,8, y para B entre 2 y 4). Otro posible método para calcular esta pertenencia viene dado por la proyección del vector de probabilidades de acceso del usuario i sobre el prototipo del grupo k; esto tiene la ventaja de que se elimina la necesidad de ajustar los parámetros A y B, y el inconveniente de que se pierde información sobre la forma del grupo, al utilizar solamente el prototipo del mismo en lugar de la distancia de Mahalanobis del usuario al grupo. Una vez determinada la pertenencia pueden recomendarse servicios de uno u otro grupo en función de la pertenencia que el usuario tenga a estos grupos, de manera que la LUT que almacena los servicios en probabilidad decreciente deberá ser cambiada para que pueda contener servicios correspondientes, en principio, a diferentes grupos. De esta forma, un usuario puede que sea recomendado con un servicio correspondiente a un grupo, y si acepta la recomendación entonces recibir una recomendación correspondiente a otro grupo.

- Un sistema como el presente puede ser aplicado a otros campos, más allá de los portales web de servicios al ciudadano, e incluso de los portales web en general. En particular, puede resultar interesante la aplicación a un campo con un prometedor futuro, como es la televisión interactiva. No obstante, para poder aplicar la metodología a este cam-

po sería necesario en primer lugar cambiar el modelo de usuario, que actualmente está desarrollado para usuarios web.

Bibliografía

Apache Web Server. `http://www.apache.org/`.

Página web de infoville. `http://www.infoville.es/`.

World Wide Web Consortium W3C. `http://www.w3c.org/`.

Agrawal, R., Gehrke, J., Gunopulos, D. y Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. En *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data* (págs. 94–105). Seattle, WA, Estados Unidos.

Agrawal, R., Imielinski, T. y Swami, A. (1993). Mining association rules between sets of items in large databases. En *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data* (págs. 207–216). Washington, D.C., Estados Unidos.

Agrawal, R., y Srikant, R. (1994). Fast algorithms for mining association rules. En *Proceedings of the 20th International Conference on Very Large Data Bases* (págs. 487–499). Santiago, Chile.

Alpaydin, E. (1998). Soft vector quantization and the E-M algorithm. *Neural Networks*, *11*, 467–477.

Alspector, J., Koicz, A. y Karunanithi, N. (1997). Feature-based and Clique-based User Models for Movie Selection: A Comparative Study. *User Modeling and User-Adapted Interaction*, *7*, 279–304.

Andersen, J., Larsen, R. S., Giversen, A., Pedersen, T. B., Jensen, A. H. y Skyt, J. (2000). *Analyzing clickstreams using subsessions* (Technical

Report TR–00–5001). Aalborg, Dinamarca: Department of Computer Science, Aalborg University.

Balaguer, E., y Palomares, A. (2003). *AI recommendation engine of Tissat, S. A.* (Internal report). València, España: Tissat, S. A.

Banerjee, A., y Ghosh, J. (2002). Characterizing visitors to a web site across multiple sessions. En *Proceedings of NGDM'02: National Science Foundation Workshop on Next Generation Data Mining.* Marriott Inner Harbor, Baltimore, MD, Estados Unidos.

Bezdek, J. C., y Pal, S. K. (1992). *Fuzzy models for pattern recognition: Methods that search for structures in data.* Nueva York, NY, Estados Unidos: IEEE Press.

Billsus, D., y Pazzani, M. (2000). User Modeling for Addaptive News Access. *User Modeling and User-Adapted Interaction*, *10*(2–3), 147–180.

Bishop, C. M., Svensén, M. y Williams, C. K. I. (1998). GTM: The generative topographic mapping. *Neural Computation*, *10*(1), 215–234.

Breese, J. S., Keckerman, D. y Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. En *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence* (págs. 43–52).

Breslau, L., Cao, P., Fan, L., Phillips, G. y Shenker, S. (1999). Web caching and zipf-like distributions: Evidence and implications. En *INFOCOM (1)* (p. 126-134).

Brin, S., y Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, *30*(1–7), 107–117.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User Adapted Interaction*, *12*, 331–370.

Cadez, I., D., H., Meek, C., Smyth, P. y White, S. (2001). *Model-based clustering and visualization and navigation patterns on a Web site* (Technical Report Nums. MSR–TR–0018). Laguna Hills, CA, Estados Unidos: Microsoft Research, Microsoft Corporation.

Camps, G., Soria, E., Pérez, J. J., Pérez, F., Figueiras, A. R. y Artés, A. (2002). Cyclosporine concentration prediction using clustering and support vector regression methods. *Electronics Letters*, *38*(12), 568–570.

Carman, C. S., y Merickel, M. B. (1990). Supervising ISODATA with an information theoretic stopping rule. *Pattern Recognition*, *23*(1–2), 185–197.

Carpenter, G. A., y Grossberg, S. (1987). A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine. *Computer Vision, Graphics and Image Processing*, *37*, 54–115.

Carpenter, G. A., y Grossberg, S. (1991). ART2: Self-Organization of Stable Category Recognition Codes for Analog Input Patterns. En *Carpenter, g. a. and grossberg, s., editors, Pattern Recognition by Self-Organizing Neural Networks, MIT Press.* Cambridge, MA, Estados Unidos.

Chang, G., Healey, M. J., McHugh, J. A. M. y Wang, J. T. L. (2001). *Mining the World Wide Web.* Norwell, MA, Estados Unidos: Kluwer Academic Publishers.

Condliff, M. K., Lewis, D. D., Madigan, D. y Posse, C. (1999). Bayesian mixed-effects models for recommender systems. En *SIGIR'99 Proceedings of the Workshop on Recommender Systems: Algorithms and Evaluation.* Berkeley, CA, Estados Unidos.

Duda, R. O., Hart, P. E. y Stork, D. G. (2000). *Pattern classification* (2nd ed.). Nueva York, NY, Estados Unidos: John Wiley & Sons.

Ester, M., Kriegel, H. P., Sander, J. y Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases. En *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining* (págs. 226–231). Portland, OR, Estados Unidos.

Everitt, B. (1981). *Cluster Analysis* (2nd ed.). Halsted Press.

Fausett, L. (1994). *Fundamentals of Neural Networks.* Upper Saddle River, NJ, Estados Unidos: Prentice-Hall.

Fayyad, U., y Uthurusamy. (1996). Data mining and knowledge discovery in databases. *Communications of the ACM*, *39*(11), 24–26.

Foltz, P. W. (1990). Using latent semantic indexing for information filtering. En *R. B. Allen, editors, Proceedings of the Conference on Office Information Systems* (págs. 40–47). Cambridge, MA, Estados Unidos.

Geyer-Schulz, A., y Hahsler, M. (2002). Evaluation of recommender algorithms for an internet information broker based on simple association rules and on the repeat-buying theory. En *Proceedings of WEBKDD'2002* (págs. 100–114). Edmonton, Canada.

Ghosh, J., Strehl, A. y Meregu, S. (2002). A consensus framework for integrating distributed clusterings under limited knowledge sharing. En *Proceedings of NGDM'02: National Science Foundation Workshop on Next Generation Data Mining* (págs. 99–108). Marriott Inner Harbor, Baltimore, MD, Estados Unidos.

González, R. C., y Woods, R. E. (2002). *Digital image processing* (2ª ed.). Prentice-Hall.

Guha, S., Rastogi, R. y Shim, K. (1999). Rock: A robust clustering algorithm for categorical attributes. En *Proceedings of the 15th International Conference on Data Engineering* (págs. 512–521). Sydney, Australia.

Hair, J. F., Anderson, R. E., L., T. R. y Black, W. C. (1999). *Análisis multivariante* (5ª ed.). Madrid, España: Prentice Hall.

Heckerman, D. (1996). Bayesian networks for knowledge discovery. En *U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, Advances in Knowledge Discovery in Data Mining* (págs. 273–305). Cambridge, MA, Estados Unidos.

Hill, W., Stead, L., Rosenstein, M. y Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use. En *CHI'95: Conference Proceedings on Human Factors in Computing Systems* (págs. 194–201). Denver, CO, Estados Unidos.

Hinneburg, A., y Keim, D. A. (1998). An efficient approach to clustering in large multimedia databases with noise. En *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining* (págs. 58–65). New York, NY, Estados Unidos.

Jennings, A., y Higuchi, H. (1993). A user model neural network for a personal news service. *User Modeling and User Adapted Interaction*, *3*, 1–25.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, *32*, 241–254.

Jolliffe, I. T. (1986). *Principal component analysis.* New York, NY, Estados Unidos: Springer-Verlag.

Karypis, G., Han, E. H. y Kumar, V. (1999). CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer*, *32*, 68–75.

Kaufman, P., y Rousseeuv, P. J. (1990). *Finding groups in data: an introduction to cluster analysis.* John Wiley & Sons.

Kohonen, T. (1984). *Self-Organization and associative memory.* Nueva York, Estados Unidos: Springer-Verlag.

Kohonen, T. (1997). *Self-organizing maps* (Second ed.).

Konstan, J. A., Riedl, J., Borchers, A. y Herlocker, J. L. (1998). Recommender Systems: A GroupLens perspective. En *Recommender Systems: Papers from the 1998 Workshop (AAAI Technical Report WS-00-04)* (págs. 60–64). Menlo Park, CA, Estados Unidos.

Krulwich, B. (1997). Lifestyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data. *Artificial Intelligence Magazine*, *18*(2), 37–45.

Lagus, K. (2000). *Text mining with the websom.* Tesis doctoral, Helsinki University of Technology, Neural Networks Research Centre, Espoo, Finlandia.

Lang, K. (1995). Newsweeder: Learning to filter news. En *Proceedings of the 12th International Conference on Machine Learning* (págs. 331–339). Lake Tahoe, CA, Estados Unidos.

Lauritzen, S. L. (1995). The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, *19*, 191–201.

Linoff, G. S., y Berry, M. J. A. (2001). *Mining the web.* Nueva York, NY, Estados Unidos: John Willey & Sons.

Lisboa, P., y Patel, S. (2004). Cluster-based Visualization of Marketing Data. En *Lecture Notes in Computer Science, IDEAL 2004: Proceedings of the Fifth International Conference.*

Mannila, H., y Toivonen, H. (1994). Efficient algorithms for discovering association rules. En *Proceedings of the AAI Workshop on Knowledge Discovery in Data Bases* (págs. 181–192). Seattle, WA, Estados Unidos.

Martín, J. D. (2003). A pseudo-supervised approach to improve a recommender based on collaborative filtering. En *Lecture Notes in Artificial Intelligence, UM2003 User Modeling: Proceedings of the Ninth International Conference* (págs. 429–431). Johnstown, PA, Estados Unidos.

Martín, J. D., Balaguer, E., Camps, G., Palomares, A., Serrano, A. J., Gómez, J. y Soria, E. (2004). Machine learning methods for one-session ahead prediction of accesses to page categories. En *Lecture Notes in Computer Science, AH2004: Proceedings of the 3rd International Conference.* Eindhoven, Holanda.

Martinetz, T. M., y Schulten, K. J. (1991). A "neural-gas" network learns topologies. En *T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, editors, Artificial Neural Networks* (págs. 397–402). Amsterdam, Holanda.

Murthy, S. K. (1998). Automatic construction of decision trees from data: A multi-disciplinary survey. *Data Mining and Knowledge Discovery*, *2*, 345–389.

Pazzani, M. J. (1999). A Framework for Collaborative, content-based and Demographic Filtering. *Artificial Intelligence Review*, *13*(5–6), 393–408.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. y Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. En *Proceedings of the Conference on Computer Supported Cooperative Work* (págs. 175–186). Chapel Hill, NC, Estados Unidos.

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

Rich, E. (1979). User Modeling via Stereotypes. *Cognitive Science*, *3*, 329–354.

Ripley, B. D. (1996). *Pattern recognition and neural networks.* Cambridge University Press.

Rosenstein, M., y Lochbaum, C. (2000). Recommending from Content: Preliminary Results from an E-Commerce Experiment. En *Proceedings of CHI'00: Conference on Human Factors in Computing.* La Haya, Holanda.

Schafer, J. B., Konstan, J. y Riedl, J. (1994). Recommender Systems in E-Commerce. En *Proceedings of the Conference on Computer Supported Cooperative Work* (págs. 175–186). Chapel Hill, NC, Estados Unidos.

Schwab, I., Kobsa, A. y Koychev, I. (2001). *Learning User Interests through Positive Examples Using Content Analysis and Collaborative Filtering* (Internal Memo). St. Augustin, Alemania: GMD.

Shardanand, U., y Maes, P. (1995). Social Information Filtering: Algorithms for Automating "word of Mout". En *CHI'95: Conference Proceedings on Human Factors in Computing Systems* (págs. 210–217). Denver, CO, Estados Unidos.

Shavlik, J. W., y Dietterich, T. G. (1990). *Readings in machine learning.* Morgan Kaufmann.

Sheikholeslami, S., Chatterjee, S. y Zhang, A. (1998). A multi-resolution clustering approach for very large spatial databases. En *Proceedings of the 24th International Conference on Very Large Data Sets* (págs. 428–439). Nueva York, NY, Estados Unidos.

Strang, G. (1988). *Linear Algebra and Its Applications.* Harcourt Brace.

Su, Z., Ye-Lu, Q. Y. y Zhang, H. J. (2000). WhatNext: A prediction system for web requests using N-gram sequence models. En *WISE 2000 Proceedings: 1st International Conference on Web Information Systems Engineering* (págs. 214–221). Hong Kong, China.

Terveen, L., y Hill, W. (2001). Human Computer Collaboration in Recommender Systems. En *J. Carroll, editor, human Computer Interaction in the New Millenium* (págs. 487–509). Nueva York, NY, Estados Unidos: Addison-Wesley.

Theodoridis, S., y Koutroumbas, K. (1999). *Pattern recognition.* Academic Press.

Towle, B., y Quinn, C. (2000). Knowledge-Based Recommender Systems Using Explicit User Models. En *Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, AAAI Technical Report WS-00-04* (págs. 74–77). Menlo Park, CA, Estados Unidos.

Wang, W., Yang, J. y Muntz, R. (1997). STING: A statistical information grid approach to spatial data mining. En *Proceedings of the 23rd International Conference on Very Large Data Bases* (págs. 186–195). Atenas, Grecia.

Yates, R. D., y Goodman, D. J. (1999). *Probability and stochastic processes.* John Wiley & Sons.

Zadeh, L. (1965). Fuzzy sets. *Information and Control*, *8*, 338–353.

Zhou, D., Weston, J., Gretton, A., O., B. y Schölkopf, B. (2003). *Ranking on data manifolds* (Technical Report Nums. TR–113). Tuebingen, Alemania: Max Planck Institute for Biological Cybernetics.

Glosario de términos de uso habitual

A continuación se van a definir una serie de conceptos de uso habitual relacionados con la web. Muchos de ellos son repetidamente utilizados en esta tesis, por lo que mediante este pequeño glosario se pretende facilitar el acceso a su significado de una manera rápida en cualquier momento de la lectura de la tesis.

Acceso a servicio web: Se dice que un usuario ha accedido a un determinado servicio web cuando encontrándose en una página web que ofrece dicho servicio a través de un hipervínculo, accede al enlace correspondiente.

***Banner*:** Aviso publicitario que ocupa parte de una página de la Web, en general, ubicado en la parte superior al centro. A través de él, el navegante puede llegar hasta el sitio anunciado.

***Caché*:** Almacenamiento intermedio o temporal de información. Por ejemplo, los navegadores poseen una memoria *caché* donde almacenan las últimas páginas visitadas por el usuario, y cuando alguna de ellas es solicitada nuevamente en un corto plazo, el navegador mostrará la que tiene guardada, en lugar de volver a buscarla en la Web. El término *caché* también se utiliza para denominar a todo depósito intermedio de datos solicitados con frecuencia.

***Chat*:** Sistema para comunicarse mediante texto en tiempo real con una o varias personas que se encuentran en otros ordenadores conectados a la red.

Contraseña o *Password*: Palabra clave que se le asigna a un usuario como contraseña de seguridad para la utilización de un ordenador, de un *e-mail* o de una zona de acceso restringido en Internet. La contraseña no debe ser

visible al escribirla.

***Cookie*:** Sistema utilizado por el servidor que consiste en guardar información acerca del navegante para su posterior recuperación. La información es proporcionada desde el navegador al servidor del World Wide Web mediante un método interactivo que hace que pueda ser recuperada nuevamente cuando se acede al servidor en el futuro.

Correo electrónico o *e-mail*: Permite el intercambio de mensajes (incluyendo archivos) entre personas conectadas a una red de forma similar al correo tradicional.

Enlace o *Link*: Sirven para saltar de una página a otra, o de un sitio web a otro, cuando se navega por Internet. También se le llama hipervínculo o hiperenlace.

Descriptor o etiqueta: En inglés también suele recibir la denominación de *page categories*. Un descriptor aglutina una serie de servicios web similares bajo una denominación común. Por ejemplo, en un periódico digital, ejemplos de descriptores son "Nacional", "Internacional" o "Deportes", entre otros.

Dirección IP: Se trata de la dirección numérica de un ordenador en Internet. Cada dirección electrónica se asigna a un ordenador conectado a Internet, siendo ésta única. Las direcciones IP son del tipo 122.548.23.91

Dominio: Conjunto de ordenadores que comparten una característica común, como el estar en el mismo país, en la misma organización o en el mismo departamento. Ejemplos: .com (comercial), .org (política y asociaciones), .es (España), .uk (Reino Unido).

Episodio: Subconjunto de accesos realizados por un usuario dentro de una misma sesión.

Hipertexto: Documento que reune imágenes, textos, sonidos o vídeos relacionados entre sí por medio de enlaces, de tal modo que al señalar una palabra o gráfico se pasa de uno a otro. La WWW es una forma de usar Internet por medio de hipertextos conectados entre sí.

Host: Es el ordenador servidor que nos provee de la información que solicitamos para realizar algún procedimiento desde una aplicación cliente.

HTML: Del inglés *Hiper-Text Markup Language*, es el lenguaje con que se escriben los documentos en Internet y se realizan casi todas las webs.

HTTP: Del inglés *HyperText Transmission Protocol*, es el protocolo para transferir archivos o documentos hipertexto a través de la red. Por ejemplo: `http://www.valencia.edu/`

Internet: También llamada la "red de redes". Es un gran conjunto de redes de ordenadores interconectados cuyos usuarios pueden compartir datos, recursos y servicios. Apareció como un experimento del ministerio de defensa americano, pero su difusión se llevó a cabo en el ámbito universitario y científico.

Intranet: Se trata de una red privada dentro de una empresa u organización que utiliza el mismo software que se encuentra en Internet, pero que es sólo para uso interno y privado.

LAN *(Local Area Network)*: Red de área local, de dimensiones limitadas. Puede tratarse de ordenadores conectados en una oficina, en un edificio, o en varios.

Listas de correo o Foros de discusión: Servicio automatizado de mensajes, normalmente moderado por un propietario en el que los suscriptores reciben mensajes dejados por otros suscriptores acerca de un tema de interés común.

***Login* o nombre de usuario:** Clave de acceso que se le asigna a un usuario para que pueda utilizar los recursos de un ordenador.

Marco o *frame*: Son subunidades dentro de una página web, que ofrecen diferente información, de tal manera que desde un mismo archivo HTML, se obtiene la información que, en principio, estaría comprendida en varias páginas. Un ejemplo de una página web con marcos es `http://www.uv.es/etse`, que contiene todo lo que serían las diferentes páginas correspondientes a la *Escola Tècnica Superior d'Enginyeria* de la *Universitat de València*, como marcos de una única página.

Página web: Archivo (generalmente HTML) que constituye una unidad de información accesible a través de un programa navegador (Explorer, Mozilla). Puede ser un texto corto o un gran conjunto de textos, fotografías, gráficos estáticos o animados, sonido, etc. La página web no es el contenido global de un sitio web sino que es una parte de dicho sitio. Por ejemplo:

`http://www.cms.livjm.ac.uk/research/snc/neural.htm`.

Portal web: Espacio web que sirve de punto de partida para navegar por Internet y que, normalmente, ofrece una gran diversidad de servicios tales como listado de sitios web, buscador, noticias, e-mail, información meteorológica, chat, grupos de discusión y comercio electrónico. Ejemplos: Yahoo, Terra, o el portal web cuyos datos se han utilizado en esta tesis (Infoville XXI).

Profundidad o longitud de sesión: Es el número de diferentes servicios accedidos durante una sesión.

***Proxy* o servidor *proxy*:** También llamado intermediario o mediador, es usado en redes locales y hace referencia a un servidor que media entre el usuario (su ordenador) y otro servidor de la red. El *proxy* puede hacer, por ejemplo, un pedido de información para un cliente en lugar de que el cliente lo haga directamente.

Recomendación: Se trata de sugerencias que se ofrecen a los usuarios de un determinado sitio web acerca de los servicios que el administrador considera que le pueden interesar.

Servicio web: Es cada uno de los objetos, o informaciones, que pueden obtenerse de un sitio web. Por ejemplo, en Infoville XXI, existen más de 2.000 posibles servicios, desde callejero o envío de SMS hasta acceso virtual a la administración pública.

Servidor web: Es una máquina conectada a la red en la que se guardan físicamente las páginas web que componen un sitio web. También se conoce con este nombre al programa que sirve dichas páginas.

Sesión: Se considera como sesión todo el período que va desde que un usuario accede a un sitio web hasta que lo abandona, teniendo en cuenta los diferentes accesos a servicios web que el usuario pueda realizar.

Sitio web: Conjunto de páginas web que tiene una dirección web única. Por ejemplo: `http://www.valencia.edu/`. Erróneamente se tiende a confundir con página web.

***Tag*:** También conocido como rótulo, etiqueta o identificador. Aunque tiene otras acepciones, la principal para la presente tesis es la que considera el *tag* como el campo clave de un registro o el conjunto de bits o de caracteres que identifica diversas condiciones acerca de los datos de un archivo, y que

se encuentra, frecuentemente, en los registros de encabezamiento de tales archivos.

URL (Universal Resource Locator): En castellano significaría Identificador Universal de Recursos. Se trata del sistema unificado de identificación de recursos en la red. Ejemplos de URL: `http://www.elpais.es` o `http://gpds.uv.es`

Usuario web: Puede definirse como aquel individuo que iteractúa con la web. En particular, si nuestro interés está centrado en el conocimiento de un determinado portal web, un usuario es cada persona que interactúa con el portal, es decir, que accede al mismo.

***World Wide Web* (WWW):** Sistema de información con mecanismos de hipertexto propuesto inicialmente por investigadores del CERN. Los usuarios pueden crear, editar y visualizar documentos de hipertexto.

XML: Del inglés Extensible Markup Language. Se trata de un meta-lenguaje que permite definir lenguajes de marcado adecuados a usos determinados. Permite a diferentes aplicaciones interactuar con facilidad a través de la red. XML fue creado al amparo del World Wide Web Consortium (W3C) organismo que vela por el desarrollo de WWW. La primera definición que apareció fue: sistema para definir, validar y compartir formatos de documentos en la web.

Determinación de tendencias en un portal web utilizando técnicas no supervisadas. Aplicación a sistemas de recomendaciones basados en filtrado colaborativo.

José David Martín Guerrero, Julio 2004