Copyright Academia Journals 2015

Sistema Recomendador con Implementación "Software as a Service"

M.S.C. Christian Javier Caldera Sanchez[1], Dr. Diego Uribe Agundis[2], Dr. Enrique Cuan Durón[3], Dra. Elisa Urquizo Barraza[4]

***Resumen*— Los sistemas recomendadores brindan a las empresas u organizaciones soluciones de procesamiento y análisis de información referente a las preferencias de sus consumidores, estas preferencias son después utilizadas para predecir patrones de conducta que son aprovechadas en mejorar la experiencia del usuario, estos sistemas tienen como base la generación de perfiles de comportamiento mediante históricos de transacciones.**

Este documento presenta los resultados de desempeño obtenidos del desarrollo de un sistema de recomendación mediante transacciones y metadatos pertenecientes a los ítems, aplicando técnicas de filtros colaborativos y de *machine learning*.

El sistema será público en la web implementando la arquitectura *Software as a Service* en la cual múltiples sistemas podrán interconectarse al mismo desde cualquier punto de internet.

***Palabras clave*— Sistema recomendador, *Software as a Service*, *Cloud Computing*, Web, API.**

Introducción

Las tecnologías de información y comunicación (TICs) han aportado grandes avances permitiendo que las organizaciones brinden mayor valor a sus servicios y/o productos, sin embargo también es cierto que aún no logran hacer más eficientes todos los procesos de la industria, ya que hay procesos críticos que involucran la intervención humana; uno de estos procesos es el de la recomendación de productos o servicios mediante información del consumidor.

En los tiempos previos a los sistemas digitales, la tarea de recomendar era exclusiva de los vendedores, mismos que mediante la observación y el análisis de información generan perfiles de sus consumidores para ofrecerles los productos o servicios que más les convenían, incrementando así la posibilidad de una venta o adquisición de servicios. Sin embargo, con la llegada de internet y de los sistemas digitales, la adquisición de bienes o servicios se ha incrementado exponencialmente mediante páginas automatizadas de ventas, mismas que por su volumen y tipo de modelo de negocio carecen de un vendedor que atienda a los consumidores personalmente, eliminando así la capacidad de las empresas de generar perfiles e incremento de ventas.

Pero, ¿Cómo lograr que un servicio de TICs satisfaga este requerimiento?, la respuesta es la Inteligencia Artificial (IA), rama de las ciencias de la computación que se dedica al desarrollo de modelos que simulen la inteligencia del ser humano.

En la actualidad la IA ha sido capaz de entregar al mundo sistemas que permitan simular y/o predecir el comportamiento humano, entre ellos los sistemas recomendadores, es decir, la recomendación de servicios o productos a los consumidores. Sin embargo, esta clase de herramientas requieren gran capacidad de procesamiento y almacenamiento de información en los servidores donde radican, lo que se traduce en una fuerte inversión por parte de las empresas.

Este artículo se enfoca en las técnicas y resultados obtenidos de la producción de un sistema para agilizar el proceso de recomendación mediante una aplicación en la nube que permitirá a cualquier otro sistema basado en internet obtener patrones de recomendación sin necesidad de adquirir más poder de procesamiento o almacenamiento de datos.

Software as a Service es el nombre de este nuevo paradigma de la computación que permite a las empresas obtener servicios o productos a través de internet sin necesidad de realizar una fuerte inversión inicial.

Descripción del Método

Método de Similitud del coseno

En este documento, se abordara el método de filtros colaborativos utilizando la similitud del coseno, de la misma forma en la que fue planteada por el equipo de *Group Lens* (Badrul Sarwar et al. 2001), en el cual se señala que la

[1] El M.S.C. Christian Javier Caldera Sanchez es Maestro en Sistemas Computacionales egresado del Instituto Tecnológico de la Laguna, Torreón, Coahuila. christian.caldera@outlook.com **(autor corresponsal)**
[2] El Dr. Diego Uribe Agundis es Profesor-investigador del Posgrado en Sistemas Computacionales del Instituto Tecnológico de la Laguna, Torreón, Coahuila. diego@itlalaguna.edu.mx
[3] El Dr. Enrique Cuan Durón es Profesor-investigador del Posgrado en Sistemas Computacionales del Instituto Tecnológico de la Laguna, Torreón, Coahuila. kcuan@itlalaguna.edu.mx
[4] La Dra. Elisa Urquizo Barraza es Profesora-investigadora del Posgrado en Sistemas Computacionales del Instituto Tecnológico de la Laguna, Torreón, Coahuila. elisaurquizo@itlalaguna.edu.mx

similitud entre dos ítems i y j pueden ser medidos mediante la función del coseno (véase la Ecuación 1), Donde i y j son vectores multidimensionales formados por los atributos que los caracterizan, estos atributos pueden ser palabras claves (metadatos) asignadas por los administradores de los sistemas o calificaciones dadas por sus consumidores.

$$similitud(i,j) = \cos(\theta) = \frac{i \,.\, j}{|i| \ * |j|}$$

Ecuación 1 - Similitud del coseno

La similitud entre ambos ítems se verá dada por valores entre 0.0 y 1.0, en el cual, 1 representara que los ítems son 100% similares y 0 que son completamente diferentes.

Supóngase el caso donde dados los ítems V_a , V_b y V_c compartiendo un conjunto de atributos A los cuales se definen en la Tabla 1.

	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12
V_a	1	1	1	1	1	1	0	0	0	0	0	0
V_b	1	1	1	1	1	0	1	0	0	0	0	0
V_c	0	0	1	0	0	0	0	1	1	1	1	1

Tabla 1 - Vectores y propiedades

De esta forma se puede visualizar que si se compara el V_a con el V_b la similitud del coseno será muy alta ya que comparten varias propiedades en común, caso contrario, si se compara el V_c con el V_a o el V_b la similitud será muy baja (cercana a cero) ya que solo comparten un solo atributo en común.

En la Figura 1 se presenta el algoritmo que muestra como calcular la similitud del coseno entre dos elementos:

Entrada: Vector V_a, Vector V_b
Salida: Similitud del Coseno

1: $\mathbf{V_p} \leftarrow V_a \cup V_b$ // Vector patrón es igual a la unión de V_a y V_b
2: **si** $Longitud(V_p) == Longitud(V_a) + Longitud(V_b)$ **entonces**
3: **devolver** 0 // Si no tienen elementos en común entonces devolver cero
4: **si no**
5: $\mathbf{V_pa} \leftarrow$ Relleno de espacios para V_a // Se llenaran de ceros los atributos de V_p que no existan en V_a
6: $\mathbf{V_pb} \leftarrow$ Relleno de espacios para V_b // Se llenaran de ceros los atributos de V_p que no existan en V_b
7: **ProductoPunto** $\leftarrow V_pa \cdot V_pb$ // Calcular el producto punto de los vectores
8: $\mathbf{\hat{V}_pa} \leftarrow V_pa/\|V_pa\|$ // Normalizar el vector V_pa
9: $\mathbf{\hat{V}_pb} \leftarrow V_pb/\|V_pb\|$ // Normalizar el vector V_pb
10: **Similitud** $\leftarrow ProductoPunto/(\hat{V_p}a * \hat{V_p}b)$ // Calcular la similitud del coseno
11: **devolver Similitud**
12: **fin si**

Figura 1 - Calculo de similitud del coseno

Supóngase ahora que se desea dar un peso específico a cada una de esas propiedades como se muestra en la Tabla 2. En este caso el valor de similitud de los V_b y V_c subiría debido a que el atributo a_3 es compartido por ambos y tiene un peso superior al de los demás atributos.

User Profiles

Los usuarios también pueden ser tratados como vectores, y ellos serán definidos por los atributos de los ítems que consuman (Jure et al. 2014), por ejemplo, si el usuario consume los ítems representados por V_b y V_c su vector

seria representado por U_i como se muestra en la Tabla 3, en donde los valores se suman para dar paso al perfil del usuario.

	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12
V_a	1	1	1	1	1	5	0	0	0	0	0	0
V_b	1	1	9	1	1	0	5	0	0	0	0	0
V_c	0	0	9	0	0	0	0	1	1	1	1	1

Tabla 2 - Vectores y propiedades con pesos

	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12
U_i	1	1	18	1	1	0	5	1	1	1	1	1

Tabla 3 - Vector del usuario

Sin embargo, debido a que los usuarios no siempre tienen gustos constantes, se deberá considerar que los elementos consumidos más recientemente por el usuario serán más importantes que los elementos más viejos que definen el perfil del mismo, para ello se utilizara la fórmula de depreciación y actualización del perfil del usuario (véase Ecuación 2). De esta forma el perfil actual decae y los ítems nuevos tienen más peso, lo que generara que los ítems más viejos se vallan desvaneciendo poco a poco.

$$Perfil_{Nuevo} = Perfil_{Actual} * (0.95) + Item_{Reciente} * (0.5)$$

Ecuación 2 - Fórmula de depreciación y actualización del perfil del usuario

Mediante la similitud del coseno y la generación de perfiles, es posible implementar tres métodos de recomendación, basado en metadatos (o atributos), basado en calificaciones y un modelo hibrido que utiliza ambas técnicas de recomendación.

Recomendaciones basadas en metadatos

En la aplicación de recomendación, los ítems estarán vinculados a las aplicaciones externas mediante códigos, estos códigos deberán ser únicos para cada aplicación externa. Por otra parte, los códigos estarán vinculados a identificadores o *Id's* dentro del sistema recomendador. De tal forma que las relaciones se definirán como indica la Figura 2, donde:

Ítem— es la entidad que representara a cualquier elemento a ser recomendado o analizado, la propiedad Id es el identificador único e indexable y Código el valor de identidad que se recibirá de la aplicación externa.

Atributo— es la entidad que representara a cualquier elemento que describa al ítem, si el ítem es una canción sus atributos podrían ser artista, genero, álbum, ritmo, duración, etc..., la propiedad Id es el identificador único e indexable y Código el valor de identidad que se recibirá de la aplicación externa.

Metadato— Dado que existe una relación entre las entidades Ítem y Atributo, es necesario representar dicha relación además de una propiedad para su indexado (id) y otra que representara el peso (Peso) del atributo, esta última puede o no ser utilizada dependiendo de la naturaleza de la aplicación del tercero.

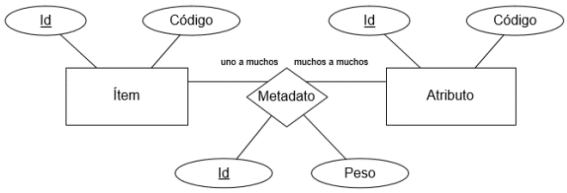

Figura 2 - UML basado en metadatos

Recomendaciones basadas en calificaciones

Para el método de recomendación basado en calificaciones, se seguirá el mismo patrón que el mostrado en la sección anterior, sin embargo, en lugar de atributos se tendrá una entidad de calificaciones y evaluadores como se muestra en la Figura 3 - UML basado en calificaciones Figura 3, donde:

Ítem— es la entidad que representara a cualquier elemento a ser recomendado o analizado, la propiedad Id es el identificador único e indexable y Código el valor de identidad que se recibirá de la aplicación externa.

Recomendador— es la entidad que representara a cualquier elemento que evalué al ítem, la propiedad Id es el identificador único e indexable y Código el valor de identidad que se recibirá de la aplicación externa.

Calificación— Dado que existe una relación entre las entidades Ítem y Atributo, es necesario representar dicha relación además de una propiedad para su indexado (id) y otra que representara la calificación dada por el usuario (Calificación) por ejemplo, si el ítem es una canción, el recomendador podrá evaluarla con estrellas con valores del 1 al 5, dicho valor será la calificación a almacenar.

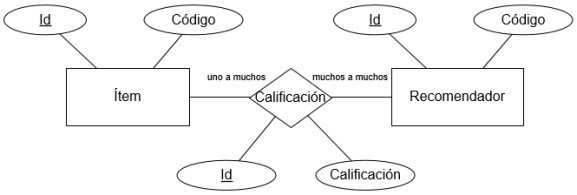

Figura 3 - UML basado en calificaciones

Recomendaciones hibridas

Existen muchos enfoques mediante los cuales abordar los métodos híbridos, algunos de ellos son presentados por Prem (Prem et al. 2010). Para esta aplicación, las recomendaciones hibridas utilizaran el mismo enfoque que las basadas en metadatos la única diferencia será que además del peso dado a cada atributo en el perfil del usuario, este valor se multiplicara por la calificación dada por el consumidor, dando apertura así a que los atributos con mayor peso y mejor calificados serán mejor posicionados en las búsquedas de recomendación.

Arquitectura de la aplicación

En esta sección, se definirá el diagrama general de las tecnologías implicadas en el desarrollo del sistema recomendador y su interacción entre ellas, facilitando así la interpretación de los procesos y transacciones a realizar, el diagrama quedara definido como se muestra en la Figura 4.

El sistema recomendador obedecerá el modelo cliente-servidor (Silvano, 1997), donde los clientes se dividirán en dos roles, los administradores de los sistemas externos que se conectaran a una interface web y las aplicaciones externas que actualizaran y consumirán los datos del sistema recomendador mediante una *Web API.* Por otra parte el servidor hospedara las interfaces de conexión de los clientes, mismas que se conectaran al sistema recomendador, y este a su vez se conectara a una base de datos con tablas cargadas en memoria, agilizando su procesamiento hasta 10 veces (Kalen, 2014), la infraestructura del servidor estará hospedada en *Windows Azure*, plataforma de procesamiento para computación en la nube (Qi et al. 2010) perteneciente a *Microsoft Corporation*.

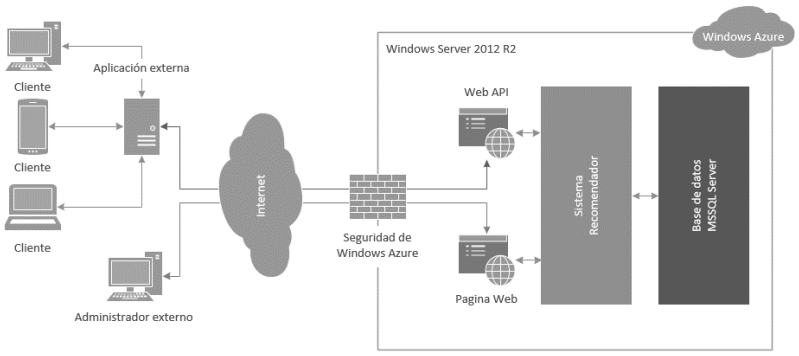

Figura 4 - Arquitectura del sistema recomendador

Dentro del sistema recomendador, se han separado las capas responsables de su funcionamiento (Simon, 2009), cada una de ellas tiene una función importante y seccionada según su aportación al mismo (véase Figura 5).

Resultados y tiempos de respuesta

Después del desarrollo del sistema recomendador se continuó con una serie de pruebas para medir su capacidad de procesamiento con grandes volúmenes de datos, así como los tiempos de respuesta entre cada operación, los resultados son presentados en la Tabla 4.

Donde cada una de las operaciones genera un tiempo menos a una operación por segundo excepto las del cálculo de similitudes que para ítems y consumidores, tardan 1 minuto y 39 segundos respectivamente, sin embargo, al ser

una operación almacenada en una fila de procesamiento, no genera tiempo de respuesta alguno para el usuario final. Siendo el alta de calificación la tarea que más demora en ser procesada con un tan solo 1.5 segundos.

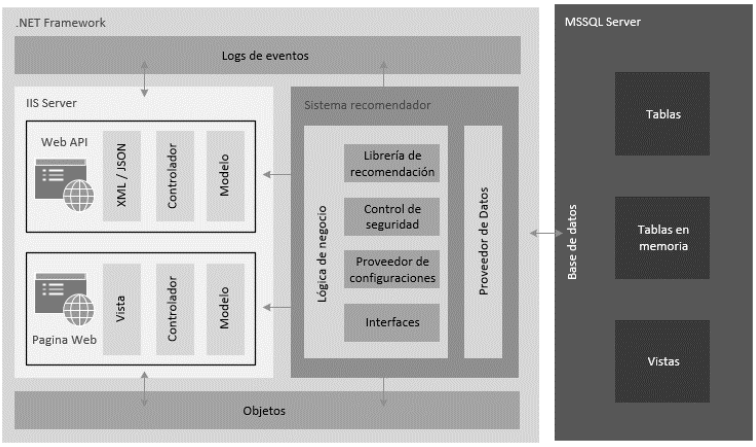

Figura 5 - Diseño del sistema recomendador

Operación	Operaciones	Tiempo total en minutos	Velocidad
Alta de ítems	765	2	6.3 operaciones / segundo
Alta de atributos	33,901	49	11.5 operaciones / segundo
Proceso de similitudes de ítems	765	1380	1 operación / minuto
Alta de consumidores	12,899	49	4.4 operaciones / segundo
Agregar calificaciones	22,998	420	0.9 operaciones / segundo
Procesar similitudes de consumidores	12,899	8352	0.65 operaciones / minuto

Tabla 4 - Tiempos de respuesta

Aplicación de demostración

Gracias a los métodos expuestos por el sistema recomendador mediante la *Web API* fue posible desarrollar una aplicación de demostración similar a *Netflix* que manejara las siguientes recomendaciones:

"Películas que pensamos podrían gustarte…"— Esta sección está basada en el método */HybridRecommendations/ForCustomer/* que provee recomendaciones para un usuario en específico basándose en su perfil, después de obtener los identificadores de los ítems a recomendar se procede a mostrarlos al usuario.

"Mejores calificadas que no has visto aun…"— Gracias al método */RatingRecommendations/GetTopRated/{id}* se pueden ofrecer al consumidor las películas mejor calificadas que aún no ha visto, estas calificaciones se basan en las evaluaciones realizadas por los usuarios del sistema y su objetivo es recomendar las mejores películas del catálogo a los usuarios sin mostrar las que ya vieron.

"Porque te gusto "x" además te comendamos…"— Esta sección utiliza el método */HybridRecommendations/ForItems* la cual muestra los top "N" ítems más similares a alguna película que el usuario califico con un valor mayor o igual a 4.

"Porque viste "x" además te recomendamos…"— Al igual que la sección anterior utiliza el método */HybridRecommendations/ForItems* sin embargo, en lugar de mostrar ítems relacionados a calificaciones altas, esta sección toma como referencia las películas más recientes que el usuario ya vio.

"Películas similares"— En cada una de las páginas de detalla de las películas se puede encontrar información adicional y al final de ella se encuentra la sección de películas similares, esto permite al usuario ver películas que son parecidas a las que ya vio y en base a ello, poder elegir una secuela o ítem que pueda llegar a ser de su agrado. Además existe un botón de comparar, que permite ver que tan similares son las películas y el porqué, como se muestra en la Figura 6.

Mediante el uso de todas estas secciones el sistema permite a los usuarios ver ítems que podrían ser de su agrado generando así un sentido de confianza y aumentando la fidelidad del usuario final con la aplicación externa.

Copyright Academia Journals 2015

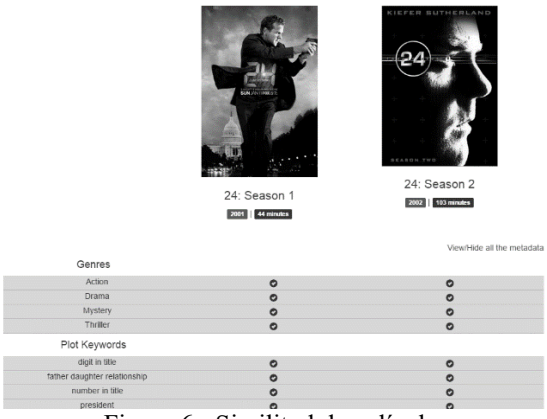

Figura 6 - Similitud de películas

Conclusiones

Como conclusión, los sistemas recomendadores son un poderoso aliado de las empresas que necesitan impulsar sus aplicaciones al siguiente nivel de fidelización, uno más personalizado con sus consumidores, mediante el ofrecimiento de elementos que les generen valor agregado.

En cuanto al desarrollo de la tesis, se ha logrado crear un sistema recomendador universal que permitirá a las PyMES en México incrementar sus ventas o simplemente el tráfico en sus sitios de una forma simple y económica.

El principal problema detectado durante el desarrollo del sistema fue que al igual que todas las aplicaciones que proveen servicios de procesamiento o almacenaje *as a service*, es el que los sistemas externos o usuarios tendrán que adaptarse a las funcionalidades que les sean ofrecidas, es decir, si una aplicación requiere una funcionalidad especial, esta no podrá ser brindada si genera un problema para la mayoría de los usuarios o si su desarrollo no es aprobado por los proveedores de servicios, un ejemplo seria, si se considera que dicha funcionalidad especial no será redituable o mantenible a largo plazo.

Recomendaciones

Los investigadores interesados en continuar esta investigación podrían concentrarse en ahondar en el uso de sistemas híbridos de más de dos algoritmos y en realizar comparaciones entre diferentes métodos de recomendación.

Referencias

Badrul Sarwar, George Karypis, Joseph Konstan y John Riedl. “Item-Based Collaborative Filtering Recommendation.”, redalyc.org (en línea), 2001, consultada por Internet el 25 de Agosto del 2015. Dirección de internet: http://recsys.cs.umn.edu/docs/Item-Based-WWW-2001.pdf.

Jure Leskovec, Anand Rajaraman y Jeffrey David Ullman. “Mining of Massive Datasets” Capitulo 9: Recommendation Systems, stanford.edu (en línea), 2014, consultada por internet el 3 de Septiembre del 2015. Dirección de internet: http://infolab.stanford.edu/~ullman/mmds/ch9.pdf.

Prem Melville y Vikas Sindhwani. “Recommender Systems”, vikas.sindhwani.org (en línea), 2010, consultada por internet el 3 de Septiembre del 2015. Dirección de internet: http://vikas.sindhwani.org/recommender.pdf.

Silvano Maffeis. “Client/Server Term Definition”, maffeis.com (en línea), 1997, consultado por internet el 3 de Septiembre del 2015. Dirección de internet: http://www.maffeis.com/articles/research/client_server.pdf.

Kalen Delaney, “SQL Server In-Memory OLTP Internals Overview”, 2014, (en linea), msdn.microsoft.com, consultada por internet el 3 de Septiembre del 2015. Dirección de internet:

Qi Zhang, Lu Cheng y Raouf Boutaba. “Cloud computing: state-of-the-art and research challenges”, http://link.springer.com/ (en linea), 2010, consultada por internet el 3 de Septiembre del 2015. Dirección de internet: http://link.springer.com/article/10.1007/s13174-010-0007-6.

Simon Malkowski, Markus Hedwig, y Calton Pu. “Experimental Evaluation of N-tier Systems: Observation and Analysis of Multi-Bottlenecks”, cc.gatech.edu (en linea), 2009, consultada por internet el 3 de Septiembre del 2015. Dirección de internet: http://www.cc.gatech.edu/systems/projects/Elba/pub/IISWC09.pdf.