UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

TIA
Tecnología, Investigación y Academia

Algoritmo SVD aplicado a los sistemas de recomendación en el comercio

SVD Algorithm Applied to Trade Recommendation Systems

Camilo Antonio Ramírez Morales[1]

Para citar este artículo: Ramírez, C. A. (2018). Algoritmo SVD aplicado a los sistemas de recomendación en el comercio. *TIA*, 6(1), pp. 18-27.

Artículo de investigación

Fecha de recepción:
05-04-2017
Fecha de aceptación:
25-07-2017

ISSN: 2344-8288
Vol. 6 No. 1
Enero - junio 2018
Bogotá-Colombia

Resumen

Se aborda la implementación del algoritmo de descomposición en valores singulares (SVD) junto con técnicas de reducción de dimensionalidad y de cálculo de mínimos, a partir del enfoque dado por Simon Funk, para la implementación en los sistemas de recomendación en el comercio. De igual manera, se realizarán pruebas de convergencia y error medio absoluto con el fin de determinar la calidad del algoritmo y su fiabilidad.

Palabras clave: descomposición matricial, Netflix price, reducción de dimensionalidad, sistemas de recomendación, SVD.

Abstract

This article discusses the implementation of the Singular Value Decomposition (SVD) algorithm along with techniques like stochastic gradient descent (SGD) and reduction of dimensionality from the approach given by Simon Funk in the recommender system for e-commerce. Likewise, tests of convergence and mean absolute error (MAE) are performed in order to check the quality and reliability of SVD.

Keywords: matrix decomposition, Netflix price, reduction of dimensionality, singular value decomposition.

[1] Tecnólogo en sistematización de datos. Estudiante de Ingeniería en Telemática, Universidad Distrital Francisco José de Caldas. Correo electrónico: camilolinchis@gmail.com

INTRODUCCIÓN

Desde el nacimiento de internet como tecnología se han abierto nuevas posibilidades que facilitan el acceso a la información, también esta se ha aumentado considerablemente. En muy poco tiempo se ha pasado de hablar de gigabytes a zettabytes. Se prevé que para 2020 se alcanzaría los volúmenes de 44 zettabytes de información almacenada [1].

Tal impacto se ha visto reflejado en todo tipo de áreas, una de ellas es el comercio, donde sitios especializados en internet ofertan millones de productos o servicios, también hay sitios donde la mayoría de datos pueden ser efímeros o de poca utilidad, produciendo un caos de información que, al contrario de satisfacer requerimientos de los usuarios, pueden complicar más su respuesta; para que sea efectiva, depende de la habilidad y experiencia que tenga el usuario para expresar sus necesidades mediante una consulta y está demostrado que en la mayoría de casos es imprecisa y vaga [2].

Por lo anterior surgen los sistemas de recuperación de información, que si bien apuntan a mejorar la forma en que se almacena, representa y organiza la información, su función no es devolver la información deseada por el usuario sino únicamente indicar qué documentos son potencialmente relevantes para dicha necesidad de información [3]. A partir de esto, surge una nueva necesidad de encontrar herramientas que se encarguen de buscar por los usuarios, que los conozcan y recomienden un conjunto limitado de opciones acordes a sus intereses.

Este es esencialmente el objetivo de los sistemas de recomendación [4]. Estos son uno de los métodos más destacados y que han ganado popularidad en los últimos años, en especial en el comercio electrónico, como sistemas inteligentes que tienen la capacidad de aprender y recopilar las preferencias, gustos o necesidades de los usuarios y, a partir de este conocimiento, brindar información limitada pero relevante que será de interés para el usuario, facilitando la digestión de información por parte del usuario.

Los sistemas de recomendación son técnicas de filtrado de información que nacen con el objetivo el objetivo de facilitar o asistir al usuario en la toma de una decisión, tal decisión en será la selección de opciones que suplirán un requerimientos [5]. En las últimas décadas se han estudiado y desarrollado diferentes tipos de técnicas que varían en términos de la información utilizada y la forma en que realizan recomendaciones [6].

Netflix, en 2007 creo un concurso llamado Netflix $1 Million Challenge [7], donde premiaba con 1 millón de dólares al mejor algoritmo de recomendación de películas que mejorará su plataforma en un 10.06%, si bien en 2009 se encontró un ganador, Netflix no uso esta implementación en su plataforma debido al esfuerzo para implementarlo, a pesar de esto otros algoritmos ganaron gran popularidad e hizo que los sistemas de recomendación tomarán fuerza. Es así como el algoritmo de *Singular Value Decomposition* (SVD) aplicado por Simon Funk [8] sea uno de los más conocidos debido a su gran exactitud, ganando notoriedad con el tiempo.

CONTENIDO

Los sistemas de recomendación

Los sistemas de recomendación son herramientas cuyo principal objetivo ayudar a los usuarios en la toma de decisiones para seleccionar un ítem que más se adecúe a sus preferencias (gustos e intereses) [9]. Un sistema de recomendación se puede definir, de manera formal, como aquel sistema que tiene como principal tarea seleccionar ciertos objetos de acuerdo con los requerimientos del usuario. Estos sistemas son muy atractivos en situaciones donde la cantidad de información que se ofrece al usuario supera ampliamente cualquier capacidad individual de exploración [10]. Se

consideran tres fases principales que se mencionan a continuación.

- Captura de preferencias: las preferencias reflejan los gustos e intereses del usuario, estas serán adquiridas a partir de la interacción del usuario con el sistema. Existen dos maneras de capturar las preferencias de los usuarios, la forma explícita e implícita. En las preferencias explícitas, el usuario conscientemente se encarga de valorar los ítems según su gustos e intereses, generalmente se suele usar una escala de puntuación como mecanismo de captura de información; por otra parte, en las preferencias implícitas, la opinión del usuario se infiere a partir del uso e interacción con el sistema.
- Extracción del conocimiento: en esta fase el sistema se encarga de interpretar la información que recopiló en la fase anterior para poder predecir gustos y preferencias del usuario.
- Recomendación: a partir del conocimiento adquirido en la fase de extracción del conocimiento, el sistema ya tiene la capacidad de seleccionar los ítems que podrían interesarle al usuario.

En la siguiente Figura 1 se resumen las fases del SR.

Técnicas de reducción de dimensionalidad

Es común encontrar que en la realidad la matriz de votaciones sea muy grande y dispersa, por esto las técnicas de reducción de la dimensionalidad tienen como objetivo transformar la matriz en una de menores dimensiones que refleje las características de la matriz original a la hora de realizar la predicción; una vez seleccionada la información, se representará con una matriz de reducidas dimensiones que represente el grado de afinidad de un usuario con las características y el grado de afinidad de un ítem a estas. Así, la idea es que dichas características representen factores latentes en la matriz original [11].

Esto permite acelerar el proceso de recomendación, pues solo habrá que considerar las características en lugar de todas las votaciones. Por otro lado, minimiza los problemas relacionados con la dispersión de la matriz y la presencia de datos erróneos [12].

Como ya se mencionó con anterioridad, el enfoque de este artículo son las técnicas de reducción de dimensionalidad de SVD, por lo tanto se tratará este tema en detalle durante las siguientes secciones.

Figura 1. Fases sistemas de recomendación

Fuente: elaboración propia.

Descomposición en valores singulares (SVD)

La descomposición singular de valores (*Singular Value Descomposition*, en adelante SVD) es una técnica de factorización de matrices que permite descomponer una matriz *A* en otras matrices *U*, *S*, *V*.

$$SVD(A) = U\ x\ S\ x\ V^t$$

Es decir, el producto matricial de *U* por *S* por V^t da como resultado la matriz *A*, con respecto a las dimensiones se parte de que la matriz *A* tendrá unas dimensiones de *nxm* donde *n* representa las filas y *m* representa las columnas. La matriz *U* será una matriz ortogonal que tendrá dimensiones de *nxn* y la matriz *V* de *mxm* también ortogonal. La matriz *S* será una matriz diagonal de dimensiones de *mxn* la cual tendrá en su diagonal lo que se denomina valores singulares, puestos en orden decreciente siendo valores mayores o iguales a cero, es decir:

$$S = \begin{bmatrix} \lambda 1 & 0 & \dots & 0 \\ 0 & \lambda 2 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & \lambda n \end{bmatrix}$$

Por lo tanto, la descomposición será igual a:

$$SVD(A)_{nxm} = U_{nxn}\ x\ S_{nxm}\ x\ V^t{}_{mxm}$$

Cabe aclarar que existe una propiedad aplicada a SVD enfocados en los sistemas de recomendación, esta consiste en que reduciendo el número de valores singulares de la matriz *S* a los primeros *k* valores, se obtendrá una aproximación de la matriz original *A*, que permite ser reconstruida a partir de las versiones reducidas de las otras matrices cometiendo un cierto error, pero disminuyendo el tamaño. Es decir:

$$SVD(A)_{nxm} \cong U_{nxk}\ x\ S_{kxk}\ x\ V^t{}_{kxm}$$

Esta importante propiedad es derivada del teorema de Eckart-Young [13] que aborda la mejor aproximación a la matriz original *A*, obteniéndola poniendo a 0 los *n* valores singulares más pequeños, así se reducirán las matrices al número de valores singulares no nulos que tenga la matriz *S*. Esto resulta entonces en la transformación de gran cantidad de datos en su representación reducida, siendo por lo tanto una propiedad muy importante que permite reducir considerablemente el tiempo de cómputo de cálculo y de uso de memoria para las tres matrices.

SVD aplicado al filtrado colaborativo

Dentro del filtrado colaborativo, es necesario el manejo de la matriz de votaciones de usuarios e ítems, por lo tanto, es posible considerar esta matriz como la base para aplicar SVD y obtener la factorización de matrices *U*, *S* y *V*. Luego, usando el teorema de Eckar-Youn, se puede reducir a *k* dimensiones. La matriz de factorización sería:

$$A_{usuarios\ x\ items} \cong U_{usuarios\ x\ k}\ x\ S_{k\ x\ k}\ x\ V^t{}_{k\ x\ items}$$

Por otro lado, es posible simplificar el proceso de SVD obteniendo solo los factores de usuarios e ítems, esto es posible descomponiendo la matriz *S* en dos matrices iguales, factores de usuarios *Ufac* y factores de ítems *Ifac* de esta manera:

$$S_{k\ x\ k} = \begin{bmatrix} \lambda 1 & & \\ & \lambda 2 & \\ & & \lambda n \end{bmatrix}$$

$$S_{k\ x\ k} = \begin{bmatrix} \sqrt{\lambda 1} & & \\ & \sqrt{\lambda 2} & \\ & & \sqrt{\lambda n} \end{bmatrix} x \begin{bmatrix} \sqrt{\lambda 1} & & \\ & \sqrt{\lambda 2} & \\ & & \sqrt{\lambda n} \end{bmatrix}$$

Por lo tanto, la matriz de votaciones se puede expresar:

$$[U_{n\,x\,k}]x\begin{bmatrix}\sqrt{\lambda 1} & & \\ & \sqrt{\lambda 2} & \\ & & \sqrt{\lambda n}\end{bmatrix}x\begin{bmatrix}\sqrt{\lambda 1} & & \\ & \sqrt{\lambda 2} & \\ & & \sqrt{\lambda n}\end{bmatrix}x[V_{k\,x\,m}]$$

Donde los factores de usuarios e ítems están representados de esta manera:

$$\begin{bmatrix} Ufact \end{bmatrix} = [U_{n\,x\,k}]x\begin{bmatrix}\sqrt{\lambda 1} & & \\ & \sqrt{\lambda 2} & \\ & & \sqrt{\lambda n}\end{bmatrix}$$

$$\begin{bmatrix} Ifact \end{bmatrix}^t = [V_{k\,x\,m}]x\begin{bmatrix}\sqrt{\lambda 1} & & \\ & \sqrt{\lambda 2} & \\ & & \sqrt{\lambda n}\end{bmatrix}$$

Por lo tanto:

$$[Votos_{n\,x\,m}] = [Ufac]\ x\ [Ifac]$$

Si bien no se tiene conocimiento claro de lo que representan los factores de usuarios e ítems, es un factor que permite predecir ítems que podrían gustar a los usuarios.

RSVD propuesta de Simon Funk

Simon Funk, en el concurso de Netflix Price, propuso un nuevo método para aproximar la factorización de matrices solucionando problemas de dispersión y escalabilidad. La idea es buscar los factores de usuarios e ítems a partir de la matriz de votación abordándolo como un problema de regresión, donde se quieren encontrar los valores de las matrices de factores.

La diferencia radica en que en este no se toma en cuenta la matriz de votos, cada voto será el producto escalar de los factores de usuarios y los factores de los ítems; así, si se considera a *qi* al vector que represente los factores de un ítem y *j* al vector que representa los factores del usuario se tendría que cumplir:

$$A_{i,j} = q_i^t \cdot p_j \cong \sum_{k=1}^{k} q_{ik}\ p_{kj}$$

Ahora solo se ajustan los factores de aquellos usuarios que hayan definido un voto reduciendo considerablemente el sistema de ecuaciones y solucionando el problema de la dispersión, cometiendo un pequeño error:

$$e_{i,j} = |\ A_{i,j} - q_i^t \cdot p_j|$$

Debido a que el error absoluto es una función complicada de tratar, se usará el error cuadrado medio:

$$(e_{i,j})^2 = (A_{i,j} - q_i^t \cdot p_j)^2 \cong (A_{i,j} - \sum_{k=1}^{k} (p_{ik} \cdot q_{kj})^2)$$

Por lo tanto, lo que plantea Simon Funk es encontrar el error mínimo para hacer la mejor aproximación de la siguiente forma:

$$Min_{p^*,q^*} = \sum\ (A_{i,j} - q_i^t \cdot p_j)^2$$

Es decir, aquella que minimice todos los errores que se pueden cometer al calcular los factores de los usuarios y de los ítems, pero es necesario hacer una regularización para evitar *Outfitting* [14] (exceso especialización), como se muestra en la Figura 2.

Así, es posible llegar a:

$$Min_{p^*,q^*} = \sum\ (A_{i,j} - q_i^t \cdot p_j)^2 + \frac{\lambda}{2}(||q_i^2|| + ||p_j^2||)$$

Dando paso a una nueva manera de ver el error:

$$(e_{i,j})^2 = (A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj})^2 + \frac{\lambda}{2}(q_i^2 + p_j^2)(1)$$

Donde λ es una constante de regularización.

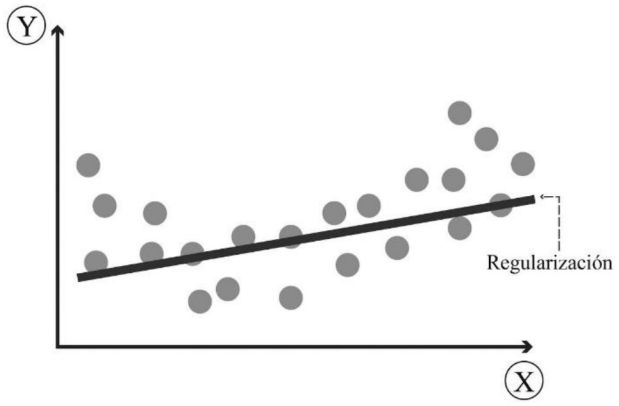

Figura 2. Ejemplo de regularización

Fuente elaboración propia.

Descenso del gradiente estocástico SGD aplicado a la minimización del error regulado [15]

Para encontrar el mínimo de la función ya detallada existen técnicas como la del descenso del gradiente estocástico (SGD, *stochastic gradient descent*) que buscan ajustar las matrices de factores poco a poco de manera automática. Esto se realizará en la función (1), con el fin de encontrar el mínimo de la función, o acercarse lo suficiente a la solución.

La técnica del SGD consiste en ir moviéndose poco a poco por la función (de *n* dimensiones) hasta encontrar el mínimo evaluando, la función en un punto dado para luego moverse una distancia hacia otro punto; ahora se calcula la derivada en ese punto y se desplaza al lado opuesto de la derivada de la función que se quiere minimizar. En dos dimensiones un ejemplo se vería como el de la Figura 3.

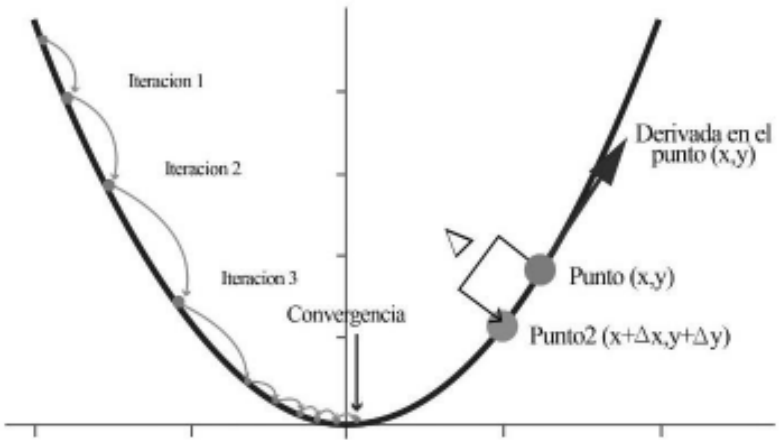

Figura 3. Ejemplo SGD

Fuente: elaboración propia.

Aplicando SGD en (1) se tiene:

$$\frac{\partial (e_{i,j})^2}{\partial\, p_{ik}} = 2\left(A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj}\right)(-q_{kj}) + \frac{\lambda}{2}(2\, p_{ik})$$

$$\frac{\partial (e_{i,j})^2}{\partial\, q_{kj}} = 2\left(A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj}\right)(-p_{ik}) + \frac{\lambda}{2}(2\, q_{kj})$$

Simplificando:

$$\frac{\partial (e_{i,j})^2}{\partial\, p_{ik}} = -2\left(A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj}\right)(q_{kj}) + \lambda(p_{ik}) \qquad (2)$$

$$\frac{\partial (e_{i,j})^2}{\partial\, q_{kj}} = -2\left(A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj}\right)(p_{ik}) + \lambda(q_{kj}) \qquad (3)$$

Teniendo ya el gradiente, solo queda aplicar las reglas de actualización tanto para p_{ik} como para q_{kj} (2), (3) con las constantes de regularización y aprendizaje. Así, la regla de actualización parte de $\sigma_{n+1} = \sigma_n - \alpha\, \nabla f(x)$, donde σ_{n+1} es el nuevo valor en la matriz de votaciones, σ_n su valor actual, α constante de aprendizaje y $\nabla f(x)$ es el gradiente ya obtenido. Dando como resultado:

Para p_{ik}

$$p'_{ik} = p_{ik} + \alpha\left(2\left(A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj}\right)(q_{kj}) - \lambda(p_{ik})\right) \qquad (4)$$

Para q_{kj}

$$q'_{kj} = q_{kj} - \alpha + \left(2\left(A_{i,j} - \sum_{k=1}^{k} p_{ik} \cdot q_{kj}\right)(p_{ik}) - \lambda(q_{kj})\right) \qquad (5)$$

El enfoque se plasma en las expresiones (4) y (5) que serán las que actualizarán los valores de la matriz de votaciones a partir de cada *Ephochs*, tanto para los factores de usuarios, como para los factores de ítems.

Implementación en Python [16][2]

Es necesario aclarar que antes de pasar las matrices *Q* y *P* (usuarios e ítems) es indispensable rellenarlas con valores aleatorios. Por otro lado, cabe mencionar que se usó la librería de *numpy* [17] para las multiplicaciones de matrices usando la función *dot*, esto permite que por cada *Ephochs* se ajusten dos factores, uno de usuarios y otro de los ítems, solo se realiza en los votos emitidos por los usuarios.

Pruebas

Al realizar la prueba es necesario contar con una cantidad considerable de datos confiables, para recolectar esta información se necesita conocer las preferencias de gran cantidad de usuarios respecto a distintos ítems, por lo tanto se usarán una de las bases de datos proporcionada por el grupo de investigación GropuLens [18] de la universidad de Minnesota, tal base de datos *Movielens1M* consiste de aproximadamente un millón de votos proporcionados por 6040 usuarios para 3900 películas, con puntajes entre uno y cinco.

Error medio absoluto (MAE) [19]

Esta prueba consistirá en definir la fiabilidad de las recomendaciones, es decir, conocer qué tan exacta serán sus predicciones, para ello se usará el *dateset* de Movielens, donde el 80% será usado como datos de entrenamiento (*Trainingtest*) y el resto como datos de prueba (*Datetest*), la idea es predecir los datos del conjunto de pruebas considerando únicamente los conjuntos de entrenamiento como datos de entrada. Por último, se comparan los datos obtenidos con los reales, para esto se usará una métrica muy conocida en los sistemas de recomendación llamada MAE (*Mean Absolute Error*) que mide la desviación media entre el valor de predicción real que el usuario asignó al ítem.

La fórmula de cálculo para el MAE es la siguiente:

$$\overline{|E|} = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N}$$

2 La implementación en el lenguaje de programación Python, puede ser vista desde este enlace: https://gist.github.com/camilortte/ebfa6d5a69a57db0a9d51bfa17e5092a

Donde:

- $|E|$= Valor absoluto promedio del error.
- p_i= Puntaje de predicción del algoritmo sobre el ítem *i*.
- r_i= Puntaje real del usuario sobre el ítem *i*.
- N= Total de elementos considerados para el cálculo de MAE.

Los resultados se muestran en la Figura 4.

Capacidad de recomendación (*coverage*) [20]

El *coverage* (cobertura) es simplemente el porcentaje de elementos para los cuales el sistema podrá generar una predicción; para este caso se puede decir que SVD tiene una capacidad de recomendación del 100%, siempre y cuando se obtengan las matrices de factores de usuarios e ítems, siendo una de las más grandes ventajas de este tipo de algoritmos.

Para esta prueba se usó un computador portátil con Ubuntu 14.04 y con el siguiente *hardware*:

- Procesador Intel Core i7 36-100M de 2.3GH.
- Memoria RAM de 8 Gb.

En este caso se usaron varias bases de datos de *Movielens*, la de 100k y la de 1M. Los resultados obtenidos se presentan en la Figura 5.

El promedio de tiempo para la ejecución de entrenamiento según la base de datos de Movielens100k fue de aproximadamente 2,2 segundos, mientras que con la base de datos de Movielens1M fue de 26,8 segundos aproximadamente.

Figura 4. Resultados error medio absoluto

Fuente: elaboración propia.

Por lo tanto, el MAE global es de 0,71847.

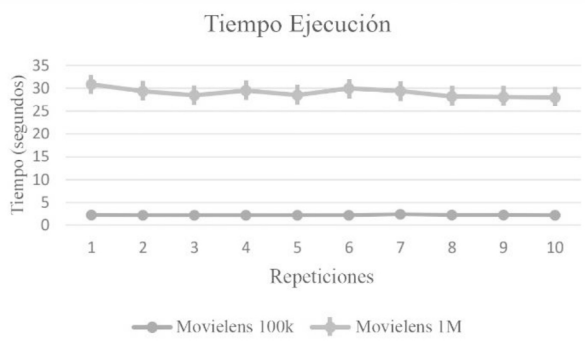

Figura 5. Resultados tiempo de ejecución

Fuente: elaboración propia.

CONCLUSIONES

En los sistemas de recomendación del filtrado colaborativo, el método de SVD genera muy buenas recomendaciones, pero en relación con el tiempo del cálculo de predicciones lo hacen una técnica poco atractiva en los entornos comerciales, es por lo tanto un objetivo pendiente, mejorar los tiempos de ejecución en la fase de predicción.

RECOMENDACIONES

Es posible mejorar los tiempos de respuesta mediante el uso de procesamiento distribuido y el uso de algoritmos de descomposición que permitan ejecutarse en paralelo y que sean compatibles con técnicas de *MapReduce*.

El algoritmo implementado tiene una complejidad muy alta, por lo tanto se sugiere usar la implementación de PyRecsys [21] creado por Oscar Celma, el cual tiene la implementación del SVD y con ciertas mejoras si se quiere llegar a usar en ambientes de producción.

REFERENCIAS

[1] Parnell, B. (2015). *De terabytes a zettabytes: el crecimiento de los datos mundiale*. Recuperado de http://www.think-progress.com/es/tendencias/de-terabytes-a-zettabytes-el-crecimiento-de-los-datos-mundiales/

[2] Rodríguez, M. (2005). *Modelos de recuperación de información basados en información lingüística difusa y algoritmos evolutivos, mejorando la representación de las necesidades de la información*. (Tesis doctoral). Granada: Universidad de Granada.

[3] G. LYS. (s.f.). *Introducción a la recuperación de información*. Recuperado de http://www.grupolys.org/docencia/ln/biblioteca/ir.pdf

[4] García, R. (2013). *SVD Aplicado a sistemas de recomendación basados en filtrado colaborativo*. (Tesis de maestría). Madrid: Universidad Politécnica de Madrid.

[5] Garxía, F. y Gil, A. (s.f.). *Personalización de Sistemas de Recomendación*. Salamanca: Universidad de Salamanca.

[6] Torres, N. (2015). Sistemas de recomendación basados en métodos de filtrado colaborativo. Santiago de Chile: Universidad Técnica Federico Santa Maóa.

[7] Verona, F. (s.f.). *Fran Verona*. Recuperado de http://franverona.com/blog/el-dia-que-netflix-rechazo-integrar-el-algoritmo-de-1-millon-de-dolares/

[8] Percy, M. (2009). *Collaborative Filtering for Netflix*. Santa Cruz: Jack Baskin School of Engineering, Santa Cruz.

[9] Castro, J. (2012). *Un nuevo modelo ponderado para Sistemas de Recomendación Basados en Contenido con medidas de contingencia y entropía*. (Trabajo tutelado). Jaén: Universidad de Jaén.

[10] Campos, L., Fernández, J., Huete J. y Rueda, M. Uso de conocimiento estructurado en un sistema de recomendación basado en Contenido. Granada: Universidad de Granada.

[11] Jiménez, A. Murillo, A., Piza, E., Villalobos, M. y Trejos, J. (2010). *Interviewees, Reducción de la dimensionalidad en análisis de datos*. Recuperado de http://www.stat.rice.edu/~jrojo/PASI/lectures/Costa%20rica/2b_Ejemplos_ACP.pdf

[12] Pepa, S. (2014). *Suite de algoritmos de recomendación de aplicaciones reales*. (Trabajo de grado). Madrid: Universidad Autonoma de Madrid.

[13] Chipman, J. S. (s.f.). Multicollinearity and reduced-ranked estimation. En: Lectures in Econometric Theory. Minneapolis: University of Minnesota. Recuperado de http://users.econ.umn.edu/~jchipman/econ8211f05/ECTBK_3.pdf

[14] Correa, D. (2015). *Regresión no lineal, Cross-Validation y Regularization*. Recuperado de https://dlegorreta.wordpress.com/2015/03/17/regresion-no-lineal-cross-validation-y-regularization/

[15] Andrew. Ng. (s.f.) Stochastic Gradient Descent. [Video]. De la lección "Large Scale Machine Learning". *Coursera*. Recuperado de https://

es.coursera.org/learn/machine-learning/lecture/DoRHJ/stochastic-gradient-descent

[16] Python. (S.f.). Python. Recuperado de https://www.python.org/about/

[17] Numpy. (S.f.). Numpy. Recuperado de http://www.numpy.org/

[18] Grouplens. (s.f.). Grouplens. Universidad de Minnesota. Recuperado de http://grouplens.org/datasets/movielens/

[19] Betarte, L., Machado, R. y Molina, V. (2006). *PGmúsica Sistema de recomendación de música*. (Trabajo de grado). Uruguay: Universidad de la República.

[20] Galan, S. (2007). Filtrado colaborativo y sistemas de recomendación. Madrid: Universidad Carlos III de Madrid.

[21] O. Celma. Pyrecsys. (s.f.). Ocelma. Recuperado de http://ocelma.net/software/python-recsys/build/html/index.html