Study of E-commerce Recomender Systems based on Implicit Feedback

Carlos Álvarez
Pontificia Universidad Católica
ctalvarez@uc.cl

Iván Wolf
Pontificia Universidad Católica
iiwolf@uc.cl

ABSTRACT

El siguiente trabajo tiene como objetivo probar y evaluar distintos métodos para resolver el problema de recomendar productos a usuarios en el contexto de un *e-commerce* real. Se desarrollan 3 modelos, uno basado en ALS, otro en una red neuronal recurrente y el último, que obtuvo los mejores resultados, en *Factorization Machines*.

El dataset sobre el cual se recomienda consta de 4 meses de registros de interacciones de usuarios en un portal. Se cuenta con la información de las vistas, adiciones al carro y transacciones que realizan los visitantes con los productos. Todos los datos cuentan con un timestamp. En promedio, hay 1.4 entradas por usuario por lo que es necesario hacer un trabajo previo al dataset para poder definir los experimentos y entrenar los modelos.

Se definen 3 experimentos y se calculan las métricas de P@10, R@10 para evaluar el desempeño de los modelos.

KEYWORDS

Recommender System, Implicit Feedback, ALS, Rrecurrent Neural Networks, Factorization Machines, Bayesian Personalized Recommendation

1. INTRODUCCIÓN

La industria del comercio electrónico va en alza, y para cada negocio es importante diferenciarse de su competencia. Para esto el uso de sistemas recomendadores se transforma en una herramienta relevante para mejorar la experiencia del usuario. Estos permiten disminuir el tiempo de búsqueda, mostrar al usuario productos relevantes para él, que no necesariamente sabía de su existencia, y mostrar complementos para los productos comprados.

Al no tener información explícita de los gustos de cada usuario, una solución es utilizar la información implícita que este entrega a través del uso de la página web. En este *paper* se estudiará tres modelos distintos para crear un sistema recomendador que utilice información de las acciones (vista, añadir al carro, transacción) del usuario en la página web.

El *dataset* utilizado para probar los modelos es el *dataset* de *Retail Rocket* publicado en Kaggle.com. Contiene información de 4 meses de todas las acciones de usuarios hechas a través de la página. Las acciones pueden ser ver un ítem, añadir un ítem al carrito y hacer una transacción de un item. Además tiene las propiedades variables de los ítem a través del tiempo y las categorías y subcategorías de estos.

2. DATASET

Primero se detalla el origen del dataset y su contenido. Luego se muestra un análisis exploratorio del dataset sobre los árboles de categorías y los eventos de vistas, añadidas al carro y transacciones que ocurren en el tiempo. Finalmente se muestra el pre procesamiento del dataset para eliminar el ruido innecesario y los criterios de selección de items y usuarios a utilizar para el entrenamiento de los sistemans recomendadores.

2.1. Origen del Dataset

El dataset tiene su origen en datos de *Retail Rocket*, un e-commerce del mundo real cuyos datos de logs de eventos están publicados en la página www.kaggle.com. Contiene datos de 2,756,505 acciones que 1,407,580 visitantes han efectuado sobre 235,061 items en su página web a lo largo de un periodo de 4 meses. Los eventos son *view*, *addtocart* o *transaction*. Además existe un archivo que contiene todas las categorías de los ítems y categorías padres de estos, y también las propiedades de cada ítem con sus variaciones en el tiempo. Las categorías y propiedades de los ítems están encriptadas con un proceso de stemming, y no se puede saber qué son concretamente.

2.2. Categorías de Productos

Se identificaron 25 árboles de categorías, con 1669 categorías en total. El factor de ramificación promedio de cada árbol es 11.15, la altura promedio es de 3.34, y la altura máxima promedio es de 3.96. Se encontró que los tamaños de los árboles en cuanto a factor de ramificación y cantidad de nodos puede ser muy variables. Además existen 1307 hojas, es decir, 1307 categorías que no tienen subcategorías. A continuación se muestran 3 ejemplos de de los 25 árboles encontrados.

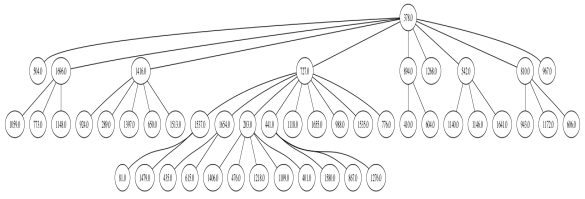

Figura 1: Árbol de categorías número 6

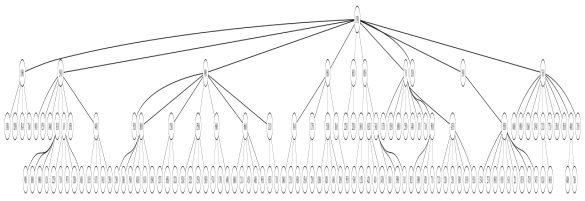

Figura 2: Árbol de categorías número 7

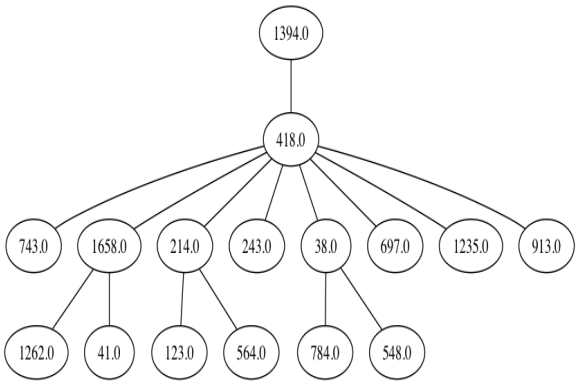

Figura 3: Árbol de categorías número 8

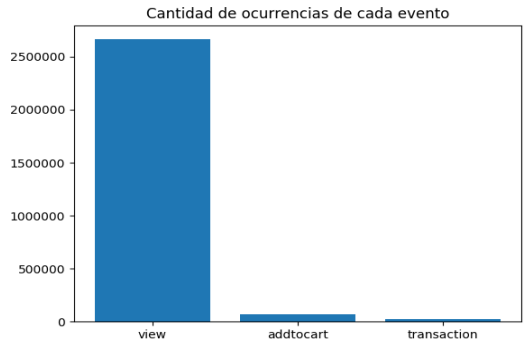

Figura 4: Cantidad de ocurrencias por evento

2.3. Análisis del Comportamiento de los Usuarios

Se analiza como es el comportamiento de los usuarios con respecto a las acciones que hacen a través del tiempo para descubrir patrones en el uso de la página, estacionalidad o algún factor de negocio relevante. El objetivo es poder intuir y dar a entender al lector qué tipo de dataset se está usando para que en trabajos futuros tenga una referencia de los resultados obtenidos a partir del trabajo realizado sobre este set de datos.

La gran mayoría de los usuarios ven pocas veces un ítem antes de comprarlo por primera vez, en promedio 1.59 veces con una desviación estandar de 1.245. También se observa que algunos usuarios ven más de 15 veces un ítem antes de comprarlo por primera vez. Esto da un indicio del comportamiento natural que tienen los usuarios en una página, la mayoría compra con decisión realizando pocas visitas al ítem antes de comprarlo, en cambio pocos deben ver el ítem muchas veces, es decir tienen cierto nivel de indecisión sobre si comprar o no.

Finalmente se analiza el tiempo en días entre la primera y la última sesión de usuarios con al menos 10 acciones. También se analiza las acciones sobre ítems que tienes más de 3, más de 10 o más de 30 acciones respectivamente. Se identifica que la mayoría de los usuarios utilizan la plataforma en un periodo de 1 día, y la minoría son usuarios recurrentes. También se aprecia que algunos ítems, independiente de su popularidad (cantidad de acciones), poseen estacionalidad dentro del periodo de tiempo en que se grabaron los datos y que la mayoría no poseen esta característica. Esto puede ser clave al momento de construir un modelo que incorpore la variable de tiempo al sistema recomendador.

2.4. Usuarios, Items y Eventos

Se analizó el dataset de los eventos, que contiene 2,756,505 de entradas, 1,407,580 visitantes únicos y 235,061 ítems únicos. El resumen de la cantidad de *views*, *addtocarts* y *transactions* se muestra en el siguiente gráfico.

Se realizó un filtrado preliminar del dataset para eliminar el ruido producido por los usuarios e items que no entregan información relevante para el análisis. Este consistió en analizar la distribución de eventos por usuarios, eventos por items y eliminar los datos no relevantes. A continuación se detalla el procedimiento.

Primero se analizó la cantidad de views, addtocart y transactions de cada usuario e ítem que correspondía a cada percentil en la cantidad total de acciones realizadas. Se vio en mayor detalle la cantidad de outliers de las distribuciones anteriores y se decidió eliminar los usuarios e ítems que se escapaban de la normalidad, y se decidió cortar el dataset en el precentil .999 para los usuarios y no cortar para los ítems.

Viendo la distribución de cantidad de acciones sobre usuarios e ítems se decidió filtrar el dataset eliminando usuarios que tengan menos de 10 acciones, y eliminar ítems con menos de 5 vistas. Además se construyeron 3 dataset distintos, uno con usuarios que tengan al menos 10 diez acciones, otro con usuarios que tengan al menos 20 acciones y otro con usuarios que tengan al menos 30 acciones. La finalidad de esto es ver la sensibilidad del modelo frente a los cambios de la cantidad de eventos, cambios en la proporción de cantidad de usuarios / ítems en el dataset en el que se basa el sistema recomendador. En la siguiente figura se muestra la cantidad de usuarios únicos e ítems únicos en cada dataset. También se muestra la cantidad de eventos totales en cada dataset.

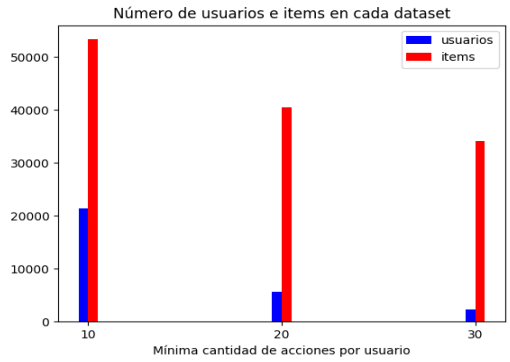

Figura 5: Cantidad de usuarios únicos e ítems únicos en los dataset para evaluación del modelo

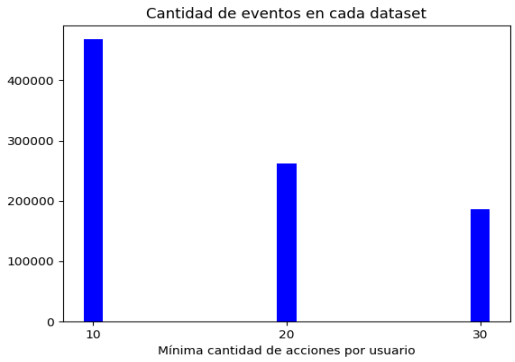

Figura 6: Cantidad de eventos totales en cada dataset

La cantidad de usuarios, ítems y eventos finales luego del filtrado de datos en cada dataset es la siguiente. En el datset de usuarios con al menos 10 acciones hay 21,469 usuarios, 53,342 ítems únicos y 427,215 eventos. En el de al menos 20 acciones hay 5,605 usuarios, 40,509 ítems únicos, y 262,436 eventos. Finalmente en el set de datos con usuarios que tienen al menos 30 acciones en la página hay 2,361 usuarios, 34,110 ítems y 165,125 eventos. Se apreció que la procporción de usuarios ítems en cada dataset va cambiando ya que hay cada vez menos usuarios, sin embargo la cantidad de ítems baja menos proporcionalmente. Esto indica que estamos recomendando sobre un gran universo de ítems, lo que da la posibilidad de que haya mayor diversidad en la recomendación.

3. EXPERIMENTOS

Para probar el rendimiento de los 3 algoritmos de recomendación se diseñaron 3 experimentos. El objetivo es, para cada usuario, recomendar 10 ítem que son los con mayor probabilidad de que el usuario interactúe con ellos. Se evalúan las métricas de P@10 y R@10. Se considera como un acierto si es que el usuario en el conjunto de test interactúa con el ítem, i.e, hay una transacción, adición al carro o vista.

- **EX1**. Primeros 3 meses de datos para entrenar y el último mes para probar.
- **EX2**. Para cada usuario, los últimos 10 eventos se usaron para el test.
- **EX3**. Para cada usuario, se hizo test con los últimos 5 ítem.

4. MODELOS

4.1. ALS Model

El primer modelo probado es en base al método ALS [1]. Se debe modelar el nivel de confianza que indique que tan seguro estamos de que al usuario u le gusta el ítem i. Primero se define la función que representa el dato observado $r_{u,i}$.

$$r_{u,i} = a \cdot views_{u,i} + b \cdot atcu, i + transactions_{u,i}$$

Donde a, b, c son reales positivos. Cada entrada de la matriz de confianza se calcula como

$$c_{u,i} = \alpha \cdot \log(q + r_{u,i})$$

.

Se probaron distintos valores de α y f (número de factores) contra el primer experimento **EX1**. Los resultados nos indican que el problema a tratar es difícil.

Model	Precision	Recall
$f = 50, \alpha = 20$	0.000622	0,003628
$f = 50, \alpha = 40$	0.000544	0.002851
$f = 50, \alpha = 50$	0.005440	0.003369
$f = 100, \alpha = 20$	0.008550	0.005201
$f = 100, \alpha = 40$	0.001010	0.006000
$f = 100, \alpha = 50$	0.000622	0.003921
$f = 200, \alpha = 20$	0.000933	0.006566
$f = 200, \alpha = 40$	0.000933	0.007128
$f = 200, \alpha = 50$	0.000933	0.006566

Cuadro 1: Resultados de ALS en el experimento EX1.

4.2. RNN

Para aprovechar la componente temporal y secuencial de los datos, se decidió implementar un modelo basado en una red neuronal recurrente. La arquitectura de la red se basa en la propuesta por [4].

El input de la red es la concatenación de los vectores *one hot* de la tupla $(usuario, accin, item)$. El output es un vector dimensión igual a la cantidad de items.

Después de que la red es entrenada, el elemento j del output $o(t+1)$, $o_j(t)$, se define como la estimación de la probabilidad de que el usuario u acceda al item j en el siguiente timestamp, dado el feedback histórico. Esto es

$$o_j(t+1) = P(v_j(t+1) = 1 | u, v(t), a(t), s(t1))$$

Para generar recomendaciones, se calcula el output del último timestamp para cada usuario. Se eligen los 10 elementos más grandes del output y los inidices de esos elementos son los ID de los ítem recomendados.

El modelo aprende mediante el algoritmo propuesto por los autores, *back propagation through time*. Para cada usuario u, se ordenan las tuplas $(u, a(t), v(t))$ por timestamp t. La parte recurrente se *desarrolla* T veces (cuando $T = 0$, la red es una red *feed foward*) y la dimensión de las capas intermedias X, W, V, Q, Y, Z es D. Se probaron distintos modelos sobre el experimento **EX2**, para valores diferentes de D y T.

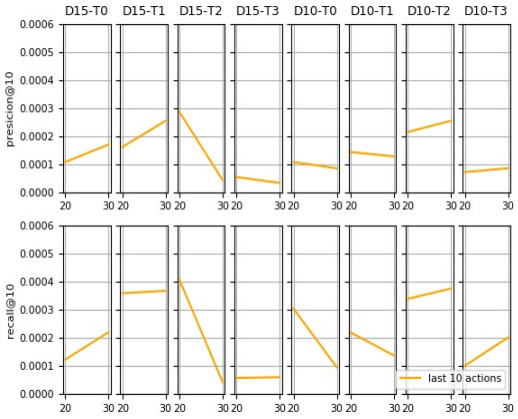

Figura 7: Resultados modelo RNN basado sobre el experimento 2

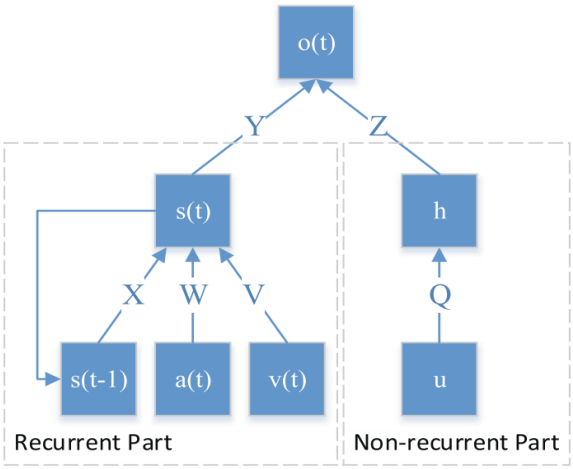

Figura 8: Arquitectura de la RNN

4.3. Factorization Machine

Para la construcción del modelo basado en máquinas de factorización se utilizaron los papers de Steffen Rendle et. al. (2009) y Steffen Rendle (2010) [3] [2]. La implementación computacional se hizo con la librería *lightfm* de python. Primero se definió una función de preferencia para representar la relación entre cada usuario y cada ítem. Se utilizó la siguiente función de preferencia.

$$pref(u,i) = \begin{cases} 1 & addtocart \text{ o } transaction \\ \min(log_5(views+1), 1) & \text{en otro caso} \end{cases}$$

Esta función se eligió empíricamente a partir del análisis preliminar del dataset. Significa que indica que si un usuario ha añadido un ítem al carrito, o lo ha comprado, tiene preferencia igual 1 por ese ítem. Si no ha hecho alguna de esas acciones sobre ese ítem, se dice que si el usuario ha visto 4 veces o más un ítem tiene preferencia 1 por ese item, y si lo ha visto entre 0 y 3 veces, tiene preferencia $log_5(views+1)$ por el ítem. Esto se hizo así debido a que un usuario puede ver un ítem por accidente o para compararlo con otro, pero no necesariamente indica una preferencia hacia este.

Una vez pre procesado el dataset con las preferencias para cada tupla (usuario, item), se entrena el modelo basado en *Bayesian Personalized Recommendation* que lo que busca es generar una lista ordenda de recomendación para cada usuario basado en *pair wise*. De esta manera el objetivo del algoritmo es dar un mejor ranking (recomendación) en lugar de dar una mejor precisión para cada ítem.

Se realizan el experimento 2 (últimas 10 acciones, naranjo) y el experimento 3 (últimos 5 ítems, azul) para evaluar el modelo. El eje x de cada gráfico representa la cantidad de acciones mínimas por cada usuario en el dataset (10, 20, 30) dado el pre proceso mencionado en la sección 2.

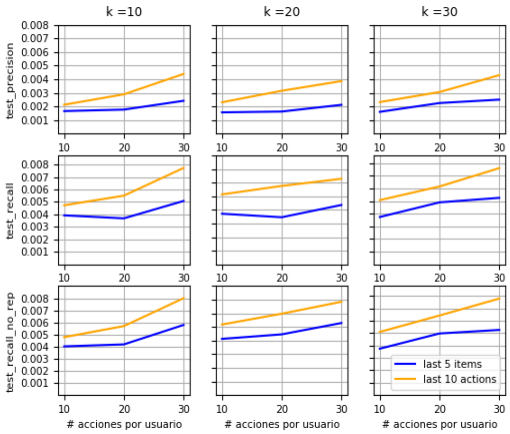

Figura 9: Resultados modelo FM basado en BPR

Este modelo obtuvo resultados de un orden de magnitud mayores comparado con los otros dos. Se observa que a medida que el hiperparámetro k crece mejora el rendimiento. Se probó hasta $k = 30$, pero podría seguir mejorando para un k mayor. Además se muestra que en general a medida que la cantidad de usuarios es menor en el dataset, el rendimiento de recomendación mejora. Esto podría ser debido a que en los dataset más pequeños se tiene mayor información sobre cada usuario, ya que han realizado un mayor número de acciones. Sin embargo, el dataset de mínimo 10 acciones por usuario le da mayor robustez al modelo debido a que puede ser usado para recomendar a usuario de los que se tiene menos información, que son la gran mayoría.

5. CONCLUSIONES Y TRABAJO FUTURO

Se mostró los rendimientos en la recomendación de ítems en un e-commerce utilizando el feedback implícito por parte de los usuarios. Se describió el contexto del dataset, su escasez de datos y el contenido de este. Luego se propuso una metodología de análisis para trabajos futuros sobre este tipo de datasets y los resultados

que cada modelo obtuvo para entregar un punto de inicio a un investigador que esté tratando con este tipo de problema.

Usar la técnica *Bayesian Personalized Recommendation* mejoró la calidad de las recomendaciones del modelo FM. El cúal fue el mejor de los tres. Esto es porque el algoritmo se basa en *pair wise*, que tiene como objetivo dar una mejor recomendación en lugar de una mejor precisión.

Los resultados obtenidos no son buenos en general, esto se debe a que la matriz usuario-item es extremadamente rala, aproximadamente dos entradas por usuario. Otro factor que afecta al rendimiento es la elección de los experimentos para cada modelo, ya que el set de ejepmlos positivos para cada test es muy escazo para cada usuario, por lo tanto .

El algoritmo de Factorization Machines fexistirá un mal rendimiento en generalue el que tuvo mejor rendimiento, a pesar de que solo se le entregó información sobre la preferencia de un usuario sobre un ítem. Como trabajo futuro se propone agregar información sobre el contexto de las acciones e ítems. Ya sea el tiempo (*timestamp*) en que se realizó cada acción o la variabilidad de las propiedades de los ítem en el tiempo. Por ejemplo, el precio o descuento de un ítem podría afectar la preferencia de un usuario hacia este.

REFERENCIAS

[1] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 263–272.

[2] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[3] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[4] Caihua Wu, Junwei Wang, Juntao Liu, and Wenyu Liu. 2016. Recurrent neural network based recommendation for time heterogeneous feedback. *Knowledge-Based Systems* 109 (2016), 90–103.