UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Máster Universitario en Investigación e Innovación en Inteligencia Computacional y Sistemas Interactivos

TRABAJO FIN DE MÁSTER

Soluciones multi-brazo en el ciclo de vida de un sistema de recomendación.

Autor: Marcos Redondo Almagro
Tutor: Pablo Castells Azpilicueta

Septiembre 2020

Marcos Redondo Almagro
Soluciones multi-brazo en el ciclo de vida de un sistema de recomendación.

Marcos Redondo Almagro

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

A mi familia y amigos.

”Las cosas no tiene que cambiar al mundo
para ser importantes”

Steve Jobs

Agradecimientos

Me gustaría agradecer a Pablo la oportunidad de trabajar con él y la ayuda prestada durante la realización del trabajo.

También hacer una mención especial a mi familia y amigos, ya que su apoyo ha sido muy importante para mí durante estos meses tan complicados debido a la situación actual y al COVID-19.

Resumen

Cada día crece más la cantidad de información que utilizan todos los sistemas inteligentes existentes en la actualidad. Entre todos esos sistemas se puede observar cómo los sistemas de recomendación han adquirido mucha importancia en los últimos tiempos con aplicaciones como Netflix o Spotify, imprescindibles en la vida de muchas personas.

El trabajo se enmarca por tanto en el ámbito de los sistemas de recomendación centrándose en la evaluación de dichos sistemas. Más exactamente mediante esta evaluación se quiere comprobar si el uso de soluciones multi-brazo en el ciclo de vida de estos sistemas produce mejores resultados que los métodos tradicionales.

Dichas soluciones multi-brazo, a diferencia de lo que se viene haciendo, donde cada uno de los brazos es un ítem, tendrán en cada uno de sus brazos un recomendador distinto, de modo que hará las veces de ensemble de recomendadores. De este modo, para cada recomendación, se escoge según la estrategia utilizada un único recomendador que será el que devuelva la recomendación.

Con ello se van a plantear diferentes ciclos de vida para la evaluación que serán analizados de forma comparativa, siendo estos una evaluación offline periódica, tests A/B con una fracción fija de tráfico, tests A/B basados en bandidos multi-brazo con una fracción fija de tráfico y una evaluación típica multi-brazo.

Para todas las pruebas planteadas se considerarán diferentes puntos del ciclo de vida de manera que se analizará el comportamiento que sigue cada variante en los diferentes momentos del ciclo de vida que puede tener un sistema de recomendación.

Palabras clave

Sistemas de recomendación, bandido multi-brazo, exploración, explotación, evaluación, ciclo de vida.

Abstract

Every day the amount of information used by all the existing intelligent systems grows more and more. Among all these systems we can see how the recommendation systems have acquired a lot of importance in recent times with applications such as Netflix or Spotify, essential in the lives of many people.

The work in the field of recommendation systems is focused on the evaluation of these systems. More precisely, this evaluation aims to check whether the use of multi-arm solutions in the life cycle of these systems produces better results than traditional methods.

These multi-arm solutions, unlike what has been done, where each of the arms is an item, will have a different recommender in each of its arms, so that it will act as an ensemble of recommenders. In this way, for each recommendation, a single recommender is chosen according to the strategy used, who will be the one used for returning the recommendation.

However, different life cycles will be proposed for the evaluation, which will be analyzed in a comparative way, being these a periodic offline evaluation, A/B tests with a fixed fraction of traffic, A/B tests based on multi-arm bandits with a fixed fraction of traffic and a typical multi-arm evaluation.

For all the tests proposed, different points of the life cycle will be considered so can be seen the behavior that follows each variant in the different moments of the life cycle that a recommendation system can have.

Keywords

Recommendation systems, multi-arm bandit, exploration, exploitation, evaluation, life cycle.

Índice

Listas

Lista de ecuaciones

Lista de figuras

1

Introducción

1.1. Motivación

Actualmente, la cantidad de información con la que trabajamos es cada vez mayor, por lo que necesitamos mecanismos que sean capaces de recolectarla, filtrarla y ofrecérnosla. Uno de los más utilizados son los sistemas de recomendación. Podemos ver su uso reflejado en nuestro día a día en aplicaciones como Netflix o Spotify, las cuales intentan ofrecernos contenido de nuestro interés entre el gran número de información con la que trabajan.

La generalización de este tipo de aplicaciones hasta convertirse en tecnologías de uso cotidiano motiva el interés por estudiar los mecanismos que dichas aplicaciones utilizan para hacer sus recomendaciones, qué tipos de algoritmos se utilizan y cómo estos se podrían mejorar para ofrecer recomendaciones más precisas y útiles.

Una de las áreas con más actividad en el desarrollo de las tecnologías de recomendación es el de la evaluación de estos. De hecho, es uno de los problemas abiertos en el campo que, junto con el desarrollo de soluciones algorítmicas cada vez más eficaces, más atención está recibiendo en los últimos años.

La evaluación es un aspecto crítico en el progreso del área, pues es lo que define la evolución algorítmica: la evaluación viene a ejercer el papel de función de bondad en un proceso de selección natural competitiva que determina qué soluciones se adoptarán y evolucionarán. Definir una buena evaluación es casi lo mismo que, o al menos está a un paso de, definir soluciones óptimas al problema de recomendación.

En cuanto a la búsqueda de soluciones algorítmicas más eficaces, el campo de los bandidos multibrazo, viene destacando en los últimos años, ya que mejora en cuanto a rendimiento y resultados a los algoritmos híbridos que se venían utilizando, lo que motivó mi elección para ser estudiado en el proceso de evaluación de un sistema de recomendación.

1.2. Objetivos

El objetivo general del trabajo es analizar en un sistema de recomendación el comportamiento de soluciones multi-brazo, por medio del contraste de diferentes alternativas de evaluación. Siendo más concretos los objetivos principales serán:

- **Estudiar el estado actual de las soluciones multi-brazo y la evaluación en sistemas de recomendación.** De esta manera se podrá conocer más acerca del estado del arte de este campo y las soluciones que se están proponiendo en la actualidad, para aplicarlo en os posteriores experimentos.

- **Implementar los algoritmos de recomendación para utilizar en las pruebas experimentales.** La implementación de dichos algoritmos permitirá tener una base sobre la que probar los diferentes métodos de evaluación. Estos algoritmos son tanto algoritmos tradicionales: popularidad, KNN y factorización de matrices, como basados en bandidos multi-brazo: ϵ-greedy, Thompson sampling, UCB y EXP3.

- **Diseñar, ejecutar y comparar los distintos experimentos para la comparación de los distintos ciclos de vida en la evaluación.** Así, se podrán obtener una buena amplitud de posibilidades y resultados que, tras su posterior estudio, permitirá sacar conclusiones acerca de los diferentes ciclos de vida planteados, así como de la forma de evaluarlos y si como se espera, los bandidos multi brazo mejoran a los métodos tradicionales.

1.3. Organización de la memoria

La memoria consta de los siguientes capítulos:

- En el capítulo 2, se expone el estado del arte de los sistemas de recomendación y su evaluación, así como de los bandidos multi-brazo, para poder contextualizar alguno de los conceptos que se abordan a lo largo de este trabajo.

- En el capítulo 3, se presentan las decisiones tomadas en la implementación de cada uno de los recomendadores, así como una pequeña explicación de cada uno para contextualizar como funciona.

- En el capítulo 4, se presenta y explica que es un ciclo de vida y se enumeran cada uno de los ciclos de vida que se van a analizar en la evaluación planteada para los experimentos.

- En el capítulo 5, se especifican detalles sobre la implementación y el funcionamiento de los diferentes experimentos, analizando y comparando posteriormente los resultados.

- En el capítulo 6, terminamos exponiendo de manera sintetizada las conclusiones obtenidas de los resultados de las pruebas y exponemos las bases de lo que podría ser un trabajo futuro.

2

Estado del arte

Una de las partes más importante a la hora de trabajar con un sistema de recomendación es la **evaluación**, ya que necesitamos una forma de cuantificar cómo de bueno es dicho sistema.

La evaluación de sistemas de recomendación tiene varias alternativas, entre las que destacan la evaluación **online** y **offline**. La primera es más rápida y económica, mientras que la segunda es más fiable pero costosa. Típicamente se aplica un primer ciclo de evaluación offline para seleccionar que sistemas o alternativas pasan a un segundo ciclo de evaluación online.

El método de evaluación online típico son los test A/B, los cuales permiten probar dos variantes de algún elemento de un sistema para comparar su funcionamiento y así poder sacar conclusiones y tomar la decisión oportuna: desplegar el sistema A o el sistema B, el que haya obtenido mejores resultados en el test A/B. Los test A/B son aplicables en recomendación, ya que se pueden comparar distintos algoritmos de recomendación y utilizar el que mejor esté funcionando.

Los test A/B son costosos, requieren tiempo (típicamente una o más semanas), e implican riesgo al exponer a usuarios reales a versiones experimentales del sistema, potencialmente subóptimas. Para paliar estos inconvenientes, se han empezado a utilizar recientemente métodos basados en **bandidos multi-brazo** que aceleran estos experimentos, y minimizan dinámicamente la asignación de tráfico a las versiones de prueba que peor funcionan.

Par aplicar estas soluciones multi-brazo, se considerará una perspectiva temporal cíclica, donde el sistema es capaz de ofrecer recomendaciones repetidamente a lo largo del tiempo y nutrirse del resultado de dichas recomendaciones para proporcionar más información al sistema de cara a futuras recomendaciones. Vemos por tanto claramente el ciclo producido, ya que está continuamente ofreciendo recomendaciones y recibiendo feedback.

2.1. Sistemas de recomendación

Los sistemas de recomendación son una serie de herramientas y técnicas que permiten proporcionar sugerencias acerca de un conjunto de **ítems** para que sean explotados por un **usuario**. Dichas

sugerencias se refieren a diversos procesos en los que es necesaria la toma de una decisión como la compra de artículos, la elección de música para escuchar o qué película escoger para ver (Ricci et al, 2015).

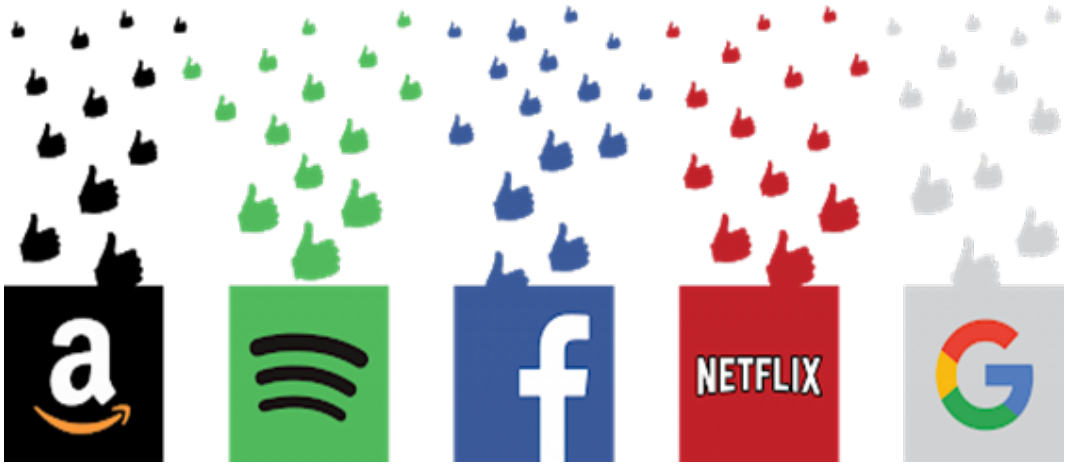

Figura 2.1: Empresas destacadas que utilizan sistemas de recomendación

Los sistemas de recomendación están dirigidos principalmente a individuos que carecen de suficiente experiencia personal o competencia para evaluar el número potencialmente abrumador de ítems diferentes que el sistema puede ofrecer. Ítem es el término general que se utiliza para denotar lo que el sistema recomienda a los usuarios.

Las recomendaciones de estos ítems se ofrecen como listas de clasificación en base los **ratings**, que son las puntuaciones que se otorgan a dichos ítems para cada uno de los diferentes usuarios que puede haber en el sistema. De esta manera se explotan los productos o servicios más adecuados en base a las preferencias y limitaciones del usuario.

Si la puntuación refleja una preferencia favorable al ítem decimos que ese ítem es **relevante** para el usuario, por el contrario, si no se adapta a lo que el usuario estaba buscando, lo consideramos **no relevante**. Los ratings pueden ser tanto binarios (relevante / no relevante) como numéricos (una escala de números de menor a mayor gusto por el ítem).

En el caso de los binarios es claro como distinguir cuales son relevantes (sí o no), pero para los numéricos es necesario utilizar algún tipo de regla para delimitar cuales se consideran relevantes y cuáles no lo son (típicamente poner uno de sus valores numéricos como umbral de relevancia).

Existen dos formas de recoger estas valoraciones. Una es de forma implícita, extrayendo la información pertinente de las acciones del usuario. Por ejemplo, el tiempo que pasa leyendo una determinada página web, los enlaces que sigue o el número de veces que se escucha una canción.

La segunda, es explícitamente, es decir el usuario asigna una puntuación a cada elemento que sería un valor numérico discreto entre un máximo y un mínimo y el límite entre relevante o no es también un valor numérico dentro de ese rango elegido arbitrariamente (Nichols, 1998).

2.1.1. Recomendación

Son varios los pasos que sigue un sistema de recomendación para poder realizar la tarea de recomendación, las cuales pueden dividirse en:

- **Observación**. Se estudia las acciones realizadas por el usuario mientras éste realiza su interacción con el sistema.
- **Detección**. Se buscan patrones de comportamiento, para identificar posibles señales de interés en el usuario.
- **Predicción**. Se obtienen posibles intereses futuros basados en indicios de intereses pasados.
- **Recomendación**. Se ofrecen al usuario las distintas opciones que el sistema cree que pueden ser más de su interés.

Para poder escoger cuál de las recomendaciones ofrecidas por el sistema se adapta mejor en cada momento es necesaria alguna forma de clasificación que, permita ordenar según un determinado criterio cada uno de los ítems que se incluye en la recomendación.

Para ello, utilizando algún criterio de similitud entre ítems, usuarios o caracterísiticas y un determinado algoritmo de recomendación, se puede generar un **ranking** que nos permitirá ordenar los ítems a ofrecer al usuario, donde el top de ese ranking es el ítem más afín al usuario en ese momento. Si el usuario necesita otra recomendación, lo más normal es que el ranking anterior cambie en función de la recomendación anterior, ya que aporta más información al sistema (Liu, 2009).

Para almacenar toda la información del sistema para cada uno de los usuarios e ítems que lo componen, así como la valoración que se le da a cada uno de los pares usuario-ítems que forman el sistema se utiliza una **matriz de puntuaciones**, ya que ese tipo de estructura permite manejar grandes cantidades de datos de manera bastante eficiente.

Este tipo de estructura por tanto consiste en una matriz en la que las filas se corresponden con los usuarios que tiene el sistema, mientras que sus columnas se corresponden con los ítems, aunque también puede ser justo, al contrario, dependiendo de cuál de los dos tiene un número más elevado de elementos en el sistema.

Por último, cada una de las entradas de la matriz se corresponden con los ratings de ese usuario para ese ítem. Si se representa la matriz, lo normal es que existan entradas vacías. Estas se corres-

ponden con las posibles recomendaciones que el sistema puede ofrecerle al usuario (Jure, 2011).

	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
U1	1	2	•	•	2	•	3	4	•	4	1	•
U2	•	•	1	5	•	5	3	1	•	5	2	1
U3	1	•	•	2	•	1	•	3	4	•	•	•
U4	•	1	4	4	•	•	3	•	5	4	•	1
U5	2	•	5	•	1	•	1	•	•	•	2	1
U6	•	•	5	2	1	•	•	4	•	1	•	2

Figura 2.2: Representación de una matriz de puntuaciones

2.1.2. Similitud en recomendación

Para poder realizar una recomendación es necesaria algún tipo de solución que permitan poder relacionar los elementos del sistema de recomendación, ya sean usuarios como ítems, es decir, se necesita buscar algo que permita medir la similitud existente entre ellos.

Para ello, los sistemas de recomendación definen una función similitud que nos permite comparar cuantitativamente estos elementos entre ellos para poder obtener relaciones que puedan utilizar posteriormente los algoritmos de recomendación que se utilicen. Existen varias funciones que trabajan con ello entre las que destacan:

- **Similitud coseno**: Este método consiste en proyectar cada uno de los elementos analizados de manera que permita comparar el coseno que forma el ángulo de sus proyecciones. Cuanto menor sea el ángulo formado entre los dos elementos, mayor será la similitud entre ellos. A continuación, se muestra la fórmula que aplica.

$$cos(\theta) = \frac{XY}{|X||Y|} = \frac{\sum_{i=1}^{n} X_i Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2 \sum_{i=1}^{n} Y_i^2}} \tag{2.1}$$

  Analizando la fórmula, podemos identificar X e Y como cada uno de los elementos que se están comparando, mientras que el denominador se corresponde con los módulos de ambos elementos. Similar a esta similitud es la correlación de Pearson que realiza la misma operación, pero centrándolo en la media (Vozalis et Margaritis, 2003).

- **Similitud Jaccard**: A diferencia del método anterior, este funciona mediante conjuntos, cuantificando como de semejantes son dos conjuntos, donde cada uno es representado por

uno de los elementos. La fórmula que aplica es la siguiente.

$$Jaccard(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|} \tag{2.2}$$

Podemos ver por tanto como se buscan elementos con componentes coincidentes, siendo el numerador la intersección entre los dos elementos y el denominador su unión (Ivchenko et Honov, 1998).

Mediante la aplicación de alguna de estas funciones de similitud, podemos relacionar los ítems o usuarios del sistema para poder aplicar los algoritmos de recomendación.

2.1.3. Algoritmos de recomendación

Al igual que en otros muchos campos, no hay un solo algoritmo que defina la tarea de en este caso recomendar, sino que existen más variantes que en función de diversos factores pueden producir mejores o peores resultados. Estos entre otras cosas se diferencian entre ellos en como eligen u ordenan los ítems que se quieren recomendar, pero si que tienen un comportamiento común.

Primeramente, al algoritmo se le pasa como entrada los ratings que ya tiene registrado el sistema, así como el usuario y el número de ítems que se quieren recomendar, lo que será el tamaño del ranking que se devolverá más tarde.

Una vez se tiene toda esta información, se aplica sobre ella el algoritmo pertinente y se le asigna a cada ítem una puntuación. La diferencia entre la mayoría de los algoritmos radica en los cálculos de dicha puntuación, ya que cada uno de ellos buscan obtener mayor beneficio sobre diferentes aspectos del sistema.

Una vez aplicado el algoritmo, se ordenan las puntuaciones obtenidas (normalmente en el top se sitúa el ítem con mayor puntación) y se devuelve el ranking del tamaño solicitado en la entrada. Dependiendo del contexto del sistema en el que se ejecuta la recomendación, podemos distinguir dos clases de algoritmos (Aggarwal, 2016):

- **Basados en contenido**: El sistema aprende a recomendar ítems que son similares a otros que ya le habían gustado antes al usuario. La similitud entre los ítems se calcula en base a las características asociadas a los ítems, es decir, si dos ítems tienen características parecidas y al usuario le ha gustado anteriormente uno de ellos, el sistema tenderá a recomendarle el otro ya que existe una similitud entre sus características. Por ejemplo, si un usuario valora positivamente las canciones de un determinado género, el sistema aprenderá a recomendarle otras canciones de ese género (Zisopoulos et al, 2008). Se puede ver la lógica de su

funcionamiento en la figura 2.3.

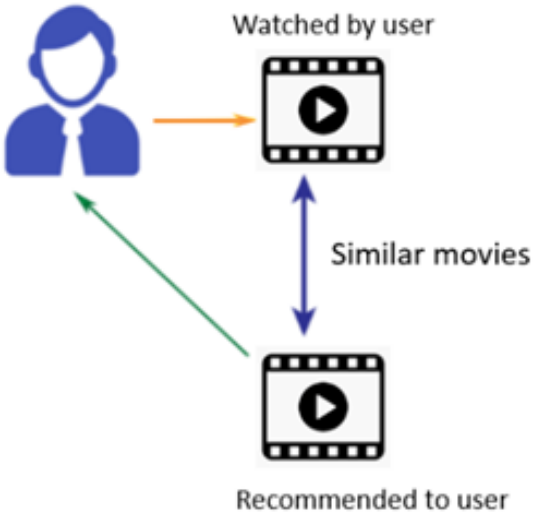

Figura 2.3: Esquema de un sistema de recomendación con algoritmo basado en contenido

- **Basados en filtrado colaborativo**: El sistema aprende a recomendar ítems que son similares a otros que ya les habían gustado a usuarios con gustos similares al usuario al que se le está recomendando. La similitud en este caso se calcula en base a pares de usuarios, según la similitud de gustos o comportamiento de ambos, es decir, la correlación que hay entre ambos. Este tipo de algoritmos son considerados los más populares a la hora de usarlos en un sistema de recomendación (Luo et al, 2013). Se puede ver la lógica de su funcionamiento en la figura 2.4.

Cuando un sistema de recomendación se encuentra en una fase temprana de uso, es decir, se dispone de poca información del usuario, o pocos usuarios en el sistema, se produce un problema conocido como **arranque en frío**. Esto provoca que, en el caso del filtrado colaborativo, no se tenga información apenas del resto de usuarios y las recomendaciones no sean muy buenas, algo que se solventa en los basados en contenido si disponemos de algo de información sobre las características del ítem.

Debido a esto, es muy común utilizar este tipo de algoritmo en combinación. Por ejemplo, si se combinan un algoritmo de filtrado colaborativo con uno basado en contenido, se puede detectar similitudes entre ítems, sin necesidad de que los usuarios hayan realizado valoraciones anteriormente (Schein et al, 2002), lo que permite obtener mejores resultados en la fase inicial del sistema.

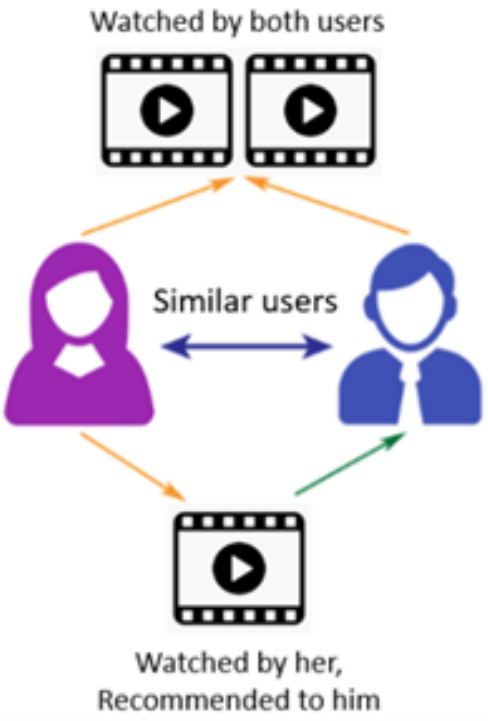

Figura 2.4: Esquema de un sistema de recomendación con algoritmo basado en filtrado colaborativo

2.1.4. Ensemble en algoritmos de recomendación

Surgen con el objetivo de cubrir las debilidades que pueden tener cada uno de los algoritmos de recomendación por separado y así, potenciar y mejorar las recomendaciones del sistema. Podemos relacionar con este tipo de algoritmo, por tanto, a los sistemas que combinen múltiples algoritmos de recomendación que produzcan una única salida.

Por tanto, los ensembles en recomendación son un caso particular de algoritmo híbrido, pero su particularidad radica en que los algoritmos que se combinan son considerados como una **caja negra**. Con esto se consigue que no haya ningún preprocesamiento previo de todos los algoritmos para que formen uno único, sino que sean las salidas de cada uno de ellos las que sean utilizadas (Aggarwal, 2016).

La ventaja por tanto de este tipo de algoritmos es que permiten la combinación de tantos algoritmos como queramos usar, pudiendo ser estos de cualquier tipo o cualquier variante de un mismo algoritmo. De esta manera, la forma más utilizada para trabajar con las salidas de estos algoritmos es combinar las puntuaciones devueltas por cada uno de los algoritmos que han sido ejecutados individualmente.

Para realizar dicha combinación existen varias técnicas, pero las más utilizadas son la suma de puntuaciones de cada salida o realizar un nuevo ranking con la combinación de las posiciones que ocupan los ítems en el ranking de cada algoritmo. Pero quizás, la que más interesa en este trabajo es otra alternativa que se viene usando y que consiste en que en cada instante que se va a recomendar

solo se utilice la salida devuelta por uno de los algoritmos, eligiendo dicho algoritmo según alguna métrica de calidad en la recomendación (Adomavicius et Tuzhilin, 2005).

2.2. Aprendizaje por refuerzo

El aprendizaje por refuerzo, el cual podemos encuadrar en el campo del **Machine Learning**, nace como alternativa a otras ramas de este campo como son el aprendizaje supervisado o no supervisado. Este tiene como objetivo el estudio de métodos con el fin de entrenar agentes inteligentes para resolver un determinado tipo de problemas de manera autónoma, siendo estos capaces de generalizar comportamientos mediante la información que se les sea proporcionada.

El objetivo que persigue es descubrir de qué manera asignar determinados comportamientos a acciones. Para ello sigue un proceso iterativo de observación, acción y recompensa, que, a diferencia de otras ramas del campo, no le recomienda al sistema la acción que debe efectuar, sino que le ofrece varias soluciones en base a una recompensa y es el propio sistema el que decide que acción realizar (Sutton et Barto, 2018).

La lógica que aplica este tipo de aprendizaje se basa en lo que se conoce como **Proceso de Decisión de Markov**. Este proceso es un tipo de modelado matemático que permite explotar el concepto de recompensa para resolver problemas de optimización basados en el concepto de aprendizaje antes comentado.

En la figura 2.5, se puede observar la abstracción del comportamiento que siguen este tipo de procesos, donde S_t y A_t son un conjunto finito de estados y de acciones respectivamente con los que interactuar en ese instante de tiempo, mientras que R_t es el conjunto de recompensas que el agente puede recibir para ese mismo instante de tiempo. En cuanto a R_{t+1} y S_{t+1} son los conjuntos comentados anteriormente, pero para el siguiente instante de tiempo que se vaya a aplicar.

2.2.1. Algoritmos de aprendizaje por refuerzo

El objetivo del aprendizaje por refuerzo es encontrar una política en la toma de decisiones que maximice la recompensa, por lo que se han desarrollado diversos algoritmos para lograr este objetivo.

En este proceso de toma de decisión, los agentes tienen dos formas de escoger la acción a realizar las cuales son conocidas como **exploración** y **explotación**.

La exploración consiste en que el agente explore entre las distintas acciones que se le presentan según los estados que le proporciona el entorno, no siendo necesariamente la mejor acción la elegida, sino la que según lo aprendido pueda maximizar la recompensa en futuras recomendaciones. Pero tiene un gran problema y es que, si no aprende adecuadamente o se abusa de ella, puede significar

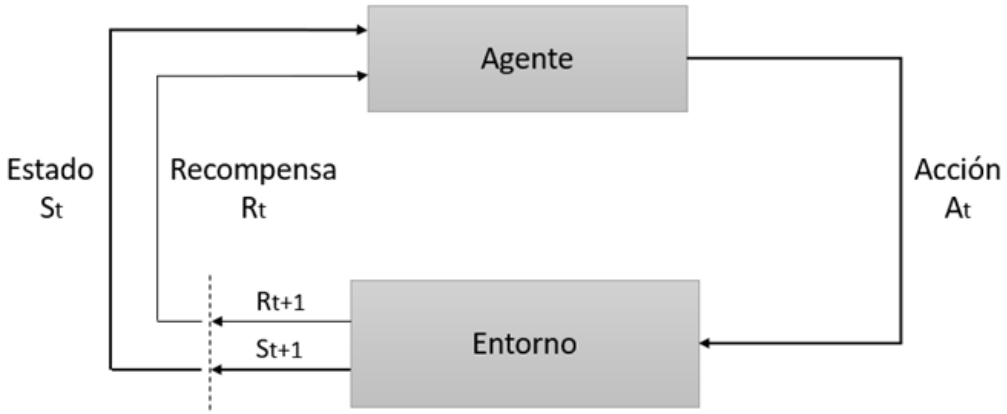

Figura 2.5: Interacción agente-entorno en MDP

que el rendimiento empeore mucho.

En cambio, la explotación se encarga de que el agente elija la acción que cree que es mejor en ese instante de tiempo, sin tener en cuenta situaciones posteriores. De esta forma, el agente premiará a la política que mejor esté actuando para ese entorno, además de obtener buenas recompensas. En este caso, como antes se ha comentado, podría haber otros caminos que mejoren la recompensa a la larga, pero con este método nos aseguramos obtener buenos resultados siempre.

Una de las principales preocupaciones para este algoritmo es realizar un buen balanceo entre estas dos posibilidades, para obtener mejores resultados con el paso de las recomendaciones (Liang, 2017). Para tratar este problema, que en el fondo es la gran diferencia que hay entre los algoritmos de recomendación de aprendizaje por refuerzo existen varios entre los que destacan:

- ϵ**-greedy**: Para cada instante de tiempo en el que es aplicado, utiliza con una probabilidad ϵ el método de exploración, mientras que con una probabilidad de $(1 - \epsilon)$ utiliza el método de explotación. Este parámetro suele variar en el rango [0,1]. Este parámetro del que hablamos puede dejarse fijo durante toda la ejecución o se puede variar en función de si está dando buenos o malos resultados (Koulouriotis et Xanthopoulos, 2008). En la figura 2.6, se puede observar la lógica seguida por el algoritmo.

- **Thompson sampling**: Basándose en las observaciones previas que ha ido haciendo, utiliza una distribución de probabilidad sobre cada una de las acciones para elegir la acción a utilizar. Para modelar la distribución, se hace uso de los parámetros, α y β, que van acumulándose en caso de acierto o fallo respectivamente para las siguientes recomendaciones.

  Utilizando dichos parámetros en cada instante de tiempo, se obtiene una muestra, que nor-

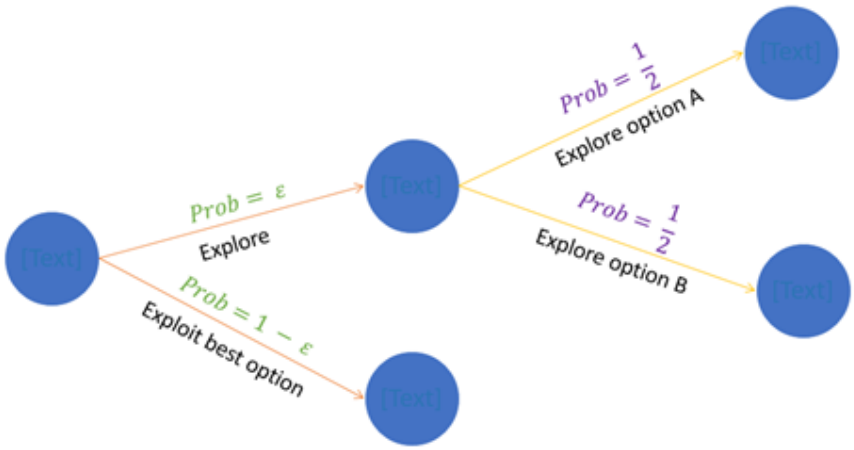

Figura 2.6: Esquema de funcionamiento del algoritmo ϵ-greedy

malmente es la media, y se escoge la acción con mayor valor muestral. Si el valor de la recompensa es binario, es decir, relevante o no relevante, se utiliza una distribución beta-bernoulli para modelar la distribución (Brodén et al, 2018). En la figura 2.7, se puede observar un esquema de como realiza la selección el algoritmo.

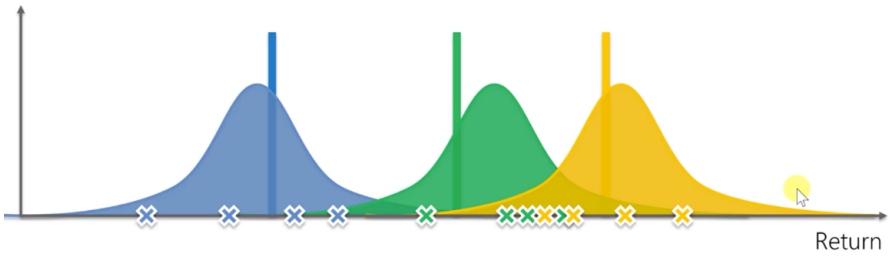

Figura 2.7: Esquema de funcionamiento del algoritmo Thompson sampling

- **UCB**: A diferencia de los algoritmos anteriores, este asume que las acciones que no ha explorado el algoritmo deben tener mejor recompensa, es decir, asume para ellos un límite de confianza alto, primando la exploración.

  Según se va estabilizando el algoritmo las recompensas obtenidas también lo harán promediándose con los resultados anteriores de manera que irá tendiendo a la explotación, siempre que el límite de confianza lo permita. Una de las cosas que permite este algoritmo también es que todas las acciones se exploren al menos una vez. (Liang, 2017). En la figura 2.8, se observa como realiza la selección el algoritmo.

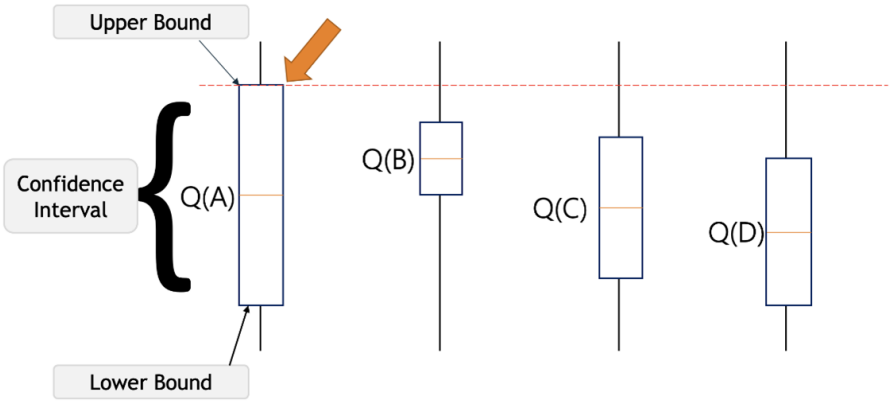

Figura 2.8: Esquema de funcionamiento del algoritmo UCB

- **EXP3**: Los algoritmos como UCB funcionan mejor en un entorno estocástico; es decir, donde los resultados de cada prueba son totalmente aleatorios. En cambio, EXP3, es capaz de ofrecer mejores resultados en situaciones no estocásticas, dónde el resultado esperado de futuras pruebas no será aleatorio, sino que pueda ser cambiado por los resultados de pruebas anteriores.

  Un ejemplo que nos permita entenderlo puede ser, por ejemplo, el caso de unos inversores que ven una acción a un precio bajo y la compran. Aunque su rendimiento en es momento no es muy alto, el mero hecho de comprar la acción hace que su precio suba y su rendimiento también. De esta manera, las ganancias esperadas de esas acciones cambian al igual que ocurre con el resultado del algoritmo cuando se aplica.

2.2.2. Bandidos multi-brazo

Estos consisten en un sistema el cual posee múltiples brazos, donde cada uno representa las acciones que el agente puede escoger y donde el objetivo del agente es intentar maximizar las probabilidades de conseguir la mayor recompensa minimizando el coste necesario para obtenerla (Katehakis et Veinott, 1987).

El agente, por tanto, va examinando las distintas acciones que puede tomar y en función de las decisiones que va tomando, irá aprendiendo de las experiencias adquiridas para ser lo óptimo posible a la hora de volver a escoger una acción a realizar.

Por tanto, se puede ver como la recompensa que tendrán dichas acciones no será la misma cada vez que se tenga que tomar una decisión, sino que este valor cambiará dinámicamente según lo acertada que haya sido dicha decisión. En la figura 2.9, se puede ver de una forma simplificada como actúa el bandido multi-brazo para escoger el brazo a utilizar.

Figura 2.9: Figura ilustrativa de la lógica de bandidos multi-brazo. Cada uno de los brazos del pulpo representa los brazos del bandido, mientras que las máquinas tragaperras simbolizan cada una de las acciones que el agente puede elegir.

Por la lógica que siguen los bandidos multi-brazo, se puede ver una clara aplicación en los sistemas de recomendación como ya otra gente empezó a ver cuándo empezó a aplicar este método en recomendación. Podemos hacer un símil de las acciones con los posibles ítems a recomendar por un sistema de recomendación, por tanto, cada uno de los brazos del bandido será un ítem del sistema.

Así, el problema que se busca solucionar con la aplicación de soluciones multi-brazo es seleccionar el ítem más conveniente para el usuario en cada instante de tiempo en el que se solicite (Li et al, 2016). En cuanto a la recompensa y su actualización para futuras recomendaciones, en este caso, variará según si la recomendación del ítem dada en el anterior instante de tiempo es acertada, por lo que vemos la importancia que tiene la relevancia de la recomendación en el proceso de recomendación del sistema.

En algunos sistemas, puede existir un contexto para la toma de decisión del brazo a escoger, lo cual en muchos casos facilita la identificación del mejor brazo. En caso de darse esta situación surgieron los bandidos multi-brazo contextuales, los cuales son capaces de trabajar con esas características extra que posee el sistema.

Estos en cada iteración, analizan un vector de características, donde se encuentra todo el contexto antes mencionado, refinando así la búsqueda y obteniendo una recompensa más precisa tras el procesamiento de las características. El vector de características también se va actualizando según el sistema realiza más recomendaciones y encuentra más relaciones entre características, consiguiendo así mejores relaciones y siendo cada vez más preciso en la elección de brazo y por tanto en las recomendaciones (Langford et Zhang, 2008).

2.3. Evaluación

En los últimos tiempos la mayoría de las investigaciones en sistemas de recomendación se centran sobre todo en el diseño de nuevos algoritmos para las recomendaciones. Pero para elegir que algoritmo es mejor es necesario una forma de medir su efectividad y así poder comprarlos. Es aquí donde vemos la importancia que tiene la evaluación en los sistemas de recomendación, ya que sin una forma de evaluar esta efectividad no tendríamos ninguna forma de compararlos (Orimo et al, 2007).

Esa efectividad o calidad del algoritmo se puede medir de diferentes formas según el objetivo que se pretenda optimizar, por lo que surgen varias métricas en función de las necesidades que tenga el sistema de recomendación, las cuales se suelen dividir en **métricas de error** y **métricas de ranking**.

Antes de explicar en qué consiste cada una cabe aclarar que para poder realizar dicha evaluación es necesario dividir el conjunto de datos que tiene el sistema en dos subconjuntos. Estos se conocen como conjunto de **entrenamiento** y de **test**, donde el primero conjunto se utiliza para intentar predecir los datos del segundo, que son tomados como reales. Así, se puede cuantificar si la predicción es más o menos acertada frente a la interacción real.

Las **métricas de error** buscan mediante diferentes formas de aplicar la media de las distancias entre los datos entrenamiento y de test (predicción-realidad), cuantificar como de cerca quedan estos pares, es decir, como de acertada es la predicción frente a la realidad. Cuanto más bajo sea ese valor, más acertada será la recomendación. Algunas de estas métricas de error son (Parra et Sahebi, 2013):

$$MAE = \frac{\sum_{i=1}^{n-1} |\hat{u}_i - u_i|}{N} \tag{2.3}$$

$$MSE = \frac{\sum_{i=1}^{n-1} (\hat{u}_i - u_i)^2}{N} \tag{2.4}$$

$$MSE = \sqrt{\frac{\sum_{i=1}^{n-1} (\hat{u}_i - u_i)^2}{N}} \tag{2.5}$$

Las **métricas de ranking**, por el contrario, se centran en la ordenación del ranking devuelto por la predicción frente a la realidad, más que en el valor exacto de dicha diferencia como las métricas de error. En este tipo de métricas se tiene muy en cuenta la **relevancia**, que ya al introducir los sistemas de recomendación se había mencionado. Se considera que, si se toma un ítem como relevante, este será tratado como un acierto. Así entre estas métricas destacan (Parra et Sahebi, 2013):

$$Precision = \frac{|Recomendados \cap Relevantes|}{Recomendados} \tag{2.6}$$

$$Recall = \frac{|Recomendados \cap Relevantes|}{Relevantes} \tag{2.7}$$

$$Med.Armnica = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.8}$$

Las dos primeras, pueden ser delimitadas, si solo nos interesa un número determinado de elementos del ranking, siendo tratadas como P@K y R@K, donde K es el tamaño de dicho ranking (Valcarce et al, 2018).

Como vemos el problema de la evaluación es un campo muy abierto debido a la cantidad de opciones sobre las que puede interesar evaluar por lo que han ido surgiendo otras métricas centradas en otro tipo de objetivos. Entre ellas podemos destacar la diversidad, la novedad y el Click-through rate (CTR).

La **diversidad** evalúa en qué medida un ítem es diferente con respecto a otro ítem o conjunto de ítems que hayan sido recomendados, mientras que la **novedad**, analiza la diferencia qué tiene un determinado ítem respecto a las recomendaciones previas de un usuario o grupo de usuarios.

Por otro lado, está el **Click-through rate**, el cual se viene utilizando mucho en sistemas de recomendación aplicados al comercio y aplica la siguiente fórmula para la obtención de su valor (Sammut et Webb, 2010):

$$CTR = \frac{Clicks}{Visualizaciones} \cdot 100 \tag{2.9}$$

Con esta métrica, por tanto, podemos medir el efecto que tienen determinados ítems al ser recomendados en una web, por ejemplo, viendo si son del interés del usuario o por el contrario son ignorados.

Para realizar una evaluación sobre un sistema de recomendación, independientemente de la métrica, hay que distinguir dos modos **offline** u **online**. El primer caso, va más enfocado a probar diferente algoritmos o versiones de ellos con el fin de escoger el más idóneo para su aplicación en un sistema real, mientras que, en el caso online, va más enfocado a probar con usuarios reales que permitan comprobar el rendimiento que tendría en un sistema en producción (Cañamares et Castells, 2018).

2.3.1. Evaluación offline

La evaluación offline se lleva a cabo utilizando un determinado conjunto de datos con información antes recolectada, que suele tener información de usuarios, ítems y sus correspondientes puntuaciones. De esta forma, utilizando este conjunto de datos podemos intentar simular el comportamiento de los usuarios que interactúan con el sistema de recomendación.

De esta forma se asume que el comportamiento del usuario o usuarios del conjunto de datos serán lo suficientemente similar a los usuarios cuando sea desplegado en producción el sistema de recomendación, para que así, podamos tomar decisiones fiables basándonos en la simulación.

Los experimentos offline son atractivos porque no requieren interacción con usuarios reales, lo que permite comparar una amplia gama de algoritmos a un bajo coste. El principal problema que presenta es que no podemos directamente medir la influencia del sistema de recomendación en el comportamiento del usuario, ya que no tenemos un feedback real (Jeunen, 2019).

Para ello se utilizan los propios datos que están en el conjunto de test como si fuera el feedback de usuario, introduciéndolos como más información en el conjunto de entrenamientos una vez recomendados. Como se puede observar esto limita el experimento, por lo que es imprescindible una evaluación con usuarios reales para confirmar el funcionamiento de los mejores resultados obtenidos.

Un ejemplo típico en el que se aplica esta evaluación es a la hora de elegir los parámetros de un determinado algoritmo, probando las distintas variantes en un experimento offline permitiendo pasar a la siguiente fase al algoritmo con los parámetros mejor ajustados.

Por lo tanto, el objetivo de este tipo de evaluación es filtrar los algoritmos u opciones que peor se adapten, dejando un conjunto relativamente pequeño de algoritmos candidatos para ser probados frente a usuarios reales.

2.3.2. Evaluación online

En la mayoría de los sistemas de recomendación se busca que dicho sistema influya en el comportamiento de sus usuarios. Por lo tanto, con este tipo de evaluación se busca analizar y medir el cambio que se produce en el comportamiento del usuario al interactuar con diferentes variantes del sistema de recomendación. El objetivo que se persigue entonces es encontrar el sistema que más se adapte a dicho usuario.

A diferencia de la evaluación offline, esta configuración nos permite probar el comportamiento de los usuarios cuando interactúan con el sistema de recomendación, y la influencia de esas recomendaciones en su comportamiento. Este caso, también permite recoger datos cualitativos de los propios usuarios, que pueden llegar a ser cruciales para interpretar los resultados cuantitativos (Gilotte et al, 2018).

Es más fiable comparar con unos pocos usuarios reales menos alternativas o algoritmos para un sistema de recomendación, que basarse en las predicciones de algoritmos que no han sido probados en la realidad, por lo que es esencial aplicar este tipo de evaluación en caso de querer poner dicho sistema en producción para el uso de usuarios reales.

Hay algunas consideraciones que deben tenerse en cuenta al realizar estas pruebas. Es importante que los usuarios sean distribuidos al azar, para que las comparaciones entre las alternativas sean justas.

También es importante si lo que queremos medir es el rendimiento de los algoritmos y no por ejemplo la interfaz que se le muestra al usuario, que las alternativas mostradas sigan el mismo diseño, siendo lo único que cambia el algoritmo que se está utilizando.

En algunos casos, puede darse que el sistema proporcione recomendaciones irrelevantes, lo que puede desalentar a los usuarios de la prueba. Por lo tanto, el experimento puede tener un efecto negativo en el sistema, que puede ser determinante en el futuro en producción que puede tener. Por estas razones, lo dos tipos de evaluaciones, offline y online se complementan para asegurarnos de un mejor funcionamiento final.

Típicamente, en los experimentos online se redirige un porcentaje de tráfico de una variante a otra de modo que se pueda analizar la efectividad que tiene esa variante con respecto a la cantidad de tráfico que ha sido desviado. Esto es conocido como **test A/B** (Siroker et Koomen, 2015) y permite por tanto comprobar si la variante del sistema B es capaz de obtener un mejor rendimiento en un determinado usuario frente a la variante A.

Figura 2.10: Lógica de los test A/B. En este caso el sistema B está obteniendo mejores resultados

Estos pueden ser aplicados no solo a dos variantes, sino a tantas como elementos se quieran comparar en la evaluación, siendo conocido esto como **test multivariante** (Kohavi et Longbotham, 2017). Hay que destacar que cuantas más variantes haya, mayor tiene que ser el volumen de datos y de usuarios, ya que si no cabe la posibilidad de que el test no sea tan preciso.

2.3.3. Uso de bandidos multi-brazo como evaluación online

Una de las grandes aplicaciones que se le está dando a las soluciones multi-brazo es utilizarlos como test A/B. Igual que hacen estas pruebas, el objetivo de los bandidos es decidir entre varias opciones, en este caso entre varios brazos, con el fin de elegir el que más rendimiento vaya a ofrecer, es decir, maximice la recompensa final.

Una prueba A/B es puramente exploratoria, es decir, los usuarios son asignados al azar a una variante, y el resultado que se puede obtener es simplemente el resultado medio de todas las variantes que, por diseño, siempre será inferior al resultado de la variante más fuerte. Por tanto, no hay una explotación consciente de la variante con mayor recompensa, existiendo una diferencia entre la política óptima que elegiríamos si conociéramos la verdadera retribución, y la acción tomada.

Los algoritmos de bandidos multi-brazo, por otra parte, tratan de lograr un equilibrio entre la exploración y la explotación explorando cada variante lo suficiente para identificar la variante más fuerte, y luego explotando principalmente la acción óptima para maximizar la recompensa total. El algoritmo también continúa explorando otras posibles variantes para descubrir si se vuelven más fuertes en el futuro. La idea es terminar con una recompensa final más, al tiempo que se permite al algoritmo seguir reuniendo datos sobre otras variantes.

Si una de las variantes es claramente peor que otra, está verá cada vez más reducido el tráfico que le es asignado. En los test A/B, esa variante que ve reducido su tráfico puede llegar a ser 0 en el caso de que la evaluación así lo considere, pero con las soluciones multi-brazo, esta situación puede controlarse, pudiendo evitar que esa opción desaparezca, ya que más adelante puede que sea mejor que la opción que está destacando en ese momento (Cañamares et al, 2019).

3

Ensembles multi-brazo

Normalmente en la literatura las soluciones multi-brazo se han utilizado para la selección de ítems, pero como se ha comentado anteriormente estas soluciones pueden ser aplicadas como test A/B. Si queremos aplicarlo de esta manera que cada una de las variantes sea un ítem no es óptimo ya que los sistemas de recomendación suelen trabajar con muchos ítems.

Así, se presenta otra alternativa mucho más atractiva para el uso de los bandidos multi-brazo como test A/B y es que cada uno de los brazos o variantes del test, sea un algoritmo de recomendación.

Mediante esta reformulación se puede utilizar la solución multi-brazo como un ensemble dinámico de recomendadores, lo cual en la mayoría de los casos va a permitir mejorar la efectividad que tendría cada uno de los algoritmos de recomendación por separado, beneficiándose de la posibilidad que tendrá el bandido de adaptarse dinámicamente al brazo que mejor esté funcionando.

Cuando un brazo sea seleccionado, se ejecutará el algoritmo correspondiente el cual acabará eligiendo mediante su lógica un ítem que será recomendado al usuario objetivo. Si ese ítem es tratado como relevante será contabilizado como acierto, siendo tratado como fallo en caso contrario, lo cual permite actualizar la recompensa de cada uno de los brazos para la siguiente selección.

Esta actualización puede ser hecha después de cada recomendación individual, o cada cierto número de recomendaciones. En este trabajo, seleccionaremos los usuarios a los que se va a recomendar, y actualizamos los brazos después de cada ronda o época, periodo en el cual se le ha hecho una recomendación a cada uno de estos usuarios.

Hay que destacar que la selección del algoritmo a recomendar en cada ronda se basa en su rendimiento en las rondas anteriores, es decir, el bandido trabaja con toda la información que tiene el sistema, no solo con la anterior ronda.

A continuación, definiremos en detalle cada uno de los recomendadores con los que trabajaremos en los experimentos. Diferenciamos entre dos tipos recomendadores de referencia y basados en bandidos multi-brazo.

En el caso de los primeros son sistemas de recomendación que emplean algoritmos tradicionales de manera individual para la recomendación. En el caso de los segundos, son sistemas de recomen-

dación que emplean alguno de los algoritmos de aprendizaje por refuerzo par modelar un bandido multi-brazo cuyos brazos son alguno de los algoritmos tradicionales mencionados en el primer caso.

3.1. Recomendadores de referencia

Para la implementación de los recomendadores de referencia, hemos utilizado varios algoritmos ya existentes y bastante utilizados en la literatura como son el algoritmo **aleatorio, voto promedio, kNN basado en usuario y factorización de matrices**. El empleo de estos algoritmos de manera individual hace que estos recomendadores sean independientes entre sí. Esto nos permitirá también compararlos con sus versiones combinadas que serán realizadas en los recomendadores multi-brazo. A continuación, analizaremos cada uno de ellos.

3.1.1. Recomendador aleatorio

Este recomendador aplica a cada ítem de los que es posible recomendar un valor aleatorio entre 0 y 1. Una vez tiene un valor cada uno de ellos, son ordenados en un ranking de recomendación en orden descendente, es decir, el ítem con el valor más cercano a 1 será el que está en el top de dicho ranking.

Como se puede observar este recomendador al asignar un valor aleatorio a los ítems, no tiene en cuenta las necesidades del usuario o las recomendaciones previas, siendo el más simple de todos los recomendadores utilizados. Al ser un recomendador sin ninguna complejidad algorítmica, en los experimentos puede servirnos como punto de referencia sobre el que todos los algoritmos deberían mejorar.

3.1.2. Recomendador voto promedio

Este recomendador tiene como objetivo ofrecer al usuario el ítem más popular para el conjunto de usuarios del sistema. Son muchas las formas de definir la popularidad en este tipo de sistemas, pero en este caso, se ha optado por centrarse en la valoración media que los usuarios hacen sobre los ítems, lo que se conoce como algoritmo de voto promedio.

Otras variantes de este tipo de algoritmos de popularidad son, por ejemplo, las que otorgan ese valor de popularidad en función de la cantidad de usuarios que valoran un determinado ítem. Otra opción podría ser otorgar esa popularidad según el número de votos que son considerados como relevantes.

Aunque vemos diversas variantes para obtener la popularidad, al final se va a utilizar esta, ya que al tener puntuaciones numéricas sobre los ítems es más fácil de cuantificar y trabajar con las

puntuaciones.

3.1.3. Recomendador kNN basado en usuario

Este recomendador se basa en seleccionar los k vecinos más similares al usuario, de manera que combinando linealmente las valoraciones de dichos vecinos se pueda obtener una predicción de la predicción final.

Podemos establecer un símil con una situación de la vida cotidiana y es que todas las personas tienen amigos más cercanos que otros, por lo tanto, a la hora de pedir una recomendación por ejemplo de un restaurante, confiaremos más en dicha persona que en una desconocida. Pues en este caso es igual, mediante la correspondiente función de similitud y los k vecinos más próximos al usuario el sistema es capaz de obtener una recomendación final.

Este tipo de método también puede ser aplicado sobre los ítems, en vez de con usuarios, pero tal y como han sido planteados los experimentos, era más interesante analizarlo de esta manera.

3.1.4. Recomendador kNN factorización de matrices

La factorización de matrices en sistemas de recomendación caracteriza tanto a los usuarios como a los ítems mediante vectores de factores, los cuales son inferidos a partir de patrones de las puntuaciones que tienen los ítems. De esta manera y mediante las operaciones propias de la factorización de matrices el sistema es capaz de predecir los ratings que se utilizarán para la posterior recomendación a cada uno de los usuarios.

Esta familia de métodos se dio a conocer como una alternativa interesante en el desafío del premio Netflix (Gábor et al, 2008), donde se observó que los resultados de la predicción podían mejorarse según los pesos asignados a los factores antes comentados y hoy se han establecido junto a los anteriores como uno de los algoritmos que mejor resultado dan en los sistemas de recomendación.

3.2. Recomendadores multi-brazo

La implementación de estos recomendadores se basa en los **bandidos multi-brazo**, siendo cada uno de los brazos, alguno de los algoritmos tradicionales mencionados anteriormente.

Para cada ronda de recomendación, estos recomendadores basados en bandidos multi-brazo escogen uno de los brazos según la recompensa producida en la ronda anterior, y una vez elegido, se utiliza dicho algoritmo para la recomendación, siendo el ranking producido el que devuelve el propio algoritmo.

Para los posteriores experimentos se han utilizado cuatro recomendadores basados en estas soluciones multi-brazo. Cada uno de estos utiliza un algoritmo de aprendizaje por refuerzo distinto siendo estos: ϵ**-greedy, Thompson sampling, UCB y EXP3.**

El objetivo por tanto que se perseguirá cuando estos sean utilizados es comparar su rendimiento entre ellos y frente a los otros recomendadores de referencia para ver cómo influye la aplicación de este tipo de soluciones frente a los métodos que más se han utilizado en la literatura.

3.2.1. Recomendador ϵ-greedy

Este recomendador aplica el algoritmo de aprendizaje por refuerzo ϵ**-greedy** para decidir qué brazo será el utilizado para la recomendación. Los brazos que componen esta solución multi-brazo son todos los recomendadores de referencia enumerados anteriormente exceptuando el aleatorio, debido a que en todos los casos produce peores resultados, por lo que se prescindió de él para todos los bandidos utilizados. Por tanto, los utilizados son: voto promedio, kNN basado en usuario y factorización de matrices.

Para elegir que brazo escoger se parte de una muestra aleatoria en el rango [0,1]. Si el valor de dicha muestra es mayor que ϵ, se elige el brazo con la mayor recompensa en ese instante. Para ello, durante cada una de las rondas en las que se hacen las recomendaciones se van guardando los aciertos y fallos con los que mediante la siguiente fórmula permite decidir el brazo a escoger para la recomendación:

$$Recompensa \rightarrow R = \frac{aciertos}{aciertos + fallos} \tag{3.1}$$

En el caso de que la muestra sea inferir a ϵ, se toma de nuevo otra muestra aleatoria en el rango [0,1] y se escoge el brazo donde caiga ese valor. Por ejemplo, en el caso de este recomendador que como se ha comentado tiene 3 brazos, se dividiría la probabilidad de cada brazo de la misma forma, es decir, un 33 %. Si el valor de la muestra cae en el primer 33 %, es decir en el rango [0,0.33] se escogería el primer brazo, siendo escogido el segundo si queda en el rango [0.33,0.66] y el tercero si cae en el rango [0.66,1].

Viendo por tanto esta lógica, se puede deducir que, si se utiliza un parámetro ϵ bajo, el bandido realizará en la mayoría de los casos un ejercicio de explotación en el que escogerá el mejor brazo en ese momento. Pero como se puede observar también en menor medida, pero con una pequeña probabilidad también aplicará exploración, de manera que pueda encontrar ciertos ítems que de otra forma no serían encontrados dando una alternativa que en muchos casos suele ofrecer un mejor rendimiento (Elahi et al, 2016).

Para los experimentos se utilizará un valor de ϵ de 0.1 para favorecer la explotación, aunque de esta manera hay momentos en el que la exploración también entra en juego permitiendo el posible descubrimiento de nuevos caminos óptimos.

3.2.2. Recomendador Thompson sampling

Este recomendador aplica otro de los algoritmos de aprendizaje por refuerzo definidos como es **Thompson sampling**. Al igual que el anterior recomendador, los brazos que hemos utilizado son cada uno de los recomendadores de referencia prescindiendo del aleatorio.

En este caso, en vez de utilizar una distribución beta, al estar trabajando con valores binarios, es decir, relevante o no relevante, acierto o fallo, se va a utilizar una variante de dicha distribución conocida como beta-bernoulli (Brodén et al, 2018).

Aplicando esta distribución, lo que el recomendador hace para la elección del brazo, es escoger una muestra cercana a la media de dicha distribución aplicada a cada uno de los brazos. Para el ajuste de esta distribución y, por tanto, para establecer el valor de dicha media se utilizan los valores de α y β, los cuales se van actualizando en cada una de las rondas. La fórmula para obtener dicha media es:

$$\mu = \frac{\alpha}{\alpha + \beta} \tag{3.2}$$

Una vez calculadas las medias, se obtiene una muestra y con la muestra para cada uno de los brazos se escoge la mayor siendo dicho brazo el utilizado para la recomendación. Una vez hechas las recomendaciones para esa ronda se actualizan los valores de α y β para la siguiente ronda. Se utilizaron como valores iniciales de α=1 y β=1 en los dos casos, para evitar el 0 al comenzar con la recomendación.

3.2.3. Recomendador UCB

Este recomendador sigue la lógica del algoritmo **UCB**. Como los dos previamente analizados, los brazos utilizados por este recomendador son cada uno de los recomendadores de referencia sin el aleatorio.

Este recomendador utiliza la incertidumbre en las predicciones del valor del brazo a escoger para equilibrar la exploración y la explotación. Para ello sigue el principio de optimismo ante la incertidumbre, lo que implica que, si no estamos seguros de una acción, debemos asumir con optimismo que esa acción es la correcta.

Por ejemplo, si se trabaja con los tres brazos comentados antes, los cuales tienen sus correspondientes incertidumbres, de acuerdo con el algoritmo UCB, escogerá de forma optimista el brazo que tenga el límite superior para esas incertidumbres más alto.

De esta manera, o bien obtendrá la recompensa más alta (explotación) obteniendo el mejor resultado para ese momento, o bien, conseguirá aprender sobre ese brazo aumentando o disminuyendo ese límite (exploración).

Hay varias variantes del algoritmo UCB, en este caso, va a ser utilizada la conocida como UCB1 (Kuleshov et Precup, 2014). La fórmula que aplica para obtener la recompensa de cada brazo i en el momento t es:

$$Recompensa \rightarrow U_{i,t} = \mu_i + \sqrt{\frac{2 \cdot log(t)}{N_i}} \tag{3.3}$$

Donde μ_i es la recompensa media estimada del brazo i, N_i es el número de veces que el brazo ha sido elegido hasta el instante de tiempo t, siendo t el instante en el que se está ejecutando el algoritmo (Nguyen-Thanh et al, 2019). Por tanto, el límite de confianza superior de un brazo está compuesto por la recompensa media estimada y el nivel de confianza de dicho brazo.

3.2.4. Recomendador EXP3

Este recomendador sigue la lógica del algoritmo **EXP3**, muy comúnmente utilizado también como algoritmo en bandidos multi-brazo. Igual que el anterior, los brazos utilizados son cada uno de los recomendadores de referencia exceptuando el aleatorio.

Este recomendador funciona manteniendo una lista de pesos asociados a cada uno de los brazos, que son utilizados para decidir aleatoriamente qué brazo escoger en la siguiente recomendación. Se escoge el que mayor recompensa tiene y una vez realizada dicha recomendación, estos pesos son actualizados aproximando a una exponencial para cada uno de los brazos aumentando o disminuyendo dichos pesos según la relevancia de la recomendación.

Además, se introduce un factor γ que se encuentra en el rango [0,1], que ajusta el deseo de elegir una acción de manera uniforme. Tener un valor de γ alto, dará lugar a un límite superior muy bajo, ya que en este caso se produciría demasiada exploración siendo la explotación insuficiente, ocurriendo lo contrario en caso de que γ sea bajo. En este caso se utilizó un valor de γ de 0.1 para favorecer la explotación, pero permitiendo que pudiera existir algún momento en el que se usara exploración.

La probabilidad final de elegir el brazo i en el momento t viene dada por:

$$Recompensa \rightarrow p_{i,t} = (1 - \gamma) + \frac{w_{i,t}}{\sum_{j=1}^{k} w_{j,t}} + \frac{\gamma}{K} \tag{3.4}$$

Donde K es el número de brazos del sistema, y $w_{i,t}$, es el peso del brazo i en el momento t. En cada momento t, el peso del brazo seleccionado es actualizado en base a una regla exponencial dada por (Nguyen-Thanh et al, 2019):

$$w_{i,t+1} = w_{i,t} \cdot exp(\gamma \frac{p_{i,t}}{p_{i,t} \cdot K}) \tag{3.5}$$

4

Evaluación según el ciclo de vida

Los sistemas de recomendación buscan ofrecer a un usuario la información que mejor se adapte a su persona, pero esto puede ser alcanzado de diferentes formas (más o menos efectivas) según el método utilizado o el ciclo de vida seguido hasta su finalización.

En el trabajo se analizarán diferentes formas de plantear dicho ciclo de vida, con el objetivo de ver cuál de ellos se comporta mejor y para comprobar si el uso de soluciones multi-brazo mejora el rendimiento de las soluciones más tradicionales. Para ello, se han planteado 4 modos de evaluar dicho ciclo de vida que más adelante serán explicados más detalladamente.

4.1. Ciclo de vida

El ciclo de vida de un sistema tecnológico, en este caso un sistema de recomendación se refiere a la utilización de una metodología con procesos claramente definidos que permitan crear un software de alta calidad con el menor coste económico y temporal posible. El ciclo suele tener unas fases definidas que pueden observarse en la figura 4.1.

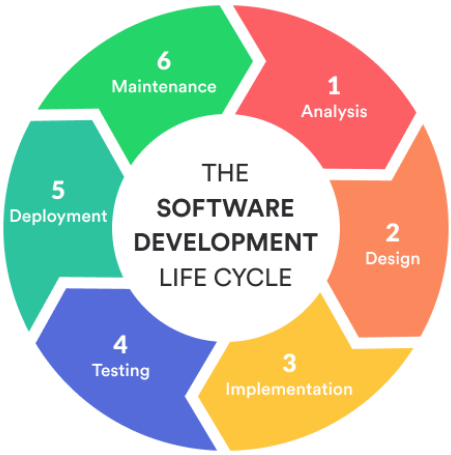

Figura 4.1: Fases del ciclo de vida de un sistema de recomendación

La mayoría de las tecnologías ya en su fase de despliegue donde ya se están utilizando, evolucionan en patrones cíclicos con ciclos más cortos en el producto ya integrado, que a su vez están dentro de un ciclo más largo, el cual engloba todas las fases que han sido mencionadas con anterioridad.

Esta situación, la podemos ver reflejada en los sistemas que se están planteando donde tenemos distintas recomendaciones que proporcionan un feedback al propio sistema (ciclos cortos) que a la larga están implicando un cambio o evolución en el desarrollo del sistema (ciclo largo).

Por tanto, una característica clave de este conjunto de ciclos que forma el ciclo de vida es el patrón evolutivo, ya que dependiendo de la forma en que sea evaluado dicho ciclo de vida, y en particular esos ciclos más cortos, permitirá al sistema ser capaz de mejorar y ofrecer mejores resultados en la tarea de recomendación, cada vez que una recomendación sea necesaria.

4.2. Modos de evaluación

Existen diferentes formas de plantear el ciclo de vida de un sistema de recomendación, lo cual provoca que a la hora de evaluar dichos sistemas se produzcan ciertas diferencias en favor de alguno de ellos. En el trabajo se van a plantear cuatro formas diferentes y a continuación se describirá cada una de ellas con más detalle:

- **Caso 1 - Evaluación offline periódica**: En cada época se evalúan cada uno de los cuatro algoritmos de referencia y se seleccione el que mejor resultado haya obtenido, es decir, el que haya registrado más aciertos. Así, en la siguiente época es ese algoritmo elegido en la anterior el que se ejecuta en las recomendaciones y así sucesivamente. En la figura 4.2, se puede observar la lógica en el ciclo de vida aplicada por este caso.

- **Caso 2 - Test A/B con fracción fija de tráfico**: Se dedica una fracción x de tráfico a evaluación con test A/B y el resto (1-x) a la ejecución de recomendaciones de la actual época. Una vez realizado el test A/B (con todas las variantes) se sustituye la versión actual por el mejor sistema resultante del test para la siguiente época, siempre que además dicho resultado sea mejor que la versión actual. Se sobreentiende que la versión que ha obtenido mejor resultado en el test tiene más tráfico (x) que las demás (1-x). En la figura 4.3, se puede observar la lógica en el ciclo de vida aplicada por este caso.

- **Caso 3 - Test A/B con fracción fija de tráfico en bandido multi-brazo**: Es como el caso anterior, pero usando bandidos multi-brazo en la parte del test A/B, es decir que la fracción de tráfico x en vez de repartirse uniformemente entre las variantes comparadas, se va modificando estocásticamente en favor de la mejor versión. Así, se ejecuta el brazo seleccionado y al terminar la época, pasa a ejecutarse el brazo con mejor resultado obtenido del test A/B

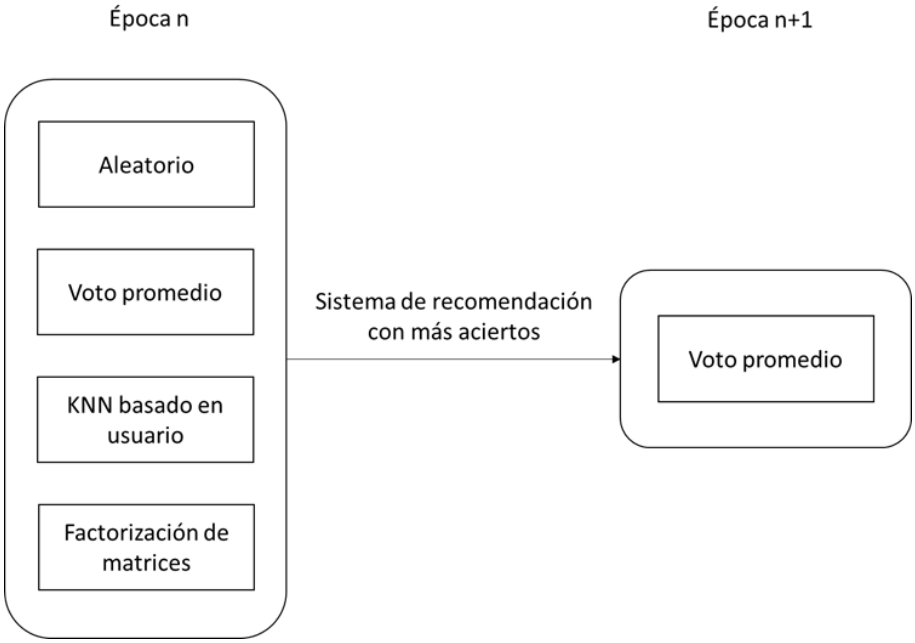

Figura 4.2: Figura ilustrativa del modo de evaluación para el caso 1

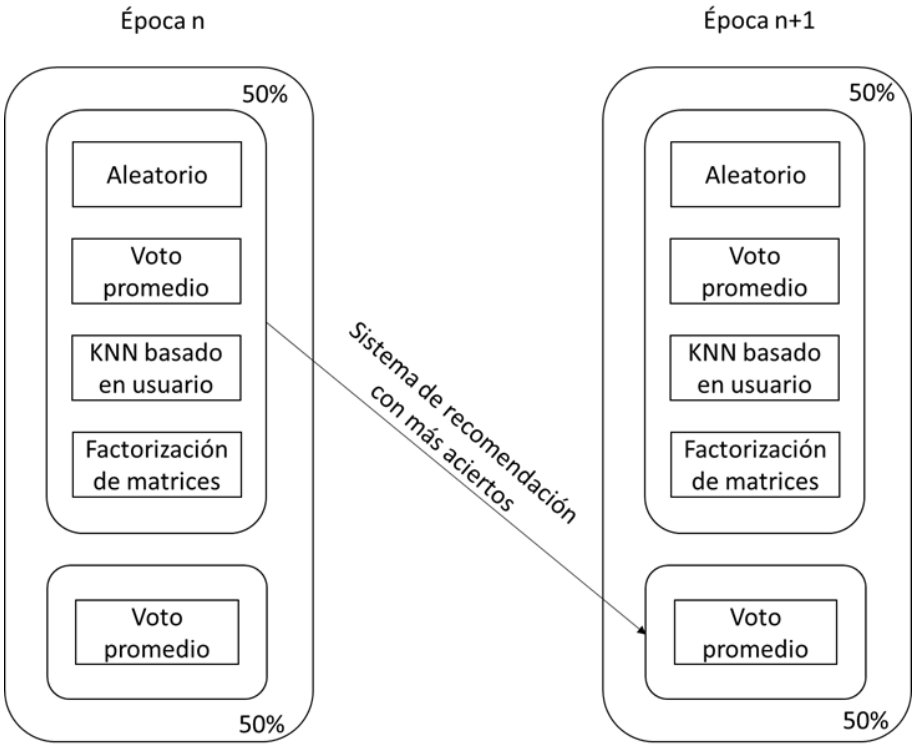

Figura 4.3: Figura ilustrativa del modo de evaluación para el caso 2

que, en este caso, realizan los propios brazos de bandido. En la figura 4.4, se puede observar la lógica en el ciclo de vida aplicada por este caso.

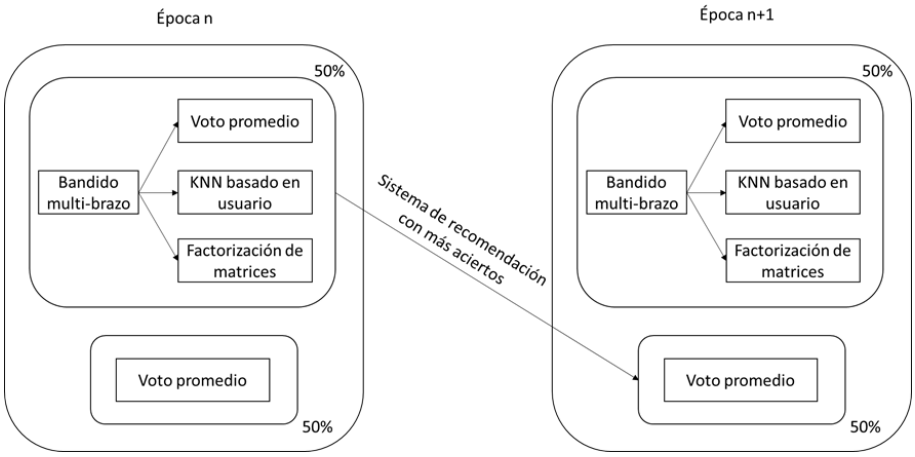

Figura 4.4: Figura ilustrativa del modo de evaluación para el caso 3

- **Caso 4 - Test A/B en bandido multi-brazo**: En este caso, se sigue la lógica del caso anterior, pero con la diferencia de que se deja el bandido multi-brazo de manera indefinida, es decir, que él elija en cada época qué brazo usar en el siguiente período, sin existir una fracción fija de tráfico. Este es el caso más común utilizado a la hora de evaluar bandidos-multi-brazo. En la figura 4.5, se puede observar la lógica en el ciclo de vida aplicada por este caso.

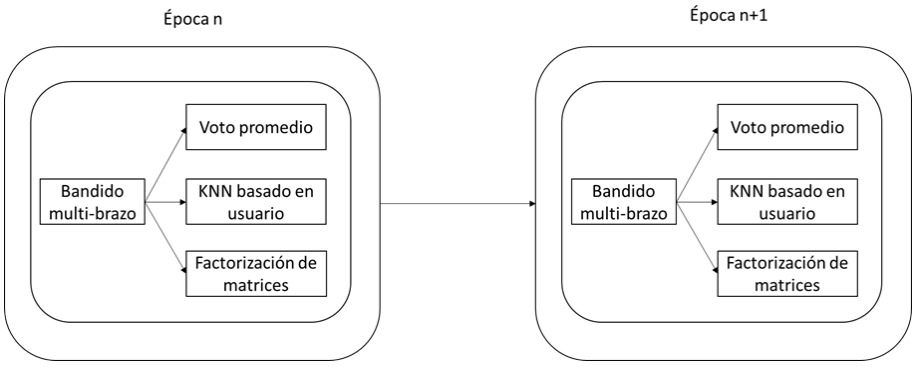

Figura 4.5: Figura ilustrativa del modo de evaluación para el caso 4

5

Experimentos

5.1. Conjunto de datos

Para la realización de los experimentos y la posterior comparación de cada uno de los modos implementados, se ha escogido como conjunto de datos un conjunto conocido como CM100k [1]. Este conjunto trabaja con las valoraciones de usuarios reales de canciones de diferentes géneros recopilado todo ello en un archivo que será con el que se va a trabajar.

Para ser más exactos, dicho archivo contiene exactamente 103.584 valoraciones de .084 canciones que fueron valoradas por 1.053 usuarios reales. El conjunto en sí está bastante equilibrado, ya que hay en la mayoría de los casos unas 100 valoraciones por usuario y de la misma manera, hay unos 100 usuarios que han valorado cada ítem.

Esto provoca un efecto positivo para trabajar con este conjunto y es que, gracias a ello, no presenta sesgos en la distribución del número de valoraciones. Esta es una de las grandes razones por las que se va a utilizar este conjunto en los experimentos, ya que, en otros conjuntos públicos, sí que existe este sesgo que afecta negativamente a los experimentos (Cremoensi et al. 2010).

Las valoraciones de canciones registradas en el conjunto van desde 1 en el caso de la valoración más baja, es decir, que no es del gusto del usuario, hasta 4 en el caso de la valoración más alta.

Como se ha comentado, es necesario tratar dichas valoraciones como relevantes o no relevantes para el posterior proceso de evaluación por lo que, en este caso, serán consideradas como relevantes aquellas valoraciones con un valor de 3 o 4, mientras que serán tomadas como no relevantes las valoraciones con un valor de 1 o 2. Así, al tratar la relevancia como un proceso binario en caso de que la valoración sea relevante será tratado como un 1, mientras que si no fuera relevante sería tomado como un 0.

Un ejemplo de cómo vienen dados los datos en el conjunto de datos original, antes de hacer todas las transformaciones comentadas anteriormente es el siguiente:

[1] El conjunto de datos está disponible públicamente en http://ir.ii.uam.es/cm100k

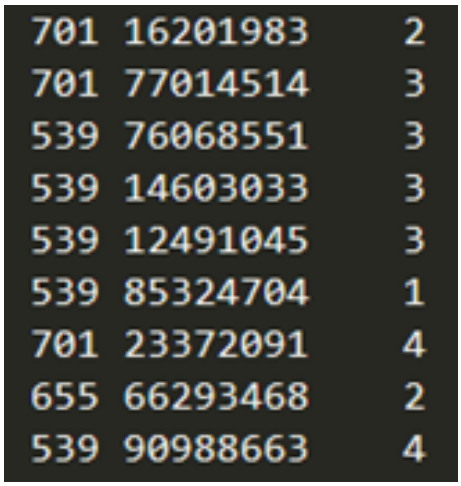

Figura 5.1: Formato del fichero del conjunto de datos

Como vemos en la figura 5.1, el fichero necesita 3 elementos, donde la primera columna representa en identificador de usuario del sistema, la segunda el identificador de ítem del sistema y, por último, la tercera, que representa la valoración dada por ese usuario para el ítem que se indica. De esta forma acumulamos cada una de las valoraciones registradas en el conjunto de datos que serán aportadas con sus correspondientes transformaciones al sistema.

5.2. Método de evaluación

Mediante los experimentos realizados se busca comprobar el rendimiento de los algoritmos presentados en el capítulo 3, dependiendo de las diferentes configuraciones que pueden ser planteadas en el ciclo de vida de un recomendador descritos en el capítulo 4, todo ello como paso previo a una evaluación online con usuarios reales.

Por tanto, en estos experimentos se busca conseguir una configuración que se asemeje lo máximo posible a unas futuras condiciones que podría tener el sistema en una evaluación online o el propio sistema ya en producción.

Por eso, en los experimentos la lógica que se sigue es intentar simular una posible interacción entre los usuarios con el sistema, de manera que simulamos de forma iterativa, las forma en la que los usuarios valoran las posibles sugerencias de ítems que le puede dar el sistema.

Esta información se va almacenando en el sistema lo que a su vez permite que este cada vez posea más información de sus usuarios y aplicando cada uno de los métodos y algoritmos comentados, sea capaz de mejorar sus futuras recomendaciones.

Para poder realizar la evaluación como se comentaba en el estado del arte, es necesario separar los datos con los que se va a trabajar en una partición aleatoria de entrenamiento y test, donde el

conjunto de entrenamiento servirá para que los recomendadores con la información que poseen den una predicción al usuario del ítem a recomendar, mientras que el conjunto de test servirá para verificar la precisión de dichas predicciones siendo incluidas en el conjunto de entrenamiento una vez recomendadas.

Como se viene comentando, se busca adecuar el sistema los más posible a un futuro sistema real, por lo que para reflejarse a ello esta partición que antes hemos comentado se aplica sobre los usuarios, porque la situación lógica es que trabajamos sobre los usuarios que se incorporan al sistema, que al final son los actores principales.

En cuanto a la división del conjunto de datos en entrenamiento y test, probamos con varios tamaños para ver cómo se comportaban según la fase del ciclo de vida en el que se encontraran. Estos tamaños son 10-90, 20-80, 50-50, 80-20, 90-10.

Estos valores quieren decir por ejemplo para el 10-90 que, de forma aleatoria, un 10 % de las valoraciones de los ítems de cada uno de los usuarios son asignadas al conjunto de entrenamiento, mientras que el 90 % de las valoraciones restante de usuarios son asigna-das a test.

Esto sería similar a un estado temprano del sistema, donde el usuario aún no ha interactuado mucho con el sistema, por lo que este no posee mucha información suya (ni de resto de usuario). En este caso, es esperable que la predicción ofrecida no sea tan precisa como en un estado más avanzado del sistema, donde el sistema sí que tendría bastante más información de cada uno de los usuarios que lo integran.

En el caso de los modos de evaluación en los que se utiliza una fracción fija del tráfico como son el caso 2 y 3, también se realiza una partición para poder trabajar en ellos. En este caso, se toma en todas las pruebas esa partición inicial como un 50-50, es decir, 50 % de los usuarios para la ejecución del algoritmo seleccionado en ese momento y un 50 % para realizar el test A/B que decide el algoritmo a utilizar en la siguiente época.

En cada ronda o época, la lógica que se sigue es que, para cada uno de los usuarios del sistema, se realice una recomendación sobre la predicción realizada y dicho elemento encontrado en test, sea añadido al conjunto de entrenamiento para que el sistema tenga más información en la siguiente época. Se considera como una época a realizar una recomendación sobre cada uno de los usuarios del conjunto de test.

En cuanto a las épocas utilizadas en la ejecución, realizamos varias pruebas para ver cómo se comportaban el sistema utilizando finalmente 500 épocas para cada uno de los sistemas probados, ya que es un número suficiente de recomendaciones para cada usuario y además con valores mayores, era mucho el tiempo utilizado en la ejecución.

5.3. Resultados

Los experimentos realizados buscan evaluar y comparar el funcionamiento de los ciclos de vida planteados para comprobar si las soluciones multi-brazo funcionan mejor que otras alternativas y si, por tanto, sería interesante usarlos de cara a una futura evaluación online.

Para ello, hay varias formas de analizar este tipo de evaluaciones. En este caso lo que se va a hacer es observar la velocidad con la que se descubren los ítems relevantes, es decir, lo rápido que es capaz de encontrar y sugerir ítems relevantes en el desarrollo de toda la ejecución del experimento. Los ítems pueden ser descubiertos en un momento diferente según el recomendador utilizado. Esto es muy importante en recomendación, ya que, si los ítems relevantes tardan mucho tiempo en ser recomendados, existen muchas probabilidades de que el usuario al no recibir recomendaciones que le interesen se termine cantando y abandonado el sistema.

Como se comenta en el apartado 5.2, se plantean varios tamaños de entrenamiento y test para cada una de las configuraciones (ciclos de vida) planteados, lo cual nos permite ver cómo evolucionan según la cantidad de información que posee el sistema.

Por ejemplo, para una configuración 10-90, el sistema aún no tiene mucha información por lo que cabe dentro de los esperable que los resultados sean más volátiles, mientras que, en su caso homónimo, 90-10, el sistema ya está llegando a su fin por lo que tiene mucha información y se espera mucho más preciso y con mejores resultados en cuanto a métricas se refiere.

La métrica que va a ser utilizada para la evaluación es el recall. Debido a que conocemos el número acumulado de aciertos y el número total de ítems relevantes, podemos representar en una gráfica la curva formada por cada uno de los recomendadores utilizando un rango [0,1] y así interpretarlo de una manera más cómoda.

De esta manera también se pueden obtener gráficas que permitan contrastar cada uno de los sistemas de recomendación planteados según su ciclo de vida, así como compararlos para ver el efecto que tienen los bandidos multi-brazo en este tipo de sistemas.

Además, estos planteamientos se pueden comparar también con los sistemas de recomendación de referencia comentados, para ver su rendimiento frente a un sistema de recomendación tradicional con el ciclo de vida natural de un sistema de recomendación.

Para realizar el análisis del ciclo de vida de cada uno de los sistemas comentados, he decidido ir analizando los diversos puntos en los que se puede encontrar el ciclo de vida del sistema para ver cómo puede influir la información que posee el sistema en el resultado final.

Así, a continuación, plantaré diferentes apartados según el punto del ciclo de vida en el que se encuentre el sistema. Estos puntos analizados tendrán un 10 %, 20 %, 50 %, 80 % y un 90 % de la información comparando cada una de las alternativas.

Todos los resultados de las pruebas que se analizarán a continuación son resultado del cálculo de la media tras 3 ejecuciones, para obtener resultados más precisos, ya que siempre puede producirse alguna diferencia entre las ejecuciones al hacerse particiones aleatorias.

5.3.1. Ciclo de vida al 10 %

En este punto el sistema tiene un 10 % de la información, es decir, el conjunto de entrenamiento del conjunto de datos con el que estamos trabajado tiene un tamaño del 10 %, mientras que el tamaño del conjunto de test se corresponde con el 90 % restante.

Esta situación se corresponde con estado temprano del sistema, donde aún tiene pocos datos por lo que aprendizaje del sistema en este punto es algo más lento. En esta situación del ciclo de vida se espera que el sistema pueda ser algo más inestable debido a la poca información que posee.

Una vez establecida la división, ejecutamos cada uno de los sistemas tanto los de referencia como una evaluación típicamente offline, como cada uno de los casos que se plantean en los diferentes ciclos de vida.

Antes de mostrar la comparación entre todas las opciones con las que estamos trabajando, se elegirá uno de los recomendadores multi-brazo (el que mejor resultados de en promedio) para comparar todos los resultados con los casos en los que se utiliza este tipo de soluciones.

Para todos ellos, como la diferencia en las primeras épocas no es muy diferente, lo que podría provocar en caso de que alguno quedara rezagado que el usuario se cansará de malas recomendaciones y abandonara, nos centramos en compararlos al final de la ejecución, es decir en la época 500.

En los casos 3 y 4, los resultados obtenidos en dicha época para cada una de las soluciones multi-brazo son los reflejados en las figuras 5.2 y 5.3 respectivamente.

Como se puede observar en el primer caso el que mejor resultado da es UCB, estando algo más destacado también en el caso 4, por lo que es la solución elegida para comparar con los demás casos.

Esto puede ser debido a que como es poca la información que se posee, establecer límites de confianza de manera que se es optimista parece funcionar mejor que otros algoritmos donde tener mucha información prima más.

Decidido esto, comparamos todas las soluciones que tras su completa ejecución producen los resultados reflejados en la figura 5.4.

Podemos ver como el aleatorio queda muy por debajo del resto de recomendadores, lo cual es esperable, ya que, al no seguir ninguna estrategia, tarda más en descubrir los ítems relevantes. Este nos puede servir de referencia para ver como mejoran cada uno de los recomendadores respecto a una recomendación base.

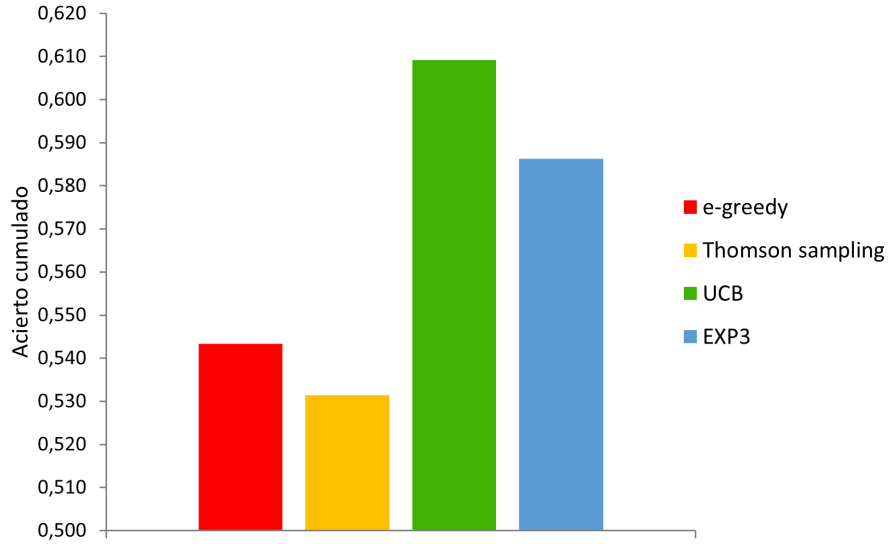

Figura 5.2: Comparativa soluciones multi-brazo con un 10 % de información para el caso 3

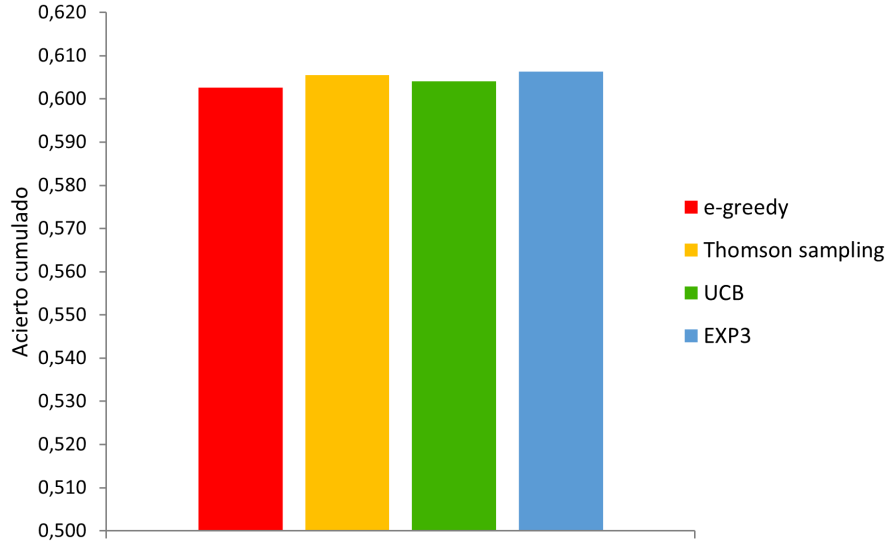

Figura 5.3: Comparativa soluciones multi-brazo con un 10 % de información para el caso 4

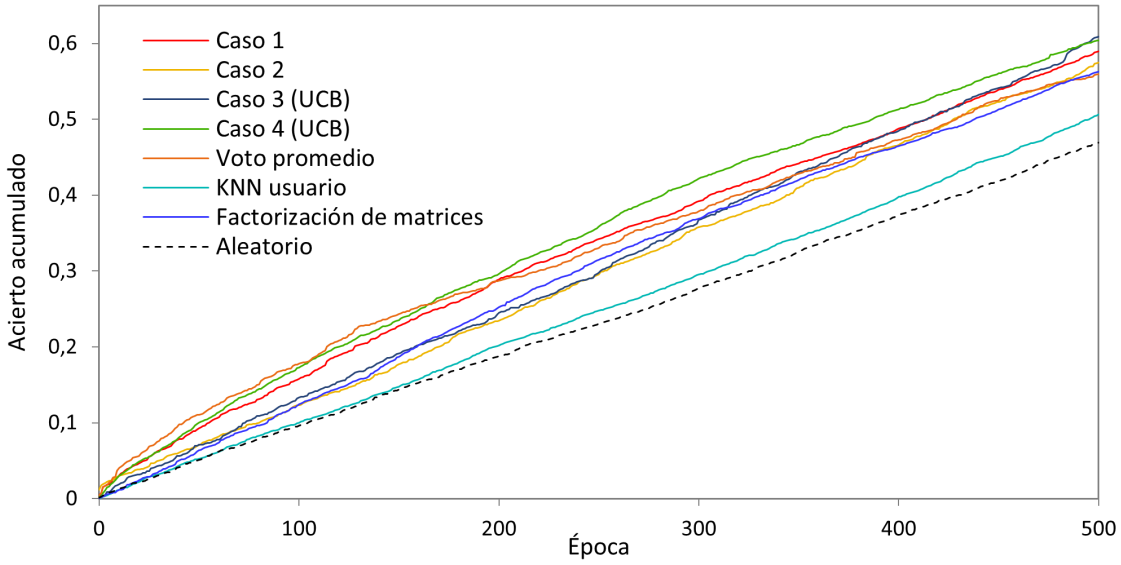

Figura 5.4: Evaluación de los sistemas con un 10 % de la información del conjunto de datos

Lo mismo pasa con el kNN basado en usuarios, ya que al haber un 10 % de la información, el algoritmo no tiene mucha información del resto de los usuarios como para compararlos, por lo que es esperable que no acierte tanto.

Entre los recomendadores restantes, sí que vemos que la diferencia existente entre ellos es menor. Por tanto, para analizar esa diferencia más en detalle, nos situamos al final del experimento (época 500), donde es esperable que hayan adquirido más información, para así poder observarlo más claramente en la figura 5.5.

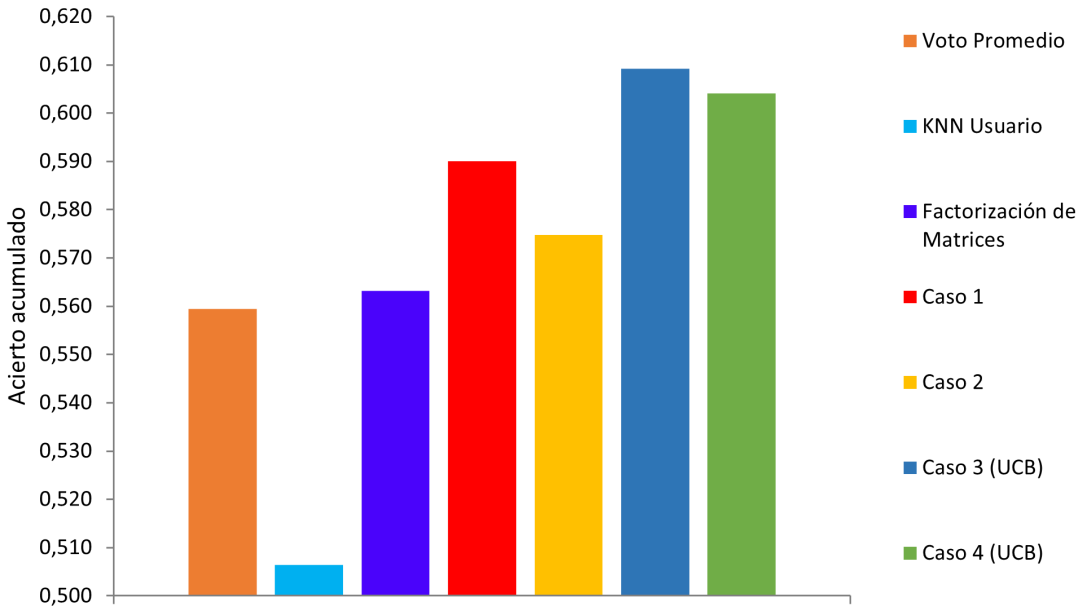

Figura 5.5: Acierto acumulado de los sistemas en la última época para el 10 % de información

Como se observa los 3 recomendadores de referencia, ya que el aleatorio no lo representamos debido a su pobre rendimiento, quedan algo por debajo de los recomendadores de cada uno de los casos que estamos planteando a lo largo del trabajo.

Esto en parte es lo esperable, ya que estos, aunque solo ejecutan uno de los algoritmos con los que trabajan sí que hacen las veces de ensemble por lo que tienen más posibilidades para mejorar la recomendación. Aunque esto no es siempre así, ya que en algún caso puede que haya un algoritmo que sea muy bueno y al utilizar en algún caso alguno de los algoritmos provoque que a la larga acabe empeorando el rendimiento.

Por otro lado, analizando los resultados de cada uno de los modos de ciclo de vida planteados en el capítulo 4.2, podemos ver como los casos en los que se emplea técnicas de bandidos multi-brazo se obtienen mejores resultados que las variantes offline y de test A/B planteadas. Esto nos permite comprobar que, de momento, para un estado temprano del sistema, este tipo de soluciones multi-brazo ya está destacando.

Parece, por tanto, que el uso de estos algoritmos de aprendizaje por refuerzo, gracias al uso que le dan al concepto de recompensa y al aprendizaje que le permite establecer esta sobre el sistema, consiguen solucionar juntos las carencias que pueden tener cada uno de los recomendadores de referencia que establecen en sus brazos.

5.3.2. Ciclo de vida al 20 %

Continuando con el ciclo de vida será analizado ahora el sistema con un 20 % de la información, es decir, el conjunto de entrenamiento tiene un tamaño del 20 %, mientras que el tamaño del conjunto de test se corresponde con el 80 % restante.

Aunque con algo más de información este punto se corresponde también con un sistema en su fase de comienzo, pero con más información como es de esperar que en el apartado anterior. El aprendizaje del sistema sigue siendo lento, pero en este caso ya se va esperando que el sistema se estabilice ya que posee algo más de información.

De nuevo, tras realizar la correspondiente división, se ejecutan cada uno de los sistemas de referencia, así como cada uno de los casos con sus variantes. De igual manera que antes, se elegirá el mejor de los recomendadores multi-brazo para comparar con el resto de las soluciones para recomendar planteadas.

Nos centramos en comparar cada una de las soluciones multi-brazo al final de su ejecución, es decir en la época 500. Así, en los casos 3 y 4, los resultados obtenidos en dicha época para cada una de las soluciones multi-brazo son los observados en las figuras 5.6 y 5.7 respectivamente.

En el caso 3 el resultado óptimo es cumplido por el recomendador que aplica EXP3, siendo el

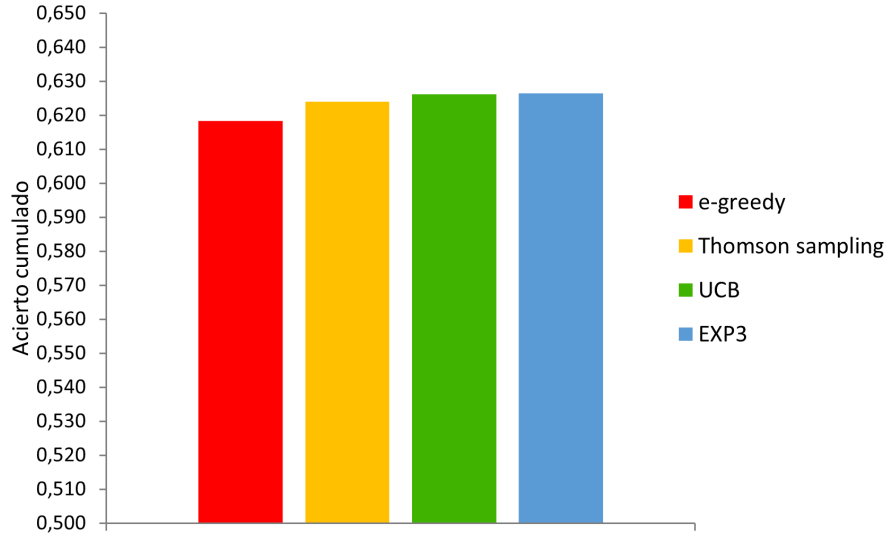

Figura 5.6: Comparativa soluciones multi-brazo con un 20 % de información para el caso 3

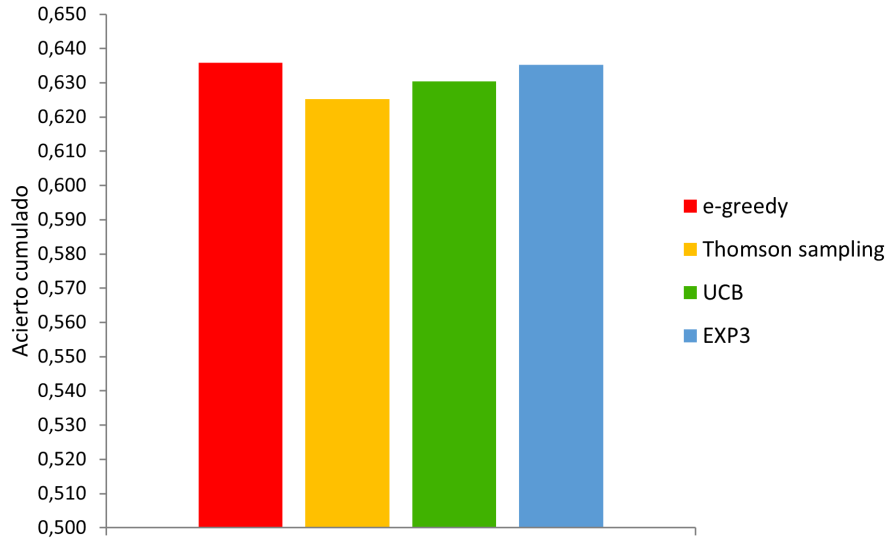

Figura 5.7: Comparativa soluciones multi-brazo con un 20 % de información para el caso 4

segundo, pero muy cerca del primero en el caso 4. El caso de ϵ-greedy, ganador en el segundo caso es el peor en el primero, por lo que se utilizará la solución EXP3 para comparar con los demás casos.

Elegido el tipo de bandido-multi-brazo, lo comparamos para cada uno de los casos con el resto de recomendadores, obteniendo los resultados mostrados en la figura 5.8.

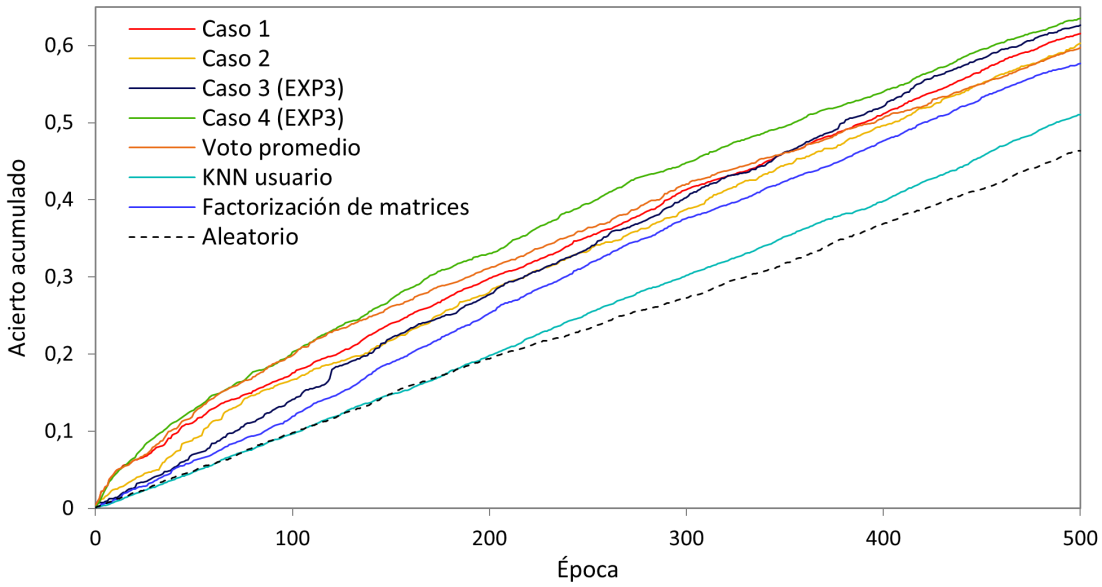

Figura 5.8: Evaluación de los sistemas con un 20 % de la información del conjunto de datos

El comportamiento de los recomendadores aleatorio y kNN, sigue siendo el mismo, pero en este caso, en la parte alta de la gráfica se pueden ver pequeñas diferencias que podemos analizar más claramente en el final del experimento como en el apartado anterior, es decir, en la época 500, resultado reflejado en la figura 5.9.

En la figura, podemos ver como en este caso, el caso 2, queda por debajo del caso 1 y muy cerca del recomendador de voto promedio. En este caso parece que el test A/B que plantea este método no es muy efectivo, lo que parece indicar que ha optado por elegir uno de los algoritmos tradicionales que peor rendimiento ofrece, posiblemente el kNN, lo que ha repercutido en su rendimiento final, frente por ejemplo el voto promedio.

Este recomendador de voto promedio como ya vamos viendo, parece ser el que mejor resultados ofrece de los algoritmos tradicionales. Esto por otra parte es esperable para un estado temprano del sistema, ya que este algoritmo ofrece los ítems más populares entre los usuarios y como aún faltan muchos por descubrir es esperable que devuelva muchos que cumplen estas condiciones.

Por otro lado, analizando las soluciones multi-brazo, lo cual es el objetivo, se puede ver que, de nuevo, son los recomendadores que mejor resultado han obtenido, por lo que al menos de momento parece que al menos para una situación temprana del sistema, es interesante utilizar este tipo de

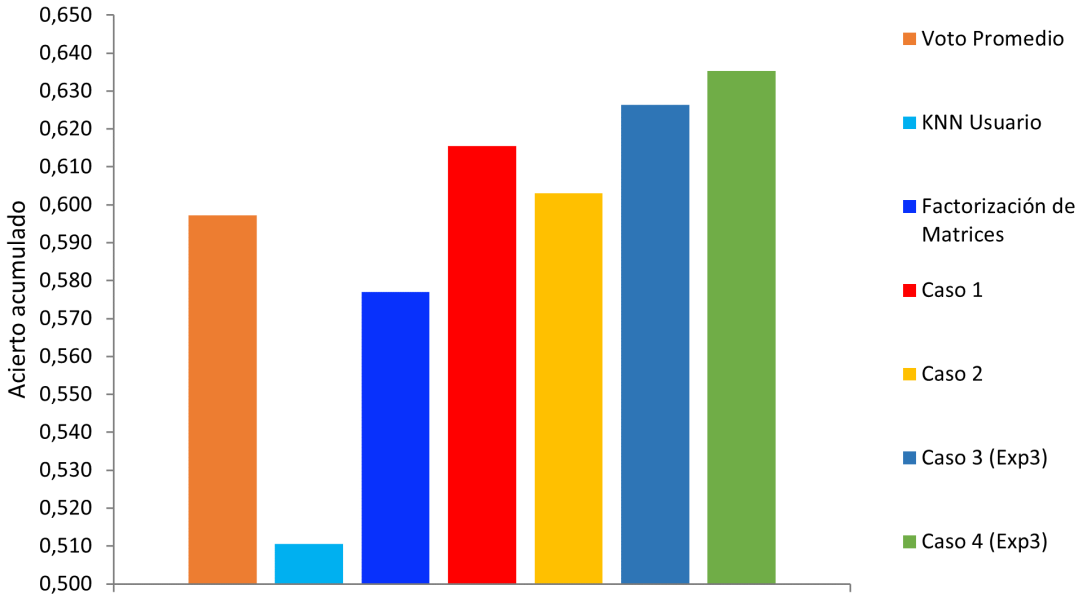

Figura 5.9: Acierto acumulado de los sistemas en la última época para el 20 % de información

soluciones en su ciclo de vida.

En cuanto a la diferencia entre ellos dos, vemos que el caso 4, donde se ejecuta sin una fracción fija de tráfico y de la manera natural que se usa ese tipo de soluciones, parece que está ofreciendo mejores resultados, aunque como hemos visto en la selección del bandido entre todas las alternativas, esto varía mucho, pero en el caso de las mejores soluciones si se está cumpliendo.

5.3.3. Ciclo de vida al 50 %

A continuación, será analizado el comportamiento del sistema con un 50 % de la información, por lo que el conjunto de entrenamiento tendrá un tamaño del 50 %, mientras que el tamaño del conjunto de test se corresponde con el 50 % restante, es decir, a partir del instante en el que el sistema ha cumplido la mitad de su ciclo de vida.

Por tanto, se puede decir que se corresponde con un sistema en su fase intermedia, donde ya posee una cantidad considerable de información, pero aún le queda lo mismo para conocer exactamente toda la información (aproximadamente ya que la partición es aleatoria, es decir, alrededor de ese valor).

Tras la partición, se ejecutan cada uno de los sistemas que estamos probando. Elegimos también como en los casos anteriores los recomendadores multi-brazo que se van a utilizar para comparar con el resto de las soluciones para recomendar planteadas. Por tanto, en los casos 3 y 4, para el final de la ejecución los resultados obtenidos para las diferentes variantes son los mostrados en la figura 5.10 y 5.11 respectivamente.

Como se puede ver en este caso la decisión sobre el recomendador multi-brazo que mejor funciona

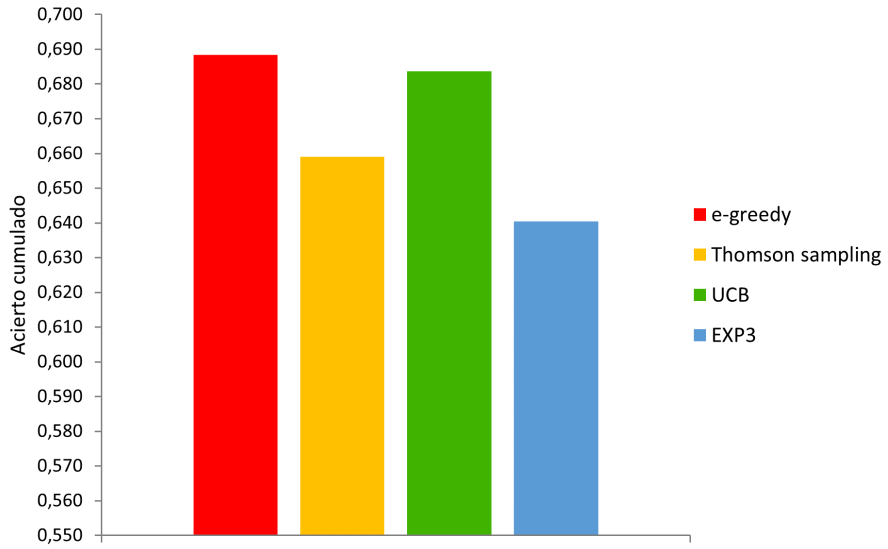

Figura 5.10: Comparativa soluciones multi-brazo con un 50 % de información para el caso 3

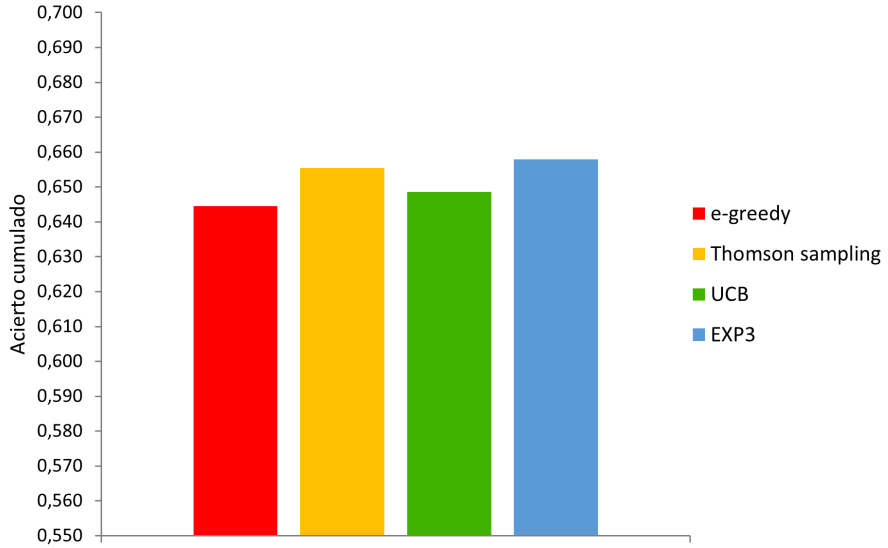

Figura 5.11: Comparativa soluciones multi-brazo con un 50 % de información para el caso 4

es complicado, ya que el que funciona muy bien en uno de ellos, baja su rendimiento en el otro.

Esto en parte puede ser fruto de que el sistema se encuentra en la mitad por lo que todos los recomendadores multi-brazo tienen comportamientos y rendimientos similares al contar con más o menos la mitad de la información.

Por tanto, en este caso, ya que en la figura 5.10, donde se representa el caso 3 hay un claro ganador que es ϵ-greedy con un valor alto y en la figura 5.11, donde se representa el caso 4 están todos bastante equilibrados, escogemos este para las posteriores comparaciones.

Comparando cada uno de estos casos con el resto variantes, obtenemos los resultados observados en la figura 5.12.

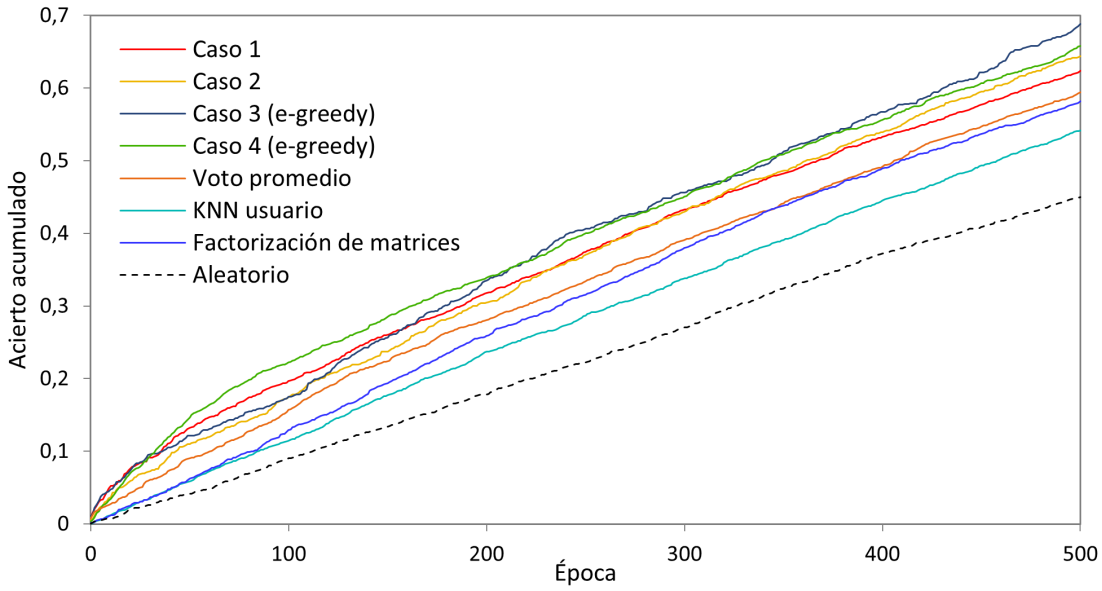

Figura 5.12: Evaluación de los sistemas con un 50 % de la información del conjunto de datos

En este caso, ya a simple vista vemos como los 4 casos planteados empiezan a marcar diferencia con los recomendadores tradicionales, lo cual ya de primeras nos empieza a confirmar algo que parece esperable pero no siempre se cumple y es que el uso de soluciones que combinan varios algoritmos, en caso de poseer suficiente información mejoran a las soluciones individuales.

Además, el recomendador de factorización de matrices cada vez está más cerca del que están siendo el mejor recomendador tradicional, voto promedio, ya que al tener más información del sistema es capaz de descubrir más factores que le permitan mejorar la recomendación.

Aun así, todos estos detalles pueden verse más claramente analizando la situación en el final del experimento como se puede ver en la figura 5.13.

Algo que comentar antes de pasar al análisis comparativo de cada una de las variantes, es que, si nos fijamos en el eje independiente de la gráfica anterior, así como del resto, en el que se mide

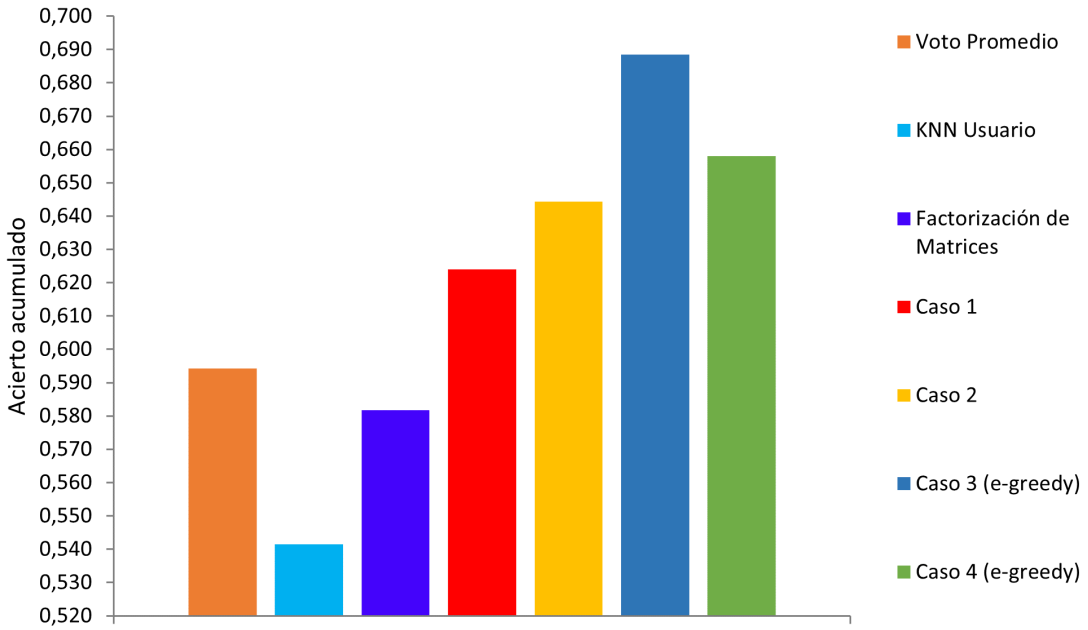

Figura 5.13: Acierto acumulado de los sistemas en la última época para el 50 % de información

el acierto acumulado, podemos ver que cuanto más avanza el ciclo de vida, el valor que se está representando es más alto.

Esto es debido a que como cada vez el sistema posee más información en el conjunto de entrenamiento, es capaz de hacer predicciones de los ítems a recomendar más precisas, lo que permite descubrir más ítems relevantes en las mismas épocas. Esto permite comprobar también que el sistema se está comportando de una manera correcta en lo que va de ciclo de vida.

Analizando los diferentes casos de ciclo de vida planteados vemos como sigue cumpliéndose las dos situaciones que se vienen observando, donde estos casos mejoran a los recomendadores tradicionales individuales y los recomendadores que aplican soluciones multi-brazo, están algo por encima de los que aplican evaluación offline y online basada en test A/B, caso 1 y 2.

En cuanto a los casos que aplican bandidos multi-brazo, se observa un cambio de tendencia que habrá que observar en el ciclo seguido, ya que el caso 3, con fracción fija de tráfico que es elegido por el bandido, está siendo mejor que el caso 4, en el que es todo el tráfico es utilizado directamente por el bandido multi-brazo.

5.3.4. Ciclo de vida al 80 %

En este apartado será analizado cómo se comporta el sistema con un 80 % de la información, siendo el tamaño del conjunto de entrenamiento del 80 % del total de información, mientras que el del conjunto de test será del 20 % restante.

Debido a esto, podemos ubicar esta situación en una fase avanzada del sistema, donde este tiene ya muchísima información, más de la que desconoce, por lo que se espera que sea capaz de hacer recomendaciones bastante acertadas.

Realizada la correspondiente división de la información, se realiza la ejecución de todos los sistemas que estamos probando y comparando. De nuevo se elige entre los casos que así lo precisan, los recomendadores multi-brazo que se van a utilizar para así poder comparar con el resto de las variantes y casos planteados. Así los resultados obtenidos al finalizar su ejecución en cada una de las variantes de los casos 3 y 4 los podemos ver en las figuras 5.14 y 5.15 respectivamente.

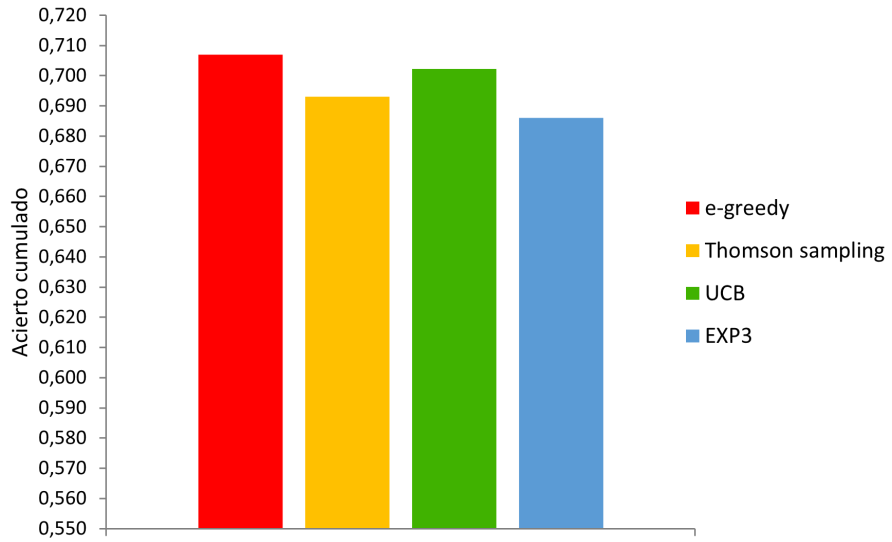

Figura 5.14: Comparativa soluciones multi-brazo con un 80 % de información para el caso 3

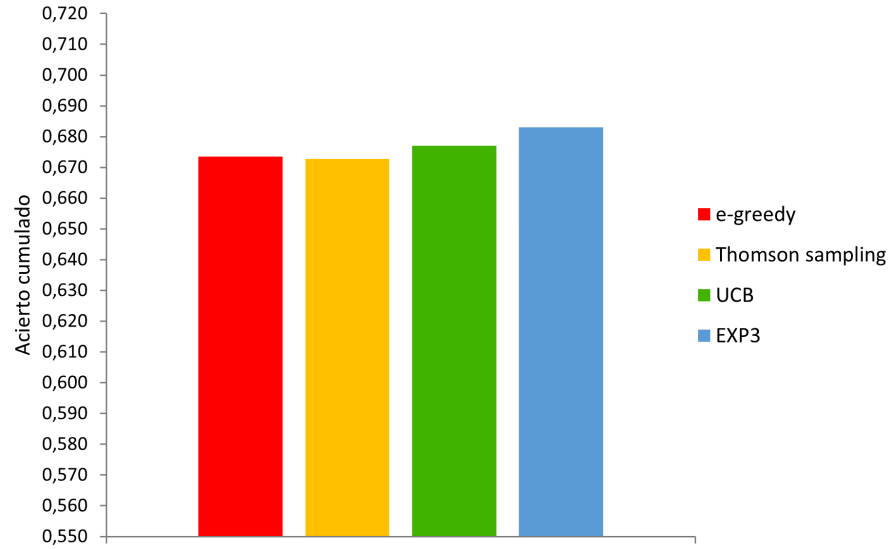

Figura 5.15: Comparativa soluciones multi-brazo con un 80 % de información para el caso 4

De nuevo la elección del mejor de los cuatro es complicada, ya que en este punto el sistema, al

poseer más información, provoca que los rendimientos sean bastante buenos pero similares, lo que complica la elección.

En este caso, parece que UCB, es el más estable ya que en los dos casos queda en segunda posición, mientras que los otros si en un caso están muy arriba, en el otro obtienen peor resultado que el resto.

Por tanto, el recomendador escogido será el que aplica UCB que comparándolo para estos dos casos junto con el resto de las opciones que se están planteando, devuelve los resultados observados en la figura 5.16

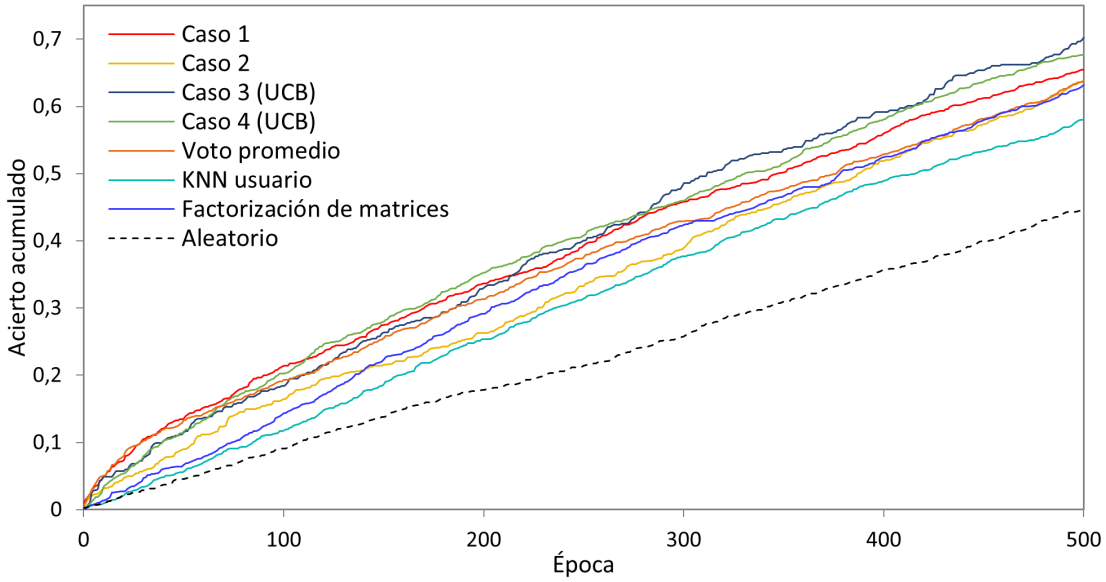

Figura 5.16: Evaluación de los sistemas con un 80 % de la información del conjunto de datos

A diferencia de los instantes del ciclo de vida analizados anteriormente, se puede observar que el caso 2, dónde se realiza el ciclo de vida sobre un test A/B con fracción de tráfico fija, tienen un rendimiento más bajo que el resto de los casos, tendiendo a tener el mismo rendimiento que los recomendadores de referencia de voto promedio y factorización de matrices.

Además, se sigue la tendencia que ya con el 50 % de información comentábamos y es que el recomendador de factorización de matrices se acerca y está prácticamente a punto de superar al de voto promedio. Esto junto a lo comentado del caso 2, podemos observarlo más claramente en el final del experimento como se puede ver en la figura 5.17.

Esto permite confirmar la tendencia de la factorización de matrices que se ha comentado, pero nos permite ver también la cercanía entre los sistemas del caso 2 y el voto promedio.

Si analizamos la figura 5.16 en detalle, se observa el pobre comienzo que tiene el caso 2, lo que puede indicar que el problema haya sido que, en las rondas iniciales, se haya escogido el algoritmo

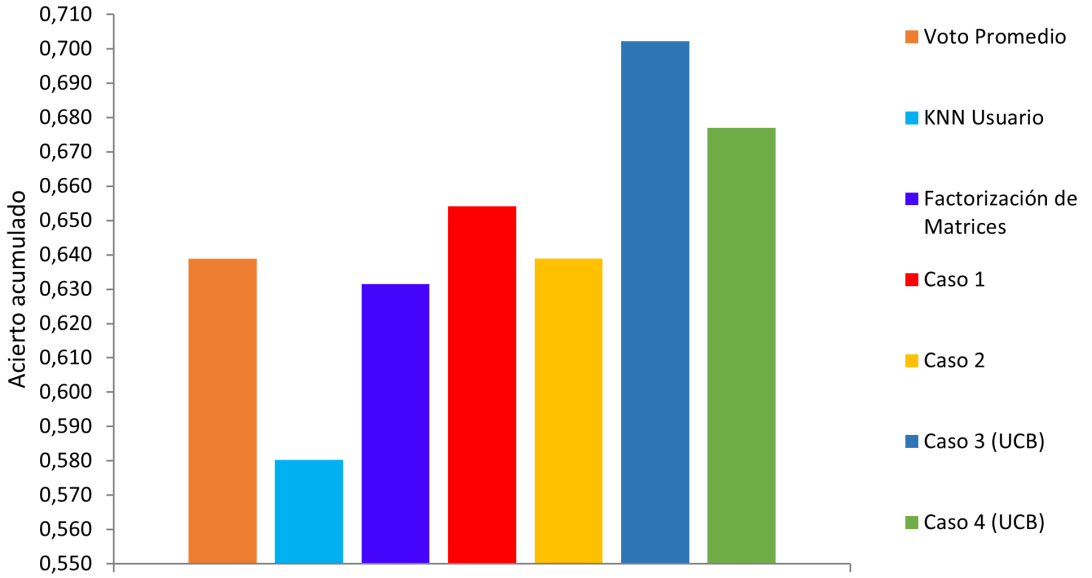

Figura 5.17: Acierto acumulado de los sistemas en la última época para el 80 % de información

de kNN, el cual como se puede ver está ofreciendo los peores resultados. Pero como se puede ver en épocas más avanzadas la fracción de tráfico de prueba ha rectificado, volviendo a algoritmos con más acierto, pero arrastrando dicho fallo que le hace alcanzar este pobre rendimiento.

Por otro lado, se confirma la tendencia del apartado anterior, ya que vemos como la distancia de los casos con bandidos multi-brazo respecto al resto se mantiene y también como el caso 3 parece que ha seguido con su tendencia ascendente mejorando los resultados del caso 4.

5.3.5. Ciclo de vida al 90 %

Finalizando con el ciclo de vida será analizado ahora el sistema con un 90 % de la información, es decir, un conjunto de entrenamiento con un tamaño del 90 % y un conjunto de test cuyo tamaño será el 10 % restante.

En este punto se puede decir que el sistema se encuentra en una fase muy avanzada, por no decir que está en su fase final. En este caso, el sistema posee ya la gran mayoría de la información por lo que el aprendizaje será mucho más rápido y efectivo.

Tras la partición, se ejecutan cada uno de los sistemas que estamos probando. Elegimos también como en los casos anteriores los recomendadores multi-brazo que se van a utilizar para comparar con el resto de las soluciones para recomendar planteadas. Por tanto, en los casos 3 y 4, tras la ejecución de ambos, para el final de la ejecución los resultados obtenidos en las diferentes variantes son los observados en las figuras 5.18 y 5.19 respectivamente.

A la hora de elegir el mejor recomendador para ambas opciones si nos fijamos sobre todo en la

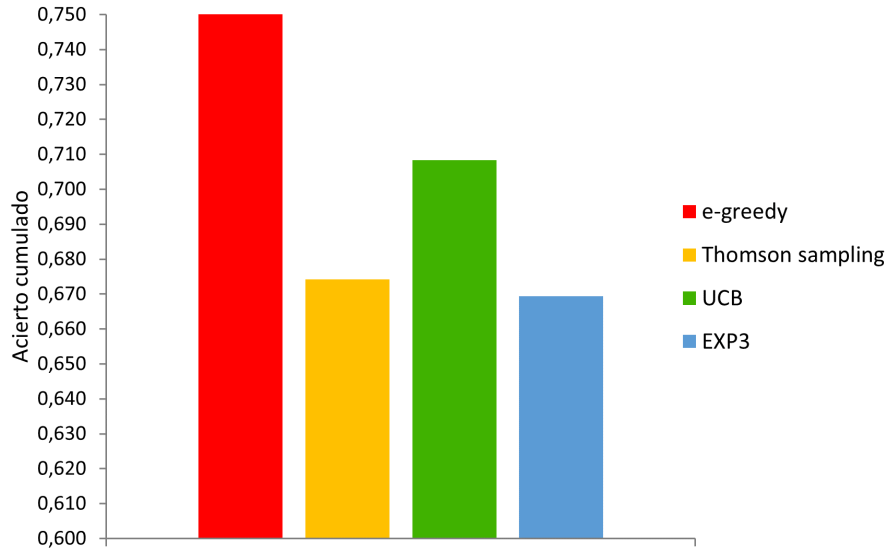

Figura 5.18: Comparativa soluciones multi-brazo con un 90 % de información para el caso 3

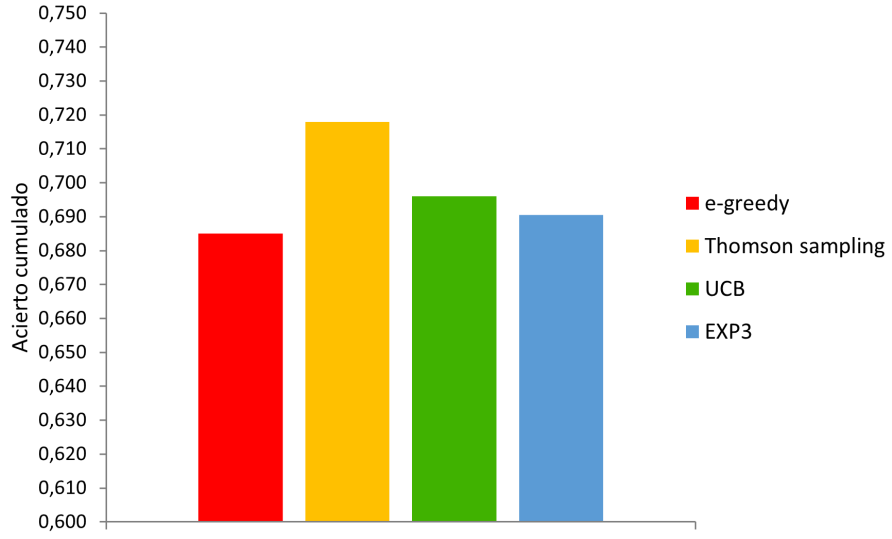

Figura 5.19: Comparativa soluciones multi-brazo con un 90 % de información para el caso 4

figura 5.18 parece claro que ϵ-greedy es el mejor, algo que además parece que es recurrente en los demás puntos del ciclo de vida para este caso y que podría ser interesante estudiar de cara a un trabajo futuro.

Pero si nos fijamos en la figura 5.19, parece que, en cambio, es el que peor se comporta para el último caso. Analizando más detenidamente se puede observar que en el caso de UCB, de nuevo es el que ofrece mejores resultados en conjunto siendo el segundo en ambos casos, por lo que es elegido. Si lo comparamos con el resto de las variantes se obtiene los resultados observados en la figura 5.20.

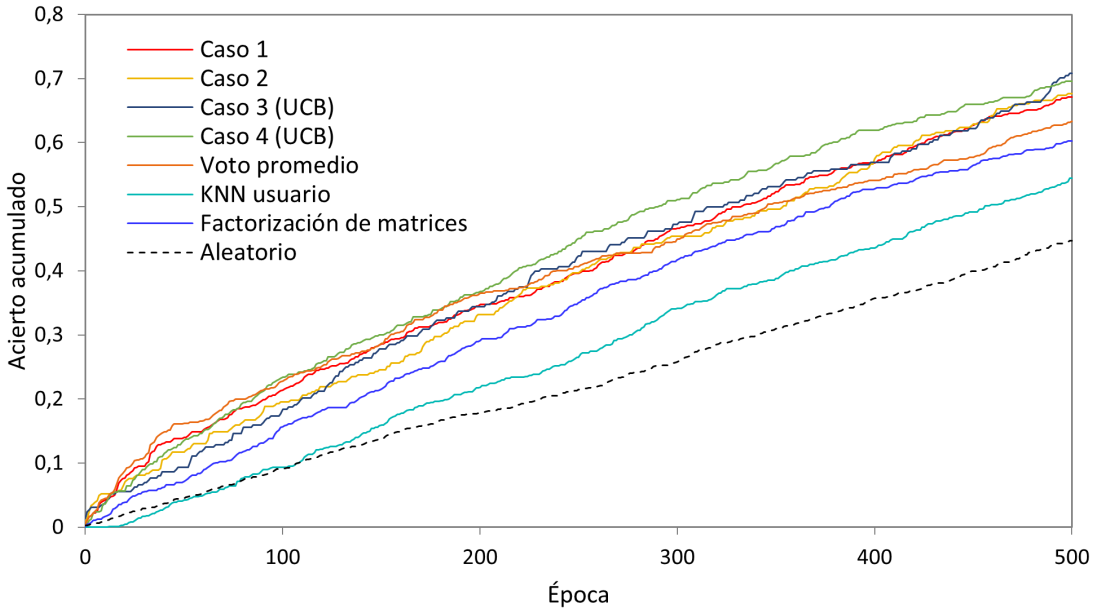

Figura 5.20: Evaluación de los sistemas con un 90 % de la información del conjunto de datos

En este punto del ciclo de vida, se observa quizás la mayor distancia entre los recomendadores de referencia y los casos con los distintos ciclos de vida planteados, lo cual por otra parte es esperable, ya que con la gran cantidad de información que posee el conjunto de entrenamiento, para sistemas con varias alternativas se espera que sea capaz de encontrar mejores caminos que los que puede tomar un sistema con un único algoritmo individual.

Estas distancias, junto a las de los propios casos planteados, se pueden ver más claramente si se representa la situación en la que se encuentran cada uno de los sistemas en el final del experimento, reflejado en la figura 5.21.

En esta gráfica se puede ver más claramente lo que se comentaba anteriormente sobre la distancia entre los casos de ciclo de vida planteados y los sistemas tradicionales donde vemos que es evidente la diferencia comentada, lo que por otro lado confirma el mejor rendimiento en los sistemas de recomendación mediante la combinación de algoritmos.

Por otro lado, también se puede observar de nuevo como los casos con ciclos de vida que inclu-

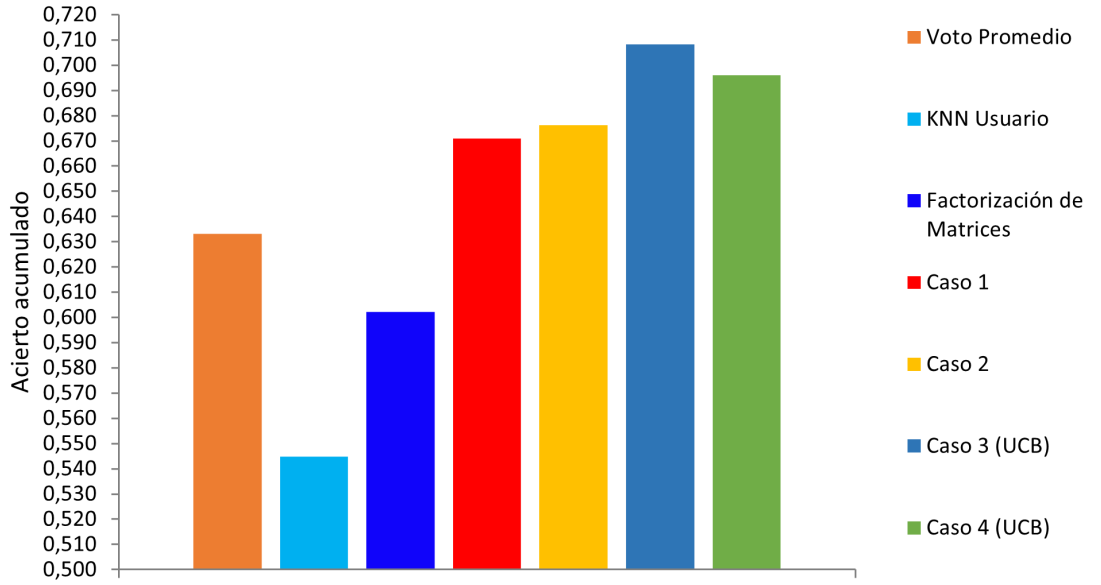

Figura 5.21: Acierto acumulado de los sistemas en la última época para el 90 % de información

yen soluciones multi-brazo son los que mejor rendimiento ofrecen lo cual nos confirma el objetivo de comprobar si efectivamente estos casos mejoraban lo que se viene haciendo tradicionalmente.

Por último, vemos como el caso 3 ha seguido su tendencia de mejora y parece asentarse como mejor alternativa para momentos avanzados del ciclo de vida donde el sistema conoce más información acerca de lo usuarios.

6

Conclusiones y trabajo futuro

6.1. Conclusiones

Observando los resultados obtenidos tras los diferentes experimentos realizados en este trabajo sobre diferentes ciclos de vida en un sistema de recomendación y cómo pueden influir las soluciones multi-brazo en dichos ciclos, se han podido obtener para el conjunto de datos utilizado las siguientes conclusiones:

- A medida que el sistema avanza en su ciclo de vida y por lo tanto el sistema posee más información, este es capaz de detectar más elementos relevantes en el mismo periodo de tiempo, observándose así su capacidad de aprendizaje.

- La combinación de algoritmos individuales ofrece mejor rendimiento que cualquiera de ellos por separado.

- El uso de soluciones multi-brazo, ofrece mejores resultados que mediante una evaluación offline o mediante un test A/B tradicional, utilizando ciclos de vida similares a estos casos.

- En el caso de que necesitemos buenos resultados al comienzo del ciclo de vida para que el usuario no abandone el sistema, parece que el caso 4 es mejor opción, mientras que si lo queremos es un resultado óptimo de cara al ciclo de vida completo el caso 3 parece más idóneo.

- En el conjunto de observaciones de cada punto en el ciclo de vida, parece que el algoritmo aplicado en los bandidos multi-brazo óptimo es el de UCB, ya que ofrece buenos resultados para ambos casos donde se aplican estas soluciones.

6.2. Trabajo futuro

De cara al futuro, existen varias opciones sobre las que se podría realizar un análisis más exhaustivo. Entre esas posibles líneas a desarrollar en un futuro destacan:

- **Ampliar el número de brazos o cambiar los existentes**, ya que como hemos visto el algoritmo kNN basado en usuario, ofrecía malos resultados lo que ha podido empeorar algo el rendimiento de los multi-brazo. Podría realizarse la prueba con este mismo basado en ítems, o distintas variantes de voto promedio y factorización de matrices que han ofrecido mejores resultados.

- **Analizar otros conjuntos de datos** para verificar que se sigue cumpliendo el mejor rendimiento de los recomendadores multi-brazo en los ciclos de vida planteados. Para ello se podría probar con conjuntos de datos de otro ámbito, como películas, anuncios o compras y también con otros conjuntos con una cantidad más elevada de datos o incluso más pequeños para ver cómo se comportan con distintas magnitudes.

- **Estudiar el algoritmo de ϵ-greedy para el caso 3**, ya que hemos visto que, a lo largo del ciclo de vida, sobre todo para un estado avanzado del sistema ofrecía resultados muy por encima del resto, por lo que la combinación de ambas técnicas puede ser interesante de estudiar para ver si en otros casos ofrece tan buen resultado. También probar con otro tamaño para la fracción fija de tráfico podría arrojar nuevos resultados que sean interesantes, para saber cómo afecta dicho tamaño al bandido.

- **Probar las soluciones multi-brazo en un sistema real**. Sería interesante probar los sistemas implementados basados en estas técnicas en un sistema en producción con usuarios reales, ya que, aunque se espera que tenga unos resultados similares a la expuesto en el trabajo, muchas veces el factor humano es impredecible.

7

Referencias

G. Adomavicius and A. Tuzhilin. *Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions*, in IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, 734-749 (2005)

C.C. Aggarwal, C.C. *Recommender Systems: The Textbook*. 8-15. (2016)

B. Brodén, M. Hammar, B. Nilsson and D. Paraschakis. *Ensemble Recommendations via Thompson Sampling: An Experimental Study within e-Commerce*. 19-29 (2018)

R. Cañamares and P. Castells. *Characterization of Fair Experiments for Recommender System Evaluation – A Formal Analysis*. In Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018) (2018)

R. Cañamares, M. Redondo, and P. Castells. 2019. *Multi-armed recommender system bandit ensembles*. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys '19). Association for Computing Machinery, New York, NY, USA, 432–436 (2019)

M. Elahi, F. Ricci and N. Rubens. *A survey of active learning in collaborative filtering recommender systems*. Computer Science Review (2016)

T. Gábor, P. István, N. Bottyán and T. Domonkos. *Matrix factorization and neighbor-based algorithms for the netflix prize problem*. In Proceedings of the 2008 ACM Conference on Recommender Systems. 267-274 (2008)

A. Gilotte, C. Calauzenes, T. Nedelec, A. Abraham and S. Dollé. *Offline A/B Testing for Recommender Systems* (2018)

G. Ivchenko and S. Honov. *On the Jaccard similarity test*. Journal of Mathematical Sciences. 88. 789-794 (1998)

O. Jeunen. *Offline Evaluation for Implicit-Feedback Recommender Systems*. (2019)

M. Katehakis and J. Veinott. *The Multi- Armed Bandit Problem: Decomposition and Computation*. Math. Oper. Res. 12. 262-268 (1987)

R. Kohavi, and R. Longbotham. *Online Controlled Experiments and A/B Testing* (2017)

D. Koulouriotis and A. Xanthopoulos. *Reinforcement learning and evolutionary algorithms for non-stationary multi-armed bandit problems*. Applied Mathematics and Computation. 913-922 (2008)

V. Kuleshov and D. Precup. *Algorithms for multi-armed bandit problems*. Journal of Machine Learning Research. 1 (2014)

J. Langford and T. Zhang. *The Epoch-Greedy Algorithm for Contextual Multi-armed Bandits*. Advances in Neural Information Processing Systems 20, Curran Associates, Inc. 817–824 (2008)

Y. Liang. *An Ensemble Approach for News Recommendation Based on Contextual Bandit Algorithms* (2017)

T. Liu (2009), *Learning to Rank for Information Retrieval*. Foundations and Trends in Information Retrieval: Vol. 3: No. 3, pp 225-331 (2009)

T. Luo, S. Chen, G. Xu, and J. Zhou. *Trust-based Collective View Prediction*. 25-51 (2013)

N. Nguyen-Thanh, D. Marinca, K. Khawam, D. Rohde, F. Vasile, E. Lohan, S. Martin, and D. Quadri. *Recommendation System-based Upper Confidence Bound for Online Advertising* (2019)

D.M. Nichols. *Implicit Rating and Filtering* (1998)

E. Orimo, H. Koike, T. Masui, and A. Takeuchi. *Analysis and Evaluation of Recommendation Systems*. 4557. 144-152 (2007)

D. Parra and S. Sahebi. *Recommender Systems: Sources of Knowledge and Evaluation Metrics.* (2013)

F. Ricci, L. Rokach, and B. Shapira. *Recommender Systems Handbook*. Springer Publishing Company, Incorporated (2015)

C. Sammut and G. Webb. *Encyclopedia of Machine Learning* (2010)

A.I. Schein, A. Popescul, L.H. Ungar and D.M. Pennock. *Methods and metrics for cold-start recommendations*. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '02, 253–260 (2002)

D. Siroker and P. Koomen. *A/B testing: the most powerful way to turn clicks into customers*. John Wiley and Sons Inc (2015)

R.S. Sutton, A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press (2018)

D. Valcarce, A. Bellogín, J. Parapar and P. Castells. *On the robustness and discriminative power of information retrieval metrics for top-N recommendation*. In Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018). 260-268 (2018)

E. Vozalis and K.G. Margaritis. *Analysis of Recommender Systems' Algorithms* (2003)

C. Zisopoulos, S. Karagiannidis, G. Demirtsoglou and S. Antaris. *Content-Based Recommendation Systems* (2008)

Anexos

A

Diagrama de clases

En este anexo se muestra el diagrama de clases del trabajo con las clases más relevante. Se omiten ciertas clases que complementan las importantes, ya que para mostrar todo sería demasiado complejo. Para que quede más claro, realizamos una división en subsistemas: *recommender*, *arm* y *bandit*. Estos a su vez tienen relaciones entre ellos las cuales son mostrados en la última figura en la que se ve todo el diagrama completo. Por otro lado, se incluye la clase TFM, la cual tiene todo el código para los casos planteados y el main de la ejecución. Podemos ver cada uno de ellos a continuación:

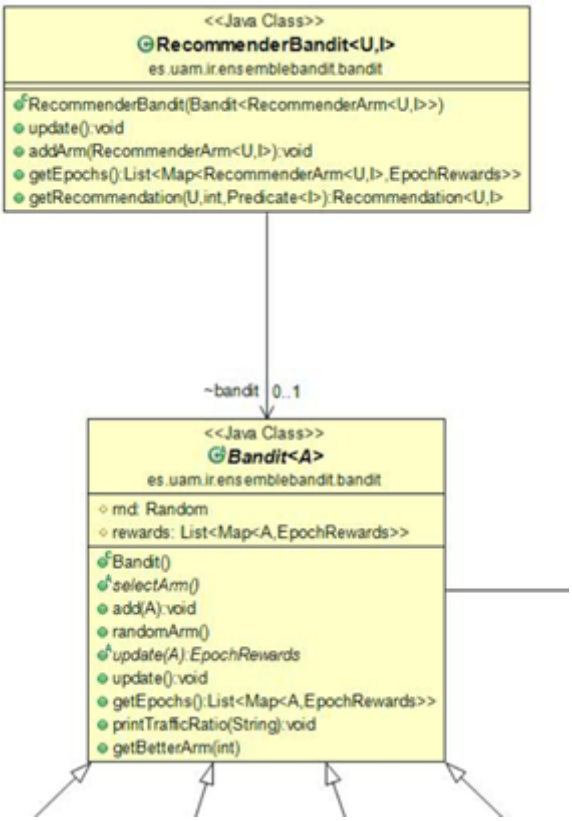

Figura A.1: Diagrama de clases del subsistema *recommender*

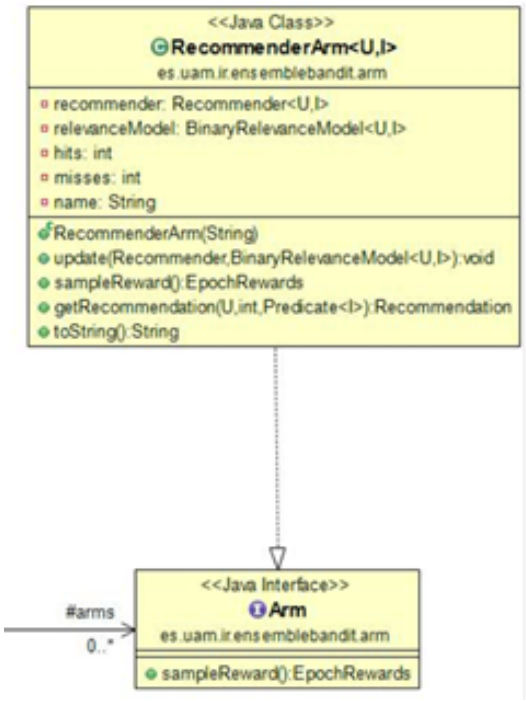

Figura A.2: Diagrama de clases del subsistema *arm*

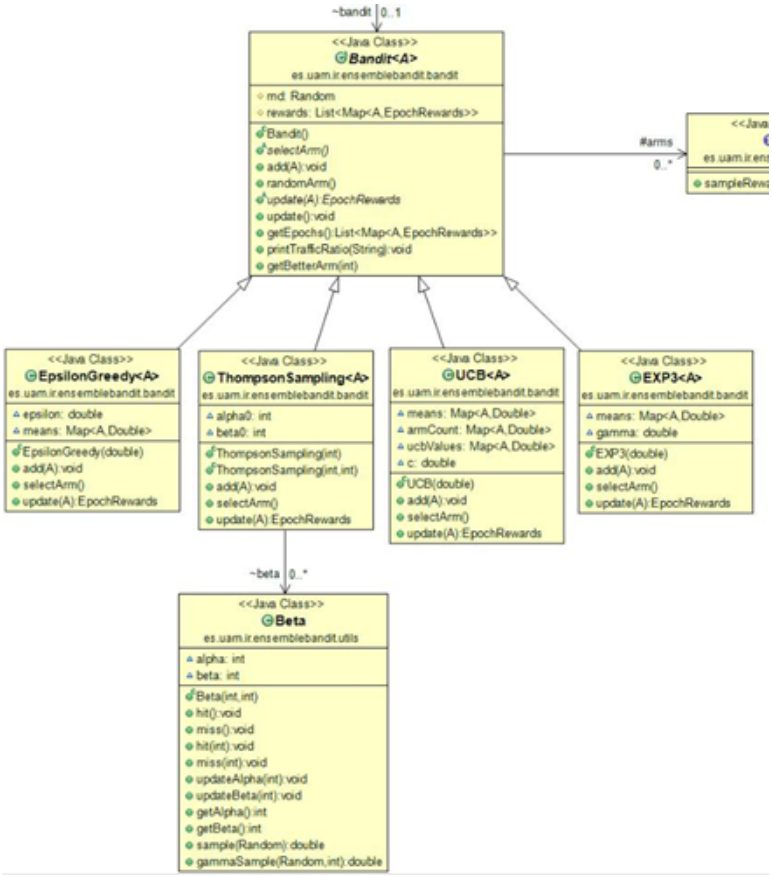

Figura A.3: Diagrama de clases del subsistema *bandit*

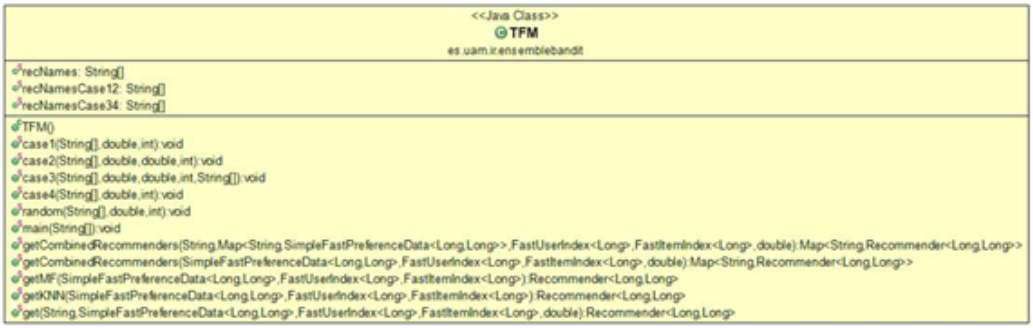

Figura A.4: Diagrama de clases del main de la ejecución

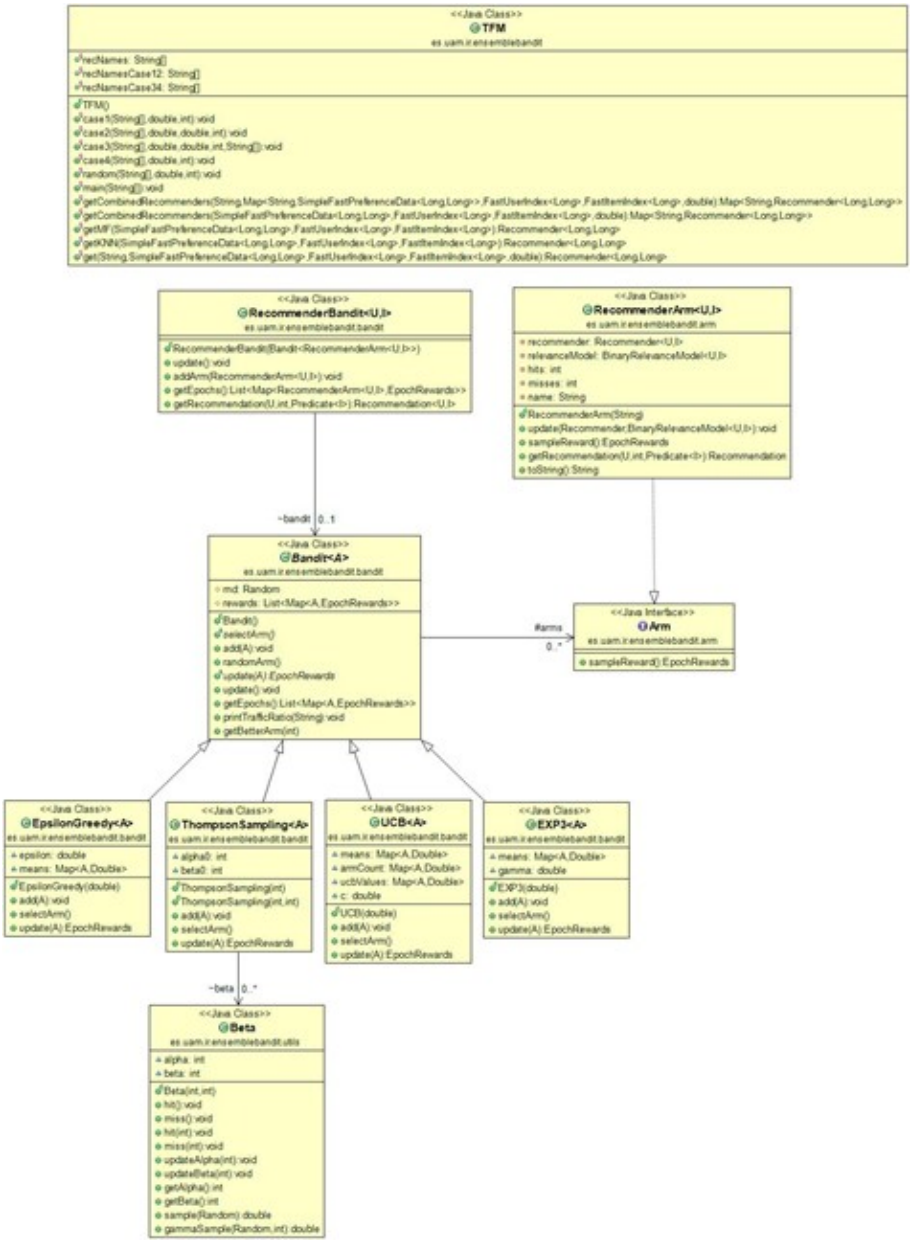

Figura A.5: Diagrama de clases del sistema completo

B

Comparativa multi-brazo caso 3

En este anexo, se van a mostrar las gráficas resultantes comparando los experimentos realizados con cada uno de los tipos de bandidos multi-brazo sobre el caso 3 de evaluación, para cada una de las situaciones de ciclo de vida analizadas.

En el trabajo están analizados cada uno de ellos sobre un punto determinado para observarlo más claramente, pero a continuación se puede observar la ejecución completa:

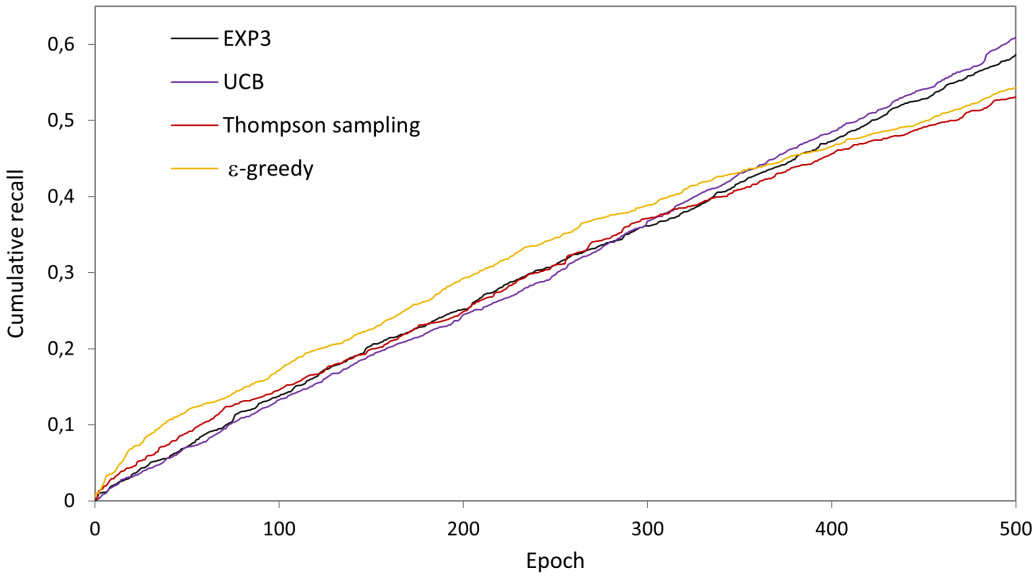

Figura B.1: Comparativa soluciones multi-brazo caso 3 para un 10 % de información

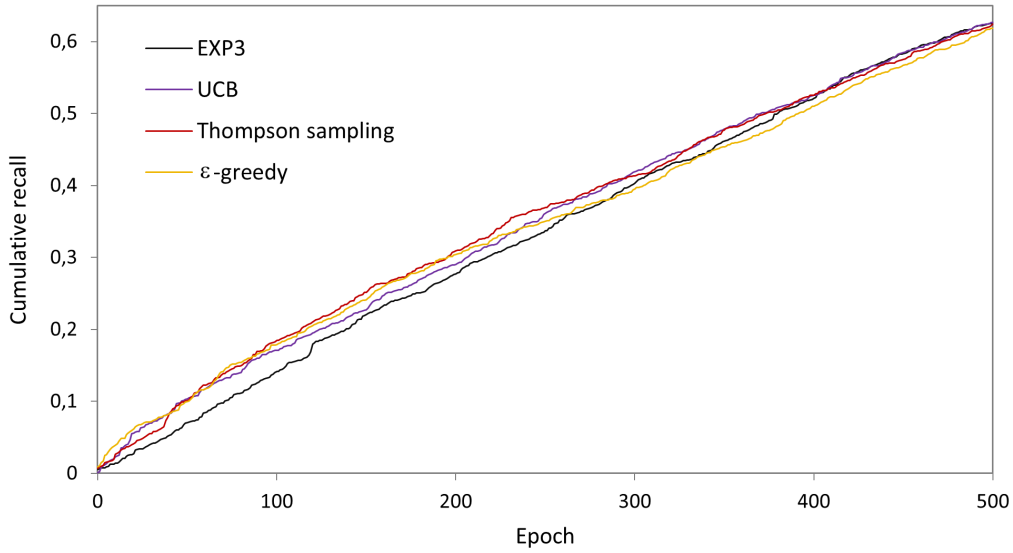

Figura B.2: Comparativa soluciones multi-brazo caso 3 para un 20 % de información

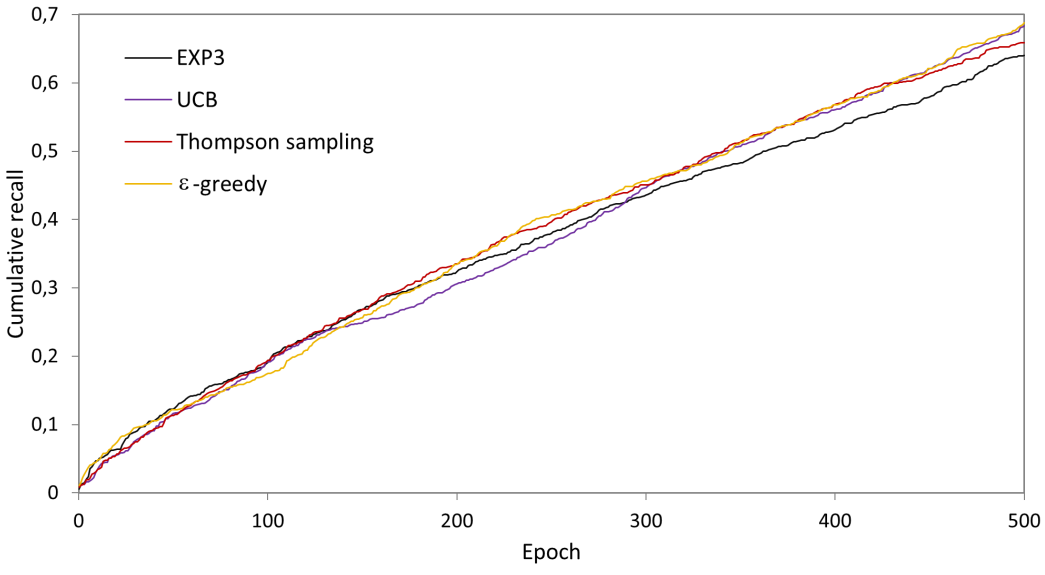

Figura B.3: Comparativa soluciones multi-brazo caso 3 para un 50 % de información

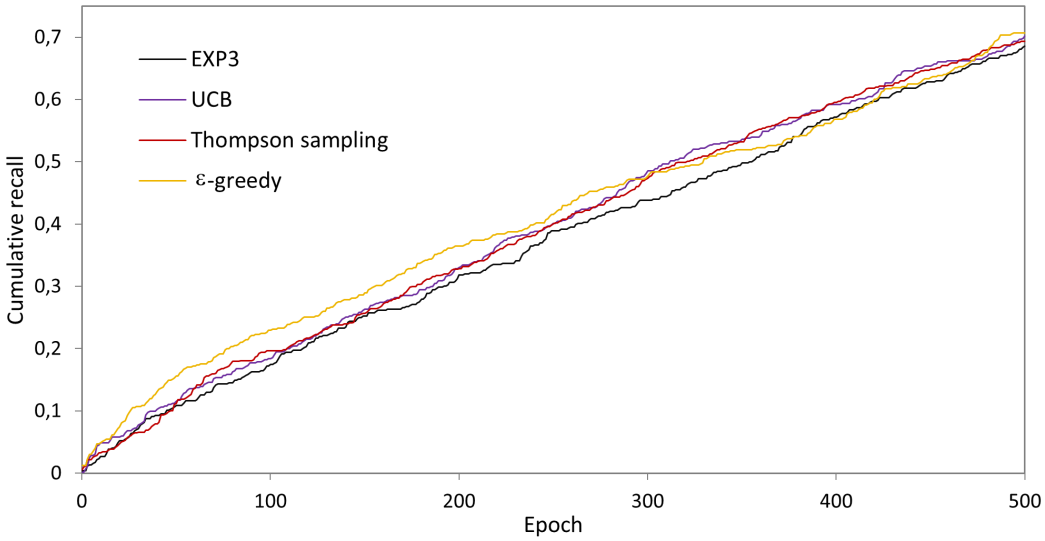

Figura B.4: Comparativa soluciones multi-brazo caso 3 para un 80 % de información

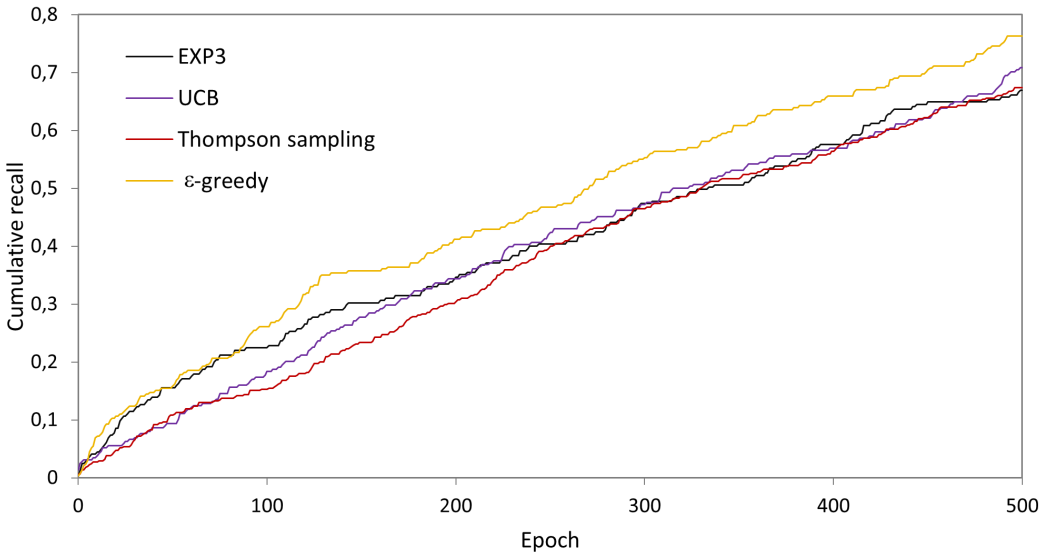

Figura B.5: Comparativa soluciones multi-brazo caso 3 para un 90 % de información

C

Comparativa multi-brazo caso 4

En este anexo, se van a mostrar las gráficas resultantes comparando los experimentos realizados con cada uno de los tipos de bandidos multi-brazo sobre el caso 4 de evaluación, para cada una de las situaciones de ciclo de vida analizadas.

En el trabajo están analizados cada uno de ellos sobre un punto determinado para observarlo más claramente, pero a continuación se puede observar la ejecución completa:

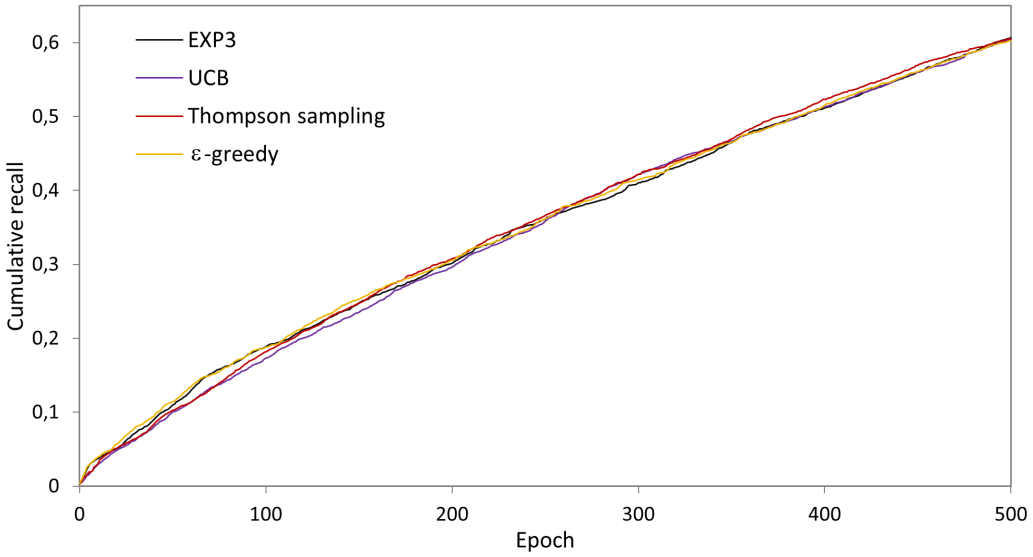

Figura C.1: Comparativa soluciones multi-brazo caso 4 para un 10 % de información

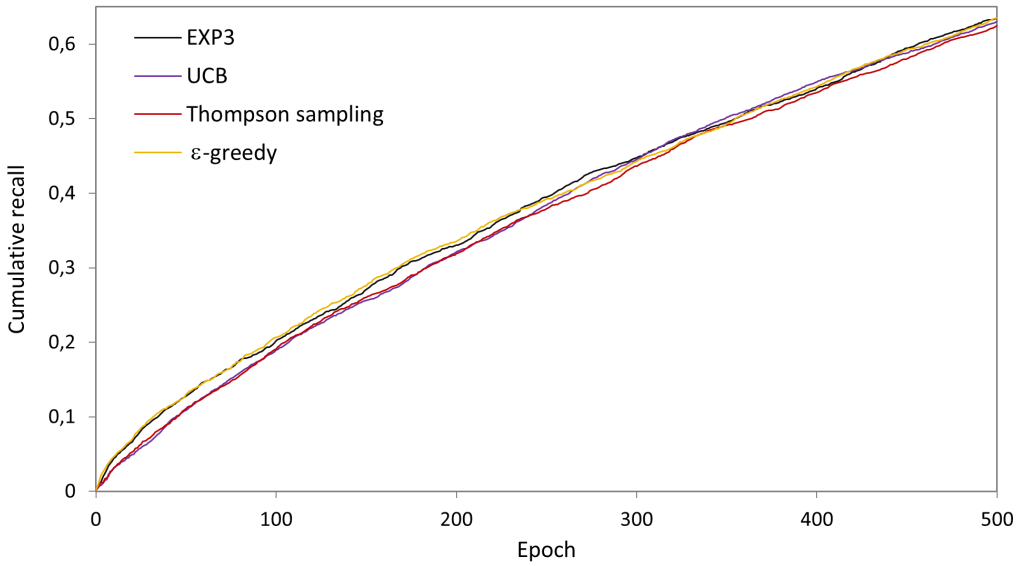

Figura C.2: Comparativa soluciones multi-brazo caso 4 para un 20 % de información

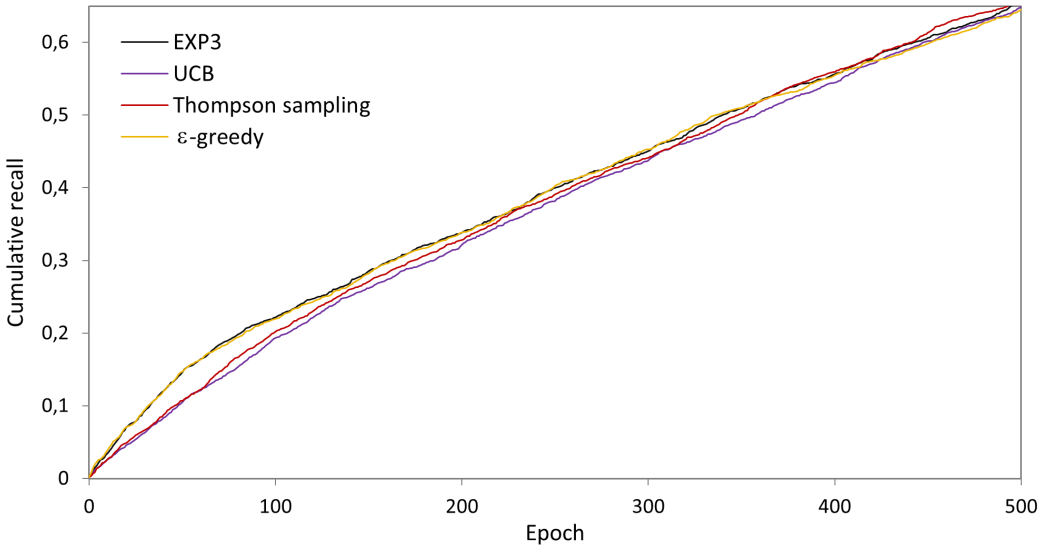

Figura C.3: Comparativa soluciones multi-brazo caso 4 para un 50 % de información

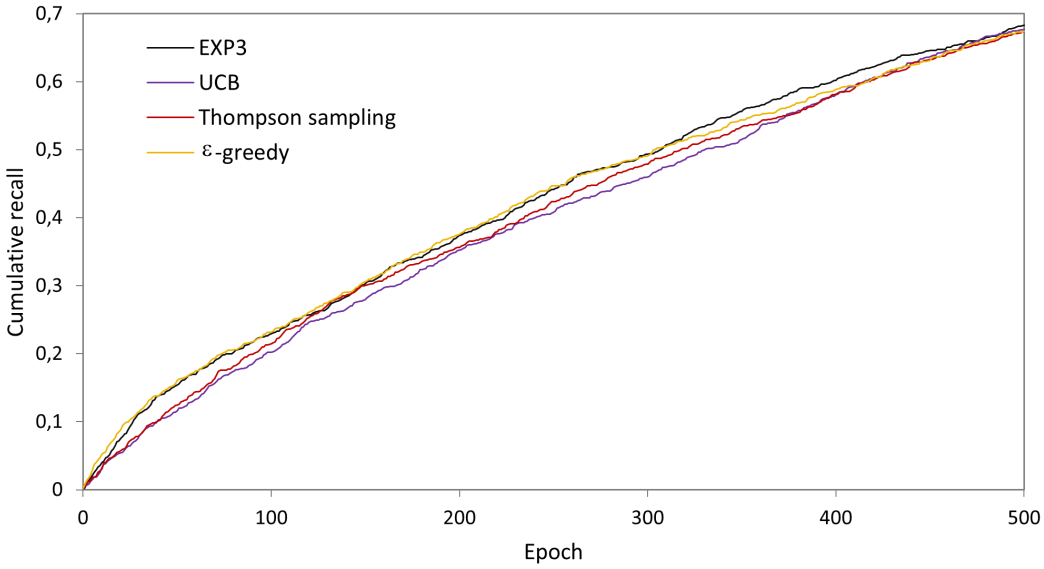

Figura C.4: Comparativa soluciones multi-brazo caso 4 para un 80 % de información

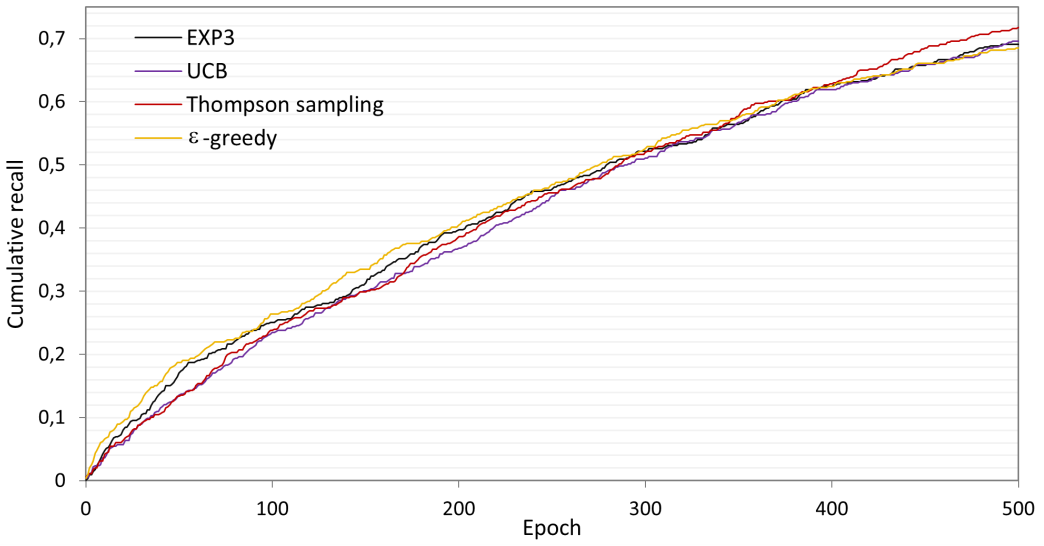

Figura C.5: Comparativa soluciones multi-brazo caso 4 para un 90 % de información

D

COMPARATIVA CASOS

En este anexo, se van a mostrar las gráficas resultantes comparando los experimentos realizados sobre cada uno de los casos de evaluación analizados, para cada una de las situaciones de ciclo de vida analizadas.

En el trabajo estas gráficas son reflejadas junto a los recomendadores de referencia para analizar su impacto frente a estos recomendadores, donde se espera un mejor rendimiento que ellos, pero a continuación se puede observar la comparativa únicamente entre los casos:

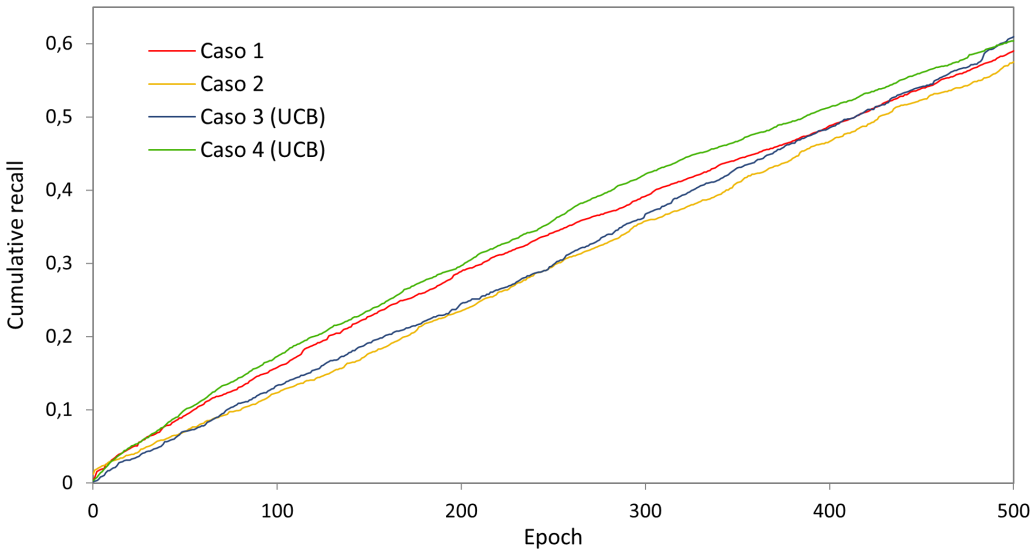

Figura D.1: Comparativa casos para un 10 % de información

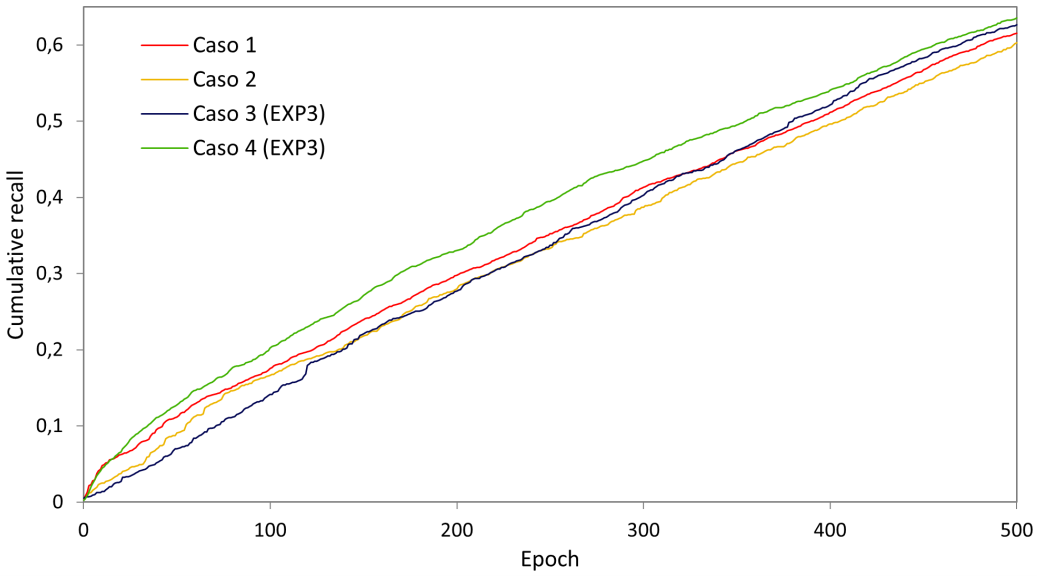

Figura D.2: Comparativa casos para un 20 % de información

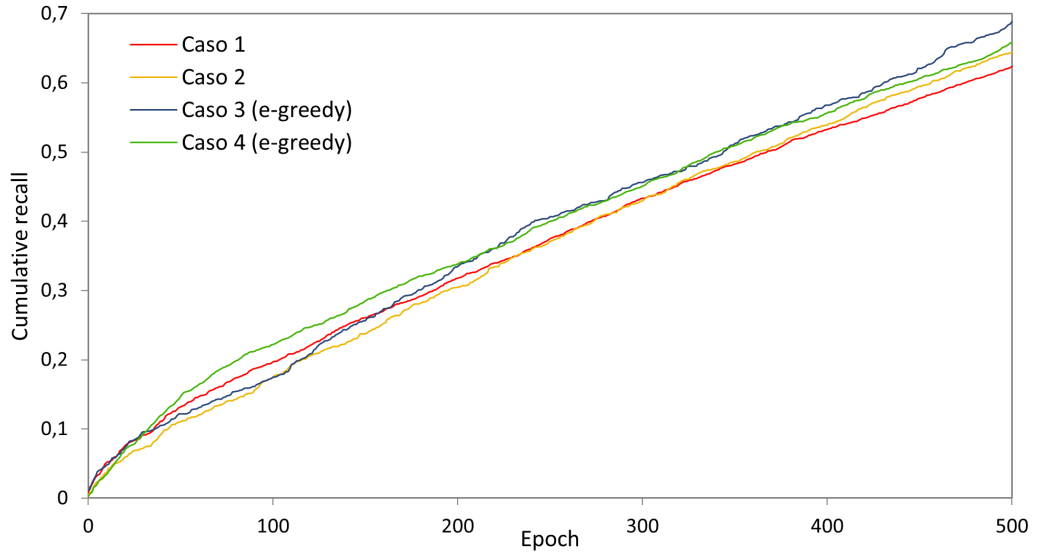

Figura D.3: Comparativa casos para un 50 % de información

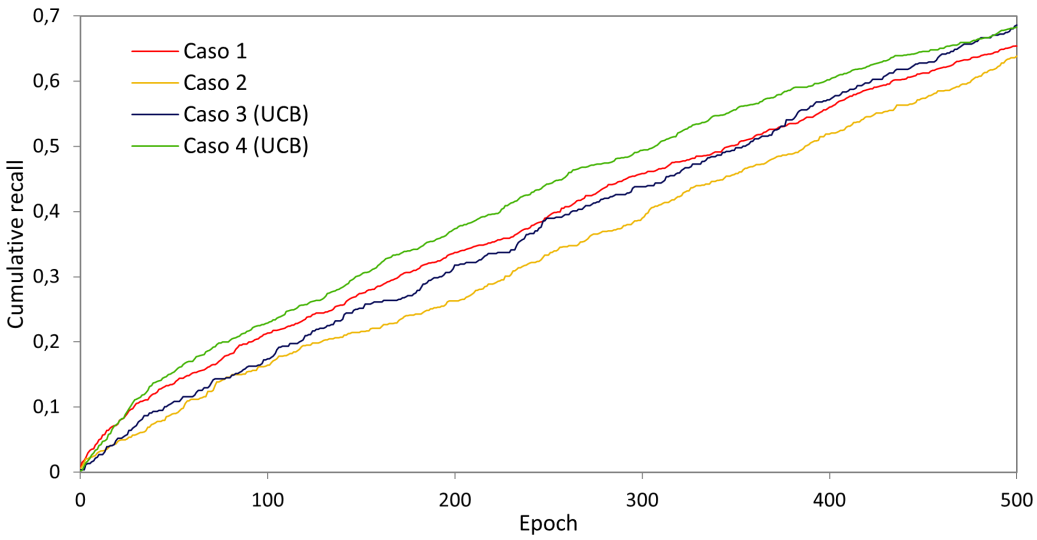

Figura D.4: Comparativa casos para un 80 % de información

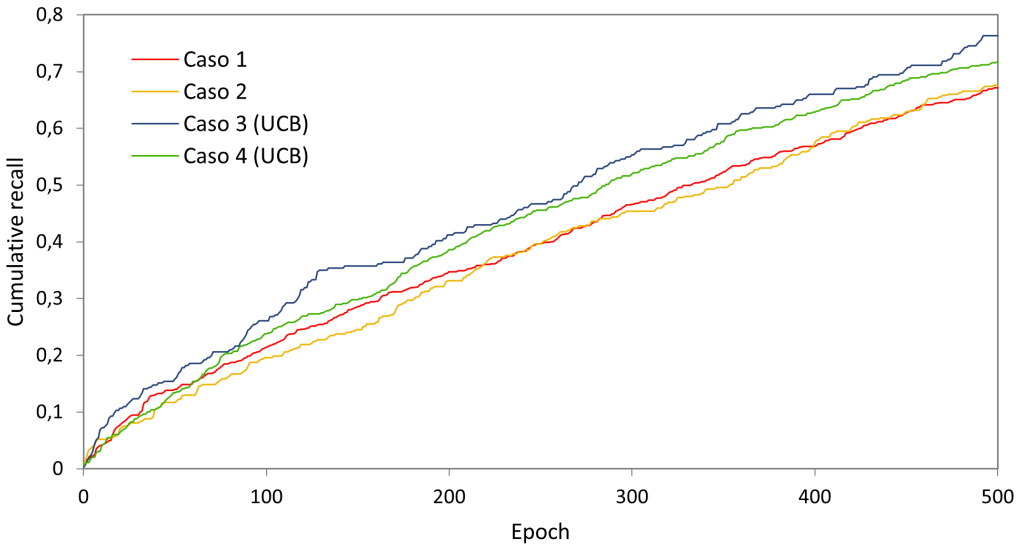

Figura D.5: Comparativa casos para un 90 % de información