

Multi-Armed Bandits Para Recomendación sobre Grupos de Usuarios Clusterizados

José Miguel Lira, Domingo Venegas, Felipe Jara

Proyecto IIC3433 - Octubre 2024

1 Abstract

Basado en las limitaciones del estudio de Claffey, A.^[1] experimentamos la implementación y evaluación off-policy (OPE) de distintas políticas de bandits, utilizando la librería de OBP (Open Bandit Pipeline)^[2], añadiendo al procedimiento la clusterización de usuarios. El dataset utilizado es ZoZo Town, que corresponde a una campaña de captura de datos de un sitio de e-commerce, basado en logs de bandits con dos políticas distintas: Random y TSampling. Este conjunto de datos consiste en una colección de eventos registrados por tiempo, donde se observa el comportamiento de los usuarios frente a distintos ítems durante 7 días (del 24 de noviembre de 2019 hasta el 30 de noviembre del mismo año). Las columnas clave son: *timestamp*, hora de registro del evento, *item_id*, identificador del ítem observado, *position*, posición del ítem en la lista vista por el usuario, *click*, booleano que indica si el usuario interactuó con el ítem, y *user_features* (4 columnas distintas) que identifican al usuario. Estas 4 columnas están hasheadas para anonimizar a los usuarios del dataset. Además de estas columnas, para cada usuario tenemos un *user_item_affinity_x*, donde *x* es el *item_id*. Esto permite analizar el interés de cada usuario por cada uno de los ítems del dataset (79 en total, 34 artículos en la sección de hombres y 45 en la de mujeres). Se puede observar el detalle en la figura 1. Durante este estudio se trabajó únicamente con los datos correspondientes al policy Random.

Campaigns	Data Collection Policies	#Data	#Items	#Dim	CTR ($V(\pi)$) $\pm 95\% \text{ CI}$	Relative-CTR
ALL	Random	1,374,327	80	84	0.35% ± 0.010	1.00
	Bernoulli TS	12,168,084			0.50% ± 0.004	1.43
Men's	Random	452,949	34	38	0.51% ± 0.021	1.48
	Bernoulli TS	4,077,727			0.67% ± 0.008	1.94
Women's	Random	864,585	46	50	0.48% ± 0.014	1.39
	Bernoulli TS	7,765,497			0.64% ± 0.056	1.84

Figure 1: Zozo Town Dataset

2 Trabajos relacionados

En [1] se evaluaron las políticas de bandit en datasets reales. Para esto utilizó dos dataset, los cuales son Deezer Dataset^[5] y ZOZOTOWN dataset^[6]. El primero contiene la probabilidad de interacción entre usuario y playlist de la aplicación de streaming francesa Deezer. Su contribución fue adaptar OBP para trabajar con dicho dataset, ya que no estaba diseñado para esto y obteniendo buenos resultados. Para el segundo dataset, implementó el estimador Self-Normalized Inverse Probability Weighting (SNIPW), estimador que normaliza la suma de los IPW. [2] crearon el framework Open Bandit Pipeline para poder aplicar bandit al dataset de Zozotown.

3 Procedimiento

El procedimiento llevado a cabo consiste en explorar la clusterización de los usuarios mediante k-means, entrenando un bandit (Thompson Sampling) On-Policy, por cada partición. Variamos la cantidad de *means*, para encontrar la cantidad óptima de particiones.

El concepto que se pretende explorar es el balance entre la cantidad y uniformidad de la información entregada al bandit. Intuitivamente, un grupo grande y diverso de usuarios puede entregar una gran cantidad de feedback, estas interacciones son suficientes para adaptar los parámetros del bandit y conseguir identificar con más certeza el brazo con mejores retornos, pero al mismo tiempo, ese feedback tiene en muchos casos señales mezcladas en cuanto a las preferencias de usuario, haciendo que el brazo óptimo identificado no necesariamente sea representativo de todo el grupo. El caso contrario es tener muchas particiones donde cada grupo de usuario tiene un bandit altamente personalizado, pero si la cantidad de interacciones no es suficiente, el bandit tendrá dificultades para identificar el brazo óptimo dada la incertidumbre que se tiene de las preferencias por los items.

Para evaluar el rendimiento de estas policies en el dataset utilizaremos los estimadores OPE. El objetivo de OPE es estimar el *policy value* de un policy en particular, sin tener que implementarlo en producción, pero a diferencia de simulaciones convencionales, los estimadores OPE están basados en un *ground-truth*. Esto se hace en base a los datos registrados (logs) de una implementación de un policy conocido, (llamado *behavioral policy*). Así se pueden realizar estudios para evaluar las mejores policies dentro de varios candidatos, y se evita poner en vivo aquellas con mal desempeño. El desafío principal que debe enfrentar la estimación de la nueva política es el sesgo o *bias* que tienen los logs originales, puesto que los items recomendados por el *behavioral policy* tienen mayores probabilidades de obtener retroalimentación (sea positiva o negativa). Luego, existen distintas formas de compensar el sesgo y obtener un valor lo más cercano posible a "lo que hubiera sido el rendimiento de nuestro nuevo policy":

1. Direct Method (DM): Consiste en primero estimar la recompensa de cada acción según el contexto con una regresión lineal (u otro modelo de aprendizaje supervisado), y usa eso para calcular el *policy value* esperado. La desventaja de esto es que, cuando la estimación de recompensa de la regresión no es buena, entonces el policy value estimado tampoco. \hat{q} es el estimador de recompensa, π_e es el policy que se está evaluando y π_b es el policy utilizado para la recolección de datos. $\mathcal{D} = (x_i, a_i, r_i)_{i=1}^n$ corresponde al "logged bandit data", donde x_i es el usuario, a_i es la acción tomada por el policy π_b para ese usuario, y r_i es el reward observado

[2].

$$\hat{V}_{DM}(\pi_e; \mathcal{D}, \hat{q}) := \mathbb{E}_{\mathcal{D}}[\hat{q}(x_t, \pi_e)]$$

2. Inverse Probability Weighting (IPW): Se aplica para contrarestar el problema de precision de *DM*. Este estimador pondera las recompensas según los *importance weights*, que se calculan en base al policy estimado y el behavior policy (que se usó para la recolección de datos). Cuando se conoce la behavior policy, las estimaciones son muy consistentes, pero tiene una alta varianza cuando el policy estimado y el behavior policy son muy distintos.

$$\hat{V}_{IPW}(\pi_e; \mathcal{D}) := \mathbb{E}_{\mathcal{D}}[w(x_t, a_t)r_t]$$

$$w(x, a) := \pi_e(a|x)/\pi_b(a|x)$$

3. Doubly Robust (DR): Combina *DM* y *IPW*, usando una versión ponderada de la recompensa, pero también usa una recompensa estimada para controlar la varianza. Mantiene la consistencia de *IPW* cuando el behavior policy es conocido, y la precision de *DM* cuando la recompensa estimada es buena.

$$\hat{V}_{DR}(\pi_e; \mathcal{D}) := \mathbb{E}_{\mathcal{D}}[\hat{q}(x_t, \pi_e) + w(x_t, a_t)(r_t - \hat{q}(x_t, a_t))]$$

El cálculo de estos estimadores se hizo con el framework open source *open bandit pipeline* que tiene métodos, modelos y funciones preestablecidos para calcular el valor estimado de un policy en base a los logs del dataset de Zozo Town. Para hacer posible la evaluación de los clusters con este framework, el procedimiento fue clasificar de antemano los logs, según las características de usuario correspondientes (user-item feedback o user features), la simulación fue ejecutada con cada partición de datos, instanciando un bandit nuevo cada vez. Por último, consideramos como baseline el desempeño de Thompson Sampling sobre la totalidad de los datos (sin clusterización), y para comparar adecuadamente el desempeño del sistema completo, consideramos el rendimiento de los clusters como un promedio ponderado, es decir: de acuerdo al tamaño de la partición resultante es la significancia de su rendimiento (resultados de OPE).

Se incluye la evaluación de la calidad de clusters de user-item affinities y user features utilizando la métrica de *silhouette score*, que a grandes rasgos: es un promedio de las distancias inter-clusters dividido por las distancias itra-cluster de cada punto de la muestra; un mayor valor en silhouette score es mejor, quiere decir que los grupos están más separados unos de otros y sus elementos son más cercanos. Junto con estos datos se presenta una visualización de la distribución de las interacciones y la asignación de clusters mediante reducciones de dimensionalidad a 2D, incluyendo: PCA y TSNE.

4 Análisis de resultados

4.1 Calidad de clusters

Para esta sección utilizaremos los datos registrados para la partición 'All', dado que representa el comportamiento del dataset de forma general. El resto de los gráficos (para las

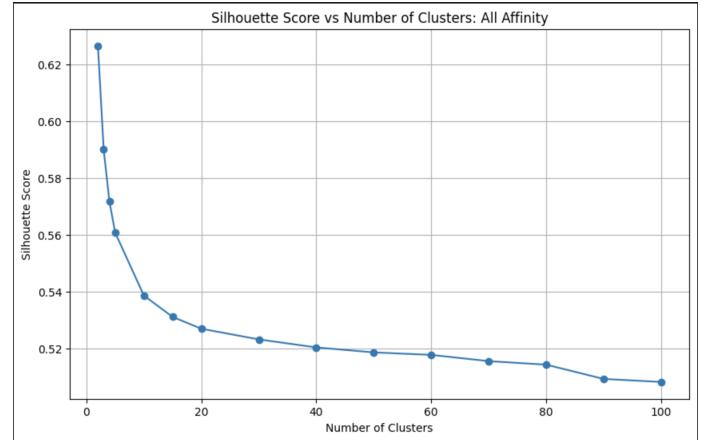


Figure 2: Silhouette scores, All Affinities

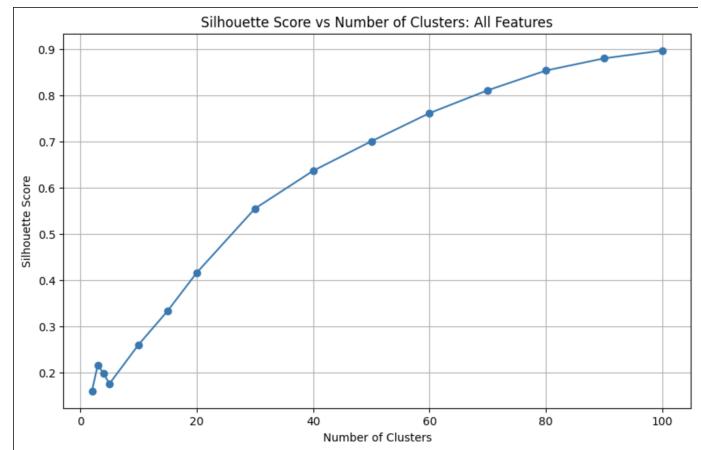


Figure 3: Silhouette scores, All Features

particiones 'Men' y 'Women') pueden ser encontrados en el apartado de anexo.

De la **figura 2** podemos observar que la calidad de la silueta disminuye asintóticamente a medida que la cantidad de particiones aumenta para los atributos de user-item affinities. Mientras tanto, lo contrario es cierto en la **figura 3** donde la calidad de la silueta aumenta para los clusters de features, a medida que la cantidad de particiones crece. Esto quiere decir de forma preliminar, que si la silueta es un buen indicador del rendimiento de los clusters, deberíamos ver mejores resultados para los cluster de affinities para cantidades bajas de particiones, y para los clusters de features, un mejor rendimiento a medida que el número de clusters aumenta.

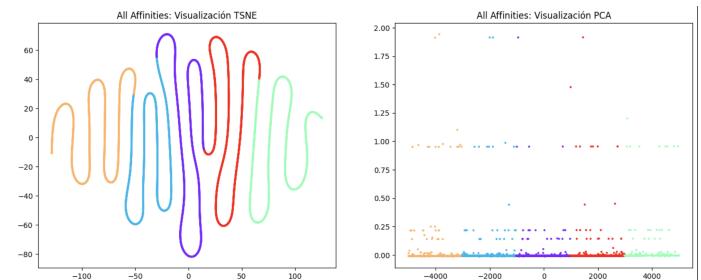


Figure 4: 2D clusters, All Affinities

Para las **figuras 4 y 5** es necesario tomar en consideración que la asignación de los clusters es previa a la visualización, por ende, el método de visualización aplicado

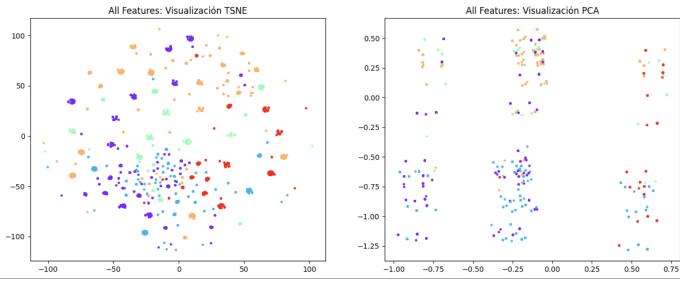


Figure 5: 2D clusters, All Features

no afecta la partición de los datos. El efecto es que algunos patrones que pueden ser evidentes en la gráfica, no necesariamente representan mejor la forma en que los vectores se discriminan en k-means, además, se utiliza un número de means igual a 5, dado que solo nos interesa visualizar un ejemplo. Con estas consideraciones, notamos que los clusters de affinities (**figura 4**) se muestran de forma muy condensada para PCA, y la partición de los datos se hace de forma horizontal. Una potencial explicación es la distribución *Long-Tail* de los datos y que se suele ver en el campo de sistemas recomendadores: pocos usuarios concentran la mayor parte de las interacciones, mientras la mayoría es aquella que interactúa poco. Se puede ver reflejado este comportamiento en la visualización de PCA, dado que hay una gran capa donde los datos se concentran y pocos datos que se encuentran muy arriba en el eje vertical, las particiones horizontales podrían representar las distintas afinidades.

El resultado de TSNE es producto de que los datos se encuentran muy condensados en una de las dimensiones (por la baja cantidad de interacciones de la mayoría de los usuarios), parte del objetivo de una reducción de dimensionalidad con TSNE es proporcionar una visualización que separa los datos unos de otros, entonces lo que vemos es la forma en que TSNE naturalmente transforma el espacio.

Los clusters mostrados en la **figura 5** son de una naturaleza completamente distinta, al no depender del número de interacciones, sino de las características de las personas, estos son naturalmente muy variados, y la distinción entre grupos es más evidente que el caso anterior.

Por último, es importante destacar cómo los valores de las siluetas se relacionan con la forma en que los clusters son visualizados. En el caso de las afinidades, los datos al estar tan condensados, aumentar la cantidad de particiones causa inevitablemente que los clusters se junten más unos con otros, y todo lo contrario es cierto para los clusters de características.

4.2 Resultados de OPE

En cuanto a la experimentación con *open bandit pipeline*, ejecutamos distintas simulaciones variando la cantidad de particiones de los datos, sobre los datasets de 'Men', 'Women' y 'All'; mostraremos los resultados de la partición 'All' dado que resumen el comportamiento del dataset, pero los demás experimentos se encuentran disponibles en el apartado de anexo. Los valores entregados corresponden al *estimated policy value* con los métodos IPW, DM y DR, el valor está representado en la probabilidad de obtener un feedback positivo (click) por cada una de las recomendaciones del bandit. Recordar que el rendimiento a través de los clusters se realiza a través de la ponderación por

los tamaños de las particiones. Por último, la cantidad de clusters de experimentación no pueden ser arbitrariamente grandes, esto porque *open bandit pipeline* requiere una cantidad mínima de experimentos para arrojar los estimadores OPE, por ende, se exploran cantidades hasta el límite permitido en cada caso.

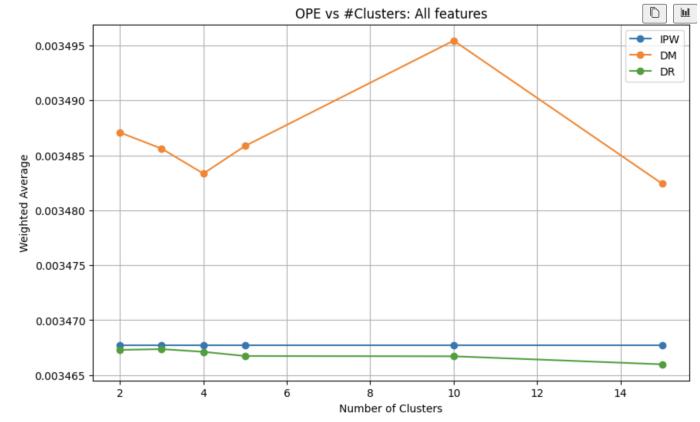


Figure 6: OPE All Features

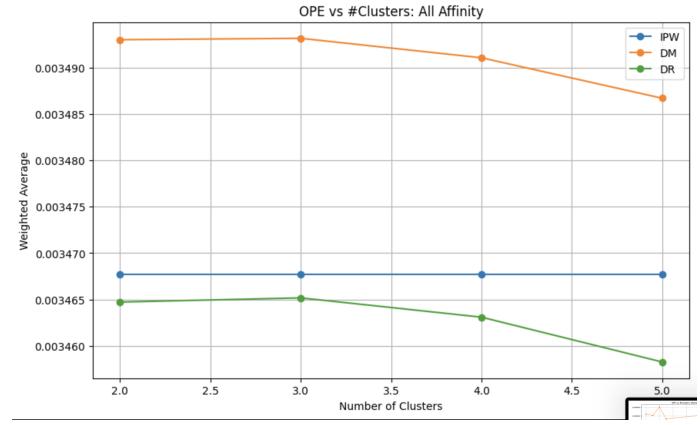


Figure 7: OPE All Affinity

Lo primero que se puede destacar es que la cantidad de particiones que soportan los *user features* es mucho mayor a las particiones que se pueden obtener con los *user-item affinities*, en concreto la **figura 6** representa la tendencia hasta 15 clusters, mientras que en la **figura 7** solo fue posible ejecutar hasta 5 clusters. Respecto a los resultados observables, notamos que la tendencia de los estimadores es que estos disminuyan a medida que los clusters aumentan. Comparando con los resultados de las siluetas en las **figuras 2 y 3**, se puede apreciar que las tendencias no coinciden para el caso de *user-features*; es decir, si bien la calidad de las siluetas de los clusters medidas con *silhouette score* tienden a aumentar para mayor cantidad de clusters, este efecto no se ve representado para las métricas de rendimiento de OPE, lo que nos lleva a concluir que no se puede confiar exclusivamente en la calidad de la clusterización para predecir el rendimiento de ejecutar múltiples bandits en cada sub-grupo.

Comparamos también el rendimiento promedio de los clusters con el baseline (Thompson Sampling sin clusterización) y marcamos en rojo en la gráfica el rendimiento de clusters que superan el baseline, junto a la cantidad de interacciones registradas para el grupo. Debido a la cercanía de los estimadores y considerando la proporción de las dife-

encias, el valor del baseline y de los puntos que escapan, se representan como el promedio de las estimaciones obtenidas por IPW, DM y DR.

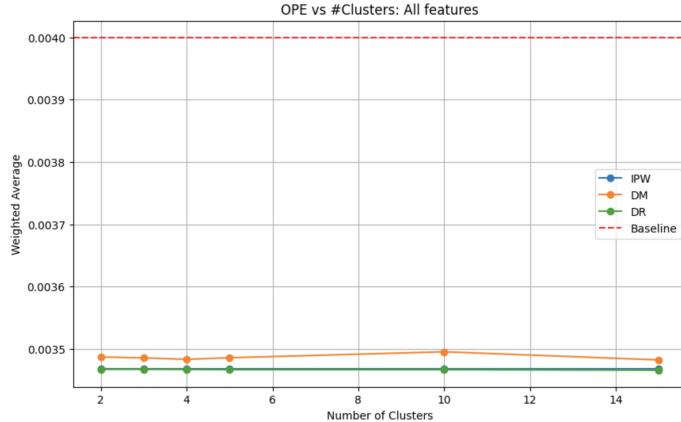


Figure 8: OPE All Features + Baseline

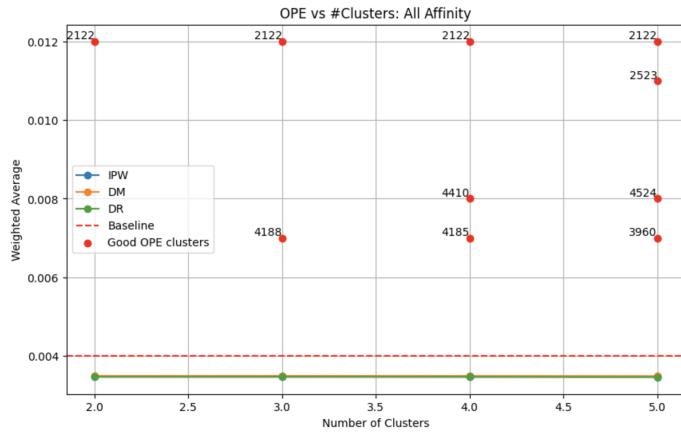


Figure 9: OPE All Affinities + Baseline

De las **figuras 8 y 9** podemos apreciar que tanto para características de usuario como afinidades, el rendimiento promedio de utilizar clusters es peor que el baseline. Además, las diferencias entre clusterizar con características o afinidades en promedio son casi nulas (fijarse en la escala del eje vertical). Lo que sí presenta una diferencia crucial entre los métodos es que para *features* no existen agrupaciones que mejoren respecto al baseline, mientras que con *affinities* notamos que cada aumento de cantidad de particiones genera un sub-grupo pequeño (comparado con los cientos de miles de interacciones del dataset) que supera ampliamente el baseline; un aspecto interesante sobre este punto es que estos grupos parecen no sub-dividirse, dado que se conserva en gran medida la cantidad de interacciones con el aumento de la cantidad de clusters.

Con el análisis anterior y tomando en cuenta los resultados de las **figuras 6 y 7** que es la representación de los OPE sin el baseline; interpretamos que en el contexto de los user-features, cada aumento en la cantidad de clusters produce una pérdida de información valiosa para el sistema en general, lo que lleva a una baja en el rendimiento promedio. No obstante, para el caso de las afinidades, se obtienen sub-grupos con excelentes resultados; la capacidad de separar los datos respecto a las afinidades depende directamente de la cantidad de interacciones del usuario, con pocas interacciones la posibilidad de distinguir los grupos es difícil y esto

se demuestra en la **figura 4** donde la distribución fue representada de forma gráfica. Dada la distribución *Long-Tail* de los datos, los grupos que son fácilmente distinguibles corresponden a aquellos que concentran gran parte de las interacciones de la plataforma; esto, sumado al hecho que las afinidades de los usuarios por los distintos ítems es mucho más informativo que sus características personales para la recomendación de los mismos ítems, nos lleva a teorizar que separar estos grupos de alta información del resto de la plataforma tiene un efecto adverso para la gran mayoría de los usuarios, aquellos que tienen menos interacciones.

En resumidas cuentas, en el contexto de *user-item affinities*, cada partición nueva separa a un grupo de alta información que se beneficia de tener recomendaciones más personalizadas, no obstante, se perjudica al bruto de los usuarios que interactúan poco y se benefician potencialmente de la exploración y feedback de los usuarios más activos. Por el lado de *user features* la calidad de la información por la que se separan los datos no es muy buena, puesto que no permite relacionar de forma directa las preferencias por ítems en particular o gustos más allá de generalizaciones que se pueden hacer sobre género, etnia, edad, etc. Entonces, cada bandit tiene menos datos a disposición, por lo que el rendimiento decrece en promedio pero sin un beneficio en las recomendaciones de ningún grupo en particular.

5 Conclusiones

5.1 Hallazgos principales

Exploramos la clusterización con k-means sobre las interacciones de usuarios basados en sus afinidades por ciertos ítems del catálogo, como por sus características personales. Medimos la calidad de los clusters utilizando la métrica de *Silhouette Score*, observando que la calidad de clusters aumenta con la cantidad de particiones para las características de usuario, y disminuye para las afinidades. Estas métricas no se condicen con el rendimiento de los bandits BTS; los cuales medimos con estimadores OPE promediando el performance de cada bandit individual y ponderando por su tamaño. En concreto, se observa que el rendimiento promedio disminuye con el aumento de particiones, pero en el caso de los clusters de afinidades, se obtienen pequeños grupos que mejoran ampliamente el baseline. Nuestra conclusión es que cada partición nueva separa a un grupo de usuarios con alta participación que se beneficia de tener recomendaciones más personalizadas, no obstante, se perjudica al bruto de los usuarios que interactúan poco y se benefician potencialmente de la exploración y feedback de los usuarios más activos. Hay ciertos aspectos que aún se deben clarificar para entender este fenómeno en su totalidad y se mencionan a continuación.

5.2 Limitaciones, Dificultades y Trabajo Futuro

Open Bandit Pipeline es el framework utilizado para la experimentación, a diferencia de una librería convencional, se encarga de todo el procedimiento necesario para obtener los estimadores OPE, desde el modelado de bandits hasta la simulación con logs del dataset *Zozo Town*. Esto trae ventajas y desventajas: por un lado, es muy sencillo y rápido comenzar a experimentar y extraer resultados de base; por

otro lado, la implementación de métodos complejos y capturar información adicional a los estimadores es muy complejo. En particular, estamos limitados al dataset de Zozo Town, y a las métricas de OPE utilizadas; para implementar la clusterización fue necesario instalar el framework localmente y sobre-escribir los archivos para obtener las particiones del dataset. Calcular métricas adicionales, pero super relevantes como diversidad, presenta su propio desafío. Vale recalcar que métricas como recall no fueron calculadas, ya que en el contexto de MAB y off policy evaluation, el estándar para evaluar nuevas técnicas de policy según la literatura son las métricas OPE.

Estas dificultades traen consecuencias y se expresan en las limitaciones de este trabajo. Primero, es de mucha importancia poder caracterizar la formación de los clusters en cuanto a los usuarios únicos que los conforman; dado que el dataset presenta interacciones de usuarios anónimos de características representadas por valores de hash, es imposible atribuir una acción a un usuario específico. Esto último es muy relevante, puesto que, se dificulta enormemente poder comprobar que los usuarios segregados reciben mejores recomendaciones por el bandit del cluster, que el baseline. En otras palabras, los resultados se expresan en términos de promedios, y no podemos resaltar interacciones particulares. Segundo, hubiera sido un buen aporte testear las recomendaciones producidas por el bandit, pero la instancia del modelo se puede ocupar únicamente para la simulación y el cálculo posterior de los OPE. Tercero, sería interesante explorar la posibilidad de aplicar *fine-tuning* con clusterización a un bandit pre-existente, observando si el decremento del rendimiento al particionar los datos se sigue expresando; esto ya que estamos limitados a inicializar un bandit desde cero con cada segmento de usuarios clusterizados.

Finalmente, se propone como trabajo futuro un modelo de recomendación que clusteriza selectivamente a los grupos con alta participación según sus afinidades por ítems, mientras que la mayoría de baja participación puede ser atendida por filtrado colaborativo, teniendo el beneficio de no perder la información valiosa de los grupos de alta participación en la plataforma. El principal desafío en este caso sería evidentemente adaptar la librería para permitir estimar el rendimiento del filtrado colaborativo con OPE, para obtener una comparación robusta frente al bandit de baseline, o bien, respecto al modelo de filtrado colaborativo.

El uso de otros datasets para estudiar el fenómeno descrito en este trabajo puede ser de mucho valor, especialmente si permiten singularizar a cada usuario, permitiendo explorar el comportamiento de los sistemas recomendadores y las interacciones del ecosistema en su generalidad.

El código del proyecto está en el siguiente repositorio de github: <https://github.com/FelipeUC2020/Proyecto-reccsys-2024-2/tree/master>

6 Bibliografía

1. Claffey, A., Ick, C., Ma, J., & Turkel, D. Evaluating Bandit Policies Across Datasets.
2. Saito, Y., Aihara, S., Matsutani, M., & Narita, Y. (2020). Open bandit dataset and pipeline: Towards realistic and reproducible off-policy evaluation. arXiv preprint arXiv:2008.07146.
3. Cavenaghi, E., Sottocornola, G., Stella, F., & Zanker, M. (2021). Non stationary multi-armed bandit: Empirical evaluation of a new concept drift-aware algorithm. Entropy, 23(3), 380.
4. Pedregosa, F., Varoquaux, Ga”el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct), 2825–2830.
5. Bendada, W., Salha, G., Bontempelli, T. (2020). Datasets from the RecSys 2020 article “Carousel Personalization in Music Streaming Apps with Contextual Bandits” [Data set]. Zenodo.
6. Data Release. (s/f). ZOZO RESEARCH. Recuperado el 9 de diciembre de 2024, de <https://research.zozo.com/data.html>
7. Samir, K., Johan, U. (2021). Adaptive normalization for IPW estimation. En arXiv [stat.ME]. <http://arxiv.org/abs/2106.07695>

7 Anexo

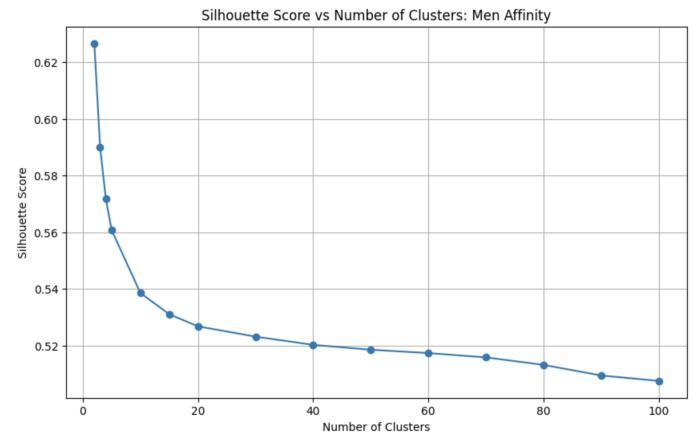


Figure 10: Silhouette scores, Men Affinities

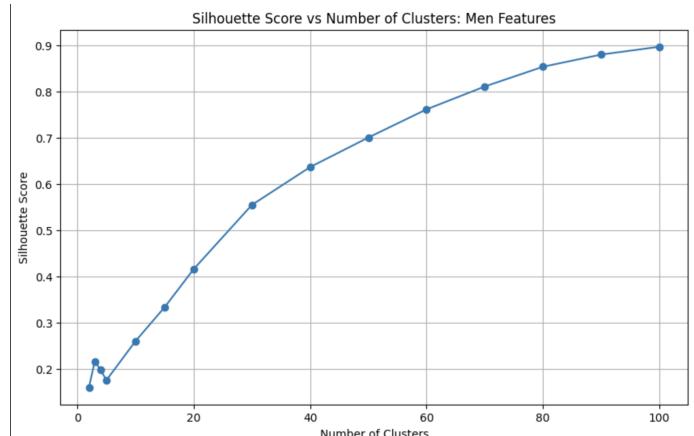


Figure 11: Silhouette scores, Men Features

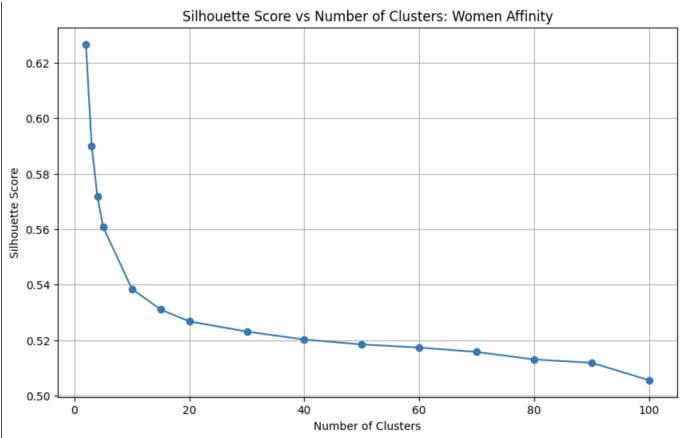


Figure 12: Silhouette scores, Women Affinities

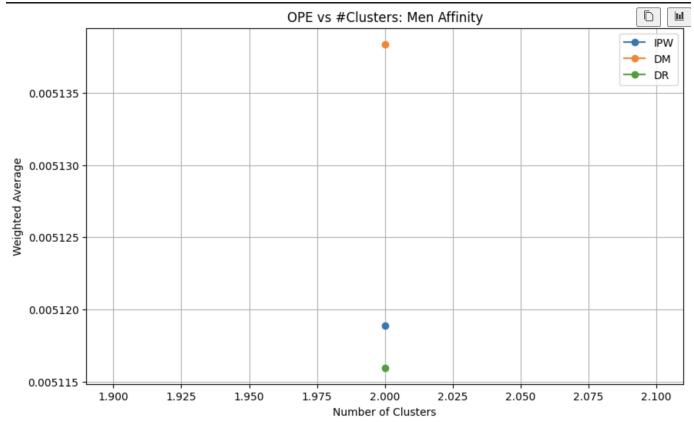


Figure 15: OPE Men Affinity

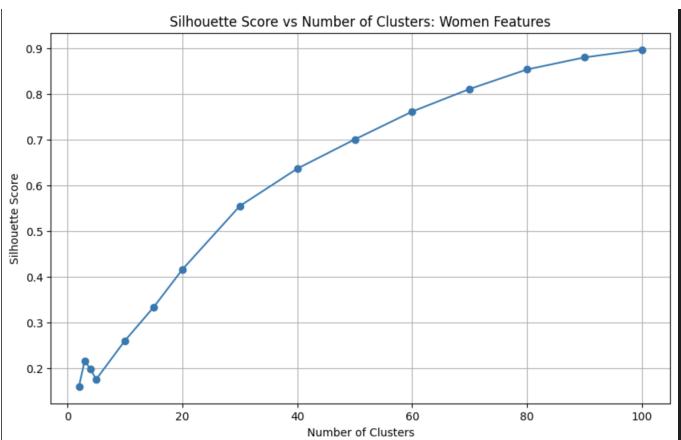


Figure 13: Silhouette scores, Women Features

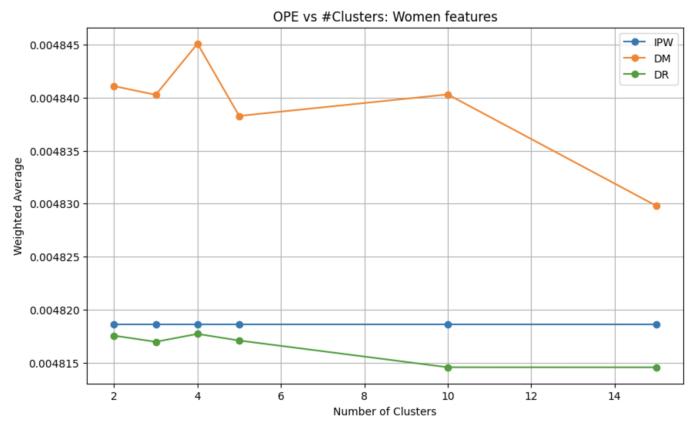


Figure 16: OPE Women Features

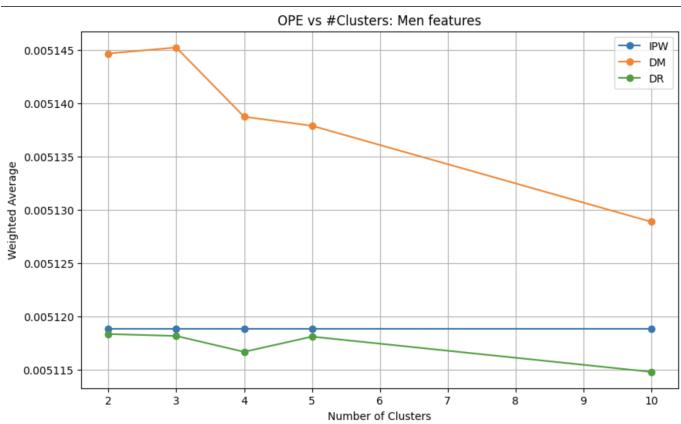


Figure 14: OPE Men Features

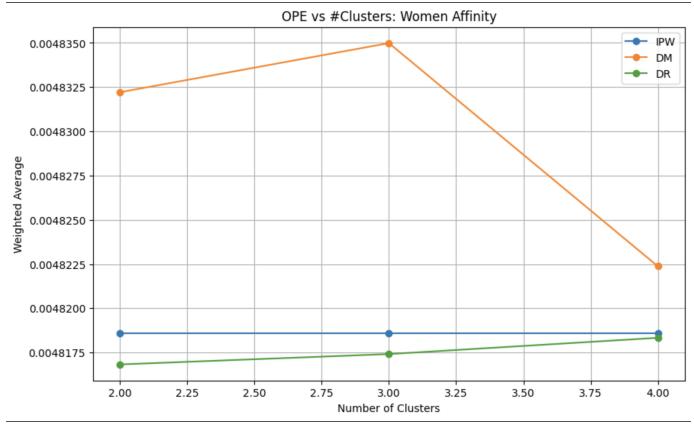


Figure 17: OPE Women Affinity

Table 1: Baseline Men

	Estimated Policy	Relative Estimated Policy
ipw	0.006010	1.172939
dm	0.005143	1.003632
dr	0.006108	1.191907

Table 3: Baseline All

	Estimated Policy	Relative Estimated Policy
ipw	0.004694	1.352924
dm	0.003702	1.067188
dr	0.004683	1.349757

Table 2: Baseline Women

	Estimated Policy	Relative Estimated Policy
ipw	0.006813	1.481190
dm	0.004555	0.990160
dr	0.006863	1.491972

Table 4: Promedio ponderado de clusters utilizando user affinity en men

Cluster	IPW	DM	DR
2	0.005119	0.005138	0.005116

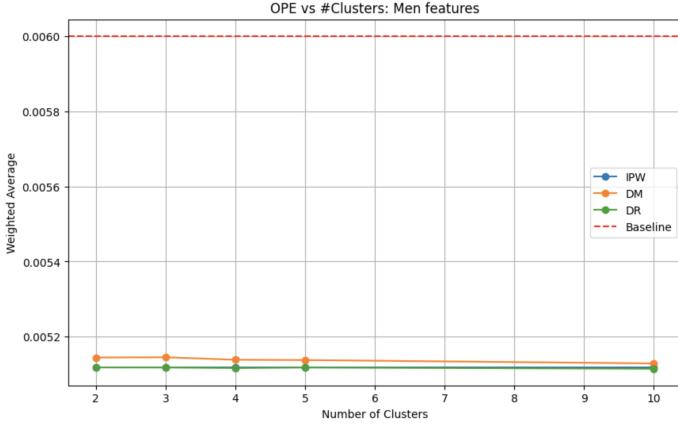


Figure 18: OPE Men Features + Baseline

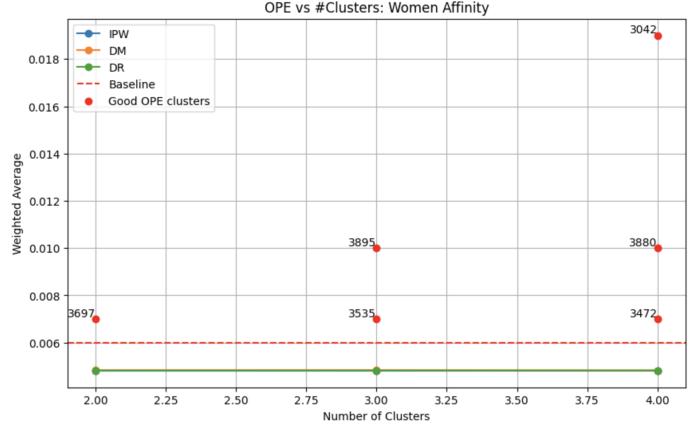


Figure 21: OPE Women Affinities + Baseline

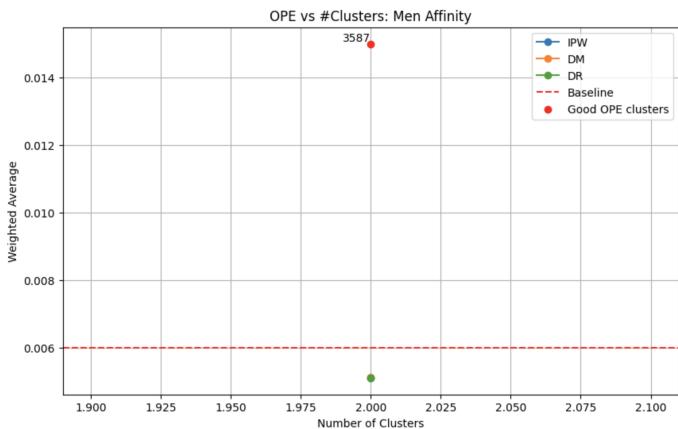


Figure 19: OPE Men Affinities + Baseline

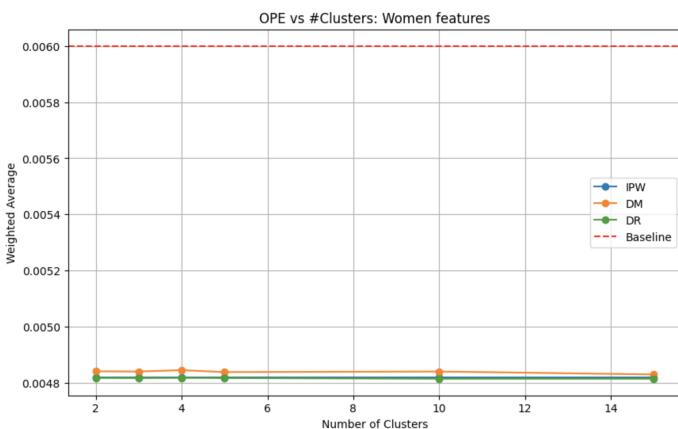


Figure 20: OPE Women Features + Baseline

Table 6: Promedio ponderado de clusters utilizando user affinity en all

Cluster	IPW	DM	DR
2	0.003468	0.003493	0.003465
3	0.003468	0.003493	0.003465
4	0.003468	0.003493	0.003463
5	0.003468	0.003493	0.003458

Table 7: Promedio ponderado de clusters utilizando user features en men

Cluster	IPW	DM	DR
2	0.005119	0.005145	0.005118
3	0.005119	0.005145	0.005118
4	0.005119	0.005139	0.005117
5	0.005119	0.005138	0.005118
10	0.005119	0.005129	0.005115

Table 8: Promedio ponderado de clusters utilizando user features en women

Cluster	IPW	DM	DR
2	0.004819	0.004841	0.004818
3	0.004819	0.004840	0.004817
4	0.004819	0.004845	0.004818
5	0.004819	0.004838	0.004817
10	0.004819	0.004840	0.004815
15	0.004819	0.004830	0.004815

Table 9: Promedio ponderado de clusters utilizando user features en all

Cluster	IPW	DM	DR
2	0.003468	0.003487	0.003467
3	0.003468	0.003486	0.003467
4	0.003468	0.003483	0.003467
5	0.003468	0.003486	0.003467
10	0.003468	0.003495	0.003466
15	0.003468	0.003482	0.003466

Table 5: Promedio ponderado de clusters utilizando user affinity en women

Cluster	IPW	DM	DR
2	0.004819	0.004832	0.004817
3	0.004819	0.004835	0.004817
4	0.004819	0.004823	0.004818

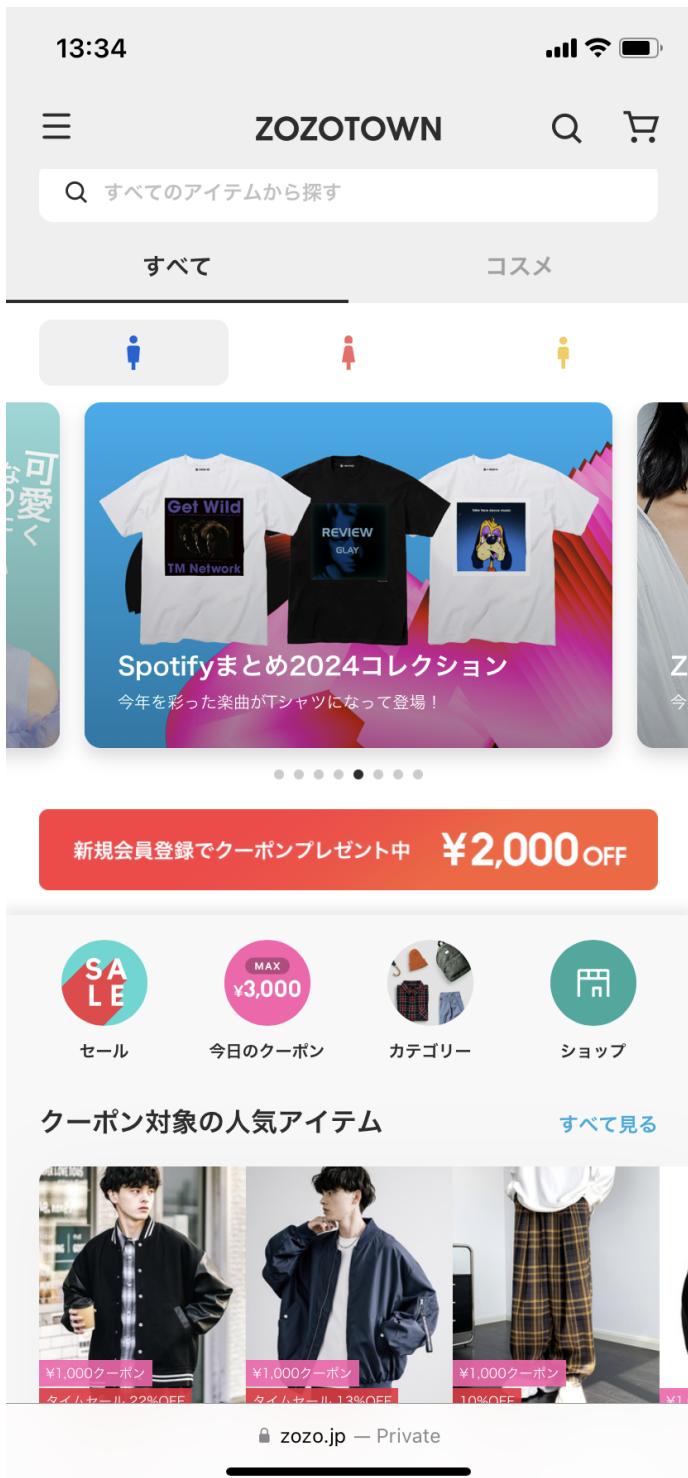


Figure 22: Zozo Town: Inicio

Table 11: Tamaño clusters usando user affinity en men

Total	#0	#1
2	132305	320644

Table 12: Tamaño clusters usando user affinity en women

Total	#0	#1	#2	#3	#4
2	860888	3697			
3	857155	3535	3895		
4	854191	3472	3880	3042	

Table 13: Tamaño clusters usando user affinity en all

Total	#0	#1	#2	#3	#4
2	1372115	2212			
3	1367927	2212	4188		
4	1363520	2212	4185	4410	
5	1361198	2122	3960	4524	2523



Figure 23: Zozo Town: Recomendaciones

Table 14: Tamaño clusters usando user features en men

Total	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
2	132305	320644								
3	132305	192952	127692							
4	132305	96745	127692	96207						
5	132305	92662	75541	55216	97225					
10	132305	70664	40429	23683	49503	28974	34167	32354	16570	24300

Table 15: Tamaño clusters usando user features en women

Total	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15
2	668605	195980													
3	341574	195980	327031												
4	314997	195980	248442	105166											
5	114711	195980	207675	105166	241053										
10	70428	195980	51827	84830	104635	88756	68086	58634	100630	40779					
15	45848	195980	17745	34667	77218	39912	57980	58661	99247	25417	21366	45134	60671	59566	25173

Table 16: Tamaño clusters usando user features en all

Total	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15
2	888334	485993													
3	311188	485993	577146												
4	235286	485993	437039	216009											
5	195060	485993	200646	216009	276619										
10	33119	485993	143690	154033	118991	98568	67794	87271	107486	77382					
15	30171	485993	117690	91524	101753	77971	43732	39682	84229	77382	60084	42564	35383	42793	43376