



TRABAJO FIN DE GRADO EN  
Máster en Big Data & Data Science

Nº GAN-XXXXXX

## **SISTEMA DE RECOMENDACIÓN PARA PEQUEÑAS TIENDAS DE VIDEOJUEGOS EN LATAM Y ESPAÑA**

---

**Autor**

RAFAEL EDUARDO DIAZ BONILLA

**Tutor**

FERRAN ARROYO

UNIVERSITAT DE BARCELONA  
**Instituto de Formación Continua – IL3**

Barcelona, septiembre 2022



# Índice

1. Introducción.....	2
1.1 Antecedentes.....	4
1.2 Objetivos.....	5
2. Sistemas de Recomendación .....	6
2.1 Filtros Colaborativos.....	7
2.1.1 Filtros Colaborativos basado en Usuarios.....	8
2.2.2 Filtros Colaborativos basado en elementos.....	10
2.2.3 FC usando datos 0-1 basado en - Usuarios y elementos.....	11
2.2.4 SR para datos 0-1 Basado en Reglas de Asociación.....	12
2.2.5 Otros métodos de filtros colaborativos.....	13
2.3 Evaluación de los algoritmos de SR .....	13
2.3.1 Evaluación de las calificaciones pronosticadas.....	14
2.3.2 Evaluación de las Top-N recomendaciones .....	14
3. Extracción de los datos - Steam.....	17
3.1 Paso 1: Extracción elementos de Steam.....	17
3.2 Paso 2: Obtener la información del videojuego .....	18
3.1.3 Paso 3: Extraer información de los usuarios.....	18
4. Metodología del proyecto .....	21
4.1 Comprensión del Negocio .....	21
4.2 Descripción de los datos .....	22
4.3 Análisis de los datos .....	24
4.4 Modelamiento de la información.....	26
4.3 Evaluación .....	27
4.4 Despliegue .....	30
5. Viabilidad Financiera.....	31
5.1 Modelo de Negocio.....	31
5.2 Benchmarking.....	34
5.3 Análisis de viabilidad .....	35
Conclusiones .....	38
Limitantes .....	38
Anexos .....	38
Referencias.....	39

# Capítulo 1

## 1. Introducción

El mercado de videojuegos de PC ha venido teniendo un enorme crecimiento en el mundo gracias a empresas como Steam. Esta plataforma fundada por Valve inicia en 2003, como compañía de videojuegos creada por dos ex-trabajadores de Microsoft, Mike Harrington y Gabe Newell.

En los 90 Valve se enfrentaba a la mentalidad de un mercado en donde los juegos se pirateaban con facilidad, y en donde todo el mundo era libre de hacer lo que quería con ellos, sin tener que crear cuentas ni conectarse online. Pero esa visión no era rentable económicamente para las compañías. Por otra parte, existían problemas adicionales como los *cheaters*, que usaban software para obtener ventajas en los juegos multijugador, y el problema de la instalación de parches manualmente, que para algunos juegos multijugador salían parches nuevos casi cada semana.

El 11 de septiembre de 2003, se lanza Steam, pero no como una tienda de videojuegos, sino como un software para instalar parches automáticamente en los juegos, y con medidas *anti-cheaters*. Aunque al principio Steam no la tuvo fácil poco a poco, se fue ganando a los usuarios por sus servicios y comodidad: ofrecía un único lugar para todos tus juegos, con instalación automática de parches, juego online, *anticheaters*, demos, y vídeos de los juegos. Esto se convirtió en un éxito lo que llevó a Valve a plantearse usar Steam para vender juegos. En 2007 la plataforma tenía 13 millones de usuarios, y cerca de 150 juegos, además, grandes compañías ya vendían sus juegos en este sitio, pues habría el acceso a nuevos jugadores que no compraban en tiendas físicas, y protegía su propiedad con medidas antipiratería.

En 2008 Steam estrena SteamCloud herramienta que permite guardar las partidas de los juegos en la nube. En el 2011 el *boom* de los juegos indie contribuye al crecimiento de la compañía, ya que estos juegos indie empezaron a ofrecer una calidad técnica y gráfica a la altura de los juegos comerciales, además de ofrecer originalidad con respecto a los juegos comerciales, que lanzaban una continuación tras otra. En el 2017 nace el servicio de Steam Direct, que permite publicar juegos directamente en la plataforma, lo que aumenta de forma brutal el catálogo de juegos de Steam.

Las cifras publicadas por SteamSpy en el año 2017, son impresionantes, pues Steam tenía 291 millones de usuarios, y el 22%, unos 63 millones, se habían unido ese mismo año. Ya que A principios de 2018 Steam limitó la forma de obtener estadísticas, por eso no hay datos tan precisos. Pero siguiendo esa progresión, actualmente Steam debe tener unos 325 o 350 millones de usuarios. Teniendo en cuenta que se han vendido unos 80 millones de consolas PS4, 40 millones de Xbox One y unos 20 millones Nintendo Switch, es fácil darse cuenta de la enormidad de Steam. Para julio de 2022, aproximadamente el 34,6 % de los usuarios de la plataforma de juegos Steam en todo el mundo utilizaban el inglés como idioma principal. El chino simplificado fue el segundo idioma más común, hablado por el 24,5 por ciento de los usuarios, seguido de ruso con un 11.3% y en cuarto lugar español-latam, lo que muestra que el mercado latino ha ido escalando a un lugar importante, por encima de idiomas como el alemán, portugués y francés. Por otra parte, el inventario de videojuegos en steam para abril del 2022 indicaba que había 50.046 juegos, y solamente el año pasado se generó un nuevo record de lanzamientos con 11784 juegos nuevos, lo que la convierte en la plataforma de videojuegos más popular del mundo.

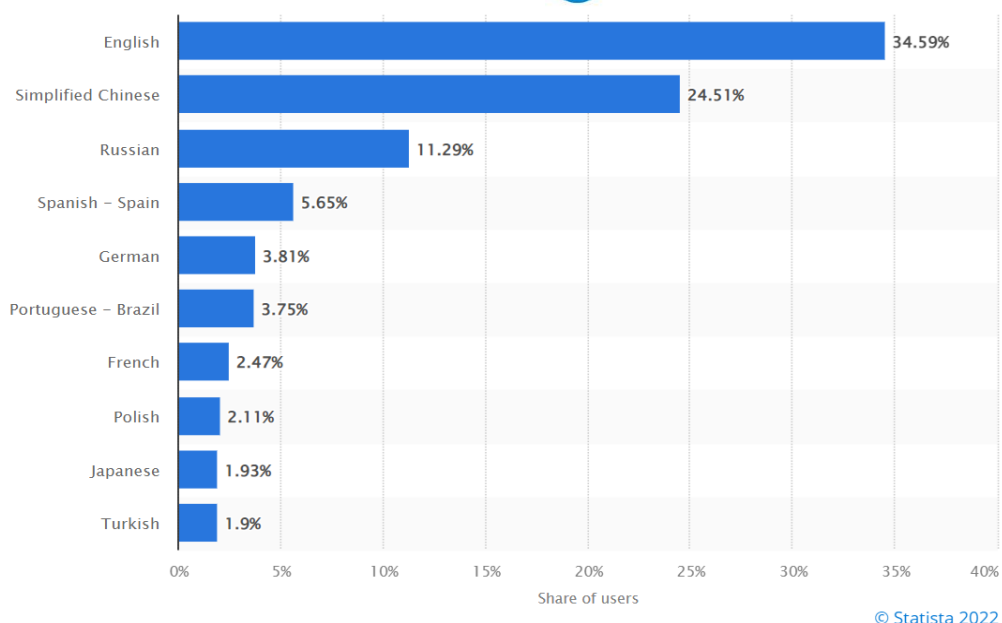


Figura 1.1. Porcentaje de usuarios por idioma en la plataforma de Steam.

Sin embargo, hay que poner estas cifras en perspectiva. La media de juegos comprados en Steam por los usuarios que se apuntaron en 2017 es de uno. Y los usuarios que llevan desde 2003 tienen una media de 15, uno por año. Muchos de los nuevos usuarios solo se apuntan para activar en Steam juegos gratuitos. Si bien Steam no inventó la rueda puso las bases del mercado de videojuegos digital, he hizo que los juegos de PC hoy vivan su mejor momento en la historia. Por ejemplo compañías japonesas publican en PC títulos como Monster Hunter World o Final Fantasy XV, solo porque existe Steam.

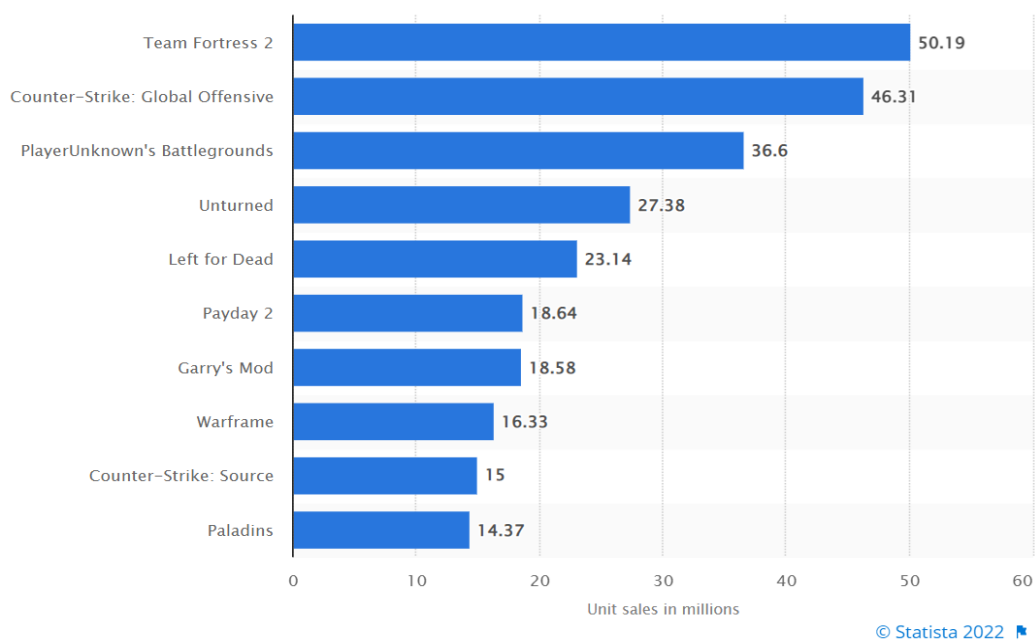


Figura 1.2 Los juegos más vendidos en Steam a partir de julio de 2018, (en millones).

Aunque hoy en día Steam es un gigante de los videojuegos, de un éxito inmenso, con más de 50.000 juegos y más de 120 millones de usuarios activos por mes, cuenta con algunos problemas.

El gigantismo es una espada de Damocles para cientos de desarrolladores cuyos juegos no se ven, pues encontrar buenos juegos entre docenas de miles de títulos, es cada vez más difícil. Y la polémica decisión de dejar publicar a todo el mundo está causando mucho malestar. En mercados saturados como este, solo los más populares ganan mucho dinero. Los 100 juegos más vendidos de Steam (eso solo son el 0.5%) se llevan el 50% de todas las ganancias, que rondan los 5.000 millones de dólares al año. Por ejemplo, de esos 5.000 millones solo PUBG (28 millones de copias vendidas) se llevó 600 millones en 2017, y GTA V (3,5 millones de copias) sobre 83 millones. Los primeros juegos para un jugador fueron Civilization VI (1,1 millones de copias) y The Witcher 3 (1,8 millones de copias), que ganaron unos 40 millones de dólares.

Sin embargo, otra de las características que impulso la expansión de Steam, como la apertura de Valve hacia los foros, el reestrenó del chat de voz en los juegos, los grupos, y las listas de amigos, permitía a los usuarios conocer gente con sus mismos gustos, quedar para jugar, crear grupos de amigos que juegan juntos. Esto ayudó a Steam no solo a fidelizar a sus clientes, y atraer a otros nuevos, sino también a empezar a implementar sistemas de recomendación interactivos para los usuarios, con los datos proporcionados por el usuario de forma directa o indirecta.

El sistema procede a analizar y procesar información histórica del usuario para transformar estos datos en conocimiento de recomendación. Basado en diferentes criterios como las valoraciones sobre el video juego, le gusta o no, en base en esto el modelo puede realizar predicciones sobre recomendaciones de videojuegos que puedan ser de utilidad o valor para el usuario, utilizando metodologías, como: filtrado demográfico, filtrado por contenido, filtrado colaborativo entre otros.

Al final con el sistema de recomendación se espera no solo mejorar la experiencia y satisfacción del usuario, sino que este compre más videojuegos y por lo tanto se incrementen las ventas de la plataforma, generando una rentabilidad mayor.

Debido a esta gran cantidad de oferta que se puede encontrar en diferentes plataformas de videojuegos, y de venta online, es difícil para el usuario encontrar productos que sean de su interés y que se adapten a sus necesidades. Debido a este problema, surgen los sistemas de recomendación que ayudan a los usuarios a encontrar los productos de su interés en este caso los videojuegos, mejorando su satisfacción y generando rentabilidad para las empresas que lo implementan. En este Trabajo de Fin de Máster se proponen elaborar un sistema de recomendación basado en filtros colaborativos, que permitan identificar los gustos de los usuarios y así crear recomendaciones. Para encontrar el mejor sistema de recomendación, se probaron diferentes algoritmos, los cuales se validaron mediante validación cruzada y se evaluaron en base de métricas como Precision, Recall y curva ROC. Para rentabilizar el proyecto se propone crear una empresa especializada en análisis de datos, cuyo primer proyecto es este sistema de recomendación el cual se les ofrecerá a plataformas de venta online de videojuegos, que no cuenten con sistemas de recomendación.

## 1.1 Antecedentes

Los sistemas de recomendación inician desde que la humanidad empieza analizar y tomar decisiones de manera inteligente, siempre se baso en recomendaciones de amigos, familiares, expertos para tomar la mejor decisión en algún tema. Pasando por estadísticas sencillas, como cual es el producto más vendido, que características son las que más gustan de un producto, etc.

Desde entonces los sistemas de recomendación (SR) han sido un campo de investigación emergente que ha crecido rápidamente y se ha vuelto popular. El aumento del interés en este tema de investigación también ha sido impulsado por las mejores de los PC y el comercio electrónico.

Cuyas empresas se han visto directamente beneficiado, por ejemplo: ayudando a los compradores sin experiencia en compras en línea, vendiendo productos de forma cruzada y mejorando la lealtad del cliente. La explosión máxima de la investigación en SR se produjo cuando Amazon lanzó su método Filtros Colaborativos (FC) a fines de la década de 1990, aumentando con éxito sus ventas, esto hizo que otras empresas en línea comenzaran a implementar SR en su sitio web. El objetivo principal de un SR es encontrar la información preferida y eliminar la información que no le gusta al usuario, el campo SR puede considerarse como un subconjunto del filtrado de información.

La idea de explotar las computadoras para recomendar el mejor artículo para el usuario ha existido desde el comienzo de la informática. La primera implementación del concepto SR apareció en 1979, en un sistema llamado Grundy ([Qomariyah, 2018](#)), un bibliotecario basado en computadora que brindaba sugerencias al usuario sobre qué libros leer. Esto siguió a principios de la década de 1990 con el lanzamiento de Tapestry, ([Terry, 1993](#)) el primer SR comercial. GroupLens, un laboratorio de investigación de la Universidad de Minnesota, EE. UU., lanzó otra implementación de SR para ayudar a las personas a encontrar sus artículos preferidos a principios de la década de 1990. Llamaron al sistema por el grupo, GroupLens Recommender System. Este sistema pretende tener un espíritu similar al de Tapestry, Ringo, BellCore y Jester. Un mayor desarrollo de los SR a fines de la década de 1990 fue la implementación de Amazon Collaborative Filtering, una de las tecnologías de SR más conocidas. Desde esta era, los SR basados en Filtrado Colaborativo se han vuelto muy populares y han sido implementados por muchos sistemas de comercio electrónico y en línea. La historia de éxito de Amazon también dio lugar al desarrollo de algoritmos con enfoque híbridos. En 1996, MovieLens a través de su empresa Net Perceptions crea el SR para Moviefinder.

Luego de la era exitosa a fines de la década de 1990, la industria ofreció una generosa financiación para implementar la investigación de SR. La competencia más popular la llevó a cabo Netflix, que consistían en mejorar en un 10% el sistema ya existente llamado Cinematch, para cuantificarlo se utilizó el error cuadrático medio (RMSE). El Premio Netflix1 en 2006 y otorgo 1 millón de dólares estadounidenses al ganador de la competencia que proporcionó el mejor sistema de recomendación para películas, anunciando al equipo ganador en 2009. En 2010, YouTube también implementó un RS en su sitio web.

## 1.2 Objetivos

El objetivo principal de este trabajo es elaborar el mejor sistema de recomendación en videojuegos, basado en filtros colaborativos que permitan lograr un crecimiento en ventas, además de mejorar la experiencia y satisfacción del usuario en la compra de videojuegos.

Como objetivos secundarios de este TFM en base al principal es:

1. Investigar los diferentes sistemas de recomendación, que son, como se utilizan y la construcción desde su parte teórica.
2. Crear una API web que permita al cliente interactuar dinámicamente, para buscar la información de los videojuegos, usuarios y cuáles son el top 10 de recomendaciones.
3. Construir un modelo de negocio rentable en base en el modelo previamente construido

## Capítulo 2

### 2. Sistemas de Recomendación

Los sistemas de recomendación aplican técnicas estadísticas y de descubrimiento de conocimiento al problema de hacer recomendaciones de productos basados en datos históricos. La creación de recomendaciones personalizadas generadas automáticamente para productos que incluyen libros, canciones, programas de televisión y películas mediante el filtrado colaborativo. Hoy en día, los sistemas de recomendación son una tecnología exitosa utilizada por los líderes del mercado en varias industrias (por ejemplo, Amazon, Netflix y Pandora). En el comercio minorista, tales recomendaciones pueden mejorar las tasas de conversión al ayudar al cliente a encontrar los productos que desea comprar más rápido, promover la venta cruzada al sugerir productos adicionales y puede mejorar la lealtad del cliente. ([Schafer, Konstan, and Riedl 2001](#)).

La mayoría de software libre disponible por lo general solo implementan una metodología de SR que parte de un proyecto de investigación. Para este TFM se utilizará el paquete **recommenderlab** disponible en el software R. Este paquete proporciona una infraestructura de investigación integral para los sistemas de recomendación. Además, **recommenderlab** se utiliza en varios cursos universitarios para demostrar los conceptos básicos del desarrollo del sistema de recomendación, y varios investigadores emplearon el paquete para desarrollar y probar sus propios algoritmos ([Hasler, 2022](#)).

Los sistemas de recomendación se agrupan en 2 grandes grupos:

1. **Basados en Contenido:** toman en cuenta el contexto y características tanto de los usuarios como los ítems. Ej: modelos de clasificación para saber si un usuario elegirá o no una película en base a sus características, las características de la película y el comportamiento pasado del usuario.
2. **Filtros Colaborativos:** modelos basados exclusivamente en el comportamiento de los usuarios con respecto a los ítems. Ej: rating dado por parte de los usuarios a las películas. Se pueden utilizar los ratings para encontrar usuarios (o películas) similares para recomendar en base a la similitud.

Este TFM se centra en las técnicas de filtrado colaborativo que se basa en la idea de que, dados los datos de calificación de muchos usuarios para muchos elementos (por ejemplo, de 1 a 5 estrellas, o me gusta o no el elemento), La premisa es que los usuarios que estuvieron de acuerdo con la calificación de algunos elementos generalmente también tienden a estar de acuerdo con la calificación de otros elementos. El paquete **recommenderlab** proporciona diferentes algoritmos populares, como los siguientes.

- **El filtrado colaborativo basado en usuarios (UBCF):** Encontrar usuarios similares y recomendar en base a los vecinos más cercanos.
- **El filtrado colaborativo basado en elementos (IBCF):** Encontrar similitud entre ítems y realizar recomendaciones en base a los ítems más similares a los preferidos por el usuario.
- **Singular Value Decomposition (SVD):** Los modelos de factores latentes usan la descomposición de valores singulares (SVD) para estimar las calificaciones faltantes usando métodos como SVD con imputación de columna, Funk SVD o mínimos cuadrados alternos.
- **El recomendador basado en reglas de asociación (AR):** usa reglas de asociación para encontrar elementos recomendados.



- **Aleternating last squares (ALS):** Factorización de matrices, utiliza el algoritmo altering last squares.
- **Popular (POPULAR):** Es un algoritmo no personalizado que recomienda a todos los usuarios los elementos más populares que aún no han calificado.
- **Aleatorio (RANDOM):** Crea recomendaciones aleatorias que se pueden utilizar como referencia para la evaluación del algoritmo de recomendación.
- **Re-recommended liked items (RERECOMMEND):** recomienda elementos que el usuario ha valorado altamente en el pasado. Estas recomendaciones pueden ser útiles para artículos que normalmente se consumen más de una vez (por ejemplo, escuchar canciones o comprar comestibles).
- **Recomendaciones híbridas (HybridRecommender):** agrupa las recomendaciones de varios algoritmos ponderándolas por un vector de pesos.

A continuación, se proporciona el detalle algunos de estos algoritmos para el resto pueden encontrar información detallada en el libro de encuestas de [Desrosiers y Karypis \(2011\)](#).

## 2.1 Filtros Colaborativos

Para comprender el uso del software, son necesarias algunas definiciones formales. A menudo daremos ejemplos para un recomendador de películas, pero los ejemplos también se generalizan a otros tipos de artículos. Sea  $\mathcal{U} = \{u_1, u_2, \dots, u_m\}$  el conjunto de usuarios  $\mathcal{I} = \{i_1, i_2, \dots, i_n\}$  el conjunto de ítems. Las calificaciones se almacenan en una matriz  $\mathcal{R} = (r_{jl})$  usuarios-elementos, de tamaño  $m \times n$ , donde cada fila representa un usuario  $u_j$  con  $1 \leq j \leq m$  y cada columna representa el ítem  $i_l$  con  $1 \leq l \leq n$ . Usamos  $\mathbf{r}_j$  para denotar el vector de filas  $\mathbb{R}$  con las calificaciones del usuario  $u_j$ . Las calificaciones usan una escala específica.

Por ejemplo, Netflix usa de 1 a 5 estrellas. y las calificaciones estimadas pueden estar dentro de un intervalo de rango coincidente (p. ej.,  $[1, 5]$ ). Por lo general, solo se conoce una pequeña fracción de las calificaciones y para muchas celdas en  $\mathbb{R}$ , faltan los valores. Los valores faltantes representan películas que el usuario no calificó y que posiblemente tampoco haya visto todavía.

El filtrado colaborativo tiene como objetivo crear recomendaciones para un usuario llamado usuario activo  $u_a \in \mathcal{U}$ . Definimos el conjunto de elementos desconocidos para el usuario  $u_a$  como  $\mathcal{J}_a = \mathcal{I} \setminus \{i_l \in \mathcal{I} \mid r_{al} \text{ no es faltante}\}$ . Las dos tareas típicas son predecir las calificaciones de todos los elementos en  $\mathcal{J}_a$  o crear una lista que contenga los mejores  $N$  ítems recomendados de  $\mathcal{J}_a$  (es decir, una lista de recomendaciones de los  $N$  principales) para  $u_a$ . Predecir todas las calificaciones que faltan significa completar la fila de la matriz de calificación donde los valores que faltan para los elementos en  $\mathcal{J}_a$  se reemplazan por calificaciones estimadas a partir de otros datos en  $\mathbb{R}$ . Desde este punto de vista, los sistemas de recomendación están relacionados con el problema de completar la matriz. La creación de una lista de los primeros  $N$  puede verse como un segundo paso después de predecir las calificaciones de todos los elementos desconocidos en  $\mathcal{J}_a$  y luego tomar los  $N$  elementos con las calificaciones predichas más altas. Algunos algoritmos omiten la predicción de calificaciones primero y pueden encontrar los  $N$  elementos principales directamente. Una lista de las mejores  $N$ -recomendaciones para un usuario  $u_a$  es un conjunto parcialmente ordenado  $T_N = (\chi, \geq)$ , donde  $\chi \subset \mathcal{J}_a$  y  $|\chi| \leq N$  ( $|\cdot|$  denota la cardinalidad del conjunto). Tenga en cuenta que puede haber casos donde la lista de los top- $N$  principales contengan menos de  $N$  elementos. Esto puede pasar si  $|\mathcal{J}_a| < N$  o si el algoritmo CF no puede identificar  $N$  artículos para recomendar. La relación binaria  $\geq$  es definida como  $x \geq y$  si y solo si  $\hat{r}_{ax} \geq \hat{r}_{ay}$  para todo  $x, y \in \chi$ . Además, se requiere que

$\forall_{x \in \mathcal{X}} \forall_{y \in \mathcal{I}_a} \hat{r}_{ax} \geq \hat{r}_{ay}$  para garantizar que la lista N superior contenga solo los elementos con la calificación estimada más alta.

Por lo general, tratamos con una gran cantidad de elementos con calificaciones desconocidas, lo que hace que predecir primero los valores de calificación para todos ellos sea computacionalmente costoso. Algunos enfoques (p. ej., enfoques basados en reglas) pueden predecir la lista de los primeros N directamente sin considerar primero todos los elementos desconocidos.

Los algoritmos de filtrado colaborativo generalmente se dividen en dos grupos, algoritmos CF basados en memoria y CF basados en modelos ([Breese, Heckerman y Kadie 1998, citado por Hahsler 2022](#)). Los CF basados en memoria utilizan toda la base de datos de usuarios (o al menos una gran muestra de ella) para crear recomendaciones.

El algoritmo más destacado es el filtrado colaborativo basado en el usuario. La desventaja de este enfoque es la escalabilidad, ya que toda la base de datos de usuarios debe procesarse en línea para crear recomendaciones. Los algoritmos basados en modelos usan la base de datos de usuarios para aprender un modelo más compacto (por ejemplo, grupos con usuarios de preferencias similares) que luego se usa para crear recomendaciones.

A continuación, presentaremos los conceptos básicos de la memoria conocida y los algoritmos de filtrado colaborativo basados en modelos. Se puede encontrar más información sobre estos algoritmos en el reciente capítulo del libro de encuestas de [Desrosiers y Karypis \(2011\)](#).

### 2.1.1 Filtros Colaborativos basado en Usuarios

El filtro colaborativo basado en el usuario ([Shardanand y Maes 1995](#)) es un algoritmo basado en la memoria que intenta imitar el boca a boca analizando los datos de calificación de muchos individuos. La suposición es que los usuarios con preferencias similares calificarán los elementos de manera similar. Por lo tanto, las calificaciones que faltan para un usuario se pueden predecir encontrando primero un vecindario de usuarios similares y luego agregando las calificaciones de estos usuarios para formar una predicción.

La vecindad se define en términos de similitud entre usuarios, ya sea tomando un número dado de usuarios más similares (k vecinos más cercanos) o todos los usuarios dentro de un umbral de similitud dado. Las medidas de similitud populares para CF son el coeficiente de correlación de Pearson y la similitud del coseno. Estas medidas se definen entre dos vectores con valores  $x$  e  $y$ .

$$m_{Pearson}(x, y) = \frac{1}{n-1} \sum_{l=1}^n \left( \frac{x_l - \bar{x}}{s_x} \right) \left( \frac{y_l - \bar{y}}{s_y} \right) \quad (1)$$

y

$$m_{Cosine}(x, y) = \frac{x \cdot y}{\|x\| \|y\|}, \quad (2)$$

donde  $n$  es el número de elementos en el vector de calificaciones y las calificaciones perdidas son saltadas para el cálculo.  $s_x$  y  $s_y$  son la desviación estándar y  $\|\cdot\|$  es la norma  $l^2$  de un vector. Para calcular la medida entre dos usuarios,  $u_x$  y  $u_y$ ,  $x = r_x$  e  $y = r_y$  representan los vectores de fila en  $\mathbb{R}$  con los vectores de perfil de los dos usuarios. Para calcular la similitud utilizando datos de calificación, solo se utilizan las dimensiones (elementos) que fueron calificados por ambos usuarios. Tenga en cuenta que el coseno y la correlación están en el rango  $[-1, 1]$ , pero las medidas de similitud deben estar en el rango de  $[0, 1]$ . Nosotros, usamos la transformación

$$s = \frac{1}{2}m + 1$$

Usando la similitud, la vecindad para el usuario activo  $\mathcal{N}(a) \subset \mathcal{U}$  se puede seleccionar mediante un umbral en la similitud o tomando los k vecinos más cercanos. Una vez que se encuentran los

usuarios en el vecindario, sus calificaciones se agregan para formar la calificación pronosticada para el usuario activo  $u_a$ . La forma más fácil es simplemente promediar las calificaciones en el vecindario. Para el artículo  $i_l$  esto es

$$\hat{r}_{al} = \frac{1}{|\mathcal{N}(a)|} \sum_{i \in \mathcal{N}(a)} r_{il}. \quad (3)$$

En la Figura 1 se muestra un ejemplo del proceso de creación de recomendaciones por CF basado en el usuario.

A la izquierda se encuentra la matriz de calificación  $\mathcal{R}$  con 6 usuarios y 8 ítems y calificaciones en el rango de 1 a 5 (estrellas). Queremos crear recomendaciones para el usuario activo  $u_a$  que se muestra en la parte inferior de la matriz. Para encontrar el  $k$ -vecindario (es decir, los  $k$  vecinos más cercanos), calculamos la similitud entre el usuario activo y todos los demás usuarios en función de sus calificaciones en la base de datos.

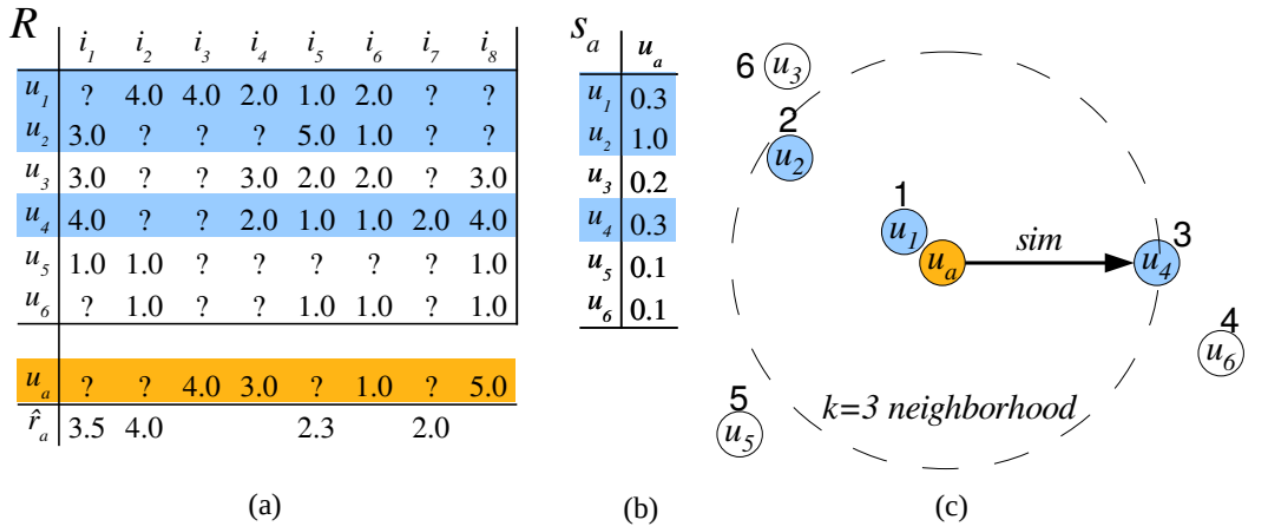


Figura 2.1: Ejemplo de filtrado colaborativo basado en usuarios con (a) matriz de calificación  $\mathcal{R}$  y calificaciones estimadas para el usuario activo, (b), similitudes entre el usuario activo y los otros usuarios  $s_a$  (distancia euclidiana convertida en similitudes), y (b) la formación del vecindario de usuarios.

y luego seleccionar los  $k$  usuarios con mayor similitud. A la derecha en la Figura 2.1, vemos una representación bidimensional de las similitudes (los usuarios con mayor similitud se muestran más cerca) con el usuario activo en el centro. Los  $k = 3$  vecinos más cercanos ( $u_1, u_2$  y  $u_3$ ) se seleccionan y marcan en la base de datos a la izquierda. Para generar una calificación estimada agregada, calculamos las calificaciones promedio en el vecindario para cada elemento no calificado por el usuario activo. Para crear una lista de recomendaciones de los  $N$  principales, los elementos se ordenan por calificación prevista. En el pequeño ejemplo de la Figura 1, el orden en la lista de los primeros  $N$  con ( $N \geq 4$ ) es  $i_2, i_1$  y  $i_5$ . Sin embargo, para una aplicación real, probablemente no recomendaríamos los elementos  $i_7$  e  $i_5$  debido a sus bajas calificaciones.

El hecho de que algunos usuarios en el vecindario sean más similares al usuario activo que otros (ver Figura 1 (b)) puede incorporarse como pesos en la Ecuación (3).

$$\hat{r}_{al} = \frac{1}{\sum_{i \in \mathcal{N}(a)} s_{ai}} \sum_{i \in \mathcal{N}(a)} s_{ai} r_{il} \quad (4)$$

$s_{ai}$  es la similaridad entre el usuario activo  $u_a$  en el vecindario. Para los datos de calificación, el rendimiento del algoritmo de recomendación se puede mejorar eliminando el sesgo de calificación del usuario, donde algunos usuarios tienden a usar siempre calificaciones más altas, mientras que otros tienden a usar calificaciones más bajas. Esto se puede hacer normalizando los datos de

calificación antes de aplicar el algoritmo de recomendación. Cualquier función de normalización  $h: \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$  puede ser usada para preprocesamiento. Dicha función debe ser reversible por  $h^{-1}$  para mapear la calificación predicha en la escala normalizada de regreso a la escala de calificación original. La normalización se usa para eliminar el sesgo de calificación individual por parte de los usuarios que constantemente siempre usan calificaciones más bajas o altas que otros usuarios. El método más popular es centrar las filas de la matriz de calificación de elementos de usuario por

$$h(r_{jl}) = r_{jl} - \bar{r}_j,$$

donde  $\bar{r}_j$  es la media de todas las calificaciones disponibles en la fila  $j$  del elemento-usuario en la matriz de calificaciones  $\mathcal{R}$ . Esto significa que las calificaciones ahora se miden para cada usuario por cuánto están por encima o por debajo de la calificación promedio del usuario. Lo contrario es simplemente,

<b>S</b>	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$i_8$	$\hat{r}_a$	$k=3$
$i_1$	-	0.1	0	<b>0.3</b>	<b>0.2</b>	<b>0.4</b>	0	0.1	-	
$i_2$	0.1	-	<b>0.8</b>	<b>0.9</b>	0	<b>0.2</b>	0.1	0	0.0	
$i_3$	0	<b>0.8</b>	-	0	<b>0.4</b>	0.1	0.3	<b>0.5</b>	4.6	
$i_4$	<b>0.3</b>	<b>0.9</b>	0	-	0	0.1	0	<b>0.2</b>	3.2	
$i_5$	<b>0.2</b>	0	<b>0.4</b>	0	-	0.1	<b>0.2</b>	0.1	-	
$i_6$	<b>0.4</b>	<b>0.2</b>	0.1	<b>0.3</b>	0.1	-	0	0.1	2.0	
$i_7$	0	<b>0.1</b>	<b>0.3</b>	0	<b>0.2</b>	0	-	0	4.0	
$i_8$	<b>0.1</b>	0	<b>0.5</b>	<b>0.2</b>	0.1	0.1	0	-	-	
$u_a$	2	?	?	?	4	?	?	5		

Figura 2.2: Filtrado colaborativo basado en elementos

$$h^{-1}(h(r_{jl})) = h(r_{jl}) + \bar{r}_j = r_{jl}.$$

En la literatura se pueden encontrar otros métodos, como la normalización del puntaje Z, que también tiene en cuenta la varianza de la calificación (ver, por ejemplo, [Desrosiers y Karypis 2011](#)). Los dos problemas principales de la FC basada en el usuario son que toda la base de datos del usuario debe mantenerse en la memoria y que debe realizarse un costoso cálculo de similitud entre el usuario activo y todos los demás usuarios en la base de datos.

## 2.2.2 Filtros Colaborativos basado en elementos

El filtro colaborativo baso en elementos ([Deshpande y Karypis 2004](#)) es un enfoque que produce recomendaciones basadas en la relación entre los ítems inferidos de la matriz de calificación. La suposición detrás de este enfoque es que los usuarios preferirán artículos que sean similares a otros artículos que les gustan.

El paso de construcción del modelo consiste en calcular una matriz de similitud que contiene todas las similitudes entre elementos utilizando una medida de similitud determinada. De nuevo son populares la correlación de Pearson y la similitud del coseno y se utilizan las ecuaciones 1 y 2. Aquí, los vectores de calificación  $x$  e  $y$  son columnas de  $\mathcal{R}$  que representan las calificaciones de dos elementos.

Todas las similitudes por pares se almacenan en una matriz de similitud  $\mathcal{S}$  de tamaño  $n \times n$ . Para reducir el tamaño del modelo a  $n \times k$  con  $k \ll n$ , para cada elemento solo se almacena una lista de los  $k$  elementos más similares y sus valores de similitud. Los elementos  $k$  que son más similares al elemento  $il$  se denotan por el conjunto  $\mathcal{S}(l)$  que puede verse como la vecindad del tamaño  $k$  del elemento. Retener solo  $k$  similitudes por elemento mejora significativamente la complejidad del

espacio y el tiempo, pero potencialmente sacrifica algo de la calidad de la recomendación ([Sarwar et al. 2001](#), citado por [Hahsler 2022](#)).

Para hacer una recomendación basada en el modelo, usamos las similitudes para calcular una suma ponderada de las calificaciones del usuario para elementos relacionados.

$$\hat{r}_{al} = \frac{1}{\sum_{i \in \mathcal{S}(l)} s_{li}} \sum_{i \in \mathcal{S}(l)} s_{li} r_{ai} \quad (5)$$

La Figura 2 muestra un ejemplo para  $n = 8$  elementos con  $k = 3$ . Para la matriz de similitud  $\mathbf{S}$ , solo se almacenan las  $k = 3$  entradas más grandes por fila (estas entradas están marcadas en negrita). Para el ejemplo, asumimos que tenemos calificaciones para el usuario activo para los elementos  $i_1, i_5$  e  $i_8$ . Las filas correspondientes a estos elementos se resaltan en la matriz de similitud de elementos. Ahora podemos calcular la suma ponderada usando las similitudes (solo se usa la matriz reducida con las  $k = 3$  calificaciones más altas) y las calificaciones de los usuarios. El resultado (debajo de la matriz) muestra que  $i_3$  tiene la calificación estimada más alta para el usuario activo.

De manera similar a los algoritmos de recomendación basados en el usuario, el sesgo del usuario se puede reducir normalizando primero la matriz de calificación del elemento del usuario antes de calcular la matriz de similitud de elemento a elemento.

La FC basada en elementos es más eficiente que la FC basada en usuarios, ya que el modelo (matriz de similitud reducida) es relativamente pequeño ( $N \times k$ ) y se puede precalcular por completo. Se sabe que la FC basada en elementos solo produce resultados ligeramente inferiores en comparación con la FC basada en el usuario y son posibles modelos de orden superior que tienen en cuenta la distribución conjunta de conjuntos de elementos ([Deshpande y Karypis 2004](#)). Además, la FC basada en artículos se aplica con éxito en sistemas de recomendación a gran escala (por ejemplo, por Amazon.com).

### 2.2.3 FC usando datos 0-1 basado en - Usuarios y elementos

Hay menos investigación disponible para situaciones en las que no se dispone de una gran cantidad de datos de calificación detallados obtenidos directamente. Sin embargo, esta es una situación común y ocurre cuando los usuarios no quieren revelar directamente sus preferencias al calificar un elemento (por ejemplo, porque consume mucho tiempo). En este caso, las preferencias solo se pueden inferir analizando el comportamiento de uso. Por ejemplo, podemos registrar fácilmente en un supermercado qué artículos compra un cliente. Sin embargo, no sabemos por qué no se compraron otros productos. La razón puede ser una de las siguientes.

- El cliente no necesita el producto en este momento.
- El cliente no conoce el producto. Tal producto es un buen candidato por recomendación.
- Al cliente no le gusta el producto. Evidentemente, un producto de este tipo no debería ser recomendado.

Mild y Reutterer (2003) presentan y evalúan algoritmos de recomendación para esta configuración. El mismo razonamiento es válido para recomendar páginas de un sitio web dados los datos de flujo de clics. Aquí solo tenemos información sobre qué páginas se vieron, pero no por qué no se vieron algunas páginas. Esta situación conduce a datos binarios o más exactamente a datos 0-1 donde 1 significa que inferimos que el usuario tiene preferencia por un elemento y 0 significa que al usuario no le gusta el elemento o no lo conoce.

[Pan, Zhou, Cao, Liu, Lukose, Scholz y Yang \(2008\)](#) llaman a este tipo de datos en el contexto del filtrado colaborativo análogo a situaciones similares para clasificadores de datos de una clase, ya

que solo la clase 1 es pura y contiene solo datos positivos. ejemplos La clase 0 es una mezcla de ejemplos positivos y negativos.

En el caso 0 – 1 con  $r_{jl} \in 0,1$  definimos:

$$r_{jl} = \begin{cases} 1, & \text{donde el usuario } u_j \text{ se conoce su preferencia por el elemento } i_l \\ 0, & \text{otro caso.} \end{cases} \quad (6)$$

Dos estrategias para manejar los datos de una clase es asumir que todas las calificaciones que faltan (ceros) son ejemplos negativos o suponer que todas las calificaciones que faltan son desconocidas. Además, [Pan et al. \(2008\)](#) proponen estrategias que representan una compensación entre las dos estrategias extremas basadas en aproximaciones ponderadas de rango bajo de la matriz de calificación y en un muestreo de ejemplo negativo que podría mejorar los resultados en todos los algoritmos de recomendación.

Si asumimos que los usuarios suelen preferir solo una pequeña fracción de los elementos y, por lo tanto, la mayoría de los elementos sin calificación serán ejemplos negativos. entonces no tenemos valores perdidos y podemos usar los enfoques descritos anteriormente para datos de calificación de valor real. Sin embargo, si asumimos que todos los ceros son valores faltantes, entonces esto lleva al problema de que no podemos calcular las similitudes usando la correlación de Pearson o la similitud del coseno ya que las partes que no faltan de los vectores solo contienen unos. Una medida de similitud que solo se enfoca en hacer coincidir unos y, por lo tanto, evita el problema con los ceros es el *índice de Jaccard*:

$$sim_{Jaccard}(x, y) = \frac{|x \cap y|}{|x \cup y|}, \quad (7)$$

donde  $x$  e  $y$  son los conjuntos de los ítems con 1 en los perfiles de usuario  $u_a$  y  $u_b$ , respectivamente.

El índice Jaccard se puede utilizar entre usuarios para el filtrado basado en usuarios y entre elementos para el filtrado basado en elementos como se describe anteriormente.

## 2.2.4 SR para datos 0-1 Basado en Reglas de Asociación

Los sistemas de recomendación que utilizan reglas de asociación producen recomendaciones basadas en un modelo de dependencia para elementos dados por un conjunto de reglas de asociación ([Demíriz 2004](#)). La matriz de perfil binario  $\mathcal{R}$  se ve como una base de datos donde cada usuario es tratado como una transacción que contiene el subconjunto de elementos en  $I$  con una calificación de 1. Por lo tanto, la transacción  $k$  se define como  $\tau_k = \{i_j \in I \mid r_{jk} = 1\}$  y toda la base de datos de transacciones es  $\mathcal{D} = \{\tau_1, \tau_2, \dots, \tau_U\}$  donde  $U$  es el numero de usuarios. Para construir el modelo de dependencia, se extrae un conjunto de reglas de asociación  $\mathbf{R}$  sobre  $\mathcal{R}$ . Las reglas de asociación son de la forma  $x \rightarrow y$  donde  $x, y \subseteq I$  y  $x \cap y = \emptyset$ . Para el modelo solo usamos reglas de asociación con un solo elemento en el lado derecho de la regla ( $|y| = 1$ ). Para seleccionar un conjunto de reglas de asociación útiles, se utilizan umbrales sobre medidas de importancia e interés. Dos medidas ampliamente aplicadas son:

$$\begin{aligned} soporte(x \rightarrow y) &= soporte(x \cap y) = \frac{Freq(x \cap y)}{|\mathcal{D}|} = \hat{P}(E_x \cap E_y) \\ confianza(x \rightarrow y) &= \frac{soporte(x \cap y)}{soporte(x)} = \hat{P}(E_y | E_x) \end{aligned}$$

$Freq(I)$  da el número de transacciones en la base de datos  $\mathcal{D}$  que contiene todos los elementos en  $I$ .  $E_I$  es el evento en el que el conjunto  $I$  está contenido en una transacción.

Requerimos  $soporte(x \rightarrow y) > s$  y  $confianza(x \rightarrow y) > c_y$  esto incluye las restricciones  $|x \cap y| \leq l$ . El conjunto de reglas  $\mathbf{R}$  que satisfacen estas restricciones forman el modelo de



dependencia. Aunque encontrar todas las reglas de asociación dados los umbrales de apoyo y confianza es un problema difícil (el modelo crece exponencialmente en el peor de los casos con el número de elementos), están disponibles algoritmos que encuentran eficientemente todas las reglas en la mayoría de los casos (p. ej., [Agrawal y Srikant 1994](#); [Zaki 2000](#); [Han, Pei, Yin y Mao 2004](#), citado por [hahsler 2022](#)). También el tamaño del modelo se puede controlar mediante  $\ell$ ,  $s$  y  $c$ .

Para realizar una recomendación a un usuario activo  $u_a$  dado el conjunto de elementos  $\tau_a$  que le gustan al usuario y el conjunto de reglas de asociación  $\mathbf{R}$  (modelo de dependencia), son necesarios los siguientes pasos:

1. Encuentra todas las reglas coincidentes  $x \rightarrow y$  para las cuales  $x \subseteq_a \tau$  en  $\mathbf{R}$ .
2. Recomendar  $N$  lados derechos únicos ( $y$ ) de las reglas de coincidencia con la confianza más alta (u otra medida de interés).

El modelo de dependencia es muy similar al FC basado en ítems con similitud basada en probabilidad condicional ([Deshpande y Karypis 2004](#)). Puede precalcularse completamente y reglas con más de un elemento en el lado izquierdo( $x$ ), incorpora efectos de orden superior entre más de dos elementos.

### 2.2.5 Otros métodos de filtros colaborativos

Con el tiempo, se han desarrollado varios otros enfoques basados en modelos. Un enfoque popular simple basado en elementos es el algoritmo Slope One ([Lemire y Maclachlan 2005](#)). Otra familia de algoritmos se basa en el enfoque de factores latentes mediante la descomposición de matrices (Koren et al. 2009). Más recientemente, el aprendizaje profundo se ha convertido en un método muy popular para la finalización flexible de matrices, la factorización de matrices y la clasificación colaborativa. [Zhang, Yao, Sun y Tay \(2019\)](#) presentan una encuesta exhaustiva. Estos algoritmos están fuera del alcance de este documento.

## 2.3 Evaluación de los algoritmos de SR

La evaluación de los sistemas de recomendación es un tema importante y [Gunawardana y Shani \(2009\)](#) presentaron revisiones. Por lo general, dada una matriz de calificación  $\mathbf{R}$ , los algoritmos de recomendación se evalúan dividiendo primero los usuarios (filas) en  $\mathbf{R}$  en dos conjuntos  $\mathcal{U}_{train} \cap \mathcal{U}_{test} = \mathcal{U}$ . Las filas de  $\mathbf{R}$  correspondientes a los usuarios de entrenamiento se utilizan para entrenar el modelo de recomendación. Entonces cada usuario  $u_a \in \mathcal{U}_{test}$  es visto como un usuario activo. Antes de crear recomendaciones, algunos elementos se retienen del perfil  $r_{u_a}$  y mide qué tan bien es la calificación pronosticada coincide con el valor retenido o, para los algoritmos  $N$  principales, si los elementos en la lista recomendada tienen una calificación alta por parte del usuario. Finalmente, se promedian las medidas de evaluación calculadas para todos los usuarios de la prueba.

Para determinar cómo dividir  $\mathcal{U}$  en  $\mathcal{U}_{train}$  y  $\mathcal{U}_{test}$  podemos usar varios enfoques ([Kohavi 1995](#)).

- **Splitting (División):** podemos asignar aleatoriamente una proporción predefinida de los usuarios al conjunto de entrenamiento y todos los demás al conjunto de prueba.
- **Muestreo Bootstrap:** podemos muestrear desde  $\mathcal{U}_{test}$  con reemplazo para crear el conjunto de entrenamiento y luego usar los usuarios que no están en el conjunto de entrenamiento como conjunto de prueba. Este procedimiento tiene la ventaja de que para conjuntos de datos más pequeños podemos crear conjuntos de entrenamiento más grandes y todavía quedan usuarios para probar.

- **Validación cruzada de k-capas:** Aquí dividimos  $\mathcal{U}$  en  $k$  conjuntos (llamados capas) de aproximadamente el mismo tamaño. Luego evaluamos  $k$  veces, siempre usando un pliegue para probar y todos los demás pliegues para inclinarse. Los  $k$  resultados se pueden promediar. Este enfoque asegura que cada usuario esté al menos una vez en el conjunto de prueba y el promedio produce resultados más sólidos y estimaciones de error.

actual / predicción	negativo	Positivo
negativo	a	b
positivo	c	d

Tabla 2.1: Matriz de confusión de 2x2

Los elementos retenidos en los datos de prueba se eligen al ([azar. Breese et al. 1998, citado por Hahsler 2022](#)) introdujo los cuatro protocolos experimentales denominados *Dado 2*, *Dado 5*, *Dado 10* y *Todos menos 1*. Para los protocolos dados  $x$  para cada usuario, se entregan  $x$  elementos elegidos al azar al algoritmo de recomendación y los elementos restantes se retienen para su evaluación. Para Todos excepto  $x$ , el algoritmo obtiene todos los elementos retenidos excepto  $x$ .

A continuación, analizamos la evaluación de las calificaciones pronosticadas y luego de las listas de recomendaciones de los  $N$  principales.

### 2.3.1 Evaluación de las calificaciones pronosticadas

Una forma típica de evaluar una predicción es calcular la desviación de la predicción del valor real. Esta es la base para el *Error Promedio Medio (MAE)*

$$MAE = \frac{1}{|\mathcal{K}|} \sum_{(i,j) \in \mathcal{K}} |r_{jl} - \hat{r}_{jl}|, \quad (8)$$

Donde  $\mathcal{K}$  es el conjunto de todas las parejas usuario-elemento  $(j, l)$  para las que tenemos una calificación predicha  $\hat{r}_{jl}$  y una calificación conocida  $r_{jl}$  que no se usó para aprender el modelo de recomendación.

Otra medida popular es el *error cuadrático medio (RMSE)*

$$RMSE = \sqrt{\frac{\sum_{(j,l) \in \mathcal{K}} (r_{jl} - \hat{r}_{jl})^2}{|\mathcal{K}|}}$$

RMSE penaliza los errores más grandes con más fuerza que MAE y, por lo tanto, es adecuado para situaciones en las que los pequeños errores de predicción no son muy importantes.

### 2.3.2 Evaluación de las Top-N recomendaciones

La lista de los top- $N$  elementos pronosticados y los elementos retenidos que le gustan al usuario (generalmente determinados por un umbral simple en la calificación real) para todos los usuarios de la prueba  $\mathcal{U}_{test}$  se pueden agregar en una *matriz de confusión* que se muestra en la tabla 2 (ver [Kohavi y Provost \(1998\)](#)) que corresponde exactamente a los resultados de un experimento estadístico clásico. La matriz de confusión muestra cuántos de los elementos recomendados en las listas top- $N$  (columna predicha positiva;  $d + b$ ) fueron elementos retenidos y, por lo tanto, recomendaciones correctas (celda  $d$ ) y cuántos eran potencialmente incorrectos (celda  $b$ ). La matriz también muestra cuántos de los elementos no recomendados (columna predicha negativa;  $a + c$ ) deberían haberse recomendado en realidad, ya que representan elementos retenidos (celda  $c$ ).



De la matriz de confusión se pueden derivar varias medidas de desempeño. Para la tarea de minería de datos de un sistema de recomendación, el rendimiento de un algoritmo depende de su capacidad para aprender patrones significativos en el conjunto de datos. Las medidas de rendimiento utilizadas para evaluar estos algoritmos tienen su origen en el aprendizaje automático. Una medida comúnmente utilizada es la precisión, la fracción de recomendaciones correctas respecto al total de recomendaciones posibles.

$$Accuracy = \frac{\text{recomendaciones correctas}}{\text{total de posibles recomendaciones}} = \frac{a + d}{a + b + c + d} \quad (10)$$

Una medida de error común es el error absoluto medio (*MAE*, también llamado desviación absoluta media o *MAD*).

$$MAE = \frac{1}{N} \sum_{i=1}^N |\epsilon_i| = \frac{a + d}{a + b + c + d}, \quad (11)$$

Donde  $N = a + b + c + d$  es el número total de elementos que se pueden recomendar y  $|\epsilon_i|$  es el error absoluto de cada elemento. Dado que tratamos con datos 0 – 1,  $|\epsilon_i|$  solo puede ser cero (en las celdas  $a$  y  $d$  en la matriz de confusión) o uno (en las celdas  $b$  y  $c$ ). Para los algoritmos de recomendación de evaluación para calificar datos, a menudo se usa el error cuadrático medio. Para datos 0-1 se reduce a la raíz cuadrada de MAE.

Los sistemas de recomendación ayudan a encontrar elementos de interés del conjunto de todos los elementos disponibles. Esto puede verse como una tarea de recuperación conocida como recuperación de información. Por lo tanto, las medidas de rendimiento de recuperación de información estándar se utilizan con frecuencia para evaluar el rendimiento del recomendador. La precisión y el recuerdo son las medidas más conocidas utilizadas en la recuperación de información ([Salton y McGill 1983](#); [van Rijsbergen 1979](#), citado por [Hahsler 2022](#)).

$$Precision = \frac{\text{elementos correctamente recomendados}}{\text{total de elementos recomendados}} = \frac{d}{b + d} \quad (12)$$

$$Recall = \frac{\text{elementos correctamente recomendados}}{\text{total recomendaciones útiles}} = \frac{d}{c + d} \quad (13)$$

A menudo, se desconoce el número total de recomendaciones *útiles* necesarias para la el recall, ya que habría que inspeccionar toda la colección. Sin embargo, en lugar de las *recomendaciones útiles totales* reales, a menudo se utiliza el número total de recomendaciones útiles conocidas. La precisión y la recuperación son propiedades en conflicto, alta precisión significa poca recuperación y viceversa. Para encontrar un equilibrio óptimo entre la precisión y el recall, se puede utilizar una medida de valor único como la *medida-E* (van Rijsbergen 1979, citado por Hahsler 2022). El parámetro  $\alpha$  controla el trade-off entre la precisión y el recall.

$$Medida - E = \frac{1}{\alpha \left( \frac{1}{Precision} \right) + (1 - \alpha) \left( \frac{1}{Recall} \right)} \quad (14)$$

Una medida popular de un solo valor es la medida-F. Se define como la media armónica entre la precisión y el recall.

$$Medida - F = \frac{2 Precision \cdot Recall}{Precision + Recall} = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \quad (15)$$

Es un caso especial de la medida-E con  $\alpha = .5$  que otorga el mismo peso tanto a la precisión como al recall. En la literatura de evaluación de recomendadores, la medida-F a menudo se denomina medida F1.

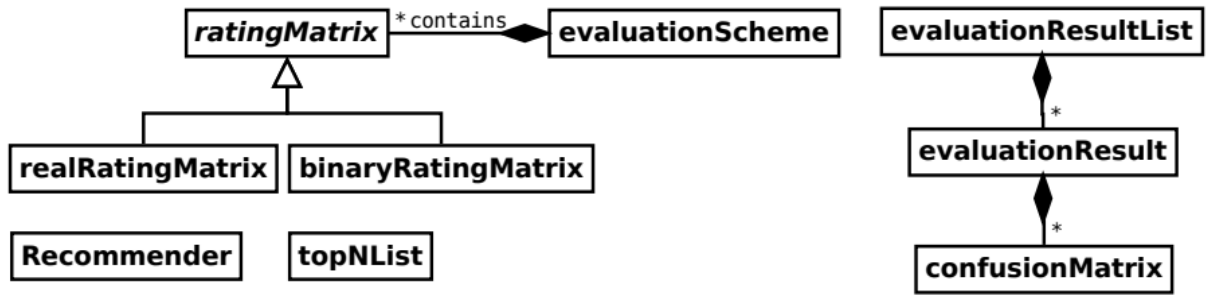


Figura 2.3: Diagrama de clases UML el paquete recommenderlab (Fowler 2004).

Otro método utilizado en la literatura para comparar dos clasificadores con diferentes configuraciones de parámetros es la *curva característica operativa del receptor (ROC)*. El método fue desarrollado para la detección de señales y se remonta al modelo Swets ([van Rijsbergen 1979](#), citado por [Hahsler 2022](#)). La curva ROC es un gráfico de la probabilidad de detección del sistema (también llamada sensibilidad o tasa de verdaderos positivos TPR, que es equivalente al recall como se define en la ecuación 13) por la probabilidad de falsa alarma (también llamada tasa de falsos positivos FPR o  $1 - \text{especificidad}$ , donde  $\text{especificidad} = \frac{a}{a+b}$  con respecto a los parámetros del modelo. Una forma posible de comparar la eficiencia de dos sistemas es comparando el tamaño del área bajo la curva ROC, donde un área más grande indica un mejor rendimiento.

## Capítulo 3

### 3. Extracción de los datos - Steam

En este capítulo se describirá la forma de extraer los datos de las reseñas de los videojuegos desde la API de Steam. Para esto se hace uso de las diferentes API's proporcionadas por la misma empresa, todas estas gratuitas y por lo general de libre acceso. Para acceder a otras API's es necesario crear una cuenta de Steam, e invertir 5.00 USD, ya que exigen el uso de la API key, tal y como se documenta en [Steam Web API Terms of Use](#). Paquetes como [SteamR](#), también permiten extraer la información, utilizando la API Key. Se debe tener en cuenta que todos los servicios de Steam están restringidos a un consumo de máximo 100.000 llamados por día o 200 cada 5 minutos lo que equivale a 1 llamado cada 1.5 segundos. Esto puede ser consultado en los [Términos y condiciones](#) de la API de Steam. Si se excede el número de llamadas la API arroja el **error 429**, esto supone una limitante grande a la hora de extraer la información.

En este TFM se utilizara la **api steampowered** documentada en [Steam Web API](#) y en [RJackson/StorefrontAPI](#). Se necesitan 3 pasos para extraer toda la información:

- **Paso 1.** Primero traer el id y nombre de cada elemento que se encuentra en Steam.
- **Paso 2.** Por cada id de cada elemento de Steam, se traer toda la información relevante y se filtra solo por videojuegos.
- **Paso 3.** Tiendo cada id de los juegos que sean pagos, se buscan todas las reseñas en español de todos los usuarios, para cada juego esto incluye si le gusto o no el video juego.

#### 3.1 Paso 1: Extracción elementos de Steam

El primer paso es utilizar la versión 2 del api (tipo GET) steampowered con url <http://api.steampowered.com/ISteamApps/GetAppList/v0002>, cuya salida es un json con dos columnas `appid` (identificador único del elemento de Steam) y `name` (nombre del videojuego) de cada elemento. El total se extrajeron 151.785 los elementos de steam esto incluye *game, application, tool, demo, dl* y *music*, e imprimimos los 10 primero objetos:

<i>appid</i>	<i>Name</i>
5	Dedicated Server
7	Steam Client
8	winui2
10	Counter-Strike
20	Team Fortress Classic
30	Day of Defeat
40	Deathmatch Classic
50	Half-Life: Opposing Force
60	Ricochet
70	Half-Life

Tabla 3.1: Encabezado listado de juegos extraído de la Api de Steam.

Del total de elementos se descarta aquellos que no tienen nombre y se utilizará el `appid`, para buscar la información de cada uno de los videjuegos.

## 3.2 Paso 2: Obtener la información del videojuego

Para extraer la información relevante de cada videojuego se accede a la siguiente api del tipo GET <http://store.steampowered.com/api/appdetails?appids=10&cc=us&l=en>. Note que en el anterior enlace se fijó el appids=10, que pertenece al juego *Counter-Strike*, pero para obtener la información de todos los videojuegos se deberá ir iterando. Esto a modo de ejemplo para ilustrar la información que traer el api, cabe agregar que en general en las apis disponibles en steam, hay dos parámetros adicionales:

- cc (Opcional) Código del país para devolver los valores de moneda apropiados.
- l (Opcional) Indicar el idioma, por defecto toma el nombre del idioma en inglés.

En este caso se ingresa cc=us y l=en, por lo que el precio de los juegos será en dólares americanos \$USD y el lenguaje en inglés, estos sirven por ejemplo para el género del juego, que indicaría Action, fantasy en vez de acción y fantasía.

La información que arroja este servicio se encuentra documentada en, el siguiente enlace [User:RJackson/StorefrontAPI - Official TF2 Wiki | Official Team Fortress Wiki](#) Desde este servicio se extrajo la información de un total de 76.612 videojuegos. En la siguiente tabla, se muestra la información más relevante.

type	name	Steam appid	Requir ed age	Is free	Supported languages	header image	develope rs	publishe rs	price	Metacritic score	categories	genres	recommenda tions	release date
game	The Apple Tree	2069800	0	FALSE	English, Russian	<a href="https://cdn.akamai.steamstatic.com/steam/apps/2069800/header.jpg?t=1658019257">https://cdn.akamai.steamstatic.com/steam/apps/2069800/header.jpg?t=1658019257</a>	Remnants of mosaic	Remnants of mosaic	NA	NA	Single player; Steam Achievements	Casual; Indie	NA	NA
game	Waitventure	2055180	0	FALSE	English, Spanish - Spain, Simplified Chinese	<a href="https://cdn.akamai.steamstatic.com/steam/apps/2055180/header.jpg?t=1660563809">https://cdn.akamai.steamstatic.com/steam/apps/2055180/header.jpg?t=1660563809</a>	Kulivszk	Kulivszk	NA	NA	Single player	Casual; Indie; RPG	NA	NA

Tabla 3.2: Información de los juegos traído de la Api de Steam.

**Nota:** como se indicó antes el api presenta una restricción de 200 llamadas cada 5 minutos y un máximo de 100.000 llamadas diarias. Esto equivale a realizar una llamada cada 1.5 segundos. Es decir que para traer la información de los 151.785 fue necesario 2d 15H 14M 38S

### 3.1.3 Paso 3: Extraer información de los usuarios

Para obtener la información de la reseña de cada uno de los juegos, se toma como base el total de los 76.612 juegos, se filtra por los que son exclusivamente pagos, ya que estos generan valor al negocio y filtramos por aquellos juegos que sean de la última década, es decir que la fecha de lanzamiento sea mayor o igual al primero de enero del 2012. Aquellos juegos sin fecha de lanzamiento o que no hayan sido reseñados se descartan, quedándonos con una base 11.362 juegos.

La api utilizada se encuentran documentados en [User Reviews - Get List](#). Este también arroja una salida formato JSON de las reseñas, para el análisis se traerán solo las reseñas que se encuentren en el idioma español, y los parámetros de salida de la api son los siguientes:

- **success:** - 1 si la consulta fue exitosa
- **query\_summary:** - Devuelto en la primera solicitud
  - **num\_reviews:** el número de revisiones devueltas en esta respuesta
  - **review\_score:** - La puntuación de la revisión
  - **review\_score\_desc:** - La descripción de la puntuación de la revisión
  - **total\_positive:** - Número total de críticas positivas
  - **total\_negative:** - Número total de críticas negativas
  - **total\_reviews:** número total de revisiones que coinciden con los parámetros de consulta.
- **cursor:** el valor para pasar a la siguiente solicitud como el cursor para recuperar el siguiente lote de revisiones
- **reseñas**
  - **recommendationid:** el ID único de la recomendación.
  - **autor**
    - **steamid:** - el SteamID del usuario
    - **num\_games\_owned:** - número de juegos propiedad del usuario
    - **num\_reviews:** - número de reseñas escritas por el usuario
    - **playtime\_forever:** - tiempo de reproducción de por vida rastreado en esta aplicación
    - **playtime\_last\_two\_weeks:** tiempo de reproducción registrado en las últimas dos semanas para esta aplicación
    - **playtime\_at\_review:** - tiempo de juego cuando se escribió la reseña
    - **last\_played:** - hora de la última vez que el usuario jugó
  - **language:** idioma que el usuario indicó al escribir la reseña
  - **review:** - texto de revisión escrita
  - **timestamp\_created:** - fecha en que se creó la reseña (marca de tiempo de Unix)
  - **timestamp\_updated:** - fecha en la que se actualizó la reseña por última vez (marca de tiempo de Unix)
  - **voted\_up:** - verdadero significa que fue una recomendación positiva
  - **votes\_up:** - el número de usuarios que encontraron útil esta reseña
  - **votes\_funny:** - el número de usuarios a los que les pareció graciosa esta reseña
  - **weighted\_vote\_score:** - puntuación de ayuda
  - **comment\_count:** - número de comentarios publicados en esta revisión
  - **steam\_purchase:** verdadero si el usuario compró el juego en Steam
  - **receive\_for\_free:** verdadero si el usuario marcó una casilla que dice que obtuvo la aplicación de forma gratuita
  - **written\_during\_early\_access:** verdadero si el usuario publicó esta reseña mientras el juego estaba en acceso anticipado
  - **developer\_response:** texto de la respuesta del desarrollador, si corresponde
  - **timestamp\_dev\_responded:** marca de tiempo de Unix de cuándo respondió el desarrollador, si corresponde

Como no toda la información es de interés, a continuación, se muestra los campos que se trajeron

<i>Recommendation id</i>	<i>Author steamid</i>	<i>Author num games owned</i>	<i>Author number views</i>	<i>Author play time forever</i>	<i>Author playtime last two weeks</i>	<i>Author playtime at review</i>	<i>Author last played</i>	<i>language</i>	<i>Timestamp created</i>	<i>Timestamp updated</i>
121017058	76561198833206080	53	1	209	209	209	2022-08-21 20:26:33	English	2022-08-21 20:28:33	2022-08-21 20:28:33
121011817	76561198805003406	84	6	55	55	55	2022-08-21 18:41:39	Ukrainian	2022-08-21 18:43:22	2022-08-21 18:43:22
121010484	76561198050493025	573	23	1223	839	1223	2022-08-21 18:00:06	English	2022-08-21 18:14:58	2022-08-21 18:14:58

<i>Voted up</i>	<i>Vote s up</i>	<i>Votes funny</i>	<i>Weighted vote score</i>	<i>Comment count</i>	<i>Steam purchase</i>	<i>Received for free</i>	<i>Written during early access</i>	<i>appid</i>	<i>Timestamp dev responded</i>	<i>Developer response</i>
FALSE	1	0	0.52471	0	TRUE	FALSE	FALSE	22340	NA	NA
FALSE	0	0	0	0	TRUE	FALSE	FALSE	22340	NA	NA
FALSE	1	0	0.52471	0	TRUE					

Tabla 3.3: Información de los usuarios y calificación de cada uno de los videojuegos.

## 4. Metodología del proyecto

Para el desarrollo de este trabajo se utilizó la metodología CRISP-DM (Cross Industry Standard Process for Data Mining), en una de las propuestas metodológicas más trabajadas alrededor del mundo para resolver los problemas de minería de datos y analítica ([Martínez-Plumed et al., 2019](#)). En el desarrollo de este trabajo se implementó la metodología iterativa CRISP-DM, la cual adapta elementos similares al de un proyecto de ingeniería de software para enriquecer el ciclo de vida de un proyecto de análisis de datos. El modelo proporciona 6 fases: comprensión del negocio, comprensión de los datos, análisis de los datos, modelamiento de la información, evaluación y despliegue; todas estas fases giran alrededor de la información y determinan las actividades que se deben desarrollar en un determinado espacio de tiempo de manera reiterada, asignando tareas concretas y definiendo los objetivos de lo que se desea lograr en cada una de ellas.

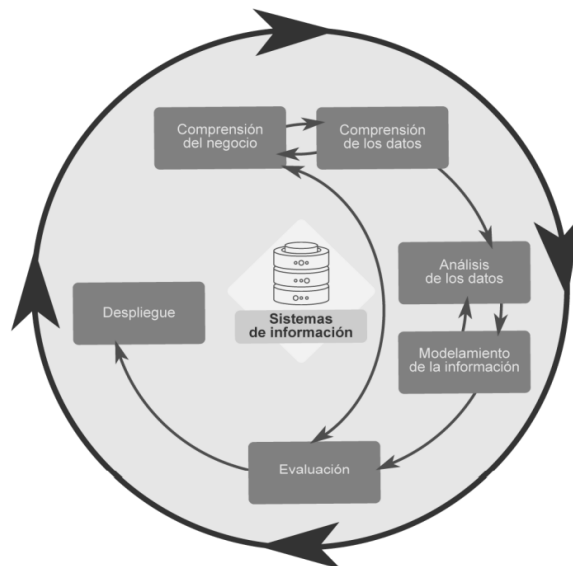


Figura 4.1 Metodología CRISP-DM.

Gracias al papel itinerante de la metodología CRISP-DM, el proceso de refinamiento permite volver a alguna fase, gracias al análisis de las necesidades en cada una de estas, permitiendo al equipo refrescar el objetivo general a través de conclusiones y oportunidades de mejora sobre los procesos y actividades desarrolladas con el fin de implementar mejoras de cara a las necesidades del negocio.

Durante el desarrollo de este TFM fue necesario la aplicación de 3 iteraciones de la metodología CRISP-DM, esto debido a los hallazgos encontrados principalmente en la parte de exploración y modelamiento de los datos, a continuación, se dará una breve explicación de cada una de las fases de la metodología CRISP-DM y utilizadas para la implementación de este sistema de recomendación.

### 4.1 Comprensión del Negocio

El objetivo principal de esta fase es definir y alinear los objetivos del proyecto de análisis de información con las necesidades del negocio, buscando que los datos sean una manera de producir un resultado que permita a la organización obtener un beneficio con el desarrollo del proyecto. Los entregables de esta fase son:

- Establecer los objetivos de negocio

- Evaluar la situación actual
- Obtener un plan de proyecto

La necesidad del negocio consiste en que las pequeñas tiendas de videojuegos online en LATAM y España, por lo general no cuentan con sistemas de recomendación para sus productos, lo que hace que pierdan una oportunidad de venta cruzada a los usuarios además de otorgar una mejor experiencia al usuario. Por esta razón se propone crear una empresa especializada en productos de analítica cuyo primer producto sea un sistema de recomendación, que cree ofertas especializadas para un grupo de usuarios de videojuegos. Esto permitirá que las tiendas mejoren sus ventas al crear campañas dirigidas o generar venta cruzada.

Para desarrollar este modelo de negocios, lo primordial es obtener la información de las reseñas que dejan los usuarios para realizar un análisis de datos que permita generar las recomendaciones.

El panorama actual muestra un crecimiento en la industria de los videojuegos solo en Colombia se mueven \$419 millones de dólares al año en ventas. Por esta razón este proyecto se centrará inicialmente en los usuarios de España y Latinoamérica como parte importante del mercado de videojuegos. Hay un total de 2.700 millones de usuarios de videojuegos en el mundo y esta cifra seguirá creciendo y en Latinoamérica hay 289 millones lo que representa más del 10%.

En el año 2020, fue un período clave en el crecimiento de esta industria a escala global, y hubo un país líder en el mercado latinoamericano: México. Según el portal Statista, la industria de los videojuegos en México generó una facturación de 1.901 millones de dólares estadounidenses. En segundo lugar, destaca Brasil, con ingresos que superan los 1.750 millones de dólares. En tercer lugar, en este “top” figura Argentina, con 507 millones.

Dado este rápido crecimiento de la industria es de vital importancia cubrir esta necesidad que tienen las tiendas de video juegos y el porque de este proyecto el centrarse en esta población.

## 4.2 Descripción de los datos

La necesidad de este proyecto nace de utilizar la información de las reseñas de los usuarios en el idioma español, la información general de los juegos y de los usuarios, por lo tanto, se tienen el siguiente modelo de bases de datos.

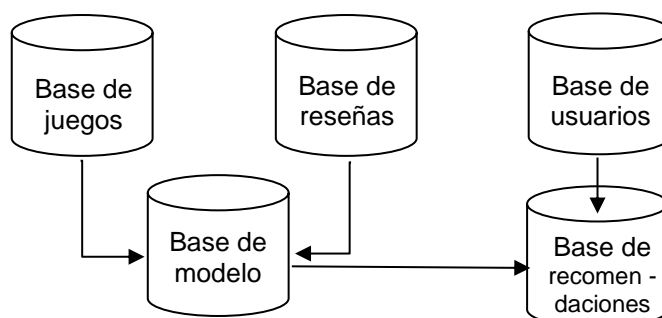


Figura 4.2 Modelo entidad – relación.

**Base de viejo juegos:** Para la base de videojuegos se tiene un total de 3.691 juegos únicos, de la última década (mayor o igual al año 2012), y que sean pagos.

Appid	name	header_image	developers	releasedate	genres
238870	Citadels	<a href="https://cdn.akamai.steamstatic.com/steam/apps/238870/header.jpg?t=1592555951">https://cdn.akamai.steamstatic.com/steam/apps/238870/header.jpg?t=1592555951</a>	Games Distillery s.r.o.	2013-07-25	Action; Strategy



Appid	name	header_image	developers	releasedate	genres
262190	Zombeers	<a href="https://cdn.akamai.steamstatic.com/steam/apps/262190/header.jpg?t=1582878998">https://cdn.akamai.steamstatic.com/steam/apps/262190/header.jpg?t=1582878998</a>	Moonbite Games; PadaOne Games	2015-01-30	Action; Indie
289360	Konung 2	<a href="https://cdn.akamai.steamstatic.com/steam/apps/289360/header.jpg?t=1656591573">https://cdn.akamai.steamstatic.com/steam/apps/289360/header.jpg?t=1656591573</a>	Fulqrum Publishing	2014-04-23	RPG
290490	The Flock	<a href="https://cdn.akamai.steamstatic.com/steam/apps/290490/header.jpg?t=1467753758">https://cdn.akamai.steamstatic.com/steam/apps/290490/header.jpg?t=1467753758</a>	Vogelsap	2015-08-21	Action; Indie
217100	Dementium II HD	<a href="https://cdn.akamai.steamstatic.com/steam/apps/217100/header.jpg?t=1474895574">https://cdn.akamai.steamstatic.com/steam/apps/217100/header.jpg?t=1474895574</a>	Memetic Games	2013-12-17	Action; Adventure; Indie

Tabla 4.1: Base de los videojuegos pagos de la última década.

**Base de las reseñas:** Base de las reseñas en español que han dejado los usuarios sobre los videojuegos, aunque se tiene mucha más información traída de la API, para la metodología que se va a aplicar de filtros colaborativos solo es necesario saber si el usuario votó positiva o negativamente cada uno de los juegos. Las columnas que tiene esta base son 3 appid (id del videojuego), el steamid (el id del usuario) y voted\_up (con TRUE y FALSE). Se tiene un total de 1.933.168 reseñas.

appid	author.steamid	voted_up
1313	76561197960914766	TRUE
1313	76561197962062015	TRUE
1313	76561197962462916	FALSE
1313	76561197962750090	TRUE
1313	76561197963002630	TRUE
1313	76561197963885198	TRUE

Tabla 4.2: Base de las reseñas.

**Base de modelo:** se fusionan la base de los juegos y de las reseñas de los usuarios, para obtener la base del modelado, la cual es una matriz dispersa y se descarta aquellos usuarios que tengan 10 reseñas o menos. La base tiene la siguiente dimensión 14058 (usuarios) x 3691 (juegos) y 259499 reseñas, siendo un objeto de R del tipo *'binaryRatingMatrix'*.

**Base de los usuarios:** Contiene toda la información de los 14.058 usuarios, esto incluye el id, si el perfil es público o privado, el nombre de usuario, nombre real, fecha de creación de la cuenta, país, número de reseñas, tiempo jugado, dinero gastado y lenguaje.

steamid	Community state	Persona name	Real name	Time created	Country code	Number of reviews	Playtime forever	price	language
76561197961967341	public	Chema ESP	Jose Maria	2003-10-11 21:32:37	España	16	115266	369	spanish

76561197968390105	public	Dimuscu1	NA	2004-08-20 12:12:10	España	77	15521	219	spanish
76561197968273060	public	Dani	Daniel García	2004-08-17 22:53:09	España	82	86828	351	spanish
76561197969817202	public	Japo32	NA	2004-10-24 15:35:29	España	18	199248	4029	spanish
76561197969523079	public	Mayc	Miguel Ángel	2004-10-12 10:32:48	España	19	21151	309	spanish
76561197968497755	public	Y4MPiR3   <sup>es</sup>	Ληγαστη ριευΞα	2004-08-24 21:53:31	Turquía	59	38907	504	spanish

Tabla 4.3: Información de los usuarios.

**Base de las predicciones:** Contiene el total de las recomendaciones dadas por el modelo a cada uno de los usuarios, como 14.058 usuarios y se dan 10 recomendaciones por cada usuario, por lo que se tiene una base de 140.580 recomendaciones, a nivel de id de usuario.

appid	author.steamid	name	genres
201810	76561197960291325	Wolfenstein: The New Order	Action
367520	76561197960291325	Hollow Knight	Action; Adventure; Indie
391220	76561197960291325	Rise of the Tomb Raider™	Action; Adventure
252410	76561197960291325	SteamWorld Dig	Action; Adventure; Indie
413150	76561197960291325	Stardew Valley	Indie; RPG; Simulation
239140	76561197960291325	Dying Light	Action; RPG

Tabla 4.4. Información de las predicciones.

## 4.3 Análisis de los datos

Como se vio en el apartado anterior, en la base de las reseñas se tiene un total 1.933.168 de votos. Como se observa en la figura 4,3 el 93% de las calificaciones del listado de juegos son positivas, mientras que el 7% son negativas es decir no recomiendan el videojuego.

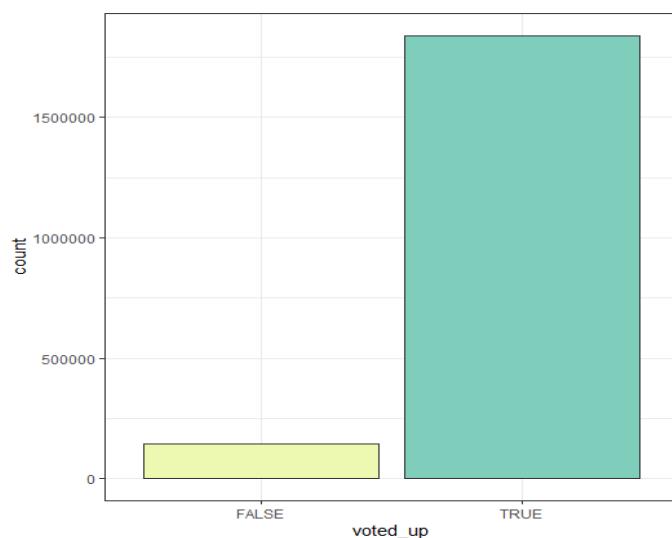


Figura 4.3: Número de votos positivos y negativos de las reseñas.

En la figura 5, se realizan dos histogramas de ratings por usuario a la izquierda, indica el número de calificaciones que ha hecho cada usuario, como vemos el grueso de los usuarios se concentra en 5 calificaciones o menos. Para encontrar recomendaciones con sentido será necesario quedarnos

con los usuarios que hayan realizado 10 calificaciones o más. Por otra parte, el histograma de la derecha muestra el numero de reseñas por películas, como hay videojuegos con una gran cantidad de calificaciones como es el caso de Fall Guys con 49.264, se hace difícil visualizar para este grafico se filtrará los juegos con 250 reseñas o menos. Y para el modelado se utilizarán películas con al menos 25 reseñas.

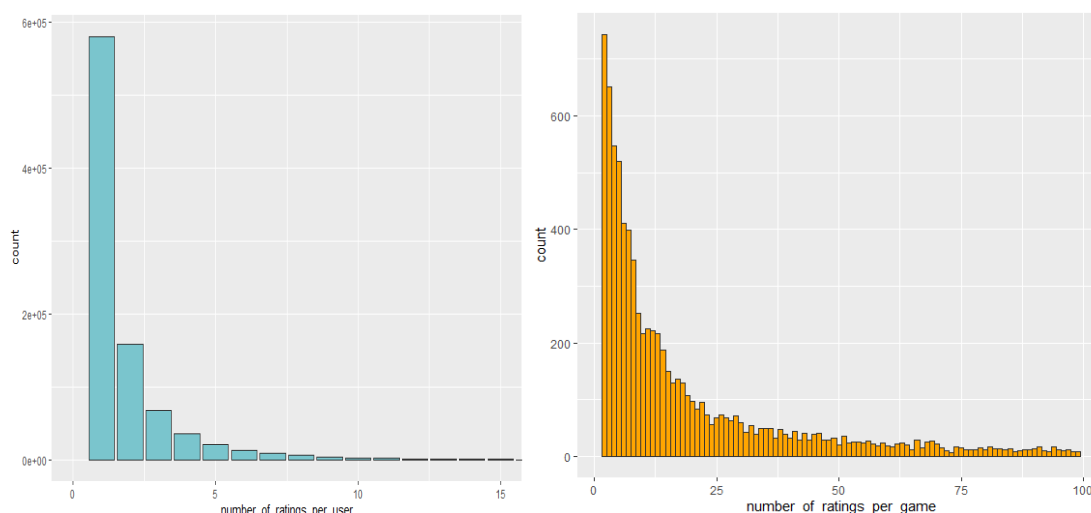


Figura 4.4: Número de ratings por usuario y por juego.

En la Figura 4.5 se visualiza la proporción de juegos por genero, donde el 26.2% son de acción, seguido de indi con un 18% y aventura con el 15.4%, completan en top 3. Mientras que los menos frecuentes son Photo Editing, Animation & Modeling y Design & Illustration con 5.1%, 5% y 4.8% respectivamente.

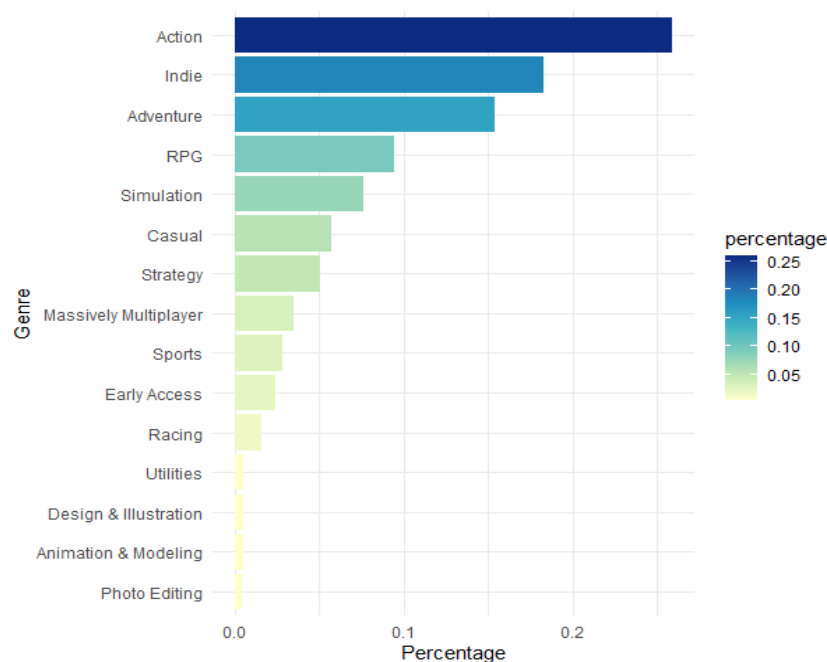


Figura 4.5: Porcentaje de juegos por género.

**Nota:** En la anterior grafica se excluyeron algunos géneros con una representación casi nula, estos son los juegos con genero accounting, nudity, sexual content, game development, education, Software training, Gore, Web publishing, video production, violent y audio production.

En la figura 4.6 se puede visualizar el top 5 de los videojuegos con mayor cantidad de reseñas, encabezado por Fall Guys con 49k reseñas, seguido de Grand Theft Auto V con 47k y el top 3 lo completa Among Us con 32k reseñas.






	image	name	genres	release_date	recommendations	rating	price
1		Fall Guys	Action;Casual;Indie;Massively Multiplayer;Sports	2020-08-03	49264	0.915	
2		Grand Theft Auto V	Action;Adventure	2015-04-13	47947	0.911	
3		Among Us	Casual	2018-11-16	32191	0.935	4.99
4		Dead by Daylight	Action	2016-06-14	29964	0.887	19.99
5		ARK: Survival Evolved	Action;Adventure;Indie;Massively Multiplayer;RPG	2017-08-27	28999	0.889	19.99

Figura 4.6: Top 5 de los videojuegos con mayor cantidad de reseñas.

Por otra parte, los juegos con mejor rating fueron Blades of Time del género acción y aventura, de 34 reseñas en español todas fueron positivas, seguido de Sherlock Holmes and The Hound of The Barskervilles con 27 reseñas, positivas y RPG Maker XP con 26 positivas, el top 5 que se muestra en la figura 4.6 tuvieron un rating de reseñas positivas del 100%.




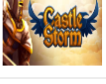

	image	name	genres	release_date	recommendations	rating	price
1		Blades of Time	Action;Adventure	2012-04-20	34	1	9.99
2		Sherlock Holmes and The Hound of The Baskervilles	Adventure;Casual	2012-04-23	27	1	9.99
3		RPG Maker XP	Web Publishing	2013-12-13	26	1	24.99
4		CastleStorm	Action;Indie;Strategy	2013-07-29	36	1	9.99
5		140	Action;Indie	2013-10-16	50	1	4.99

Figura 4.6: Top 5 de los videojuegos con mejor rating.

Los juegos con peor rating fueron: en primer lugar, Devil's Hunts con solo un 13.8% de votos positivos, seguido de Towns con un 18.2% y Edge of Space con un 19.2%

## 4.4 Modelamiento de la información

Para el ajustar el modelo se utilizó el paquete de R **recommenderlab**, cuya matriz binaria

14045 (usuarios) x 3691 (películas) rating matrix of class 'binaryRatingMatrix' with 259499 ratings. Se probaron 7 algoritmos de recomendación diferentes. El proceso de modelado para los diferentes algoritmos, incluyo como punto de partida el aleatorio para comparar las recomendaciones generadas por los demás algoritmos. El popular para recomendar a los usuarios los juegos más populares que no han sido calificados por los usuarios. El altering las square, que busca minimizar una función de costo utilizando factorización de matrices. Las Reglas de Asociación que busca hechos que ocurren en común dentro del conjunto dato. Para filtro colaborativo basado en usuarios y filtro colaborativo basado en elementos, se utilizó el índice de Jaccard de la ecuación 7 como medida de similitud. No fue posible aplicar el algoritmo de descomposición en valores singulares, pues esta metodología no está implementada para calificaciones binarias. A continuación, se enumeran los algoritmos utilizados

- RANDOM: Se ajusto con los parámetros por defecto.
- POPULAR: Se ajusto con los parámetros por defecto.
- ALS: Que utiliza factorización de matrices con los parámetros por defecto.
- AR: Reglas de asociación con soporte del 0.001, confianza del 0.35 y longitud máxima de las reglas de 2.
- UBCF: Filtros colaborativos basado en usuarios con la distancia de Jaccard y  $k = 30$ , para el algoritmo de vecinos más cercanos.
- IBCF: Filtros colaborativos basado en elementos con la distancia de Jaccard y  $k = 25$ , para el algoritmo de vecinos más cercanos.
- HYBRID: El algoritmo hibrido como mezcla de las reglas de asociación con una ponderación del 75% y filtros colaborativos basado en elementos con un peso del 25%, ambos con los parámetros mencionados anteriormente.

Además, para el esquema de entrenamiento, se utilizó la función `evaluationScheme` del paquete `recommenderlab`, con el parámetro `given` 10, que restringe a que los usuarios hayan calificado mínimo 10 video juegos, se particiono el conjunto de datos en dos partes, entrenamiento un 80% y test 20%, además de aplicar validación cruzada de 5 capas para evitar el sobreajuste del modelo. En el modelado se fijó una semilla para que el ejercicio sea replicable y evaluando diferentes valores de  $n$ . Este proceso demoro exactamente 7 horas 33 minutos y 14 segundos.

## 4.3 Evaluación

En la Tabla 4.5, se muestra el resultado obtenido después de aplicar cada uno de los algoritmos, donde el  $n$  indica el número de recomendaciones que se darían, es decir 3, 5, 7, 10 o 15 videojuegos. como se esperaba el aleatorio es el de peor rendimiento en cuanto a precisión y recall, con valores del 0.2% y 0.07%. Mientras que reglas de asociación es el de mejor performance con hasta un 18% de precisión (para 3 juegos) y hasta un 7% de recall si se recomendaran 15 juegos. En cuanto a los algoritmos de recomendación por filtros colaborativos de usuarios y elementos, el rendimiento es mejor para el método del filtrado por elementos, con el mayor recall del 15.8% si se recomendaran 15 juegos. Finalmente, el hibrido una meza del IBCF y el AR tiene las métricas más niveladas entre precisión y recall.

	TP	FP	FN	TN	N	PRECISION	RECALL	TPR	FPR	N
RANDOM	0.0060412	2.993959	8.449325	3669.551	3681	0.0020137	0.0007807	0.0007807	0.0008152	3

0.0102345	4.989765	8.445132	3667.555	3681	0.0020469	0.0012477	0.0012477	0.0013587	5
0.0159204	6.984080	8.439446	3665.561	3681	0.0022743	0.0019942	0.0019942	0.0019017	7
0.0223881	9.977612	8.432978	3662.567	3681	0.0022388	0.0028225	0.0028225	0.0027168	10
0.0339730	14.966027	8.421393	3657.579	3681	0.0022649	0.0042684	0.0042684	0.0040751	15
<b>POPULAR</b>									
0.2212509	2.778749	8.234115	3669.766	3681	0.0737503	0.0355614	0.0355614	0.0007565	3
0.3379531	4.662047	8.117413	3667.883	3681	0.0675906	0.0536785	0.0536785	0.0012692	5
0.4410803	6.558920	8.014286	3665.986	3681	0.0630115	0.0697177	0.0697177	0.0017857	7
0.5881308	9.411869	7.867235	3663.133	3681	0.0588131	0.0924902	0.0924902	0.0025624	10
0.8218195	14.178180	7.633547	3658.366	3681	0.0547880	0.1271445	0.1271445	0.0038601	15
<b>ALS</b>									
0.3927505	2.607249	8.062615	3669.937	3681	0.1309168	0.0651118	0.0651118	0.0007097	3
0.5927505	4.407249	7.862616	3668.137	3681	0.1185501	0.0955206	0.0955206	0.0011997	5
0.7622601	6.237740	7.693106	3666.307	3681	0.1088943	0.1212204	0.1212204	0.0016980	7
0.9841507	9.015849	7.471215	3663.529	3681	0.0984151	0.1545182	0.1545182	0.0024543	10
1.2977967	13.702203	7.157569	3658.842	3681	0.0865198	0.1991590	0.1991590	0.0037302	15
<b>AR</b>									
0.3273632	1.411585	8.128003	3671.133	3681	<b>0.1833382</b>	0.0523784	0.0523784	0.0003843	3
0.3743426	1.651883	8.081024	3670.893	3681	0.1790511	0.0576685	0.0576685	0.0004497	5
0.3887704	1.729567	8.066596	3670.815	3681	0.1778799	0.0590435	0.0590435	0.0004709	7
0.3987918	1.772566	8.056574	3670.772	3681	0.1775978	0.0598789	0.0598789	0.0004826	10
0.4049751	1.803909	8.050391	3670.741	3681	0.1774924	0.0603385	0.0603385	0.0004911	15
<b>UBCF</b>									
0.0482587	2.951741	8.407107	3669.593	3681	0.0160862	0.0057518	0.0057518	0.0008037	3
0.0820896	4.917910	8.373276	3667.627	3681	0.0164179	0.0100028	0.0100028	0.0013390	5
0.1196162	6.880384	8.335750	3665.664	3681	0.0170880	0.0150997	0.0150997	0.0018734	7
0.1766880	9.823312	8.278678	3662.721	3681	0.0176688	0.0220515	0.0220515	0.0026747	10
0.2764037	14.723596	8.178962	3657.821	3681	0.0184269	0.0360639	0.0360639	0.0040089	15
<b>IBCF</b>									
0.3038380	2.696162	8.151528	3669.848	3681	0.1012793	0.0475632	0.0475632	0.0007340	3
0.4611230	4.538877	7.994243	3668.006	3681	0.0922246	0.0725236	0.0725236	0.0012356	5
0.5953802	6.404620	7.859986	3666.140	3681	0.0850543	0.0928030	0.0928030	0.0017436	7
0.7772566	9.222743	7.678109	3663.322	3681	0.0777257	0.1208777	0.1208777	0.0025108	10
1.0253731	13.974627	7.429993	3658.570	3681	0.0683582	<b>0.1578295</b>	0.1578295	0.0038045	15
<b>HYBRID</b>									
0.4117271	2.588273	8.043639	3669.956	3681	0.1372424	0.0664518	0.0664518	0.0007045	3
0.5670931	4.432907	7.888273	3668.112	3681	0.1134186	0.0899248	0.0899248	0.0012067	5
0.6972992	6.302701	7.758067	3666.242	3681	0.0996142	0.1092622	0.1092622	0.0017158	7
0.8686567	9.131343	7.586709	3663.413	3681	0.0868657	0.1347919	0.1347919	0.0024859	10
1.1004264	13.899574	7.354940	3658.645	3681	0.0733618	0.1687342	0.1687342	0.0037840	15

Tabla 4.5: Resultado al aplicar los diferentes algoritmos de recomendación.

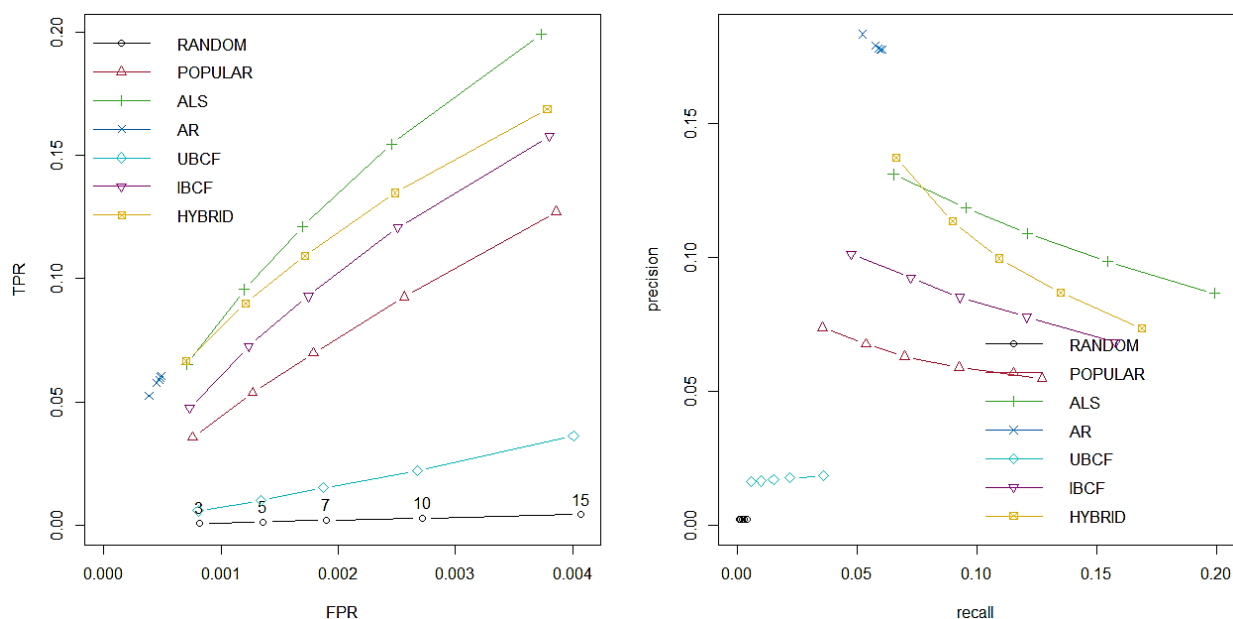


Figura 4.7: Curva ROC (izquierda) y precisi3n vs recall (derecha).

En la figura 4.7 se visualiza la curva ROC para los 7 algoritmos siendo el de mejor comportamiento el ALS llegando a un AUC del 20%, el modelo hibrido en amarillo tiene un comportamiento cercano. Algo similar ocurre en la grafica de la derecha de precisi3n vs recall, solo que en un punto el modelo hibrido parece funcionar mejor.

Se utilizar3 como modelo definitivo el modelo de recomendaci3n hibrido. Este sistema utiliza un concepto similar al stacked de machine learning, donde se pueden fusionar varios modelos d3biles para obtener uno m3s robusto.

A modo de ejemplo se muestra la predicci3n de ejemplo se utilizada en el usuario con id 76561197960284889, que ha calificado 17 todos positivamente, para este usuario se recomiendan los siguientes 5 juego utilizando el modelo hibrido.

	author:steamid	appid	name	voted_up	release_date
14	76561199136513271	1817070	Marvel's Spider-Man Remastered	true	2022-08-12
12	76561199136513271	1506830	FIFA 22	true	2021-09-30
13	76561199136513271	1544360	LEGO® Builder's Journey	true	2021-06-22
10	76561199136513271	1196590	Resident Evil Village	true	2021-05-06
11	76561199136513271	1404210	Red Dead Online	true	2020-12-01
8	76561199136513271	418370	Resident Evil 7 Biohazard	true	2017-01-23
9	76561199136513271	498240	Batman - The Telltale Series	true	2016-08-02
7	76561199136513271	359550	Tom Clancy's Rainbow Six® Siege - A Ubisoft Original	true	2015-12-01
5	76561199136513271	292030	The Witcher® 3: Wild Hunt	true	2015-05-18
4	76561199136513271	271590	Grand Theft Auto V	true	2015-04-13
6	76561199136513271	307780	Mortal Kombat X	true	2015-04-13
2	76561199136513271	239140	Dying Light	true	2015-01-26
1	76561199136513271	209000	Batman™: Arkham Origins	true	2013-10-24
3	76561199136513271	249130	LEGO® Marvel™ Super Heroes	true	2013-10-22

Tabla 4.6. Calificaciones dadas por el usuario con id 76561199136513271 a los video juegos. Para este usuario las recomendaciones son:











image	name	genres	developers	release_date	recommendations	rating
All	All	All	All	All	All	All
1	 Wallpaper Engine	Casual; Indie; Animation & Modeling; Design & Illustration; Photo Editing; Utilities	Wallpaper Engine Team	2016-11-20	23752	0.9918
2	 Resident Evil 2	Action	CAPCOM Co., Ltd.	2019-01-24	6210	0.9879
3	 The Forest	Action; Adventure; Indie; Simulation	Endnight Games Ltd	2018-04-30	20018	0.975
4	 Batman: Arkham City - Game of the Year Edition	Action; Adventure	Rocksteady Studios	2012-09-07	2787	0.9727
5	 LEGO® Batman™ 3: Beyond Gotham	Action; Adventure	TT Games Ltd	2014-11-11	271	0.9483
6	 LEGO® Batman™ 2: DC Super Heroes	Action; Adventure	TT Games	2012-06-22	289	0.9446
7	 Batman: The Enemy Within - The Telltale Series	Adventure	Telltale	2017-08-08	291	0.9416
8	 Batman™: Arkham Knight	Action; Adventure	Rocksteady Studios	2015-06-23	3802	0.94
9	 LEGO® Marvel Super Heroes 2	Action	TT Games; Feral Interactive (Mac)	2017-11-14	388	0.9201
10	 Resident Evil 3	Action	CAPCOM Co., Ltd.	2020-04-02	2581	0.8861

Figura 4.8: Top 5 recomendaciones para el usuario, con id 76561197960284889.

En la Figura 4.8 se observan las recomendaciones las cuales son totalmente coherentes con respecto a los gustos del usuario, por ejemplo, jugo dos Resident Evil el Village y el 7 biohazard, el sistema recomienda el Resident Evil 2 y 3 del desarrollador Capcom, ambos de la misma saga de video juegos de terror. Por otra parte, también jugo Batman – The Telltale Series y Batman Arkham Origins, el sistema recomienda al usuario Batman The enemy within. Este usuario Tambien jugo LEGO Builders Journey y LEGO Marvel Super Heroes, recomienda LEGO Marvel Super Heroes 2, como secuela de la parte I, y LEGO Batman 2, LEGO Batman 3. lo cual es totalmente coherente.

Así e pueden encontrar muchas más recomendaciones, solo es cuestión de interactuar con las recomendaciones dadas por el modelo.

## 4.4 Despliegue

Para desplegar el modelo de recomendación y ser utilizado en tiempo real se puede desplegar como una Api Restful y utilizar diferentes servicios en la nube, como Google Cloud Platform, Amazon Web Service o Azure. Por facilidad se utilizará r Shiny, ya que su interfaz es amigable y permite al usuario interactuar con los diferentes componentes. Se puede desplegar gratuitamente en [shinyapps.io](https://shinyapps.io), por lo que se hace en este servicio en el siguiente enlace

[https://rafael-eduardo-diaz.shinyapps.io/recommender\\_systems/](https://rafael-eduardo-diaz.shinyapps.io/recommender_systems/)



## 5. Viabilidad Financiera

La viabilidad Financiera del Proyecto tiene como objetivo determinar la rentabilidad del proyecto gracias al análisis de una inversión inicial, unos beneficios y unos costos de la ejecución del mismo. De modo que, al evaluar el proyecto, estaremos poniendo en duda su viabilidad. El análisis de viabilidad tiene como objetivo

- Realizar un estudio económico y financiero para determinar la factibilidad y rentabilidad de la inversión.
- Realizar la evaluación económica y financiera del proyecto a fin de estimar un plazo para el retorno de la inversión necesaria.
- Realizar la estimación de indicadores financieros que nos permitan prever costo y retorno de la inversión.

### 5.1 Modelo de Negocio

El modelo de negocio de se basa en crear una empresa con sede en principal en Colombia especializada en crear soluciones de analítica, esto incluye el sistema de recomendación, análisis de datos, modelos de machine learning, consultorías y desarrollo de páginas web. El primer proyecto será el sistema de recomendación de video juegos, y a medida que crezca la empresa se elaboraran diferentes productos de analítica.

Esta empresa con nombre AnalitycSify iniciara operaciones en el II semestre del 2023, con sede principal en Bogotá, Colombia. Ubicada en el sector de Santa Bárbara Occidental, en el Centro de Negocios [Bluework](#) en la calle 118 #19A-33. La oficina cuenta con un espacio de 19 mts. Incluye: 3 salas de espera, parqueaderos, auditorio de 60 m2 con video y audio surround, cocina, aseo, seguridad y baños. Acceso fácil al transporte público, restaurante, bancos y comercio.

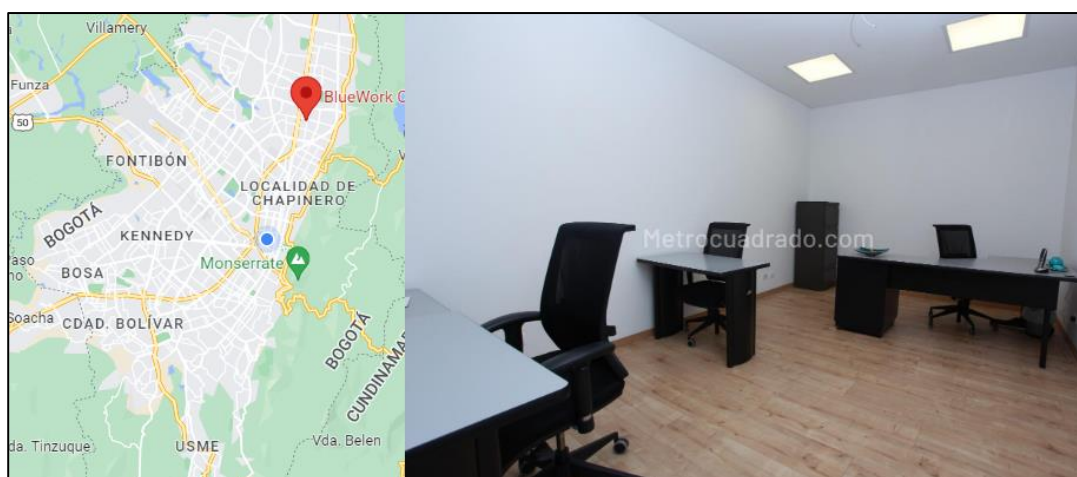


Figura 5.1 Bluework, coworking ubicado en el norte de la ciudad de Bogotá, Colombia.

A continuación, se busca las empresas similares a Steam, que venden videojuegos. A continuación, se muestran 8 de las empresas más grandes en el mundo especializada en venta de videojuegos:

Nombre	Logo	Información adicional	Descripción	SR
Steam		Página web: <a href="https://store.steampowered.com/">https://store.steampowered.com/</a> Empresa: Valve Lanzamiento: 2003	Steam es una plataforma de distribución digital de videojuegos ofrece protección contra piratería, 45 servidores de emparejamiento, transmisiones de vídeo y servicios de redes sociales. También proporciona al usuario la instalación y la actualización automática de juegos y características de comunidad como grupos y listas de amigos, guardado en la nube, etc.	Si cuenta con un avanzado sistema de recomendación.
Good Old Games (GOG)		Página web: <a href="http://www.gog.com/">http://www.gog.com/</a> Empresa: CD Project Lanzamiento: 2008	Es un servicio de venta y distribución de videojuegos. Los videojuegos comprados a través de GOG.com no usan gestión digital de derechos (DRM), por lo tanto el usuario no tiene que instalar un cliente especial para descargar o ejecutar los videojuegos, aunque el servicio ofrece un gestor de descargas.	Si, un grupo de especialistas que recomiendan los juegos.
Origin		Página web: <a href="https://www.origin.com/">https://www.origin.com/</a> Empresa: Electronic Arts Lanzamiento: 2011	Origin es una alternativa muy interesante para los aficionados a los juegos de Electronic Arts. La plataforma es propiedad de la famosa desarrolladora y ofrece solo los juegos que esta hace, lo que puede ser un punto positivo o negativo, dependiendo del caso.	Si cuenta con un sistema de recomendación.
Green man gaming		Página web: <a href="https://www.greenmangaming.com/">https://www.greenmangaming.com/</a> Empresa: Paul Sulok Lanzamiento: 2009	Green Man es un minorista, distribuidor y editor de videojuegos en línea con sede en Gran Bretaña. Green Man Gaming ha ganado 4,7 millones de usuarios desde su lanzamiento original.	Cuenta con un sistema de recomendación.
Kinguin		Página web: <a href="https://www.kinguin.net/">https://www.kinguin.net/</a> Empresa: Electronic Arts Lanzamiento: 2011	Esta plataforma es algo diferente a las otras opciones de las que hemos hablado hasta ahora, ya que ofrece un lugar seguro en el que los jugadores pueden vender, comprar e intercambiar juegos, objetos y skins con otras personas, lo que la convierte en una de las mejores opciones.	Cuenta con un sistema de recomendación.
Humble Bundle		Página web: <a href="https://www.humblebundle.com/">https://www.humblebundle.com/</a> Empresa: Jhon Graham, Jeff Rosen Lanzamiento: 2010	Ofrece muchos juegos estupendos disponibles, y una gran parte de las ganancias se destina a instituciones benéficas. Lo mejor es que cada mes la tienda crea paquetes de al menos seis juegos que puedes conseguir por el precio que elijas. Por lo general, estos paquetes son temáticos.	Cuenta con un sistema de recomendación.
Gamers Gate		Página web: <a href="https://www.gamersgate.com/">https://www.gamersgate.com/</a> Empresa: Paradox Lanzamiento: 2006	Tienda que ofrece un gran catálogo de juegos, que va desde los mayores lanzamientos de la industria hasta algunos de los indies más interesante. Además, ofrece un programa de recompensas que depende mucho de la participación e interacción en la web.	Cuenta con un sistema de recomendación.
Epic Games		Página web: <a href="https://www.epicgames.com/">https://www.epicgames.com/</a> Empresa: Electronic Arts Lanzamiento: 1991	Es muy probable que ya conozcas o al menos hayas oído hablar de Fortnite, uno de los battle royales más famosos desde que salió a la venta en 2017. Solo puedes conseguirlo en Epic Games Store. Una marketplace que ha ido dejando una gran huella en el mercado de los videojuegos.	Cuenta con un sistema de recomendación.

Tabla 5.1 Principales plataformas de venta de video juegos en el mundo.

Por lo general estas grandes empresas que están consolidadas en el mercado, cuentan con equipos de analítica, los cuales elaboran los sistemas de recomendación. Por lo tanto, no tiene mucho sentido enfocar el modelo de negocio en estas compañías. Por eso el foco del sistema de recomendación propuesto en este TFM serán las pequeñas y nuevas tiendas virtuales que no cuenten con estos sistemas.

Como nicho de mercado, se buscarán primero las tiendas virtuales en Colombia, seguido de LATAM y España, que se dediquen a la venta y distribución de videojuegos. La primera opción de ofrecer este proyecto es a la empresa [Juegos Digitales Colombia](#), que es una pequeña tienda virtual de videojuegos para diferentes plataformas y se fundó el 11 de agosto del 2020. Esta tienda lleva apenas 2 años en el mercado y no cuenta con un sistema de recomendación. En España un posible cliente será la página de [Xtralife](#), la cual existe hace más de una década con sede y almacén en Barcelona. Ofrece videojuegos, consolas, accesorios, merchandising y mucho más, empaquetadores oficiales de universos de las principales marcas gaming, ofrece productos virtuales y en físico.

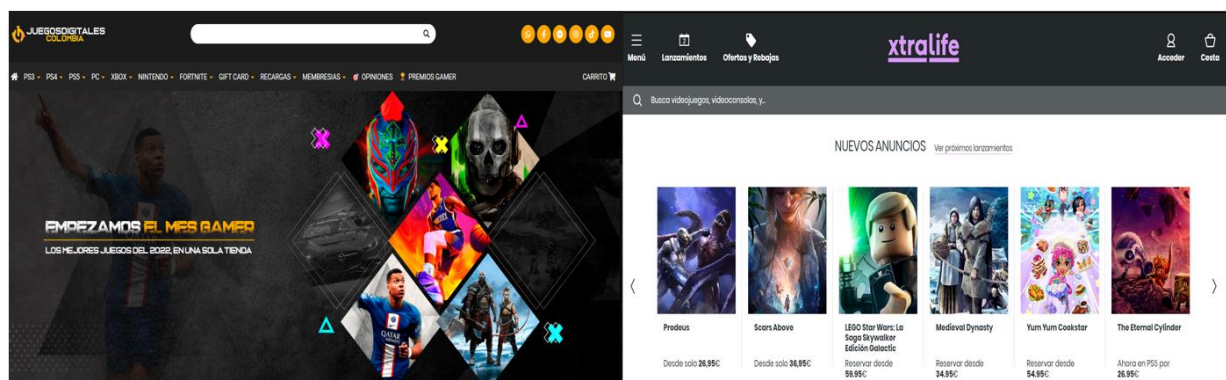


Figura 5.2 Izquierda, página de la empresa juegos digitales Colombia, y derecha de Xtralife.

Por otra parte, no se descartan las grandes compañías, pues se espera que la empresa se vaya especializando en mejorar cada vez mas el sistema de recomendación propuesto, y si logra dar mejores recomendaciones que los ya existentes, se les ofrecería a las grandes compañías tal y como le paso a Netflix en el 2007 que ofreció un premio de 1 millón de dólares aquel que lograra construir un mejor sistema de recomendación que Cinematch.

El otro modelo de negocio será ofrecer un segundo producto a las tiendas virtuales de videojuegos. El cual consiste en segmentar a los clientes mediante técnicas de clustering para crear campañas dirigidas. Para construir este producto, será necesario recolectar información demográfica del usuario y su histórico de compras, esto no solo permitirá dar recomendaciones más acertadas, sino que permitirá identificar aquellos usuarios con mayor valor, esto en basado en la cantidad de dinero que hayan gastado en videojuegos.

La fuente de ingresos de la compañía se basará en proporcionar un api Restful que consumirá el cliente que adquiera el producto. La idea no es solo ofrecer un producto sino un servicio integral que permita medir el rendimiento del sistema de recomendación. De esta forma este proyecto tendrá una mejora sustancial ya que, no se vende el producto una vez, si no que, se paga regularmente por el servicio brindado por la compañía.

En cuanto al precio se fijarán unas tarifas ya sean mensuales o por llamada que se haga a la API. Note que el cliente podrá integrar la API directamente a su sitio web, para dar recomendaciones directas al usuario.

## 5.2 Benchmarking

El estudio profundo realizado a la competencia, indica que hay un gran número de empresas que se dedican a comercializar sistemas de recomendación, además de ofrecer soluciones de analytics, AI, captación de usuarios, customer acquisition funneling, entre otros servicios. Para este TFM nos hemos enfocado únicamente en empresas ubicadas en LATAM y España, el cual será nuestro nicho de mercado inicial.

Nombre	Logo	Descripción
<b>Smarthint</b> 		Fundada en 2017, SmartHint ayuda a miles de tiendas virtuales a brindar a sus usuarios una experiencia individualizada, completa y única, a través de un sistema inteligente de búsqueda y recomendación. Mediante inteligencia artificial, análisis de datos y comportamiento del consumidor dentro de la tienda virtual, nuestra tecnología identifica intereses y personaliza toda la experiencia online de los usuarios, potenciando factores como: ganancia de competitividad, fidelización de clientes e incremento de ventas. Con el propósito de elevar la experiencia del cliente y transformar el viaje de compras en línea, hoy SmartHint es parte del ecosistema Luizalabs, el laboratorio de tecnología e innovación de Magazine Luiza.
<b>Recommendix</b> 		Recommendix ofrece una plataforma para surtido basado en datos y gestión de precios. Esta empresa propone un enfoque completamente diferente para la búsqueda en sitios de comercio electrónico. Básicamente le hace al usuario un corto cuestionario, para entender lo que esta buscando, inmediatamente la herramienta convierte las respuestas de los usuarios en un conjunto de filtros. Se obtiene una selección de 10 productos que coinciden con sus criterios de búsqueda.
<b>BrainSINS</b> 		Es una empresa que se enfoca en desarrollar sistemas de recomendación de compras personalizadas, para los ecommerce. Se basa en recomendaciones en función del stock, en función de fechas, en base a proveedores o marcas que más se quieran comercializar, entre otros. Ofrecen un software exclusivo de la compañía, para que el cliente pueda analizar los resultados y rendimiento de las estrategias de recomendaciones, y evaluar cuánto está aportando el sistema a su negocio. Aseguran un incremento en las ventas de un 17%.
<b>SoluSoft</b> 		Solusoft es una compañía de Consultoría y Servicios de Tecnologías de la Información con una amplia experiencia que alcanza ya los 28 años y que está repleta de casos de éxito en la implantación de productos y herramientas informáticas al servicio de muchas empresas y organizaciones en general. Ofrece el servicio de crear sistema de recomendación utilizando algoritmos de filtros colaborativos.

Tabla: 5.2 Benchmarking de las empresas en Colombia (LATAM) y España.

## 5.3 Análisis de viabilidad

Para la constitución de una empresa, según la Cámara de Comercio de Bogotá (CCB), se debe contar con la mayoría de edad; las personas naturales como es mi caso deben iniciar con la creación de empresa en la Cámara de Comercio a través del Registro Mercantil. “Entre la información solicitada estará a qué se va a dedicar, dirección, correo electrónico y demás información básica. Hay que matricular el establecimiento de comercio y le solicitarán un estimado de los activos”, anotó. La CCB, a su vez, adicionó que para el Registro Mercantil es necesario tener diligenciado y firmado el Registro Único Empresarial y Social (Rues).

Por otra parte, el Registro Único Tributario de la Dirección de Impuestos y Aduanas Nacionales (Dian), también debe ser tramitado por las sociedades. Hay que tener claro que, como esa empresa operará en un municipio, tiene que inscribirse en la administración tributaria de esa región. En Bogotá se llama el Registro de Información Tributaria (RIT) y esos trámites son necesarios para nacer y sacar la autorización de facturación.

La CCB ofreció un estimado de cuánto dinero podría necesitarse para formalizar una empresa. Para las personas naturales y jurídicas, los derechos de matrícula cuestan \$8.4 USD para capitales de hasta \$52.971 USD y el formulario Rues tendría un costo de \$1.5 USD.

**Nombre de la Empresa:** AnliticSify

**Slogan:** El poder de los datos

**Logo:** El logo de elaboración propia ha sido creado en la página [Tailor Brands](#).



**Ubicación:** Edificio BlueWork, que es un coworking ubicado en Calle 118 #19A-33, Bogotá.

**Numero de empleados:** 3 personas

- Data Scientist
- Desarrollador web
- Call Center & Support

**Nota:** Para efectos de este ejercicio los valores serán expresados en dólares americanos (USD) con una tasa representativa del mercado (TRM) de \$4.600 COP.

Para realizar el análisis de viabilidad hemos considerado los siguientes costos, primero los salarios de los empleados, que incluye un científico de datos, que creara el sistema de recomendación, dará consultorías en todo lo relacionado con datos, y elaborara modelos de machine learning. El desarrollador web se encargará principalmente de crear paginas web a demanda de los clientes y dará soporte en los temas de TI de la empresa. Una persona encargada de manejar el call center, chat y correo corporativo, además de recepcionar PQRS y soporte que se necesite por parte de los clientes.

Cargo	No	Sueldos	Prestaciones	Total Sueldo	Año 1	Año 2	Año 3	Año 4	Año 5
Data Scientist	1	\$ 1.300	\$ 674	\$ 1.974	\$ 23.690	\$ 30.797	\$ 40.035	\$ 52.046	\$ 67.660
Desarrollador Web	1	\$ 800	\$ 415	\$ 1.215	\$ 14.578	\$ 18.952	\$ 24.637	\$ 32.028	\$ 41.637
Call Center & Support	1	\$ 450	\$ 233	\$ 683	\$ 8.200	\$ 10.660	\$ 13.858	\$ 18.016	\$ 23.421
<b>Total</b>	<b>3</b>	<b>\$ 2.550</b>	<b>\$ 1.322</b>	<b>\$ 3.872</b>	<b>\$ 46.468</b>	<b>\$ 60.409</b>	<b>\$ 78.531</b>	<b>\$ 102.091</b>	<b>\$ 132.718</b>

Tabla 5.2 Salarios de los empleados, desde el año 1 al 5, con incremento anual del 3%.



Costos Fijos					
Conceptos / Año	Año 1	Año 2	Año 3	Año 4	Año 5
Salarios	\$ 46.468,1	\$ 60.408,6	\$ 78.531,2	\$ 102.090,5	\$ 132.717,7
Arriendo	\$ 4.440,0	\$ 4.688,6	\$ 4.951,2	\$ 5.228,5	\$ 5.521,3
Servicio de Agua	\$ 180,0	\$ 190,1	\$ 200,7	\$ 212,0	\$ 223,8
Servicio de Luz	\$ 360,0	\$ 380,2	\$ 401,4	\$ 423,9	\$ 447,7
Servicio de Internet + Telefonía	\$ 421,8	\$ 445,4	\$ 470,4	\$ 496,7	\$ 524,5
Posicionamiento web	\$ 213,6	\$ 225,6	\$ 238,2	\$ 251,5	\$ 265,6
<b>Subtotal</b>	<b>\$ 52.083,5</b>	<b>\$ 66.338,4</b>	<b>\$ 84.793,1</b>	<b>\$ 108.703,1</b>	<b>\$ 139.700,6</b>

Costos Variables					
Conceptos / Año	Año 1	Año 2	Año 3	Año 4	Año 5
Uso de Google Cloud Computing	\$ 7.200,0	\$ 7.603,2	\$ 8.029,0	\$ 8.478,6	\$ 8.953,4
Transportes	\$ 100,0	\$ 105,6	\$ 111,5	\$ 117,8	\$ 124,4
Mantenimiento PC	\$ 200,0	\$ 211,2	\$ 223,0	\$ 235,5	\$ 248,7
Comisiones ejecutivo comercial	\$ 3.900,0	\$ 4.118,4	\$ 4.349,0	\$ 4.592,6	\$ 4.849,8
<b>Subtotal</b>	<b>\$ 11.400,0</b>	<b>\$ 12.038,4</b>	<b>\$ 12.712,6</b>	<b>\$ 13.424,5</b>	<b>\$ 14.176,2</b>

**Nota:** Todos los valores están dados en USD con una TRM de \$4.600 COP

<b>Costos Totales</b>	<b>\$ 63.483,5</b>	<b>\$ 78.376,8</b>	<b>\$ 97.505,6</b>	<b>\$ 122.127,6</b>	<b>\$ 153.876,8</b>
-----------------------	--------------------	--------------------	--------------------	---------------------	---------------------

Tabla 5.3 Costos fijos y variables de la empresa desde el año 1 al año 5.

En la anterior tabla se visualizan los costos fijos que incluye los salarios de los empleados, el canon de arrendamiento correspondiente al coworking, los servicios de agua, luz, internet y telefonía, además de un servicio de posicionamiento web para la página de la empresa.

Por otra parte, entre los costos variables se encuentra el uso de la nube GCP, que se utilizarán diferentes servicios como App Engine, Compute Engine, SQL, Big Query, Cloud Storage, entre otros el cual será a demanda y por lo tanto su costo es variable. Se incluyen el valor de los transportes, cuando sea necesario visitar a un posible cliente que no pueda asistir a nuestra oficina, además del mantenimiento de los computadores en caso de que alguno falle y finalmente las comisiones que se le darán a un ejecutivo comercial que trabaje bajo el esquema de comisión por venta de los productos de la compañía. No es necesario incluir mobiliario ya que el coworking lo proporciona, sin embargo, en el balance general se tiene en cuenta la compra de 3 computadores, 2 portátiles y 1 de escritorio con su respectiva depreciación para los cálculos. Para los costos totales se aplicó una inflación correspondiente al 5.6% anual.

Ingresos x Venta de Servicios					
Conceptos / Año	Año 1	Año 2	Año 3	Año 4	Año 5
Sistema de recomendación	\$ 10.000,0	\$ 21.120,0	\$ 44.605,4	\$ 70.655,0	\$ 74.611,7
Soporte	\$ 6.000,0	\$ 12.672,0	\$ 26.763,3	\$ 42.393,0	\$ 44.767,0
Comisión por venta	\$ 7.000,0	\$ 17.740,8	\$ 39.029,8	\$ 64.296,1	\$ 67.896,6
Consultorías	\$ 1.000,0	\$ 4.224,0	\$ 5.018,1	\$ 5.887,9	\$ 6.217,6
Creación Soluciones ML	\$ 20.000,0	\$ 26.400,0	\$ 33.454,1	\$ 35.327,5	\$ 37.305,8
Desarrollo de sitios web	\$ 6.000,0	\$ 6.336,0	\$ 8.363,5	\$ 8.831,9	\$ 9.326,5
<b>Total</b>	<b>\$ 50.000,0</b>	<b>\$ 88.492,8</b>	<b>\$ 157.234,2</b>	<b>\$ 227.391,4</b>	<b>\$ 240.125,3</b>

Tabla 5.4 Ingresos por ventas de servicios.

Como se indicó la empresa ofrecerá diferentes servicios, para aumentar los ingresos y que el negocio sea sostenible y rentable. Para el primer año se contempla vender 1 sistema de recomendación por un valor de \$10.000 USD que contempla la puesta en marcha del mismo, el soporte mensual de este tiene un valor de \$500 USD y por cada venta que haga la plataforma gracias

a este medio supone una comisión del 2.5% sobre el valor del videojuego. A parte se prestarán servicios de consultoría para asesorar empresas en proyectos de datos, esto supone un valor de \$500 USD por cada una, para el primer año se harán 2. Las soluciones ML hará parte también del ingreso de la empresa, la elaboración de cada modelo predictivo tendrá un valor de \$5.000 USD y para el primer año se espera vender 4 modelos. Finalmente se espera vender el desarrollo de 4 paginas web el primer año con un valor promedio de \$1.500 USD cada una.

Para la constitución de la empresa se deberán pagar \$9.9 USD de derecho de matrícula en la cámara y comercio de Bogotá, más el formulario RUES. Los socios aportarán un capital inicial de \$2.500 USD, de los cuales \$500 USD irán a caja y \$2.000 USD al banco. Además, se pedirá un préstamo al banco por valor de \$20.000 USD con una tasa EA del 14.7%, a un plazo de 5 años. Dinero que se utilizara para comprar los 3 equipos de cómputo, pagar el arriendo y salarios de los primeros meses.

ESTADO DE RESULTADOS						
	Año 0	Año 1	Año 2	Año 3	Año 4	Año 5
Ventas		\$ 50.000,0	\$ 88.492,8	\$ 157.234,2	\$ 227.391,4	\$ 240.125,3
-Costo de Ventas		\$ 63.483,5	\$ 78.376,8	\$ 97.505,6	\$ 122.127,6	\$ 153.876,8
<b>=Utilidad en ventas</b>		<b>-\$ 13.483,5</b>	<b>\$ 10.116,0</b>	<b>\$ 59.728,5</b>	<b>\$ 105.263,8</b>	<b>\$ 86.248,5</b>
- Egresos operacionales		\$ 32.828,5	\$ 34.141,7	\$ 35.507,3	\$ 36.927,6	\$ 38.404,7
		\$ 32.083,5	\$ 33.396,7	\$ 34.762,3	\$ 36.182,6	\$ 37.659,7
Depreciación		\$ 745,0	\$ 745,0	\$ 745,0	\$ 745,0	\$ 745,0
Amortización		\$ -	\$ -	\$ -	\$ -	\$ -
<b>=Utilidad operacional</b>		<b>-\$ 46.312,1</b>	<b>-\$ 24.025,7</b>	<b>\$ 24.221,2</b>	<b>\$ 68.336,2</b>	<b>\$ 47.843,8</b>
+Otros ingresos		\$ -	\$ -	\$ -	\$ -	\$ -
-Otros Egresos		\$ -	\$ -	\$ -	\$ -	\$ -
<b>=Utilidad Antes de impuestos</b>		<b>-\$ 46.312,1</b>	<b>-\$ 24.025,7</b>	<b>\$ 24.221,2</b>	<b>\$ 68.336,2</b>	<b>\$ 47.843,8</b>
-Impuesto de renta 30%	30%	\$ -	\$ -	\$ 7.266,4	\$ 20.500,9	\$ 14.353,1
<b>=Utilidad neta</b>		<b>-\$ 46.312,1</b>	<b>-\$ 24.025,7</b>	<b>\$ 16.954,8</b>	<b>\$ 47.835,3</b>	<b>\$ 33.490,6</b>

Tabla 5.5 Estado de resultados de la empresa.

BALANCE GENERAL						
	Año 0	Año 1	Año 2	Año 3	Año 4	Año 5
<b>ACTIVO</b>	<b>2.510</b>	<b>- 26.796</b>	<b>- 7.932</b>	<b>36.390</b>	<b>76.401</b>	<b>50.344</b>
Caja	500	- 45.067	- 68.348	- 43.382	18.433	46.521
Bancos	2.000	2.000	2.000	2.000	2.000	2.000
Oficina						
Muebles y enseres	-	-	-	-	-	-
Maquinaria y equipo	-	-	-	-	-	-
Equipo de computo	3.750	3.750	3.750	3.750	3.750	3.750
Dep.acum activo	-	745	1.490	2.235	2.980	3.725
Cargos diferidos preoperativos	-	-	-	-	-	-
<b>PASIVO</b>		<b>17.016</b>	<b>13.594</b>	<b>9.668</b>	<b>5.565</b>	<b>-</b>
Obligaciones financieras	-	17.016	13.594	9.668	5.565	-
<b>PATRIMONIO</b>	<b>2.510</b>	<b>- 43.812</b>	<b>- 21.526</b>	<b>26.721</b>	<b>70.836</b>	<b>50.344</b>
Impo por pagar		-	-	7.266	20.501	14.353
Derechos de Matricula + RUES	10					
Capital	2.500	2.500	2.500	2.500	2.500	2.500
Utilidades	-	- 46.312	- 24.026	16.955	47.835	33.491
<b>Utilidad Neta Acumulada</b>	<b>-</b>	<b>- 46.312</b>	<b>- 70.338</b>	<b>- 53.383</b>	<b>- 5.548</b>	<b>27.943</b>

Tabla 5.6 Balance general de la empresa.

Como se observa en el estado de resultado y el balance general los dos primeros años dan utilidad negativa, pero a medida que se va consolidando la empresa el beneficio va aumentando, debido a que se van adquiriendo nuevos clientes y desarrollando otros productos.

Para el quinto año de operación ya se ha recuperado la inversión inicial, por lo que se empezaran a recibir los beneficios netos. También a partir del año 5 se espera un crecimiento sostenido del 20%, que es cuando ya se ha consolidado la empresa.

## Conclusiones

En este documento, se describió la teoría de los principales sistemas de recomendación que utilizan filtros colaborativos, además se mostró como extraer la información de Steam desde las diferentes Api's públicas que están disponibles en la web. Para el modelado se utilizó el software R junto con el paquete **recommenderlab**, que cuenta con toda la infraestructura necesaria, desde algoritmos, evaluación y predicción.

Se creo un sistema de recomendación de filtros colaborativos hibrido, a partir de los algoritmos reglas de asociación y el IBCF, que fueron los de mejor performance en términos de precisión y recall. El modelo se desplegó utilizando el framework r shiny, a modo mockup para los clientes.

También se propone crear una empresa especializada en soluciones de analítica, que comercialice como producto el sistema de recomendación para pequeñas tiendas de videojuego en LATAM y España, donde el primer clienta será juegosdigitalescolombia.com, para posteriormente expandirse a otros países.

Se realizo el análisis de viabilidad del proyecto, y para que este sea rentable es necesario ofrecer otro tipo de soluciones relacionadas con la analítica. De esta forma la inversión realizada por los socios y el préstamo solicitado al banco se recupera en el quinto año dejando beneficios netos a los socios.

## Limitantes

Se encontraron dos limitantes en este proyecto, relacionado con el proceso de extraer la información de la API de Steam. Primero porque para algunos endpoint es necesario contar con la API Key Steam. La solución fue abrir una cuenta que es gratuito, e invertir \$5.00 USD en algún video juego lo cual habilita el servicio. La segunda limitante es todas las API's de Steam solo permiten una llamada cada 1.5 segundo, por lo que si se quiere extraer la información de 50.000 juegos se necesitaría 20 H 50 M. La solución a este problema fue crear dos instancias de Windows en GCP (Compute Engine), más mi computador eran 3 PC, extrayendo la información al tiempo.

## Anexos

El código utilizado, se encuentra alojado en un repositorio público de Github, en la siguiente ubicación: [https://github.com/RafaelEduardoDiaz/sistema\\_recomendacion](https://github.com/RafaelEduardoDiaz/sistema_recomendacion). Allí, encontrara el archivo Excel con el análisis de viabilidad financiera y este mismo archivo en formato Word.

Además de los script en R con los códigos del repositorio, que se dividen en diferentes archivos:

1. Extraer la data desde la API de Steam
2. Modelación del Sistema de Recomendación
3. Análisis descriptivo de los datos



# Referencias

- Demiriz A (2004). "Enhancing Product Recommender Systems on Sparse Binary Data." *Data Mining and Knowledge Discovery*, 9(2), 147–170. ISSN 1384-5810.
- Deshpande M, Karypis G (2004). "Item-based top-N recommendation algorithms." *ACM Transactions on Information Systems*, 22(1), 143–177. ISSN 1046-8188.
- Desrosiers C, Karypis G (2011). "A Comprehensive Survey of Neighborhood-based Recommendation Methods." In F Ricci, L Rokach, B Shapira, PB Kantor (eds.), *Recommender Systems Handbook*, chapter 4, pp. 107–144. Springer US, Boston, MA. ISBN 978-0-387-85819-7
- Gunawardana A, Shani G (2009). "A Survey of Accuracy Evaluation Metrics of Recommendation Tasks." *Journal of Machine Learning Research*, 10, 2935–2962.
- Hahsler, M. (2022). *recommenderlab: An R Framework for Developing and Testing Recommendation Algorithms*. *arXiv preprint arXiv:2205.12371*.
- Kohavi R, Provost F (1998). "Glossary of Terms." *Machine Learning*, 30(2–3), 271–274.
- Koren Y, Bell R, Volinsky C (2009). "Matrix Factorization Techniques for Recommender Systems." *Computer*, 42, 30–37. doi:<http://doi.ieeecomputersociety.org/10.1109/MC.2009.263>.
- Lemire D, Maclachlan A (2005). "Slope One Predictors for Online Rating-Based Collaborative Filtering." In *Proceedings of SIAM Data Mining (SDM'05)*.
- Martínez-Plumed, F., Contreras-Ochando, L., Ferri, C., Hernández Orallo, J., Kull, M., Lachiche, N., Flach, P. A. (2019). Crisp-dm twenty years later: From data mining processes to data science trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 1-1. doi: 10.1109/TKDE.2019.2962680
- Mild A, Reutterer T (2003). "An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data." *Journal of Retailing and Consumer Services*, 10(3), 123–133.
- Schafer JB, Konstan JA, Riedl J (2001). "E-Commerce Recommendation Applications." *Data Mining and Knowledge Discovery*, 5(1/2), 115–153.
- Shardanand U, Maes P (1995). "Social Information Filtering: Algorithms for Automating 'Word of Mouth'." In *Conference proceedings on Human factors in computing systems (CHI'95)*, pp. 210–217. ACM Press/Addison-Wesley Publishing Co., Denver, CO.
- Pan R, Zhou Y, Cao B, Liu NN, Lukose R, Scholz M, Yang Q (2008). "One-Class Collaborative Filtering." In *IEEE International Conference on Data Mining*, pp. 502–511. IEEE Computer Society, Los Alamitos, CA, USA. ISSN 1550-4786.
- Pascual, J. (2018). 15 años de Steam: Evolución y Anécdotas. Hobby Consolas. <https://www.hobbyconsolas.com/reportajes/15-anos-steam-evolucion-anecdota-312159>
- Terry, D. B. (1993, December). A tour through tapestry. In *Proceedings of the conference on Organizational computing systems* (pp. 21-30).
- Qomariyah, N. N. (2018). *Pairwise Preferences Learning for Recommender Systems* (Doctoral dissertation, University of York).
- Zhang S, Yao L, Sun A, Tay Y (2019). "Deep learning based recommender system: A survey and new perspectives." *ACM Computing Surveys (CSUR)*, 52(1).