Streamly

Recomendación de películas impulsado con IA

Por: Miguel Ángel Vázquez, Samuel Robayo y Abel Pérez

1. Introducción a Streamly

Streamly es una innovadora plataforma que busca transformar la experiencia de descubrimiento de películas para los usuarios.

A través de la integración de diversos modelos de inteligencia artificial y procesamiento de lenguaje natural, la aplicación y el sitio web ofrecen sugerencias altamente personalizadas, adaptándose a los gustos individuales y fomentando la exploración de nuevos contenidos.

El proyecto se ha desarrollado con un enfoque multiplataforma, asegurando la accesibilidad desde dispositivos móviles y navegadores web.

1.1 Entendimiento del negocio

Streamly es una plataforma digital de entretenimiento enfocada en la distribución de contenido audiovisual personalizado para cada usuario. Su propuesta de valor se basa en ofrecer una experiencia dinámica, inteligente y adaptativa, donde el contenido que se presenta a cada usuario responde a sus gustos, hábitos de consumo y contexto de uso.

En este contexto, diferenciarse a través de una personalización más precisa y una interfaz intuitiva es clave para captar y retener usuarios. Streamly reconoce que el conocimiento profundo del comportamiento de sus usuarios no solo permite mejorar las recomendaciones, sino también ofrecer al usuario una mejor experiencia en su inversión en entretenimiento.

1.2 Objetivos del negocio

Con el fin de cumplir con los objetivos de **Streamly**, se ha realizado un análisis detallado de los datos de uso de la plataforma para identificar patrones de visualización y preferencias de contenido de nuestros usuarios. A partir de estos análisis, se busca predecir los tipos de contenido que podrían interesarles en el futuro, con el objetivo de desarrollar un sistema de recomendación más preciso y personalizado. Esto permitirá aumentar el nivel de actividad en la plataforma y mejorar la experiencia del usuario, alineándose con la misión de Streamly de ofrecer un servicio de entretenimiento más dinámico y adaptado a cada persona. Los objetivos que se han determinado son los siguientes:

1. **Conseguir más actividad sobre los usuarios ya registrados:**

   Fomentar un mayor nivel de compromiso (retención) entre los usuarios registrados mediante el incremento en la frecuencia de uso, la duración de las sesiones y la interacción con las funcionalidades de la plataforma.

2. **Conseguir una nueva base de usuarios para mejorar los modelos:**

   Generar crecimiento orgánico de la base de usuarios mediante la recomendación entre pares (boca a boca), con el objetivo de obtener un conjunto de datos más representativo y robusto que permita optimizar la precisión y generalización de los modelos predictivos. Esto facilitará un proceso iterativo de mejora continua, orientado a una personalización más eficaz y, en consecuencia, a una experiencia de recomendación más relevante para el usuario.

3. **Optimizar la oferta de contenido en función de las preferencias y patrones de consumo detectados:**

   Utilizar los datos analizados para tomar decisiones informadas sobre la adquisición, producción y promoción de contenido en la plataforma. Esto permitirá priorizar aquellos géneros, formatos o temáticas con mayor potencial de engagement, contribuyendo tanto a la mejora de la experiencia del usuario como a la eficiencia operativa y comercial de Streamly.

1.3 Análisis DAFO

Para comprender mejor el negocio y su posicionamiento en un entorno altamente competitivo, se ha llevado a cabo un análisis DAFO (Debilidades, Amenazas, Fortalezas y Oportunidades). Este análisis permite identificar los factores internos y externos que pueden influir en el rendimiento de Streamly, y sirve como base para definir estrategias más efectivas que contribuyan al cumplimiento de los objetivos establecidos. Los resultados del análisis se presentan como anexo al final de este documento.

2. Entendimiento de los datos

2.1 Descripción de los datos

Aquí podemos observar el contenido de cada tabla de nuestra base de datos y el tipo de datos de cada campo

Tabla Géneros		
Variables	Description	Data Type
id	Identificador único del género	int64
name	Nombre del género	object

Tabla Ratings		
Variables	Description	Data Type
user_id	Identificador único del usuario	int64
movie_id	Identificador único del usuario	int64
rating	Puntuación del rating	float64
timestamp	Fecha	int64
description	Texto del Rating	object

Tabla Users		
Variables	Description	Data Type
userId	Identificador único del usuario	int64
username	Nombre con el que se le conoce en la app	object
firstName	Nombre	object
lastName	Apellido	object
email	Email del usuario	object
passwordHash	Contraseña con Hash	object
password	Contraseña	object
preferred genres	Géneros favoritos	object
age	Edad del usuario	int64
occupation	Trabajo del usuario	object

Tabla Films		
Variables	Description	Data Type
movie_id	Identificador único de la película	int64
tmdb_id	Identificador único de la película en tmdb	int64
imdb_id	Identificador único de la película en imdb	object
title	Título del a película	object
original_title	Título original de la película	object
overview	Descripción de la película	object
tagline	Tags de la película	object
release_date	Fecha de lanzamiento	object
runtime	Duración de la película	int64
budget	Coste de la película	int64
revenue	Beneficios de la película	int64
popularity	Popularidad de la película	float64
vote_avarage	Media de ratings en tmdb	float64
vote_count	Cantidad total de votos en tmdb	int64
status	Estado de la película	object
adult	Identificador de sí la película es solo para publica adulto	int64
video	Link del trailer	int64
poster_path	Link del Poster	object

backdrop_path	Link del background	object
homepage	Link de la película	object
original language	Idiomas originales	object

Tabla Views		
Variables	Description	Data Type
user_id	Identificador único del usuario	int64
movie_id	Identificador único de la película	int64
view_date	Descripción de la película	object

Tabla Películas - Género		
Variables	Description	Data Type
movie_id	Identificador único de la película	int64
genre_id	Identificador único del género	int64

2.2 Verificación de la Calidad de los Datos

Una vez que tenemos los datos, el primer paso es limpiar y normalizar, es decir localizar valores vacíos, duplicados, perdidos, nulos, columnas o campos en distintos idiomas o diferentes unidades de medida. Analizando la base de datos, no encontramos valores nulos ni vacíos en estas tablas, como era de esperar en este caso tampoco encontramos diferentes unidades de medida y está todo en inglés, pero hemos detectado una mala práctica en cuanto a los géneros de los usuarios en el dato:

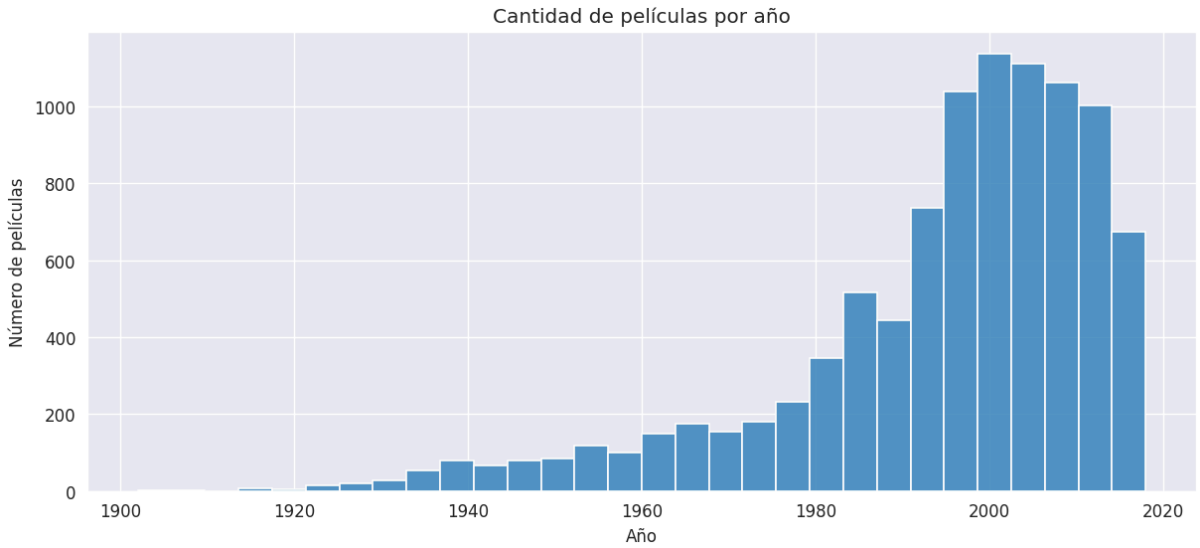
'preferred_genres'

Ya que es un String con muchos géneros con un separador “|”, por lo que hemos decidido transformar este campo en una nueva tabla auxiliar con los géneros favoritos del usuario.

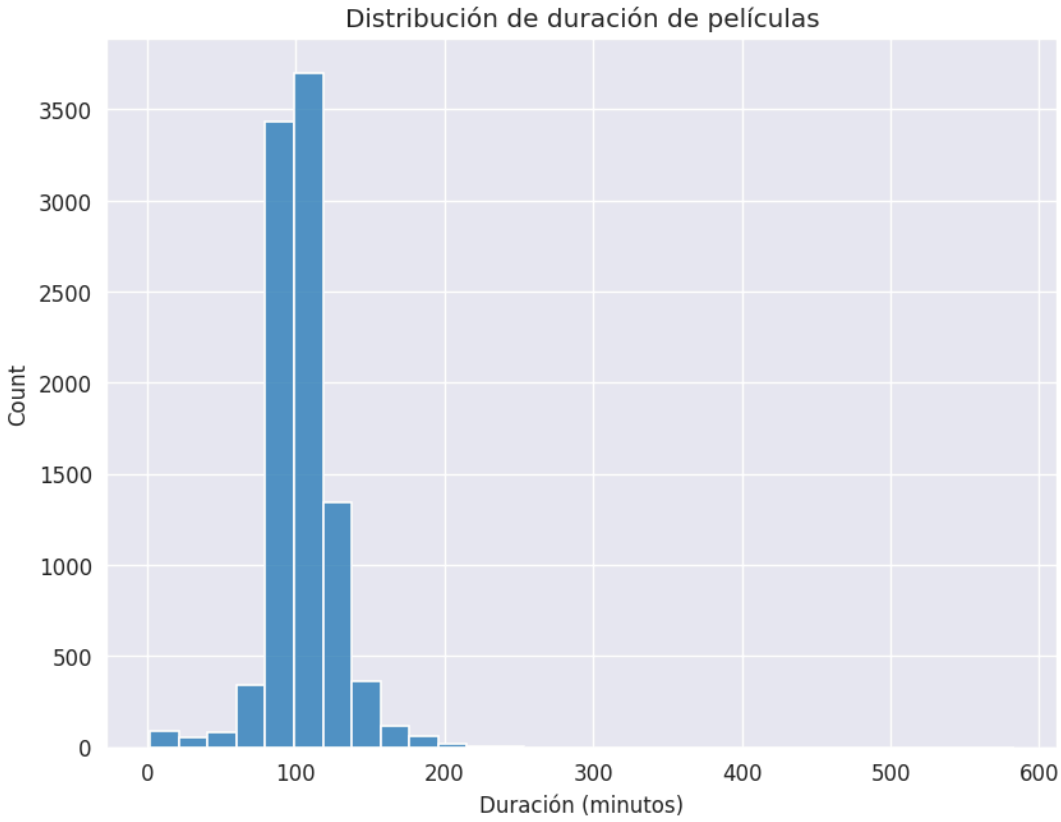
Tabla User - Género		
Variables	Description	Data Type
user_id	Identificador único de la película	int64
genre_id	Identificador único del género	int64

A continuación describiremos las variables numéricas:

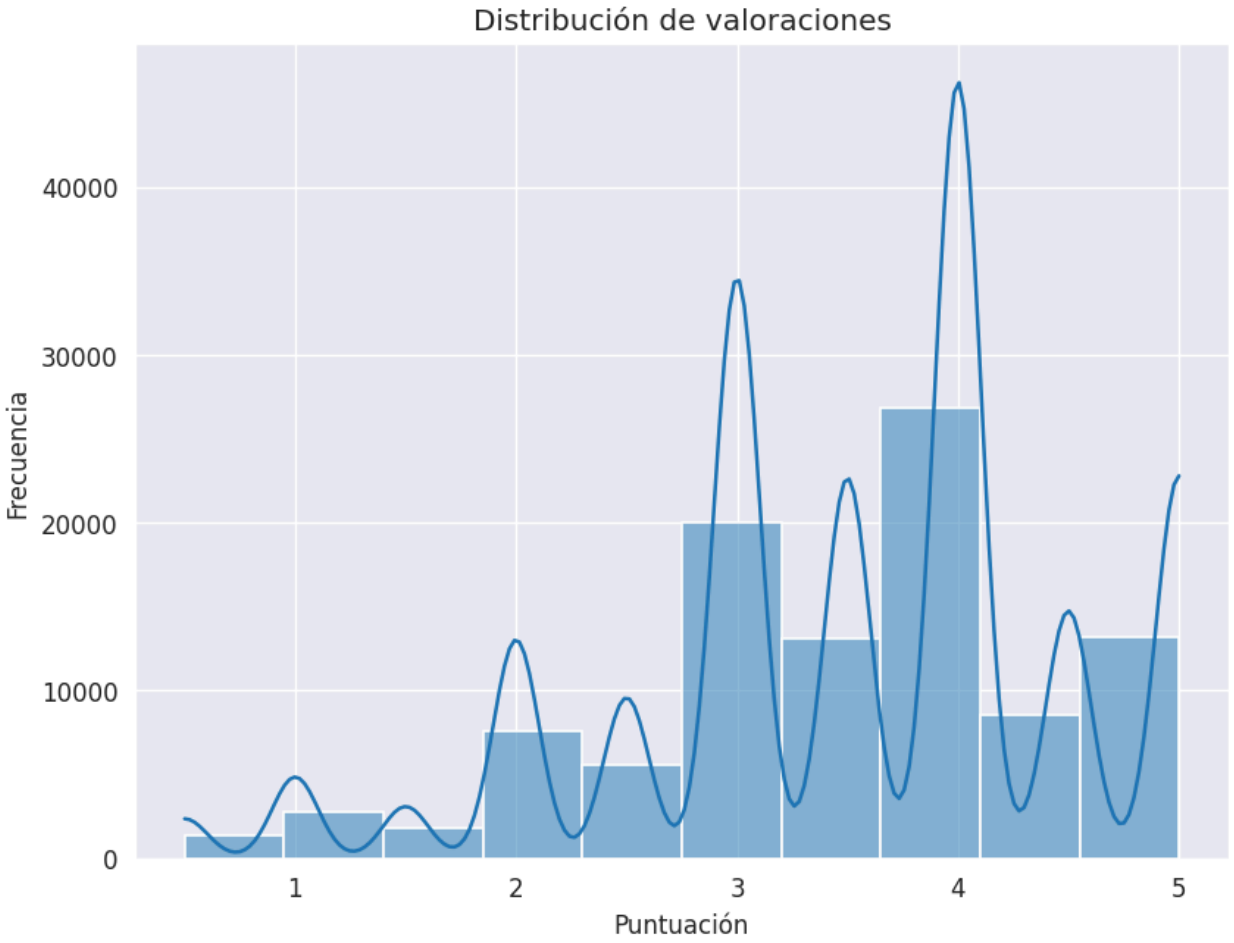
- Cantidad de películas por año: Con esta gráfica podemos observar cuales son los años en los que más películas se publican.



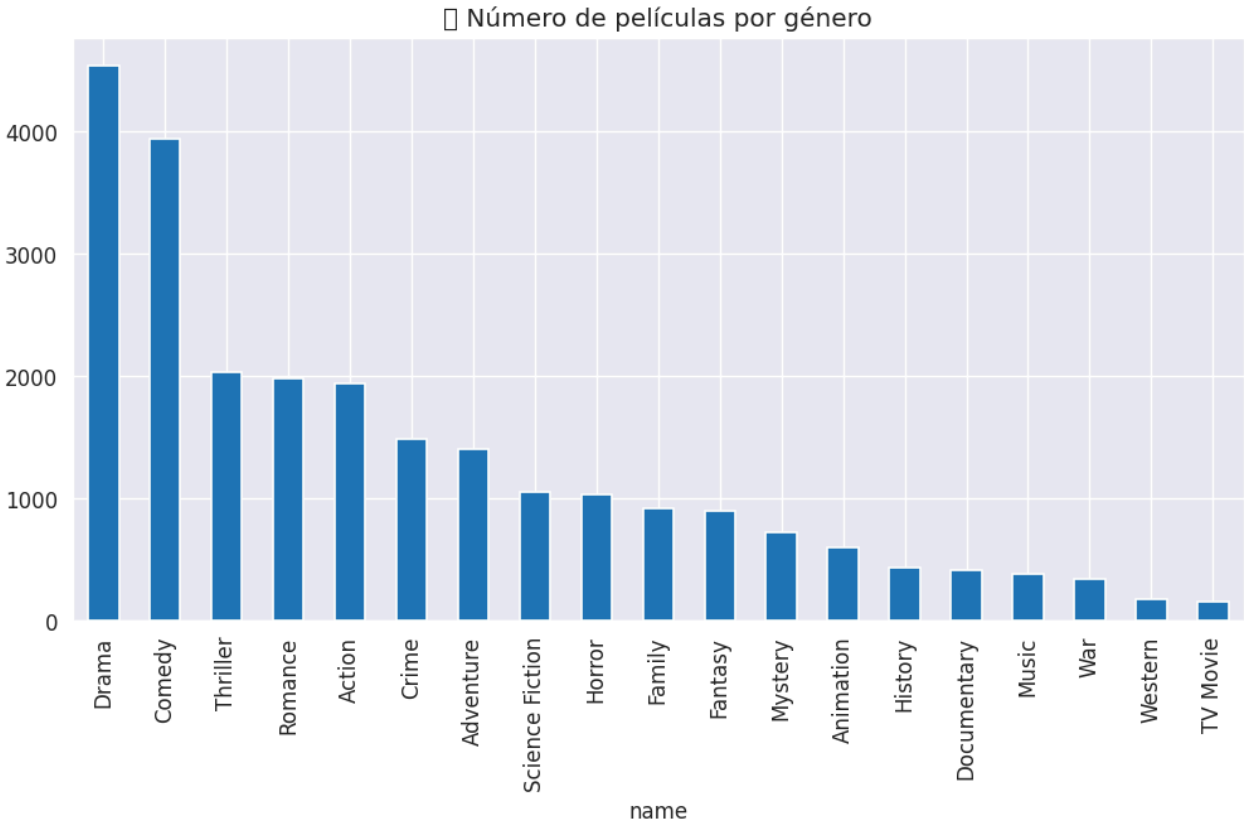
- Cantidad de películas por duración: se puede observar la distribución de la duración de las películas.



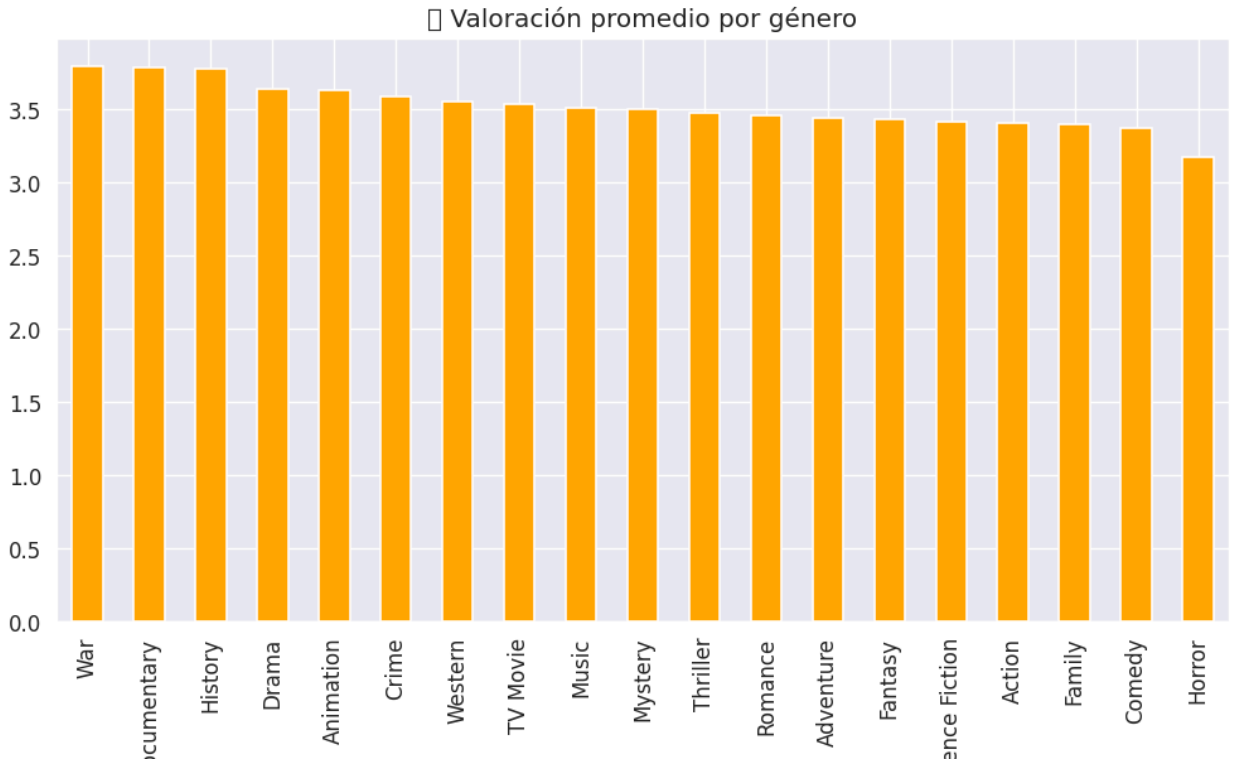
- Cantidad de ratings por puntuación: podemos la distribución de las notas que se han publicado



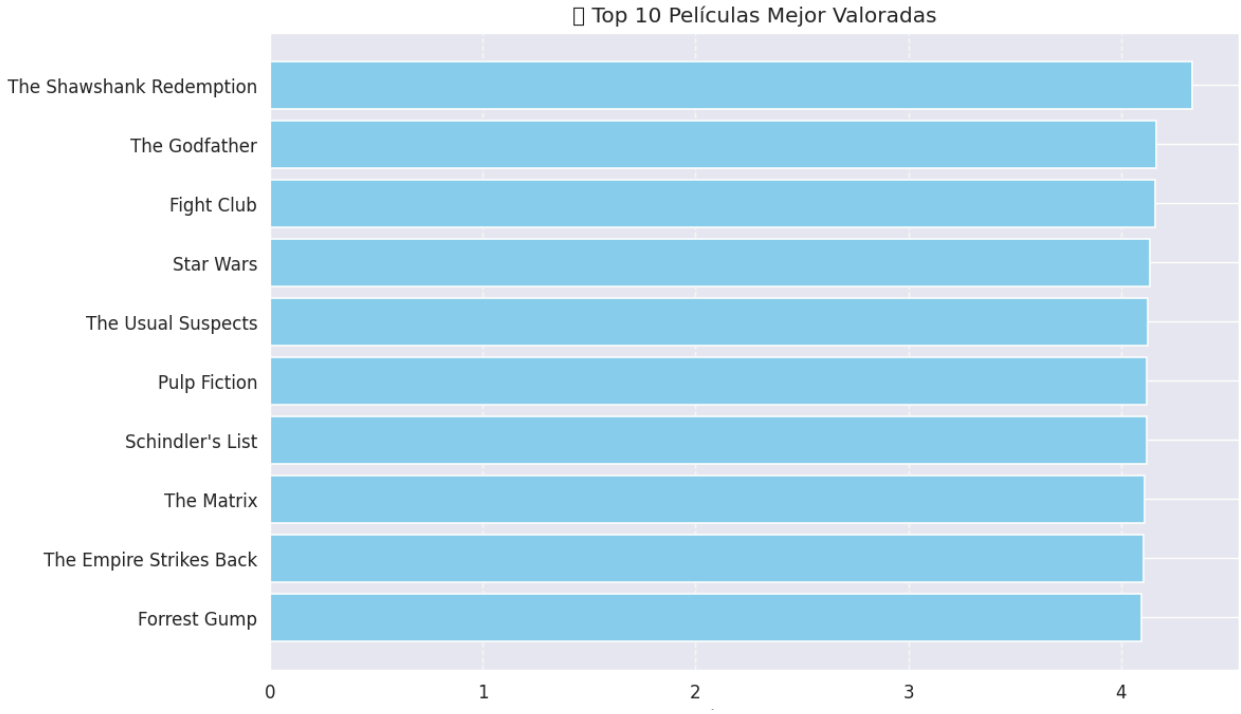
- Cantidad de películas por género: Conteo de las películas por cada uno de los géneros que existen en la base de datos.



- Valoración media por género: Según los ratings de los que nosotros disponemos, podemos saber cuales son los géneros mejor valorados.



- Películas mejor valoradas: Podemos saber las películas mejor valoradas, siempre teniendo en cuenta un mínimo de reseñas, por lo que hemos utilizado una fórmula como la de "imdb" para así poder filtrar aquellas películas que solo tengan una reseña que sea muy buena.



3. Preparación de los Datos

3.1 Seleccionar Datos

En este apartado exponemos y argumentamos la selección de los datos más relevantes e importantes para usar en nuestro modelo SVD y NeuralCF. Los datos son:

- Películas
- Views
- Users
- Ratings
- Géneros

A partir del análisis de estos datos tendremos en cuenta las películas de las que disponemos y también los usuarios, entrelazamos lo que ha visto cada usuario, incluyendo los ratings y géneros.

3.2 Limpieza de los Datos

Antes de pasar a la fase de creación del modelo es necesario realizar una limpieza y/o construcción de la base de datos para elevar o garantizar la calidad de los mismos. Esto a través de eliminación de valores innecesarios, valores duplicados, conversión de tipos de datos, imputación de datos faltantes, para poder entrenar de una mejor manera el modelo. Para este caso, se ha detectado que algunos valores como el tag y el Link de las películas tienen valores nulos, pero no usaremos estos datos en ninguno de nuestros modelos.

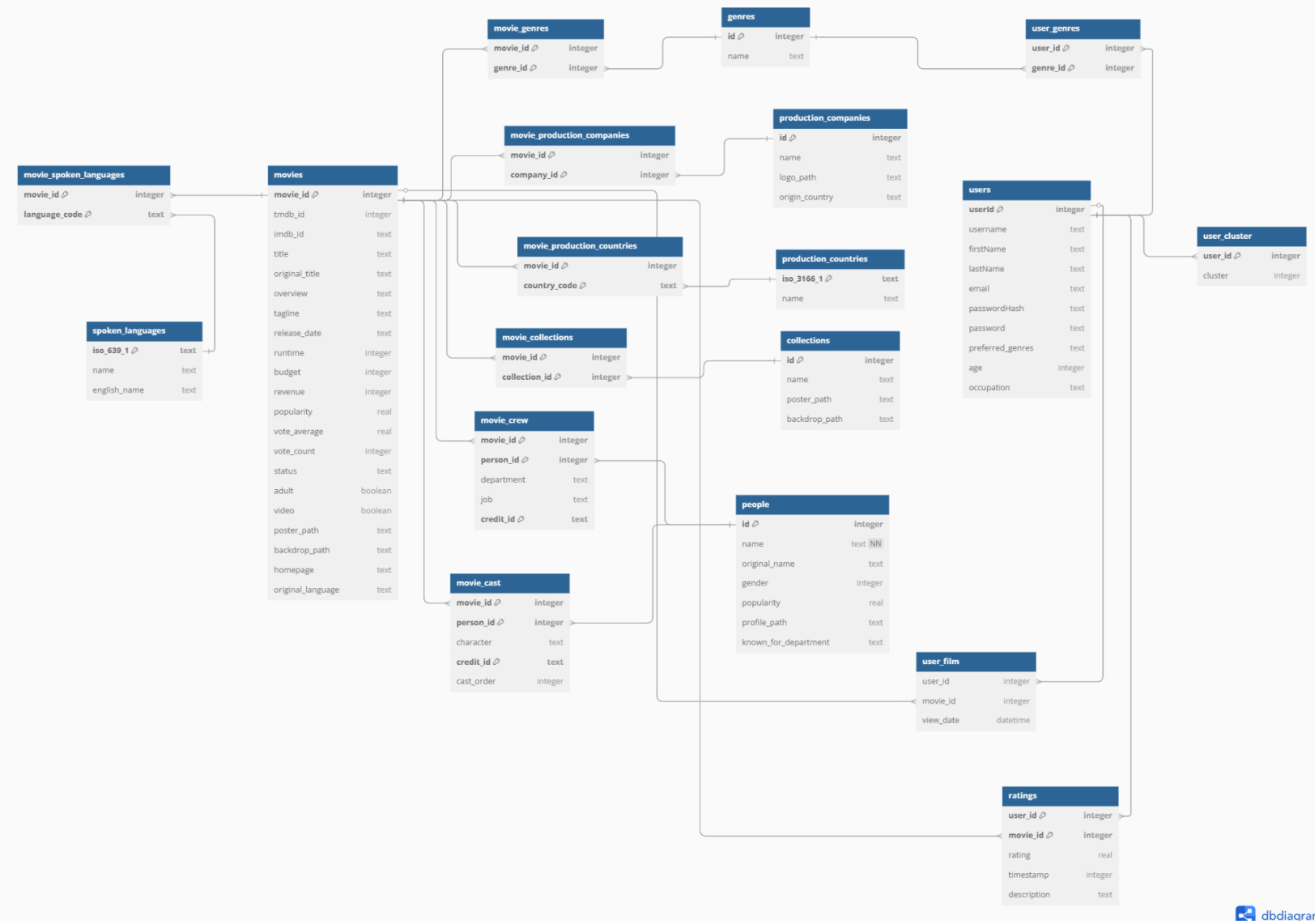
3.3 Estructura de datos

Esta base de datos, tmdb_movies.db, está diseñada para almacenar y gestionar una amplia gama de información relacionada con películas y sus usuarios.

En su núcleo, la BD detalla películas con campos como título, presupuesto, ingresos y fechas de estreno. Se extiende para clasificar películas por géneros y registrar sus compañías y países de producción, así como los idiomas hablados.

Además, incluye información sobre las personas involucradas en la industria cinematográfica, distinguiendo entre el elenco y el equipo de cada película.

Para los usuarios, la base de datos almacena perfiles personales y sus preferencias de género. Rastrea las películas vistas por cada usuario, sus calificaciones y los agrupa en clusters para análisis. En resumen, es una solución integral para explorar y gestionar datos de películas y la interacción de los usuarios con ellas.



3.4 Segmentación de clientes

Con el objetivo de identificar grupos de clientes con comportamientos similares y así mejorar las estrategias de marketing y personalización, se ha llevado a cabo un proceso de segmentación mediante clustering, utilizando la metodología RF (Recency, Frequency).

3.4.1 Metodología RF

La segmentación RF se basa en dos métricas clave del comportamiento del cliente:

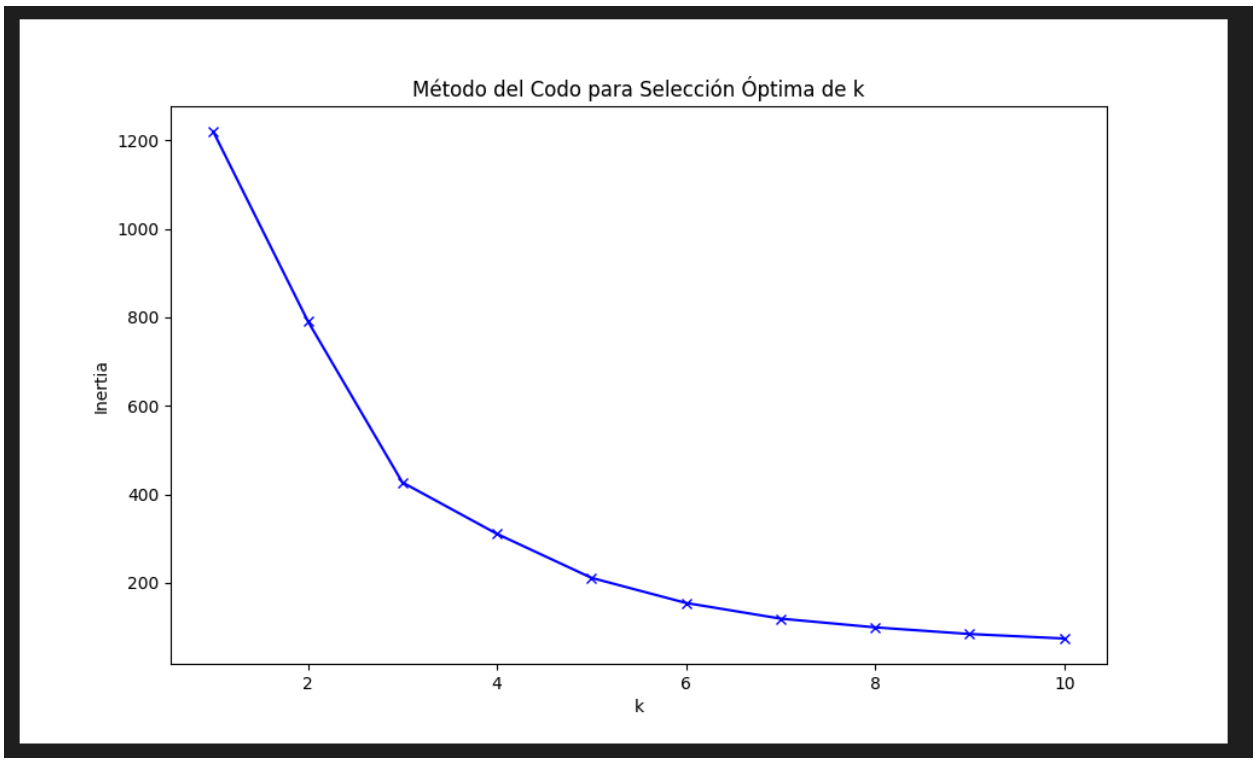
- Recency (Recencia): Tiempo transcurrido desde la última vista del usuario.
- Frequency (Frecuencia): Número total de vistas realizadas en un período determinado.

Estas variables se calcularon a partir del histórico de vistas y ratings, estos fueron normalizadas mediante StandardScaler para asegurar una escala uniforme en el análisis de clustering.

3.4.2 Selección del número óptimo de clusters

A la hora de elegir el número ideal de grupos (K) en el modelo de clustering, hemos empleado el método del codo.

A partir del gráfico resultante, se observó que el valor óptimo de K era 3, lo que indica la presencia de tres segmentos diferenciados de clientes.



3.4.3. Resultados de la Segmentación

Gracias a que hemos utilizado el método del codo, podemos determinar que un número óptimo de Clusters de clientes serían 3, cada uno con patrones distintos de recency, frequency. Esta segmentación nos ha permitido analizar los siguientes segmentos:

⚪ Cluster 0 – Usuarios Activos

- Recency: 5.35 días.
- Frequency: 285 películas.
- Descripción:
  Este grupo representa usuarios que han visto películas recientemente y mantienen un nivel de visualización constante. Son usuarios comprometidos con la plataforma y con un hábito de consumo regular.

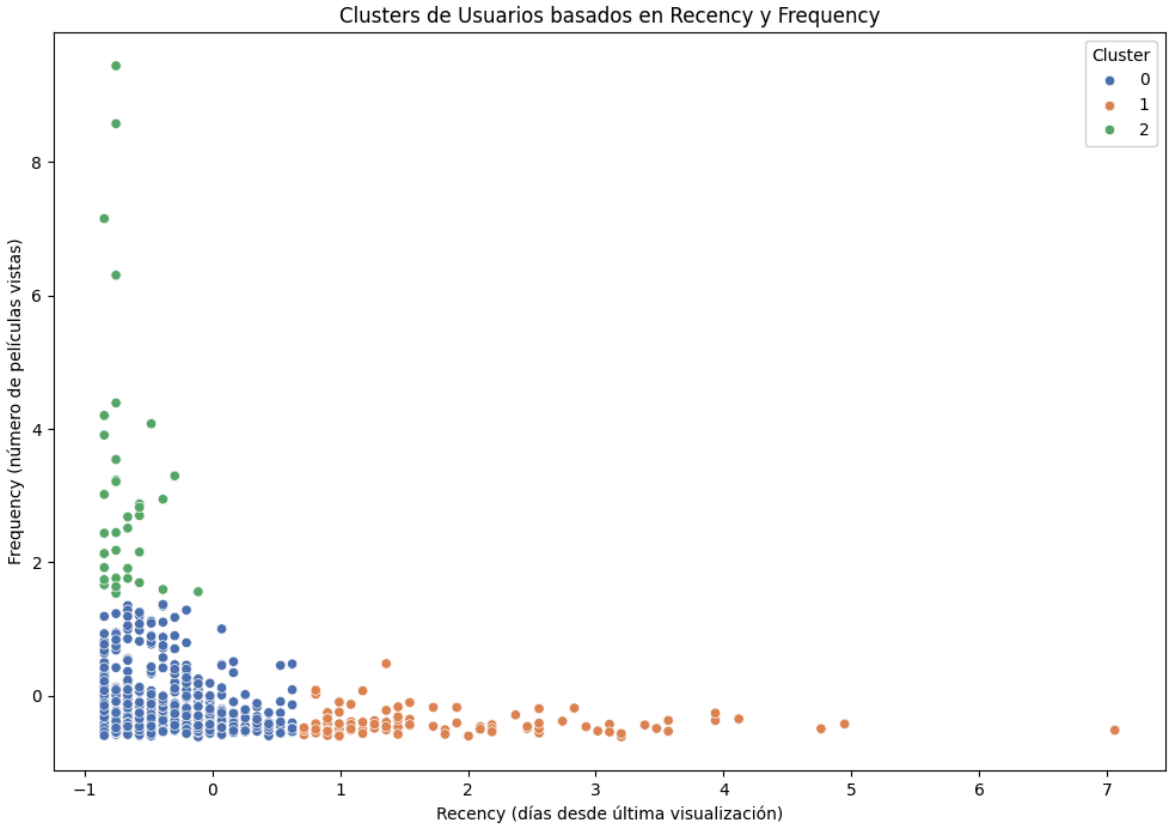
🟡 Cluster 1 – Usuarios poco activos

- Recency: 28.31 días
- Frequency: 215 películas.
- Descripción:
  Son usuarios que han reducido su actividad recientemente, aunque en el pasado han visualizado una cantidad considerable de películas. Tienen potencial de reactivación con recomendaciones personalizadas.

🟢 Cluster 2 – Usuarios muy activos

- Recency: 1.75 días
- Frequency: 1159.12 películas
- Descripción:
  Este es el grupo más valioso. Tienen una frecuencia de visualización extremadamente alta y su actividad es muy reciente. Son usuarios intensivos, muy comprometidos con la plataforma y clave para el lanzamiento de nuevas funcionalidades o contenidos exclusivos.

Esta segmentación constituye una herramienta poderosa para la toma de decisiones estratégicas centradas en el cliente, aumentando la efectividad de las acciones comerciales.



4. Arquitectura y Tecnologías Principales

La infraestructura de Streamly se basa en una arquitectura de microservicios, donde los modelos de recomendación y procesamiento de lenguaje natural se exponen a través de una API RESTful.

- Backend API (FastAPI): La columna vertebral del sistema es una API desarrollada con FastAPI. Este framework de Python fue elegido por su alta velocidad, facilidad de uso y la generación automática de documentación interactiva (Swagger UI), lo que facilita el consumo de los servicios desde las aplicaciones cliente. La API gestiona la autenticación de usuarios, el acceso a la base de datos y la interacción con los modelos de IA.

- Base de Datos (SQLite): La información de películas, usuarios, géneros, valoraciones y datos relacionados con TMDB se almacena en una base de datos SQLite. Esta elección proporciona una solución ligera y eficiente para la gestión de datos locales del proyecto.

- Frontend Web (Laravel): La plataforma web de Streamly ha sido desarrollada utilizando Laravel, un popular framework de PHP. Laravel facilita la creación de aplicaciones web robustas y escalables, gestionando la interfaz de usuario y comunicándose con la API de FastAPI para obtener las recomendaciones y otros datos.

- Frontend Móvil (Flutter): Para la aplicación móvil, se ha optado por Flutter, el kit de desarrollo de UI de Google. Flutter permite construir aplicaciones nativas compiladas para iOS y Android desde una única base de código, garantizando una experiencia de usuario consistente y de alto rendimiento en ambos sistemas operativos. La aplicación móvil también interactúa con la API de FastAPI para todas sus funcionalidades.

Esta arquitectura ha sido diseñada con un enfoque centrado en la experiencia del usuario, ofreciendo acceso a recomendaciones personalizadas tanto desde plataformas móviles como web. Al habilitar múltiples puntos de acceso, se garantiza una experiencia consistente y accesible, mientras que la modularidad del sistema permite una alta escalabilidad, favoreciendo la evolución y expansión del negocio.

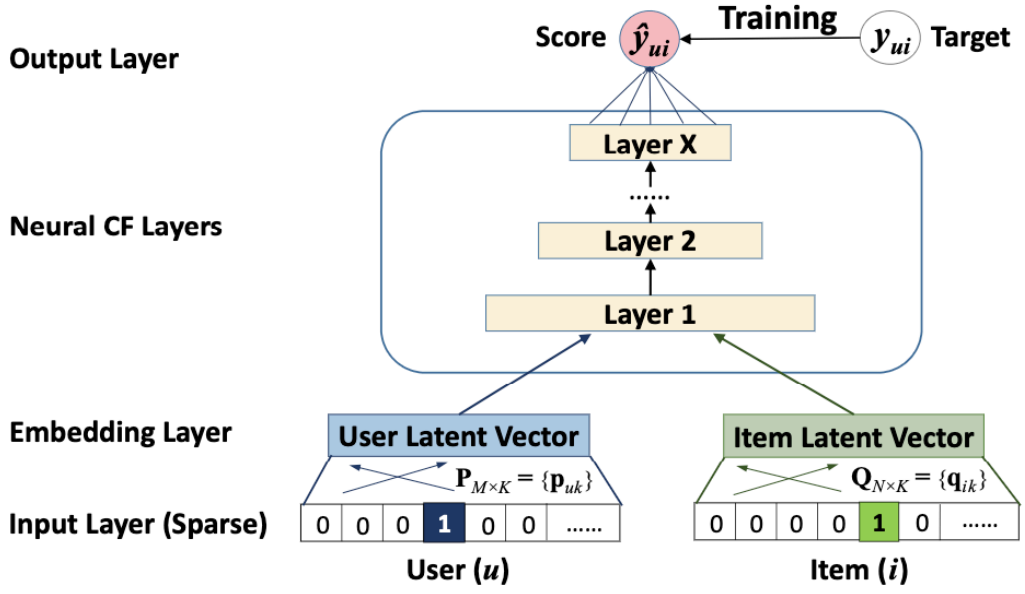
5. Sistema de Recomendación Multialgorítmico

Streamy implementa un enfoque multi algorítmico para ofrecer una variedad de recomendaciones personalizadas, utilizando diferentes modelos de inteligencia artificial según el contexto:

- Neural Collaborative Filtering (NeuralCF): Este modelo de redes neuronales se utiliza para las secciones de "Personalized Recs" y "Maybe you will like". A diferencia de los métodos colaborativos tradicionales que se basan en la factorización de matrices, NeuralCF utiliza redes neuronales profundas para aprender interacciones complejas entre usuarios y elementos. Su enfoque colaborativo permite identificar patrones de gustos entre usuarios y predecir las preferencias de un usuario por películas que aún no ha visto, basándose en el comportamiento de usuarios similares y en las características implícitas aprendidas por la red.

  La forma en la que funciona NeuralCF es mediante el uso de redes neuronales para modelar directamente las interacciones entre usuarios e ítems. En lugar de utilizar una descomposición lineal de la matriz de interacciones, como ocurre en la factorización matricial tradicional, NeuralCF aprende representaciones latentes no lineales a través de embeddings y capas ocultas entrenadas de forma supervisada.

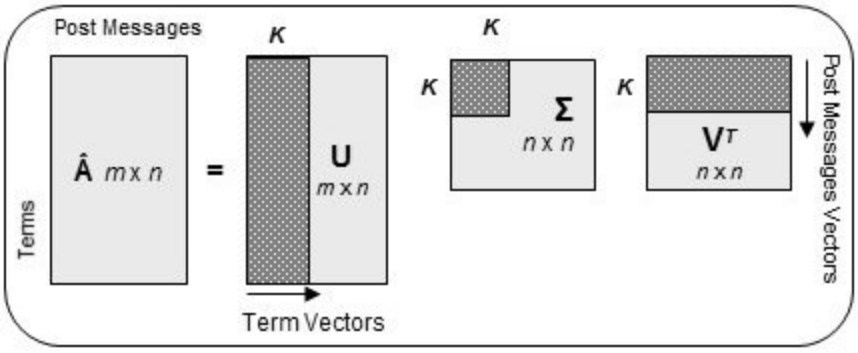
  Durante el entrenamiento, el modelo ajusta sus pesos internos minimizando una función de pérdida Esto le permite aprender tanto patrones generales de comportamiento como preferencias individuales.



- Truncated Singular Value Decomposition (TruncatedSVD): Este algoritmo de factorización de matrices es una técnica de reducción de dimensionalidad . Su objetivo es identificar y preservar las estructuras latentes más relevantes de los datos, eliminando el ruido y reduciendo la complejidad computacional. La diferencia que tiene comparado con el clásico modelo SVD es que, en este caso, se conservan únicamente los componentes principales, de tal manera que se obtiene una aproximación de menor rango de la matriz original. Esto nos permite representar únicamente las relaciones más significativas entre filas y columnas, lo que facilita la detección de patrones subyacentes sin procesar toda la información redundante o irrelevante.

  Esto resulta muy útil aplicado al caso concreto ya que no se dispone de una alta capacidad computacional y las relaciones que podemos obtener inferir provienen exclusivamente de las reseñas de los usuarios sobre las películas en el dataset.

  En este proyecto se ha utilizado para 1 caso en concreto:



    - "Users like you watched": Identifica películas populares entre usuarios con perfiles de gustos similares en el mismo cluster, ofreciendo recomendaciones basadas en la sabiduría colectiva. Encuentra usuarios con patrones de valoración similares de manera más rápida y robusta, incluso en conjuntos de datos dispersos.

- Singular Value Decomposition (SVD): Este algoritmo de factorización de matrices es una técnica clave en la reducción de dimensionalidad y descubrimiento de patrones latentes. Su objetivo es descomponer la matriz original en tres submatrices que capturan las relaciones más importantes entre usuarios y elementos (en este caso, géneros de películas). A través de esta descomposición, se pueden identificar factores latentes que representan tanto las preferencias de los usuarios como las características subyacentes de los géneros, este modelo requiere más potencia computacional debido a que puede trabajar con más factores que TruncatedSVD

    - Recomendador de Géneros: Se utiliza para sugerir películas basadas en los géneros preferidos que el usuario ha seleccionado en su perfil. **SVD (Singular Value Decomposition)** descompone la matriz de interacciones usuario-película en componentes latentes que capturan relaciones subyacentes entre usuarios y géneros. Al aplicar esta descomposición en una matriz densa, se obtiene una representación compacta que retiene la información más relevante, eliminando el ruido. Esto permite generar recomendaciones eficientes al identificar similitudes entre los perfiles de género de los usuarios y las características latentes de las películas.

- Recomendación Basada en Contenido (Red Neuronal): Dentro de las fichas de películas individuales, se utiliza un modelo de red neuronal que considera exclusivamente el contenido de la película (como sinopsis, géneros, actores, etc.) para recomendar títulos similares. Esta red aprende a mapear las características de las películas a un espacio de representación donde las películas con contenido similar están cerca, permitiendo a los usuarios profundizar en un estilo o temática específica que les interese.

- Métricas de Evaluación: Para asegurar la calidad y precisión de los modelos de recomendación, se utilizan diversas métricas:
    - Error del Modelo:
        - Mean Squared Error (MSE): Mide el promedio de los cuadrados de los errores, es decir, la diferencia cuadrada promedio entre los valores predichos por el modelo y los valores reales. Un MSE más bajo indica un mejor ajuste del modelo a los datos.
        - Root Mean Squared Error (RMSE): Es la raíz cuadrada del MSE. Proporciona una medida del error en las mismas unidades que la variable de respuesta, lo que lo hace más interpretable. Un RMSE bajo indica que el modelo predice los valores de manera precisa.
    - Calidad de la Recomendación: Precision@K, Recall@K, Normalized Discounted Cumulative Gain (NDCG@K), Mean Average Precision (MAP@K) y Hit Rate@K, que miden la relevancia y la efectividad de las recomendaciones en los primeros K resultados. Estas métricas son cruciales para evaluar cómo de bien el sistema recomienda elementos que el usuario realmente valoraría.
- Segmentación de Usuarios (Clustering): Los usuarios se dividen en 3 clusters, lo que permite una categorización y un tratamiento más específico de los perfiles de usuario, posiblemente para afinar las recomendaciones o personalizar la experiencia. Este agrupamiento se realiza basándose en patrones de comportamiento o preferencias de los usuarios.
- Recomendaciones Exploratorias y Gamificación: Para fomentar el descubrimiento de nuevos gustos, Streamly incluye un apartado de "Top 10 recomendaciones exploratorias". Este selecciona las películas menos recomendadas dentro del top de predicciones personalizadas, animando al usuario a salir de su zona de confort. Además, esta funcionalidad se ha gamificado mediante una "ruleta" en la

aplicación, añadiendo un elemento lúdico al proceso de descubrimiento.

5.1. Sistemas desechados

Para este proyecto se han probado distintos modelos que finalmente se han descartado, como el modelo de recomendación KNN. Esto se debe a que la matriz de usuarios y películas es demasiado dispersa para que este enfoque sea efectivo.

Tras probar con SVD y TruncatedSVD, se decidió abandonar la idea de utilizar KNN.

6. Chatbot de conversación sobre películas

Para poder ofrecer al usuario una manera interactiva de descubrir nuevos gustos o información adicional sobre las películas de su interés, se ha desarrollado un chatbot inteligente entrenado con la información de las películas del set de datos. Este chatbot es un sistema conversacional basado en lenguaje natural que permite a los usuarios obtener información y recomendaciones de películas a través de preguntas en español.

A través de un lenguaje accesible y natural, el usuario puede consultar desde detalles específicos sobre una película hasta obtener sugerencias personalizadas basadas en filtros como género, año, calificación o popularidad. El objetivo es simular una experiencia conversacional fluida y útil, acercando al usuario a una navegación más intuitiva, sin necesidad de explorar menús o aplicar filtros manuales.

Esta implementación se ha realizado mediante microsoft/DialoGPT-medium como modelo de generación de lenguaje. Este modelo recibe una consulta del usuario y realiza el siguiente procedimiento para proporcionar una respuesta:

1. Extracción del título de la película:

   Se analiza la consulta para identificar si el usuario hace referencia a una película específica, por ejemplo: “Dame información sobre Jumanji”. En ese caso, el chatbot busca en la base de datos y devuelve detalles como el año, sinopsis, géneros y calificación.

2. Filtrado de la información:

   Se utilizan expresiones regulares para extraer filtros de la consulta realizada por el usuario, como pueden ser:

   - Año de lanzamiento.
   - Géneros
   - Calificación máxima o mínima.

3. Respuesta del bot

Una vez extraída la información relevante, el sistema formatea los datos en una respuesta legible y estructurada. En el caso de una película concreta, se muestran detalles como el título, año de lanzamiento, géneros, valoración promedio, número de votos y descripción.

Si la consulta incluye filtros generales (como “las mejores películas de acción”), se devuelven hasta tres recomendaciones ordenadas por relevancia según los criterios extraídos. En caso de no encontrar coincidencias, el sistema sugiere ejemplos de consultas válidas para guiar al usuario. Este paso final garantiza que la interacción sea clara, útil y adaptada a la intención original del usuario.

Finalmente, se ha acotado el modelo para que solo hable de películas, ya que es el único propósito de su implementación, para ello se ha aplicado un filtro que hace que el bot, al no reconocer información de películas en la solicitud del usuario, devuelva un texto genérico, de tal forma que informa al usuario de que su petición no ha sido correctamente interpretada.

7. Procesamiento de Lenguaje Natural (PLN)

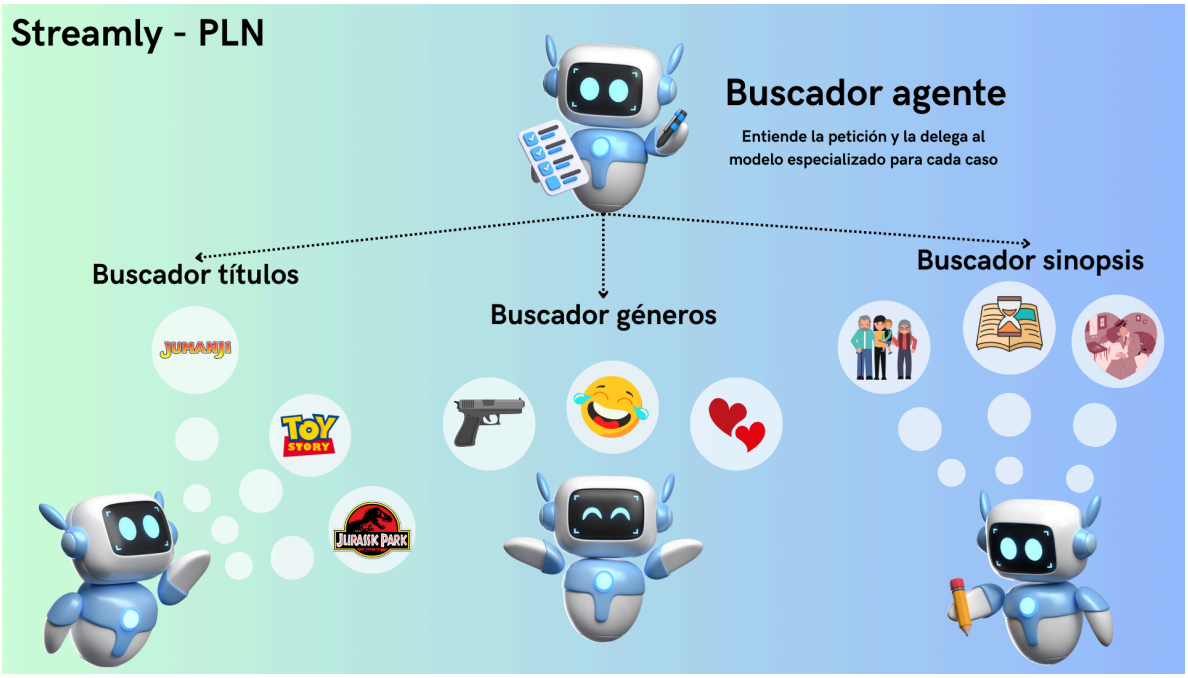
El sistema de PLN de Streamly es un componente crucial para la interacción del usuario, permitiendo búsquedas inteligentes y flexibles:

7.1 Arquitectura de Agentes:

El corazón del PLN es un sistema basado en la arquitectura de agentes. Un "agente de búsqueda" principal (buscador_agente.py) es el encargado de analizar la consulta del usuario y dirigirla al modelo especializado más adecuado. Este agente actúa como un orquestador inteligente de las diferentes capacidades de búsqueda.

El enfoque LLM tiene sus limitaciones: cada consulta trataría el texto de forma genérica (título, género o sinopsis) y no permitiría afinar el tratamiento específico de cada tipo de búsqueda.

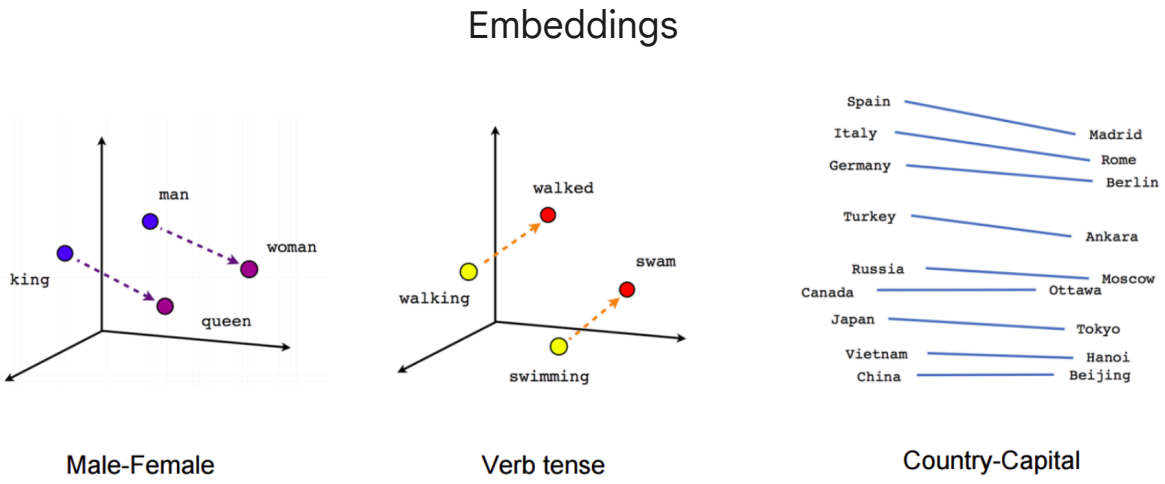
El enfoque de agente, en cambio, enruta la petición a modelos especializados, de esta manera podemos tratar cada caso según las necesidades de este, por ejemplo, las stopword(Las stopwords son palabras muy frecuentes y de escaso valor semántico (como “y”, “el” o “de”) que se eliminan para optimizar el análisis de texto.) que pueden tener un valor importante en el título, pueden no aportar demasiado a la hora de encontrar una película por su sinopsis.



7.2 Modelos Especializados:

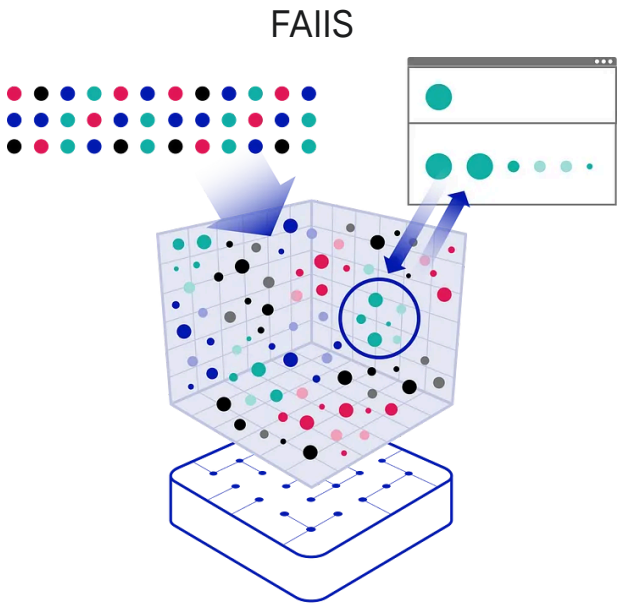
7.2.1 Búsqueda por Título:

Utiliza Sentence Transformers para generar embeddings. Los embeddings son representaciones vectoriales de las palabras, frases o documentos completos. En esencia, transforman el texto en un conjunto de números (coordenadas en un espacio multidimensional) donde las palabras o frases con significados similares están más cerca unas de otras. Por ejemplo, "katana", "espada" y "sable" tendrían embeddings muy cercanos, lo que permite al sistema entender la similitud conceptual incluso si el usuario se equivoca en el término exacto. Estos embeddings se utilizan junto con FAISS (Facebook AI Similarity Search) para realizar búsquedas de similitud eficientes. FAISS es una biblioteca para una búsqueda eficiente de similitud y agrupamiento de vectores densos. Permite buscar rápidamente los vectores más cercanos (y por lo tanto los títulos más similares) a una consulta dada en un conjunto de datos grande, lo que es crucial para la rapidez en la búsqueda de títulos. Esto permite encontrar películas incluso con errores tipográficos o variaciones en la consulta.



7.2.3 Búsqueda por Sinopsis:

Similar a la búsqueda por título, emplea Sentence Transformers para generar embeddings de las sinopsis de las películas y FAISS para encontrar títulos relevantes basándose en descripciones de la trama. El modelo paraphrase-multilingual-MiniLM-L12-v2 se utiliza específicamente para generar los embeddings. Este modelo está preentrenado para entender el significado de frases en múltiples idiomas, lo que permite el procesamiento de consultas en diversos idiomas y la comprensión de sinopsis sin importar el idioma original, facilitando la búsqueda por contenido semántico.



7.2.4 Búsqueda por Género

El sistema dispone de un motor dedicado que extrae de la consulta del usuario los géneros cinematográficos mencionados (por ejemplo, “horror”, “thriller”, “drama”) y los normaliza para garantizar una coincidencia precisa. A continuación, realiza la búsqueda en tres niveles de relevancia creciente:

1. Coincidencia exacta de géneros
   Se devuelven primero las películas cuyo conjunto de géneros coincide exactamente con los solicitados.

2. Coincidencias ampliadas
   Películas que incluyen todos los géneros buscados más uno o más géneros adicionales. De este modo, enriquecemos la búsqueda sin perder fidelidad al criterio original.

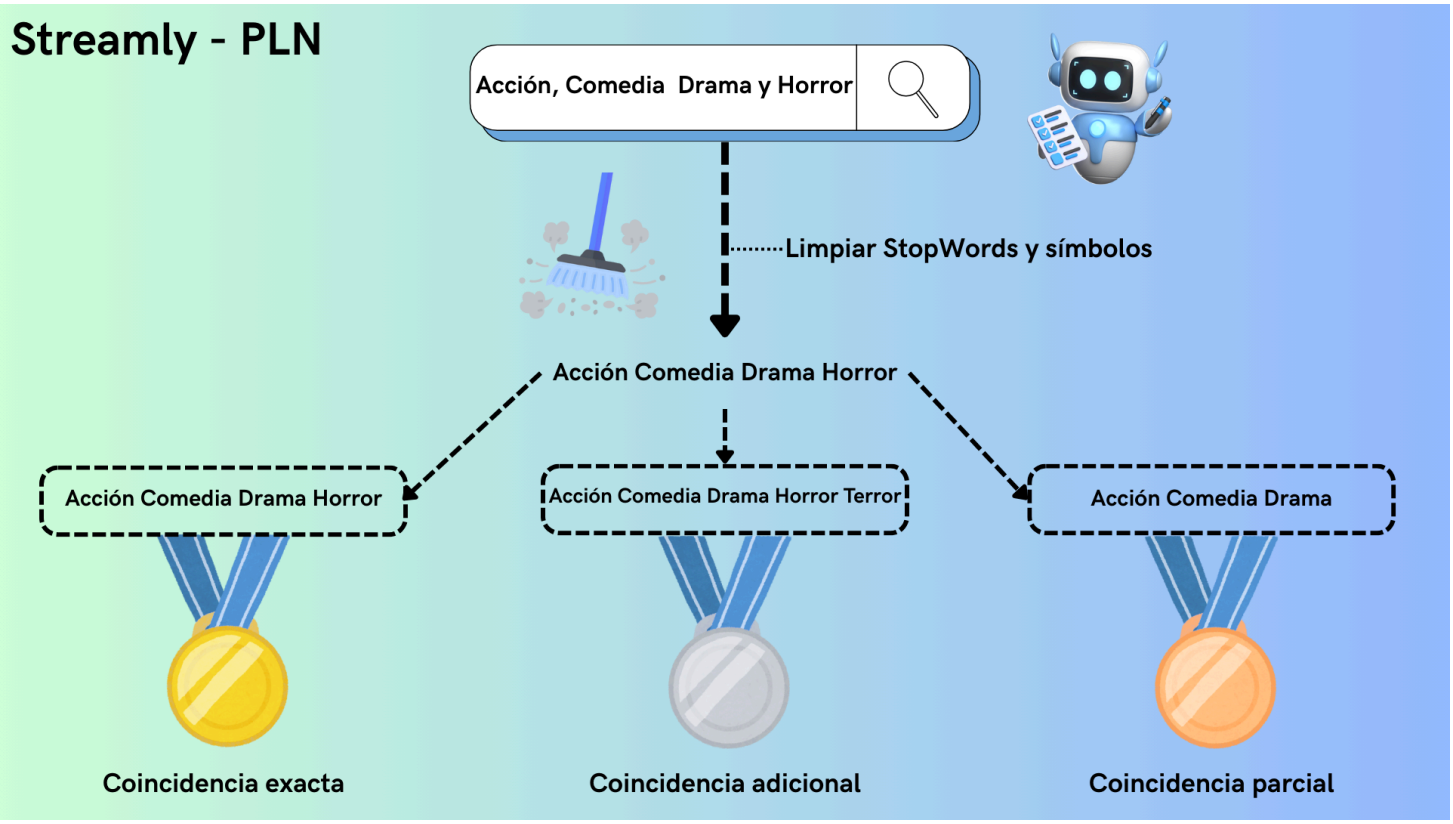
3. Coincidencia parcial
   Finalmente, aparecen títulos que comparten la mayoría de los géneros indicados, permitiendo que falte alguno de ellos o que se incluyan menos géneros. Esto amplía las opciones cuando no hay suficientes resultados de los niveles anteriores.

Por ejemplo, si un usuario busca “horror, thriller, drama, comedia”:

- Primero obtendrá cualquier película que tenga exactamente esos cuatro géneros.

- Después, aquellas que sumen un quinto género (p. ej. “family”) además de los cuatro iniciales.

- Por último, títulos que cumplan tres de los cuatro criterios, garantizando así una lista variada y ordenada según relevancia.



- Flexibilidad de Consulta: Gracias a esta arquitectura, el usuario puede buscar películas por título, género o sinopsis, en varios idiomas y con cierta tolerancia a errores de escritura, todo desde la misma caja de búsqueda, asegurando la accesibilidad para el usuario y a nosotros como desarrolladores nos permite desarrollar cada caso de forma específica, utilizando diferentes librerías o técnicas de tratado de texto o PLN en cada caso.

8. Evaluación Modelo SVD de recomendación de géneros

8.1 Evaluación de los Resultados

Para obtener el mejor rendimiento en los modelos, se ha realizado una evaluación en dos fases. En primer lugar, se llevó a cabo un análisis de resultados utilizando los siguientes indicadores:

- Reporte de Clasificación: Proporciona métricas como precisión (precisión), exhaustividad (recall) y F1-score para cada clase.

- Exactitud (Accuracy): Se utilizó como una métrica global para medir el porcentaje total de predicciones correctas.

Estas métricas nos permiten evaluar el rendimiento de los modelos en un escenario binario, es decir, cuando el resultado de una recomendación puede clasificarse como correcta o incorrecta (acierto o fallo). En este contexto, métricas como la precisión, el recall, el F1-score y la exactitud (accuracy) son útiles para medir la capacidad del modelo para distinguir entre recomendaciones relevantes y no relevantes.

Para el modelo SVD se utilizan métricas adicionales orientadas a sistemas de recomendación como **Precision@K, Recall@K, NDCG@K, MAP@K y Hit Rate@K**, que ofrecen una visión más detallada del rendimiento del modelo en rankings personalizados.

Para poder conseguir estas métricas se ha hecho una división del conjunto de datos en todos los modelos, el cual se ha hecho de la siguiente forma:

- 80% para entrenamiento.
- 20% para test.

Debido a la naturaleza de este sistema de recomendación, se le ha dado más **importancia** a las métricas no binarias. ya que con estas podemos saber la cantidad de recomendaciones relevantes.

Precisión

- Es la proporción de verdaderos positivos sobre el total de elementos clasificados como positivos. Mide cuántas de las predicciones positivas fueron realmente correctas.

Accuracy (Exactitud)

- Proporción de predicciones correctas sobre el total de predicciones

Recall

- Proporción de verdaderos positivos sobre el total de positivos reales.

F1 - Score

- Media armónica entre precisión y recall. Proporciona un equilibrio cuando ambos errores (falsos positivos y falsos negativos) son importantes.

MSE

- Promedio de los errores al cuadrado entre los valores reales y los predichos.

RMSE

- Raíz cuadrada del MSE, con la diferencia de que tiene la misma unidad que la variable objetivo.

En el SVD, estas son las métricas que se han añadido, además de una explicación.

Precision@K

- Proporción de elementos relevantes entre los primeros *K* elementos recomendados, evalúa la precisión de las recomendaciones.

Recall@K

- Proporción de elementos relevantes que aparecen dentro del top-*K* recomendaciones, respecto al total de relevantes disponibles, mide la cobertura de las recomendaciones. Dice cuánto del comportamiento real del usuario se ha logrado capturar.

NDCG@K (Normalized Discounted Cumulative Gain)

- Métrica que mide la calidad del orden en el que aparecen los ítems relevantes dentro del top-*K*. Premia que los ítems relevantes estén más arriba en la lista.

MAP@K (Mean Average Precision)

- El promedio de las precisiones acumuladas cada vez que aparece un ítem relevante dentro del top-*K*, proporciona una medida global de la calidad del ranking.

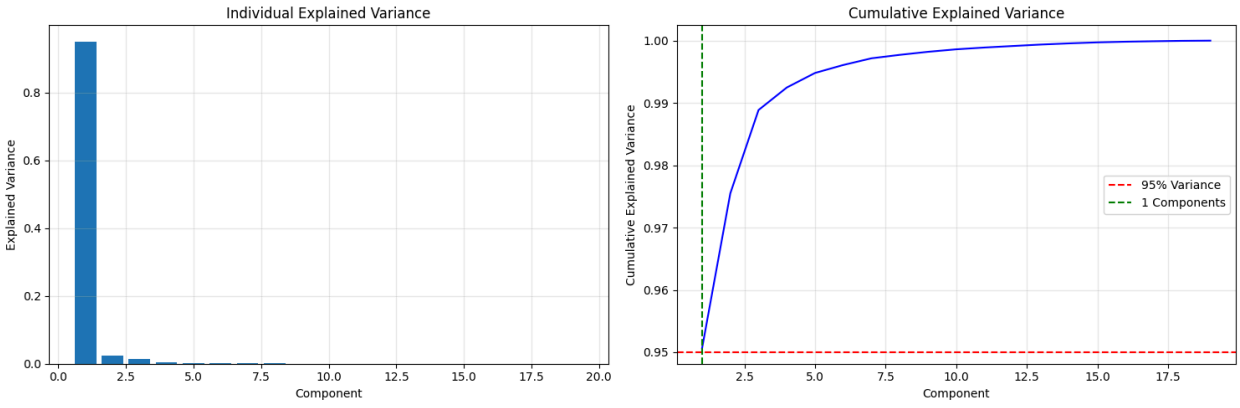
Hit Rate@K

- Proporción de usuarios para los cuales al menos un ítem relevante aparece en sus *K* recomendaciones, mide cuán frecuentemente tu sistema acierta al menos una vez.

8.2 Testing

SVD

En este primer modelo, además de las métricas descritas anteriormente, se han utilizado MSE y RMSE para evaluar el error del modelo. Asimismo, con el objetivo de determinar la cantidad óptima de componentes a utilizar, se han generado dos gráficas que nos permitan obtener la mejor cantidad de componentes.



Gracias a estas gráficas, se ha podido probar el valor óptimo de N que en este caso es 1.

Métrica	N = 1
MSE	0.01
RMSE	0.13
Precision	0.82
Accuracy	0.92
Recall	0.88
F1-Score	0.85
Precision@K	0.94
Recall@K	0.94
NDCG@K	0.89
MAP@K	0.84
Hit Rate@K	0.99

Gracias a estos resultados, podemos seleccionar un valor de N que represente un equilibrio óptimo entre el tiempo de computación y el error tolerado por el modelo. Se ha elegido N = 1,

ya que con esta cantidad de componentes se alcanza un corte del 90%-95% de la varianza en la gráfica. A partir de este punto, el modelo muestra mejoras marginales, y cualquier aumento en el número de componentes solo incrementa significativamente la carga de trabajo sin ofrecer mejoras sustanciales en el rendimiento.

Mientras más componentes pongamos en el modelo, las métricas aumentarán pero a su vez nos arriesgamos a sobreajuste y pérdida de capacidad de generalización.

8. Evaluación Modelo SVD de recomendación de películas

8.1 Evaluación de los Resultados

Al igual que en el caso del anterior modelo SVD, este segundo modelo ha sido evaluado mediante un enfoque en dos fases también. Se utilizaron los mismos indicadores de evaluación ya descritos (como precisión, exhaustividad, F1-score y exactitud) para valorar su rendimiento en términos de clasificación binaria.

Además, dado que se trata también de un sistema de recomendación, se emplearon las métricas propias del dominio, como Precision@K, Recall@K, NDCG@K, MAP@K y Hit Rate@K, con el objetivo de analizar la calidad de las recomendaciones generadas en rankings personalizados. Estas métricas permiten comparar de forma consistente el rendimiento entre ambos enfoques y valorar la capacidad del modelo para identificar elementos relevantes en las primeras posiciones de la lista recomendada.

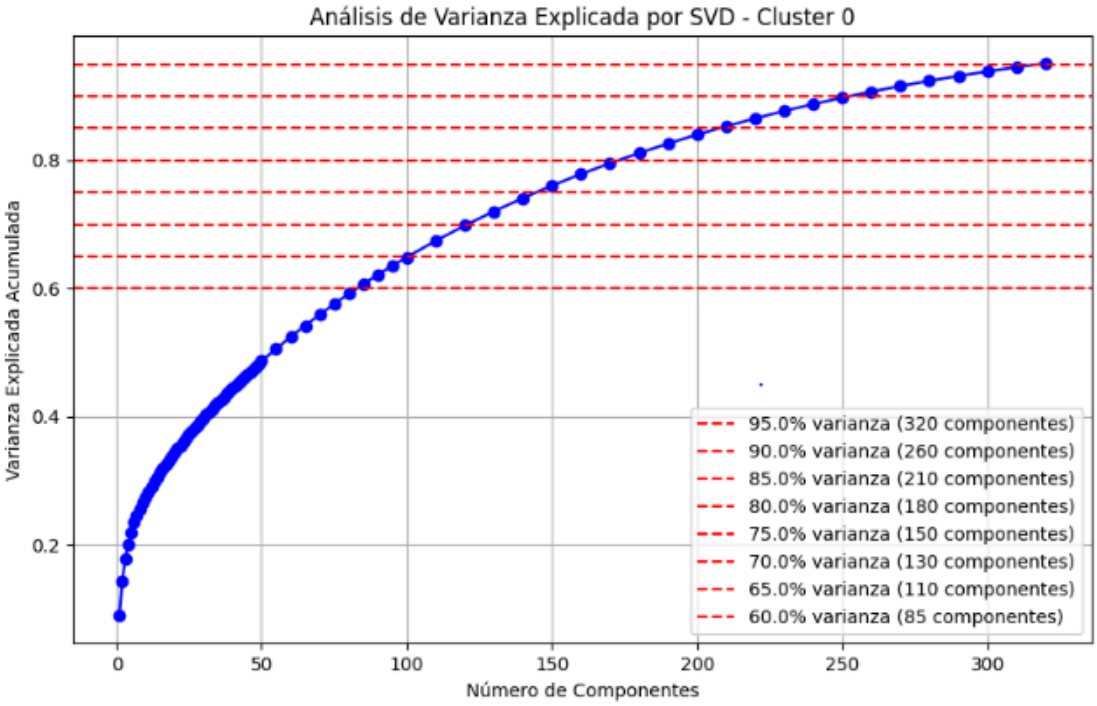
De la misma forma que el anterior modelo, se le ha dado más **importancia** a estas últimas métricas no binarias, ya que con ella podemos saber la cantidad de recomendaciones relevantes en un grupo.

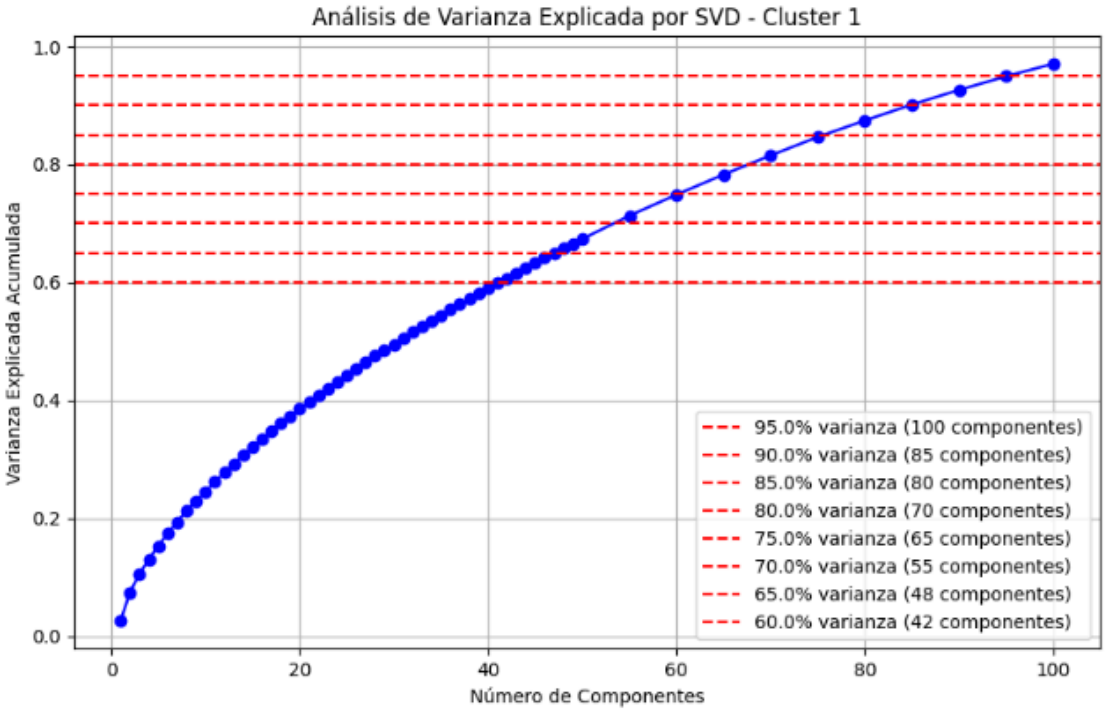
En cuanto a la distribución de datos para entrenar los modelos y conseguir las métricas, se ha dividido de la misma forma que el anterior modelo:

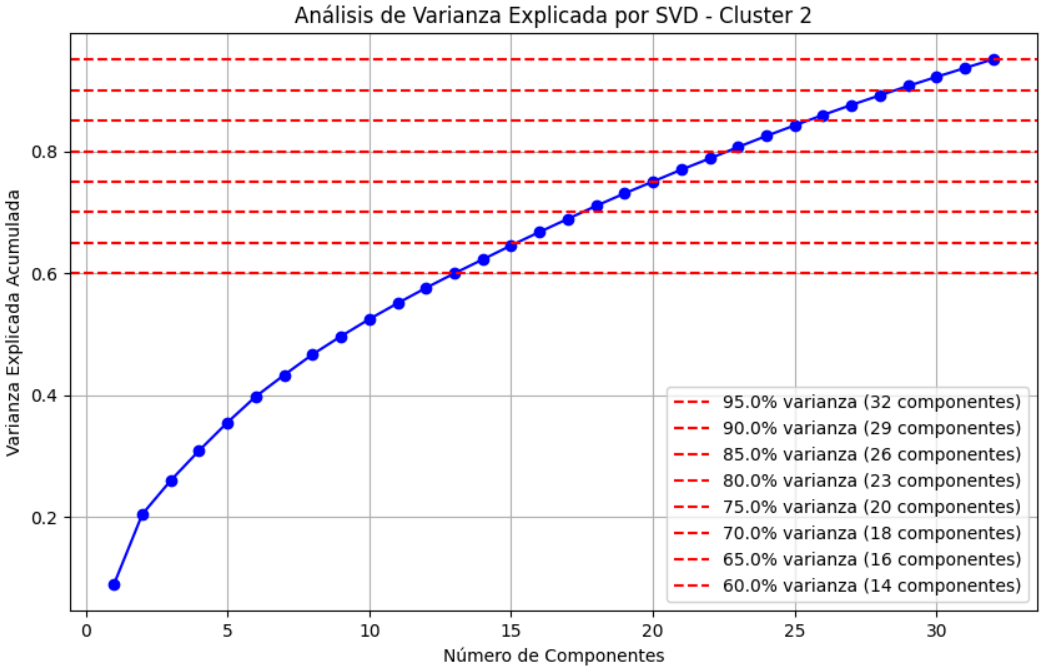
- 80% para entrenamiento.
- 20% para test.

8.2 Testing

En este primer modelo, además de las métricas descritas anteriormente, se han utilizado MSE y RMSE para evaluar el error del modelo. Asimismo, con el objetivo de determinar la cantidad óptima de componentes a utilizar, se han generado gráficas que permitan saber la mejor cantidad de componentes por Cluster.


Análisis de Varianza Explicada por SVD - Cluster 0
Varianza Explicada Acumulada
Número de Componentes
0.2
0.4
0.6
0.8
0
50
100
150
200
250
300
95.0% varianza (320 componentes)
90.0% varianza (260 componentes)
85.0% varianza (210 componentes)
80.0% varianza (180 componentes)
75.0% varianza (150 componentes)
70.0% varianza (130 componentes)
65.0% varianza (110 componentes)
60.0% varianza (85 componentes)


Análisis de Varianza Explicada por SVD - Cluster 1
Varianza Explicada Acumulada
Número de Componentes
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
95.0% varianza (100 componentes)
90.0% varianza (85 componentes)
85.0% varianza (80 componentes)
80.0% varianza (70 componentes)
75.0% varianza (65 componentes)
70.0% varianza (55 componentes)
65.0% varianza (48 componentes)
60.0% varianza (42 componentes)



Gracias a estas gráficas, se han podido probar diferentes valores de N (número de componentes). En base a los resultados y la varianza observada, se ha usado el valor óptimo del 95% en los tres clusters.

Métrica	Cluster 0	Cluster 1	Cluster 2
MSE	0.007	0.003	0.002
RMSE	0.08	0.06	0.021
Precision	0.51	0.51	0.52
Accuracy	0.50	0.51	0.52
Recall	0.51	0.50	0.51
F1-Score	0.60	0.67	0.67

Precision@K	0.90	0.94	0.99
Recall@K	0.70	0.80	0.99
NDCG@K	0.90	0.92	0.99
MAP@K	0.86	0.89	0.99
Hit Rate@K	0.99	0.99	0.99

Gracias a estas gráficas, podemos seleccionar un valor de N que represente un equilibrio óptimo entre el tiempo de computación y el error tolerado por el modelo. Se han elegido los valores de N óptimos según los componentes de las gráficas anteriores . A partir de este punto, el modelo muestra mejoras marginales, y cualquier aumento en el número de componentes solo incrementa significativamente la carga de trabajo sin ofrecer mejoras sustanciales en el rendimiento.

Mientras más componentes pongamos en el modelo, las métricas aumentarán pero a su vez nos arriesgamos a sobreajuste y pérdida de capacidad de generalización.

9. Evaluación Modelo Red Neuronal de recomendación de películas

9.1 Evaluación de los Resultados

Las métricas de este modelo se han enfocado de una forma parecida a los modelos de SVD.

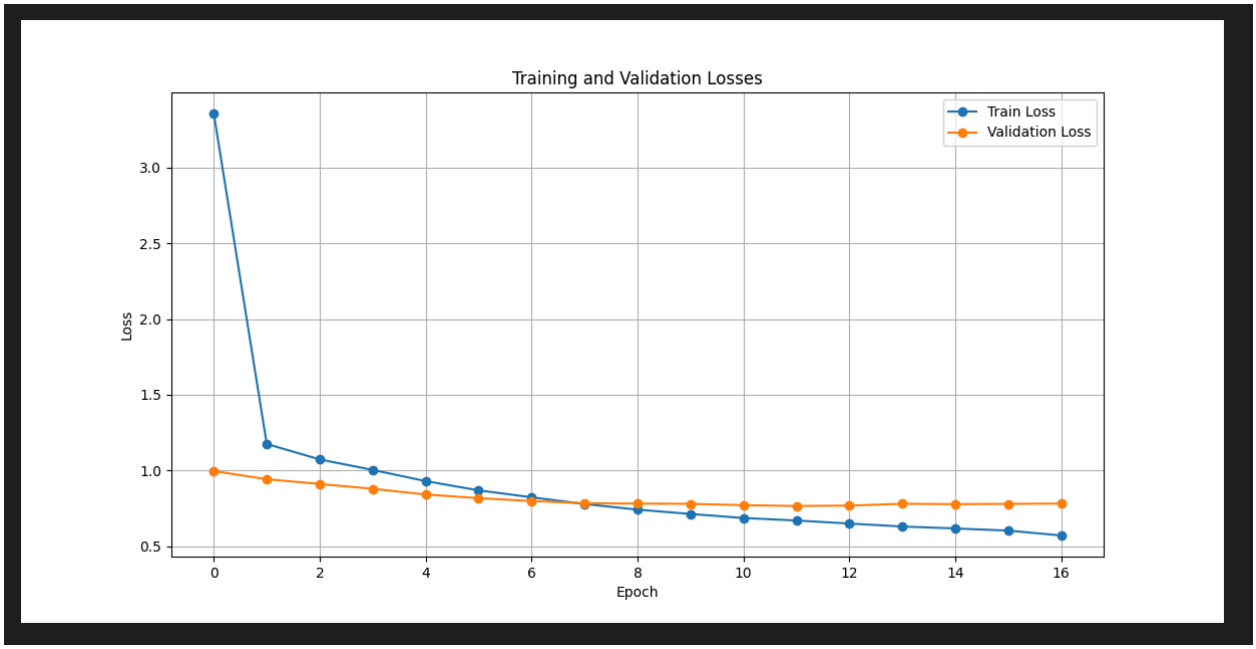
Debido al enfoque que se le ha dado, se han usado métricas como el MSe RMSE, Precision@K, Recall@K, NDCG@K, MAP@K y Hit Rate@K, además de que para saber cuándo se debe parar de entrenar el modelo, ya que se entrena por "épocas" siendo cada una de estas, una vuelta al aprendizaje del modelo, se ha utilizado un "early stopping", cuando el modelo detecta que en X cantidad de épocas no se aprende más, este para de entrenar, así no se pierde capacidad de computación, ya que son demasiadas vueltas sin ningún tipo de mejora.

En cuanto a la distribución de datos para entrenar los modelos y conseguir las métricas, se ha dividido de la misma forma que el anterior modelo:

- 80% para entrenamiento.
- 20% para test.

9.2 Testing

En este primer modelo, además de las métricas descritas anteriormente, se han utilizado MSE y RMSE para evaluar el error del modelo. Asimismo, con el objetivo de determinar la cantidad óptima de componentes a utilizar, se han generado dos gráficas que permiten comparar los resultados obtenidos.



Esta gráfica muestra la evolución de la función de pérdida (“loss”) del modelo a lo largo de las épocas. Nos permite identificar el punto en el que el modelo deja de mejorar, lo cual es útil para detener el entrenamiento a tiempo y evitar el sobreajuste. En este caso, observamos que el valor más bajo de pérdida se alcanza en la época 11. A partir de ahí, no se registran mejoras significativas durante las siguientes cinco épocas, hasta la época 16, momento en el que se decide finalizar el entrenamiento.

Métrica	Epoch = 5	Epoch = 11	Epoch = 20
Loss	0.84	0.74	0.78
MSE	0.95	0.67	0.60
RMSE	0.97	0.81	0.76
Precision	0.71	0.80	0.83
Accuracy	0.59	0.67	0.69
Recall	0.27	0.50	0.47
F1-Score	0.39	0.60	0.60
Precision@K	0.90	0.92	0.92
Recall@K	0.90	0.99	0.99
NDCG@K	0.72	0.89	0.85
MAP@K	0.90	0.94	0.98
Hit Rate@K	0.99	0.99	0.99

Como vemos, las métricas del entrenamiento con 11 épocas son muy similares al entrenamiento con 20, aun con el doble tiempo de entrenamiento, por lo que se ha decidido utilizar el número de épocas que hemos sacado con la gráfica anteriormente presentada, es decir, entrenamiento con 11 épocas.

10. Despliegue

Una vez entrenados los modelos, es posible desplegar un servicio en la nube que exponga la API desarrollada, lo que permite centralizar la lógica del proyecto y facilitar su consumo desde distintos clientes. Esta arquitectura permite integrar los modelos tanto en una aplicación web desarrollada con Laravel como en una aplicación móvil construida con Flutter, garantizando así una experiencia coherente y escalable en múltiples plataformas.

La aplicación móvil incorpora un robusto sistema de autenticación basado en JWT (JSON Web Tokens). Este sistema no solo permite a los usuarios registrar e iniciar sesión de forma segura, sino que también facilita el mantenimiento de la sesión de forma automática mediante tokens. Esto significa que, una vez que un usuario inicia sesión, la aplicación puede utilizar un token para verificar su identidad en futuras solicitudes a la API, evitando la necesidad de reintroducir credenciales repetidamente y mejorando significativamente la experiencia del usuario.

11. Propuestas de mejora

Como propuesta de mejora, se podría crear datos artificiales, ya que como la base de datos que se ha utilizado no tiene muchos usuarios, a la hora de crear un sistema de recomendación por SVD, la matriz está muy dispersa, incluso utilizando técnicas como el TruncatedSVD para poder reducir estas dimensiones.

Como propuestas de mejoras, se considera oportuno crear datos artificiales debido a la falta de usuarios, de esta manera podríamos acotar mucho mejor la matriz creada por nuestro recomendador hecho mediante TruncatedSVD.

Otra posible mejora sería adquirir, ya sea de manera sintética o mediante datos reales, diferentes tipos de interacciones de los usuarios con las películas, como por ejemplo las horas de visualización, los clics, las búsquedas realizadas o el tiempo que pasan navegando en ciertas secciones del servicio. Incorporar este tipo de datos permitiría enriquecer significativamente la matriz de interacciones, pasando de un enfoque puramente explícito (como valoraciones o reseñas) a uno más completo que incluye feedback implícito.

Esto resulta especialmente relevante al implementar sistemas de recomendación de contenido audiovisual, ya que es común que muchos usuarios no dejen reseñas o puntuaciones sobre los contenidos que consumen. Al integrar métricas de interacción más diversas, se puede mitigar la escasez de datos explícitos y mejorar la cobertura del sistema, permitiendo a los modelos aprender a partir de patrones de comportamiento más sutiles y frecuentes. En consecuencia, se logra una mayor robustez en las recomendaciones, así como una mejor personalización incluso en escenarios con poca participación activa del usuario.

12. Conclusiones

En cuanto al apartado de modelos, se ha logrado implementar con éxito una estructura de recomendación basada en la utilización de diversos algoritmos de recomendación con diversos fines, así como utilizar métodos de reducción de dimensionalidad como Truncated SVD para lograr reducir al máximo el coste computacional del proyecto.

Hemos conseguido muy buenas métricas de forma no binaria (Precision@K, Recall@K, NDCG@K, MAP@K y Hit Rate@K) en los conjuntos de prueba, lo que indica que el sistema es adecuado para su uso cotidiano. Estas métricas reflejan un buen rendimiento en tareas de recomendación y validan la utilidad del modelo en escenarios reales. Las métricas binarias no han alcanzado valores tan altos, los resultados obtenidos se consideran con un buen rendimiento, aunque no igual que las no binarias, se espera que el rendimiento de las métricas binarias mejore progresivamente a medida que se acumule más información a través del uso real. Una mayor cantidad de datos permitirá afinar las predicciones y optimizar el comportamiento del modelo.

En lo que respecta a la arquitectura del proyecto, se ha conseguido implementar con éxito un sistema robusto basado en modelos unificados, expuestos a través de una API REST. Esta estrategia permite que los distintos módulos de inteligencia artificial puedan ser consumidos fácilmente desde múltiples plataformas, tanto web como móviles.

Este enfoque modular y desacoplado no solo optimiza el mantenimiento y la escalabilidad del sistema, sino que también facilita su evolución futura, permitiendo incorporar nuevos modelos o funcionalidades sin afectar al resto de la infraestructura. Al centralizar la lógica del negocio en una única API y distribuirla eficientemente a través de diversos canales de acceso, se logra una integración fluida y una experiencia de usuario coherente, rápida y adaptable. Todo ello contribuye a sentar las bases de una arquitectura preparada para escalar en contextos reales de uso y alineada con los estándares actuales del desarrollo de software moderno.

13. Anexos

Repositorio backend, API y modelos de recomendación + ipynb análisis:

https://github.com/Miguelstucom/StreamlyBack

Repositorio aplicación móvil desarrollada con Flutter

https://github.com/Miguelstucom/streamly

Repositorio aplicación web desarrollada con Laravel

https://github.com/SJRobayo/streamly-web-app.git

Video de explicación de endpoints y demos del proyecto.

https://youtu.be/HAhMq-re9lo

Análisis DAFO Streamly

DAFO.pdf