Desarrollo de un sistema de recomendaciones personalizadas en la plataforma LMS KME360

Carolina Samacá Romero

Enrique José Peña Fajardo

Proyecto Aplicado para optar al título de
Magister en Ciencia de Datos

Dirigida por
Gloria Inés Álvarez

Codirigida por
Diego Luis Linares

Pontificia Universidad Javeriana
Facultad de Ingeniería y Ciencias
Maestría en Ciencias de Datos
Santiago de Cali
13 de julio de 2025

Pontificia Universidad JAVERIANA Cali

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

INTRODUCCIÓN

En la era digital, el aprendizaje continuo se ha consolidado como un pilar fundamental para el desarrollo profesional y la competitividad organizacional. Las plataformas de Gestión del Aprendizaje (*LMS*, por sus siglas en inglés, y *LXP*, *Learning eXperience Platforms*) han democratizado el acceso a la formación, ofreciendo catálogos de recursos educativos cada vez más amplios. Sin embargo, esta abundancia de contenido ha generado un desafío paradójico: la sobrecarga de información. Los usuarios, enfrentados a cientos o miles de opciones, a menudo experimentan dificultades para identificar los recursos que mejor se alinean con sus necesidades específicas, sus roles laborales y sus objetivos de aprendizaje, un fenómeno que puede llevar a la frustración, la desmotivación y, en última instancia, a bajas tasas de finalización de cursos [1].

El problema se agudiza en entornos de aprendizaje abiertos y diversos, como los ofrecidos por la Cámara de Comercio de Cali a través de *LXP KME360*. En este contexto, la diferencia entre los perfiles de usuario, que abarcan desde emprendedores noveles hasta empresarios con experiencia, hace que un enfoque de "talla única" para la presentación de contenidos sea particularmente ineficaz. Adicionalmente, los sistemas de recomendación habituales a menudo enfrentan el desafío del *arranque en frío*, donde son incapaces de ofrecer sugerencias pertinentes a usuarios nuevos por falta de un historial de interacciones [2].

Para abordar esta problemática, el presente proyecto se enfoca en el diseño, desarrollo e implementación de un sistema de recomendaciones inteligente y personalizado para la plataforma *LMS KME360*. El objetivo es transformar la experiencia del usuario de una navegación pasiva por un catálogo a un descubrimiento guiado y relevante. Para lograrlo, se implementa una metodología de ciencia de datos basada en *CRISP-DM*, utilizando un enfoque híbrido basado en modelos que combina técnicas de Procesamiento del Lenguaje Natural (*NLP*) y aprendizaje no supervisado.

La solución propuesta aprovecha el historial de interacciones de los usuarios para construir perfiles vectoriales que capturan sus intereses latentes. Posteriormente, se aplican algoritmos de *clustering* para segmentar a los usuarios en grupos con afinidades similares, permitiendo generar recomendaciones colaborativas de manera eficiente y escalable. Este documento detalla cada fase del proyecto, desde la comprensión del negocio y la preparación de los datos, hasta el entrenamiento y la evaluación cualitativa del modelo en un despliegue piloto. Se espera que el sistema desarrollado no solo mejore la satisfacción y el compromiso de los usuarios, sino que también ofrezca una solución robusta y adaptable que siente las bases para futuras mejoras en la personalización del aprendizaje en la plataforma *KME360*.

1. DEFINICIÓN DEL PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

En el actual entorno corporativo, la capacitación y el desarrollo continuo del personal son cruciales para la competitividad y la adaptación de las organizaciones. Las plataformas de Gestión del Aprendizaje (*Learning Management Systems* o *LMS* por sus siglas en inglés, o *LXP*, *Learning eXperience Platforms*) se han convertido en herramientas fundamentales para la distribución y el consumo de contenido formativo, ofreciendo acceso a una amplia gama de cursos estructurados y recursos complementarios.

Sin embargo, estas plataformas, incluida *KME360*, a menudo albergan una vasta cantidad de opciones, lo que puede abrumar a los usuarios al dificultar la identificación de contenido que se alinee con sus necesidades individuales, roles laborales específicos, intereses particulares y objetivos de desarrollo profesional. Esta sobrecarga de información (*Information overload*) [2] no solo genera frustración y pérdida de tiempo valioso para el empleado, sino que también puede mermar la efectividad de los programas de formación, afectando la motivación, el compromiso y, en última instancia, las tasas de finalización de cursos, impactando así el retorno de la inversión en capacitación para la empresa.

La plataforma *KME360*, en particular, atiende a una clientela diversa, que va desde empleados de empresas locales y nacionales, hasta Cámaras de Comercio que ofrecen cursos y recursos al público general. Cada una cuenta con catálogos de cursos y perfiles de empleados con características y necesidades de aprendizaje distintas. Un enfoque genérico para la presentación de contenidos resulta, por tanto, insuficiente para atender esta heterogeneidad de manera efectiva.

Adicionalmente, surge el desafío inherente a los sistemas de recomendación conocido como "arranque en frío" (*cold start*) [3]: ¿cómo se pueden ofrecer sugerencias pertinentes y valiosas a usuarios nuevos que carecen de un historial de interacciones previo en la plataforma, o cómo se introducen eficazmente cursos recién incorporados al catálogo que aún no han sido consumidos o valorados?

Actualmente, *KME360* podría no contar con un mecanismo dinámico, adaptativo y suficientemente granular que personalice proactivamente la oferta formativa a cada individuo, considerando su contexto y evolución dentro de la organización. Por ello, existe una necesidad clara de desarrollar e implementar un sistema de recomendación de contenidos y cursos que sea inteligente, personalizado y escalable para la plataforma *KME360*. Este sistema de recomendación debe considerar las características individuales

de cada empleado, a partir de los cursos que ya ha completado, para ofrecer alternativas que amplíen, mejoren, profundicen, expandan y complementen sus intereses, ofreciendo una experiencia de aprendizaje más significativa experiencia.

1.2. FORMULACIÓN DEL PROBLEMA

¿Cómo puede la plataforma de gestión de aprendizaje *KME360* proporcionar recomendaciones personalizadas de cursos y recursos que se adapten a las necesidades y preferencias individuales de los usuarios de uno de sus clientes?

2. OBJETIVOS DEL PROYECTO

2.1. OBJETIVO GENERAL

Desarrollar un sistema de recomendación en la plataforma de gestión de aprendizaje *KME360* para uno de sus clientes, que permita proporcionar a cada usuario sugerencias de cursos y recursos que se adapten a su perfil.

2.2. OBJETIVOS ESPECÍFICOS

- OE1: Identificar las características disponibles en el perfil de los usuarios que permitirán la construcción del sistema de recomendaciones.
- OE2: Diseñar la arquitectura del sistema de recomendación, asegurando su integración fluida con la plataforma existente.
- OE3: Seleccionar los algoritmos de aprendizaje automático más adecuados para proporcionar recomendaciones personalizadas.
- OE4: Entrenar los modelos seleccionados utilizando datos de la plataforma y evaluar su precisión y efectividad mediante pruebas con datos reales.
- OE5: Seleccionar el modelo con mayor precisión.
- OE6: Realizar un despliegue piloto del modelo en una instancia de prueba de la plataforma de gestión de aprendizaje *KME360*.

3. MARCO DE REFERENCIA Y ANTECEDENTES

Los sistemas de recomendación han emergido como herramientas cruciales para mitigar la sobrecarga de información en diversas plataformas digitales. Este fenómeno, donde la abundancia de opciones dificulta la toma de decisiones efectivas, es particularmente palpable en entornos de e-learning y plataformas de gestión del aprendizaje (*LMS*/*LXP*) [1]. En el contexto del e-learning, como se señalan en [4], estos sistemas son particularmente valiosos al ayudar a los estudiantes a descubrir contenidos y actividades de aprendizaje pertinentes, considerando su perfil, tipo de aprendizaje, conocimientos previos y nivel de experticia. El objetivo principal de estos sistemas es optimizar la experiencia de aprendizaje, fomentar la consecución de objetivos y aumentar la retención de usuarios, evitando tasas de deserción que suelen afectar a estas plataformas [5].

3.1. MARCO TEÓRICO

La Metodología *CRISP-DM* es el estándar *de facto* en la industria para proyectos que buscan extraer valor de los datos, y es el marco de referencia que da estructura a la metodología con la que se estructuró este proyecto. En la siguiente sección se ofrece una descripción general de dicha metodología. En las secciones que le siguen, se ofrece una descripción de los diversos enfoques para construir sistemas de recomendación, y se explican sus fortalezas y debilidades, al igual que sus retos y metodologías de evaluación pertinentes.

3.1.1. Metodología *CRISP-DM*

Para guiar el desarrollo de este proyecto de manera estructurada y sistemática, se ha adoptado la metodología *CRISP-DM* (*Cross-Industry Standard Process for Data Mining*). Esta metodología es el estándar de facto en la industria para la ejecución de proyectos de análisis de datos y minería de datos, proporcionando un marco de trabajo flexible pero robusto que organiza el proceso en un ciclo de vida iterativo [6] [7]. Su popularidad radica en que se enfoca no solo en los aspectos técnicos del modelado, sino también en la comprensión del contexto del negocio y la implementación práctica de la solución, asegurando que los resultados generen un valor real.

CRISP-DM descompone el ciclo de vida de un proyecto en seis fases principales [6], las cuales no son estrictamente secuenciales y a menudo implican retroceder a fases anteriores a medida que se obtienen nuevos conocimientos. La fase de Evaluación podría dar como resultado que deba volverse a fases anteriores para refinar su contenido, decisiones y resultados. A continuación, se presenta un diagrama de la metodología y se

describen brevemente cada una de estas fases.

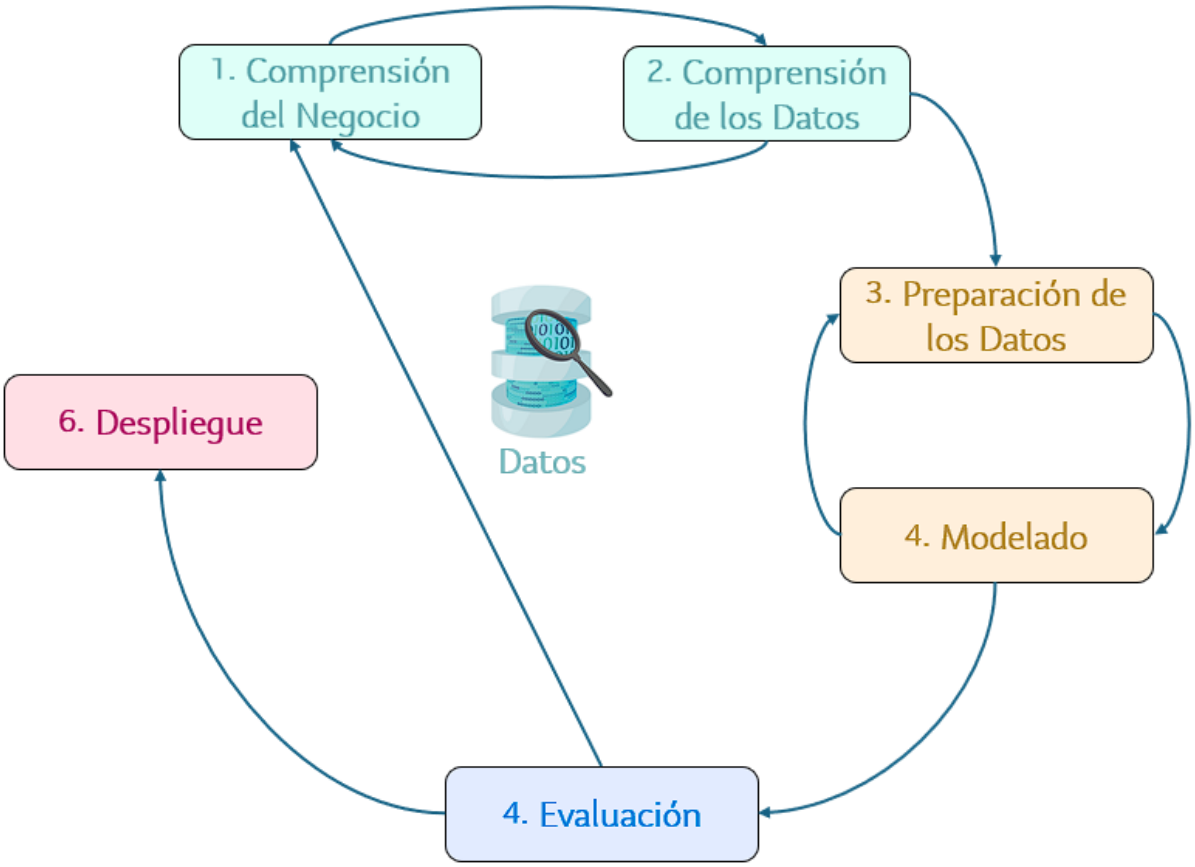


Figura 1: Ciclo de vida de la Metodología CRISP-DM

1. **Comprensión del Negocio (*Business Understanding*):** Esta fase inicial se centra en comprender los objetivos y requisitos del proyecto desde una perspectiva de negocio. Se traducen las metas organizacionales, como mejorar la experiencia del usuario o aumentar la retención, en un problema de minería de datos claramente definido y se establecen los criterios de éxito para evaluar el resultado final.
2. **Comprensión de los Datos (*Data Understanding*):** Inicia con la recolección de los datos disponibles y su exploración preliminar. El objetivo es obtener una primera impresión de los datos, identificar su calidad, descubrir patrones iniciales y formular hipótesis que guiarán las fases posteriores.
3. **Preparación de Datos (*Data Preparation*):** Esta es la fase que a menudo consume el mayor tiempo durante el ciclo. Cubre todas las actividades para construir el conjunto de datos final que alimentará los modelos. Las tareas típicas incluyen la

selección de variables relevantes, la limpieza de datos, el tratamiento de valores nulos, las inconsistencias o el ruido, la construcción de nuevas características y la transformación de los datos al formato requerido por las herramientas de modelado.

4. **Modelado (*Modelling*):** En esta fase se seleccionan y aplican diversas técnicas de modelado, como algoritmos de aprendizaje automático. Se calibran sus parámetros para optimizar el rendimiento y es común que se construyan y evalúen múltiples modelos para el mismo problema, permitiendo comparar sus fortalezas y debilidades.
5. **Evaluación (*Evaluation*):** Antes de proceder al despliegue, los modelos construidos se evalúan rigurosamente en el contexto de los objetivos de negocio definidos en la primera fase. Se verifica si el modelo cumple con los criterios de éxito y si los resultados son comprensibles y útiles. Esta fase puede revelar que es necesario volver a fases anteriores para refinar el enfoque.
6. **Despliegue (*Deployment*):** La fase final consiste en poner el modelo en un entorno operativo para su uso por parte de los usuarios finales. El despliegue puede variar desde la generación de un informe estático hasta la integración del modelo en un sistema de producción, como un servicio web. Esta fase también incluye la planificación del monitoreo y mantenimiento continuo del modelo para asegurar su rendimiento a lo largo del tiempo.

El carácter cíclico de *CRISP-DM* permite que el conocimiento adquirido en cada fase retroalimente las anteriores, promoviendo un proceso de mejora continua y asegurando que la solución final esté firmemente alineada con las necesidades del problema original. En este proyecto, la aplicación de esta metodología proporciona la estructura necesaria para pasar de los datos brutos de la plataforma *KME360* a un sistema de recomendación funcional.

3.1.2. Sistemas Basados en Contenido (*Content-based Filtering*):

Los sistemas de recomendación basados en contenido operan bajo el principio fundamental de recomendar ítems que son similares a aquellos que un usuario ha valorado positivamente o con los que ha interactuado en el pasado [2]. A diferencia de otros enfoques, su lógica se centra exclusivamente en las propiedades de los ítems y las preferencias de un único usuario, sin depender de los datos de otros miembros de la comunidad. Por ejemplo, si un usuario en una plataforma *LMS* ha completado con éxito un curso sobre "Introducción a Python", el sistema inferirá un interés en el tema de "programación" y podría recomendar cursos sobre "Análisis de Datos con Python" o

"Desarrollo Web con Django", ya que comparten características semánticas.

El proceso para generar estas recomendaciones se puede descomponer en tres fases principales:

1. **Análisis de Contenido y Representación de Ítems:** El primer paso consiste en transformar el contenido de los ítems, para nuestro caso, de los cursos, en un formato estructurado que la máquina pueda entender. Esto se logra mediante la extracción de características clave (metadatos) de cada curso, como palabras clave, descripciones, nombres, categorías o metodologías.
    - Una técnica clásica para esta tarea es *TF-IDF (Term Frequency-Inverse Document Frequency),* que representa cada curso como un vector basado en la importancia de las palabras que contiene. La importancia de una palabra aumenta con su frecuencia en un documento, pero se penaliza si es muy común en todo el corpus de documentos [2].
    - Un enfoque más avanzado, es el uso de *word embeddings* (como *Word2Vec* o *GloVe*). Estas técnicas aprenden representaciones vectoriales densas para las palabras a partir de grandes *corpus* de texto, capturando su significado semántico y relaciones contextuales. De este modo, cada curso puede ser representado por un vector numérico que encapsula la semántica de su contenido textual, permitiendo comparaciones más ricas que la simple coincidencia de palabras clave.
2. **Construcción del Perfil de Usuario (*Profile Learning*):** Una vez que los ítems están representados vectorialmente, se construye un perfil para cada usuario. Este perfil es, en esencia, un resumen de sus intereses, inferido a partir de su historial de interacciones explícitas (calificaciones positivas) o implícitas (cursos completados o tiempo de visualización). Un método común para crear este perfil es agregar los vectores de los ítems que le han gustado al usuario, por ejemplo, calculando el vector promedio de todos los cursos con los que ha interactuado [8]. Este vector de perfil resultante reside en el mismo espacio vectorial que los vectores de los ítems.
3. **Generación de Recomendaciones (*Filtering Component*):** Con los perfiles de usuario y las representaciones de los ítems, el paso final es encontrar coincidencias. El sistema compara el vector de perfil de un usuario con los vectores de todos los ítems del catálogo que el usuario aún no ha consumido. Se puede utilizar una métrica de similitud, como la similitud del coseno *(cosine similarity),* para calcular qué tan "cercanos" están los vectores de los ítems al vector del perfil del usuario. Finalmente, se presenta al usuario una lista clasificada o ranking de los ítems más similares, que constituyen las recomendaciones [2]. Esta lista, por supuesto, no puede incluir ítems

que el usuario ya haya consumido.

Los sistemas basados en contenido presentan varias ventajas significativas:

- Independencia del Usuario: Pueden generar recomendaciones personalizadas incluso para usuarios con gustos únicos, ya que no dependen de encontrar "vecinos" con preferencias similares.
- Explicabilidad: Las recomendaciones son transparentes y pueden justificarse fácilmente, al estilo: "Te recomendamos el curso X porque es similar al curso Y que ya tomaste". Esto lo podemos ver en algunas recomendaciones de la plataforma *Netflix,* que sugiere unas películas especificas porque se ha visto otra película.
- Manejo de Ítems Nuevos (*Ítem Cold Start*): Un nuevo curso puede ser recomendado inmediatamente después de ser añadido al catálogo, siempre que sus características puedan ser analizadas, sin importar si ha sido valorado por otros usuarios, lo cual es una ventaja crucial sobre el filtrado colaborativo [3].

Sin embargo, estos sistemas también enfrentan desafíos importantes, entre los que podemos mencionar los siguientes: [2]

- Análisis de Contenido Limitado: La calidad de las recomendaciones depende enteramente de la calidad de los metadatos y del contenido textual disponible. Puede ser difícil extraer características útiles de ítems con poco texto o metadatos pobres.
- Sobre especialización (*Overspecialization*): Tienden a recomendar ítems muy similares a los que el usuario ya conoce, lo que limita la capacidad de descubrir intereses nuevos o inesperados. A esta condición se le conoce como *Serendipia*.
- Problema de Usuario Nuevo (*User Cold Start*): El sistema es ineficaz para usuarios nuevos que no tienen un perfil de interacciones. Sin un historial, es imposible inferir sus preferencias y, por lo tanto, no se pueden generar recomendaciones personalizadas.

3.1.3. Sistemas de Filtrado Colaborativo (*Collaborative Filtering* - CF):

El Filtrado Colaborativo (CF) es uno de los enfoques más populares y exitosos en la historia de los sistemas de recomendación [2]. Su filosofía se basa en el principio de la "sabiduría de las multitudes" o la homofilia social: la idea de que si una persona A comparte intereses con una persona B, es probable que A también se interese en otros ítems que le gustan a B. A diferencia de los métodos basados en contenido, el CF no requiere analizar las

características de los ítems en sí mismos; en su lugar, se basa exclusivamente en los patrones de comportamiento e interacción de una comunidad de usuarios [3], por ejemplo, las calificaciones, las compras realizadas o las visualizaciones.

El pilar de este enfoque es la matriz de interacciones usuario-ítem, una representación de los datos donde las filas corresponden a los usuarios, las columnas a los ítems y las celdas contienen la interacción, por ejemplo, una calificación de 1 a 5, o un valor binario que indica si se ha completado o no un curso. Esta matriz suele ser extremadamente grande y dispersa (*sparse*), ya que un usuario promedio interactúa solo con una pequeña fracción del catálogo total de ítems [2]. El objetivo del CF es predecir los valores faltantes en esta matriz, es decir, estimar la preferencia de un usuario por un ítem con el que aún no ha interactuado. Los principales paradigmas dentro del Filtrado Colaborativo son los siguientes:

- **Filtrado Colaborativo Basado en Memoria (*Memory-Based CF*):**

Estos métodos, también conocidos como algoritmos de vecinos más cercanos (*k-nearest neighbors* o k-NN), operan directamente sobre la matriz completa de interacciones usuario-ítem para calcular similitudes. Se dividen en dos categorías principales: [2]

- Basado en Usuarios (*User-Based CF*): El algoritmo identifica un conjunto de "vecinos" (otros usuarios) que tienen perfiles de interacción muy similares al usuario activo. La similitud se mide comúnmente con métricas como la Correlación de Pearson o la Similitud del Coseno. Una vez que se encuentra el vecindario o clúster, la predicción para un ítem no visto se calcula como un promedio ponderado de las calificaciones que los vecinos han dado a ese ítem.
- Basado en Ítems (*Item-Based CF*): En lugar de encontrar usuarios similares, este enfoque busca ítems similares. La similitud entre dos ítems se calcula basándose en cómo han sido calificados por el mismo conjunto de usuarios. Por ejemplo, los cursos que tienden a ser completados o valorados positivamente por los mismos usuarios se consideran similares. La predicción para un ítem se basa en las calificaciones que el usuario activo ha dado a ítems similares [3]. Este enfoque a menudo ofrece mejor escalabilidad y resultados más estables que el basado en usuarios.

- **Filtrado Colaborativo Basado en Modelos (*Model-Based* CF):**

En lugar de operar sobre toda la matriz de datos, estos métodos utilizan técnicas *de machine learning* para aprender un modelo compacto a partir de los patrones de interacción. Este modelo luego se utiliza para realizar las predicciones, lo que suele ser mucho más rápido y escalable.

- Factorización de Matrices (*Matrix Factorization*): Esta familia de técnicas, que incluye métodos como la Descomposición en Valores Singulares (SVD) y el *Alternating Least Squares (*ALS), descompone la matriz dispersa usuario-ítem en dos matrices de factores latentes de menor dimensión: una para los usuarios y otra para los ítems. Estas matrices representan características latentes (o implícitas) de usuarios e ítems. La predicción de una calificación se obtiene calculando el producto punto de los vectores de factores latentes del usuario y del ítem correspondiente [8].
- *Clustering*: Este enfoque agrupa a los usuarios en clústeres basados en la similitud de sus perfiles de interacción. La predicción para un usuario activo se realiza promediando las calificaciones de los usuarios que pertenecen al mismo clúster. Algoritmos como *K-Means*, son un ejemplo claro de este paradigma. Aquí, el modelo es la asignación de cada usuario a un clúster específico.

Entre las ventajas del Filtrado Colaborativo están las siguientes:

- Independencia del Contenido: No requiere un análisis profundo de los ítems, lo que le permite recomendar ítems complejos como videos y audios, sin necesidad de metadatos ricos.
- Capacidad de Serendipia: Puede descubrir intereses nuevos y diversos para un usuario al recomendar ítems populares en otros "grupos de gusto", algo que los sistemas basados en contenido difícilmente logran [2].

Sin embargo, al igual que otros sistemas, enfrenta desafíos importantes como los siguientes:

- Problema de Usuario Nuevo (*User Cold Start*): Es ineficaz para usuarios nuevos, ya que no se pueden encontrar "vecinos" si no existe un historial de interacciones.
- Problema de Ítem Nuevo (*Item Cold Start*): Un ítem nuevo no puede ser recomendado hasta que haya sido calificado por un número suficiente de usuarios para ser incluido en los cálculos de similitud o en el entrenamiento del modelo.
- Escalabilidad y *Sparsity*: Como se mencionó, la dispersión de la matriz usuario-ítem puede dificultar encontrar usuarios o ítems con suficientes interacciones en común. Además, los métodos basados en memoria pueden ser computacionalmente costosos a medida que crecen los datos [8].

3.1.4. Sistemas Basados en Conocimiento (*Knowledge-based RS)*

Los sistemas de recomendación basados en conocimiento van un paso más allá de los patrones de interacción (Filtrado Colaborativo) y de las características superficiales de los ítems (Basado en Contenido). Este enfoque utiliza conocimiento explícito sobre el dominio, los ítems, y las necesidades del usuario para inferir recomendaciones. Se fundamenta en la creación de una base de conocimiento que contiene reglas, restricciones y relaciones complejas, permitiendo generar recomendaciones que no solo son precisas, sino también altamente justificables [2].

Estos sistemas son especialmente útiles en dominios donde el número de interacciones, como por ejemplo, las calificaciones, es bajo o inexistente, como en la recomendación de productos complejos y costosos, como automóviles o servicios financieros, o en contextos educativos especializados, donde se requiere una comprensión profunda de los prerrequisitos y objetivos de aprendizaje [9]. Su lógica no se basa en el "gusto" sino en la "adecuación funcional" de un ítem a los requisitos explícitos del usuario. El proceso se apoya en un motor de inferencia que utiliza la base de conocimiento para encontrar soluciones.

Dentro de este paradigma, se pueden distinguir varios subtipos: [9]

- Sistemas Basados en Restricciones (Constraint-based): El usuario especifica un conjunto de requisitos o restricciones explícitas, por ejemplo "necesito un curso sobre marketing digital, de nivel principiante, que dure menos de 10 horas". El sistema utiliza estas restricciones para filtrar el catálogo de ítems y devolver aquellos que cumplen con todos los criterios. Si no se encuentra una solución exacta, el sistema puede proponer ajustes a las restricciones para encontrar ítems cercanos.
- Sistemas Basados en Casos (Case-based): Estos sistemas recomiendan ítems basándose en la similitud con casos o problemas resueltos en el pasado. Se busca un ítem que haya satisfecho necesidades similares a las del usuario actual. Por ejemplo, si se sabe que un usuario con un perfil de habilidades "X" se benefició del curso "Y", se podría recomendar el mismo curso a otro usuario con un perfil "X" similar.
- Sistemas Basados en Demografía: Este es un subenfoque que agrupa a los usuarios basándose en sus atributos demográficos explícitos, como edad, género, ubicación, nivel educativo o rol laboral. Las recomendaciones se generan a partir de lo que es popular o relevante dentro de un grupo demográfico específico [3]. Aunque es

sencillo de implementar, puede llevar a estereotipos y a recomendaciones poco personalizadas a nivel individual.

- Sistemas Basados en Reglas y Ontologías: Estos sistemas utilizan un conjunto de reglas predefinidas, a menudo estructuradas en una ontología que modela el dominio. Las reglas pueden ser del tipo "SI un usuario ha completado el curso A, *entonces* recomendar el curso B (prerrequisito)" o "SI el rol del usuario es 'Ventas', *entonces* mostrar cursos de 'Negociación'". Este enfoque, mencionado por [10] y explorado en trabajos como el de [11], permite codificar conocimiento experto y políticas de negocio directamente en el motor de recomendación.

Las principales ventajas de los sistemas basados en conocimiento son:

- **Solución al Problema de Arranque en Frío (*Cold Start*):** Son efectivos desde el primer momento, ya que no dependen de un historial de interacciones. Pueden recomendar ítems nuevos y atender a usuarios nuevos siempre que se puedan especificar sus requisitos.
- **No producen sobre especialización:** Evitan la tendencia de recomendar únicamente ítems similares a los ya consumidos, ya que su lógica se basa en la satisfacción de necesidades explícitas.
- **Alta Explicabilidad y Confianza:** Las recomendaciones son transparentes, ya que se pueden justificar con las reglas o restricciones que las generaron, lo que aumenta la confianza del usuario.

A pesar de sus ventajas en contraposición a los sistemas anteriores, también presentan desafíos importantes como los siguientes:

- **Costo de Adquisición de Conocimiento:** El principal cuello de botella es la creación y el mantenimiento de la base de conocimiento (reglas, ontologías, catálogos de características), que suele ser un proceso manual, costoso y que requiere de expertos en el dominio [2].
- **Menor Serendipia:** Aunque evitan la sobre especialización, no suelen promover el descubrimiento de ítems inesperados, ya que se limitan a satisfacer los requisitos explícitos del usuario.
- **Rigidez:** Si la base de conocimiento no es lo suficientemente flexible, el sistema puede ser incapaz de encontrar soluciones si el usuario define restricciones demasiado estrictas.

3.1.5. Filtrado Basado en Reglas (Rules-based Filtering)

Los sistemas de recomendación basados en reglas constituyen una categoría pragmática dentro de los sistemas basados en conocimiento. Este enfoque utiliza un conjunto de reglas lógicas predefinidas, generalmente en formato "SI-ENTONCES" (IF-THEN), para filtrar y priorizar los ítems que se presentan al usuario [2]. Estas reglas pueden ser definidas por los administradores de la plataforma, expertos en el dominio, o incluso por los propios usuarios, y permiten codificar conocimiento heurístico y políticas de negocio directamente en el motor de recomendación.

Como menciona [10], un sistema basado en reglas puede emplear tecnologías más complejas como las ontologías para crear reglas semánticas robustas. Una ontología define formalmente las entidades, propiedades y relaciones dentro de un dominio específico, por ejemplo, el dominio de los cursos de una empresa. Esto permite crear reglas más inteligentes que operan sobre conceptos en lugar de sobre atributos aislados.

El propósito principal de este enfoque es guiar al usuario a través de un catálogo de ítems, asegurando que las recomendaciones cumplan con criterios específicos que van más allá de la simple similitud o popularidad. En el contexto de una plataforma *LMS*, las reglas podrían ser:

- **Reglas de Secuencia/Prerrequisito:**
    - SI un usuario ha completado "Introducción a la Contabilidad", ENTONCES recomendar "Contabilidad Intermedia".
    - SI un usuario se inscribe en "Curso Avanzado de *AWS*", ENTONCES verificar si ha completado "Fundamentos de Cloud Computing". Si no, sugerir este último primero.
- **Reglas Basadas en el Perfil del Usuario:**
    - SI el rol de un usuario es "Gerente de Ventas", ENTONCES priorizar en sus recomendaciones los cursos etiquetados como "Negociación" y "Liderazgo".
    - SI el departamento de un usuario es "Recursos Humanos", ENTONCES recomendar cursos sobre "Legislación Laboral" y "Gestión del Talento".
- **Reglas de Negocio o Promocionales:**
    - SI un curso es nuevo en el catálogo, ENTONCES mostrarlo en la sección de "Novedades" para todos los usuarios del grupo relevante.
    - SI un curso tiene una tasa de finalización baja, ENTONCES no recomendarlo

tan frecuentemente.

Este enfoque a menudo se combina con otras técnicas para formar sistemas híbridos. Por ejemplo, se puede utilizar un algoritmo de minería de datos como FP-Growth para descubrir reglas de asociación a partir del historial de interacciones de los usuarios, por ejemplo, ”los usuarios que toman el curso A también suelen tomar el curso C", y luego utilizar estas reglas descubiertas para generar recomendaciones, como se explora en el trabajo de [11].

Las ventajas de los sistemas basados en reglas son:

- **Control y Transparencia:** Los administradores tienen un control directo y explícito sobre la lógica de las recomendaciones, lo cual es fundamental para alinear las sugerencias con los objetivos de negocio o pedagógicos.
- **Eficiencia Computacional:** Suelen ser muy rápidos, ya que la lógica de filtrado es directa y no requiere cálculos complejos de similitud sobre grandes matrices de datos.
- **Solución al Problema de Arranque en Frío:** Al igual que otros sistemas basados en conocimiento, funcionan bien para ítems y usuarios nuevos, siempre que se puedan aplicar las reglas existentes.

Los principales desafíos incluyen:

- **Creación y Mantenimiento de Reglas:** La definición de un conjunto de reglas completo y coherente puede ser un proceso laborioso y que requiere de conocimiento experto. Las reglas pueden volverse obsoletas y necesitan un mantenimiento constante.
- **Rigidez y Falta de Personalización Dinámica:** Un sistema basado puramente en reglas puede ser demasiado rígido y no adaptarse a los gustos individuales y cambiantes de un usuario. Las recomendaciones pueden parecer genéricas si un gran número de usuarios cae bajo la misma regla.
- **Escalabilidad del Conjunto de Reglas:** A medida que el catálogo de ítems y la complejidad del dominio crecen, el número de reglas necesarias para cubrir todos los escenarios puede volverse inmanejable.

Debido a estas limitaciones, los sistemas basados en reglas rara vez se utilizan de forma aislada. Su verdadero poder se manifiesta cuando se integran en sistemas híbridos, donde pueden actuar como una capa de filtrado o de ajuste sobre las recomendaciones generadas por métodos como el Filtrado Colaborativo o el Basado en Contenido [2].

3.1.6. Sistemas Híbridos

Dado que cada enfoque de recomendación individual presenta, tal como se presentó en las secciones anteriores, fortalezas y debilidades inherentes, la investigación y la práctica industrial han convergido hacia la creación de sistemas híbridos. Un sistema de recomendación híbrido es aquel que combina dos o más técnicas de recomendación para lograr un mejor rendimiento, aprovechar las sinergias entre los métodos y mitigar las desventajas de cada uno de ellos por separado [12].

La motivación principal para la hibridación es superar problemas comunes como el arranque en frío (*cold start*), la dispersión de datos (*Data sparsity*) y la falta de diversidad o serendipia en las recomendaciones. Por ejemplo, un sistema híbrido podría utilizar filtrado basado en contenido para generar recomendaciones a usuarios nuevos, mitigando así el problema de *user cold start* del filtrado colaborativo, y luego usar un enfoque colaborativo a medida que el usuario acumula un historial de interacciones [3].

En [12], el autor propone una taxonomía de siete estrategias para el uso de sistema híbridos fundamentales:

1. **Ponderado (*Weighted Hybrid*):** Los resultados de diferentes sistemas de recomendación se combinan numéricamente. Se calcula un puntaje para cada ítem a partir de cada recomendador utilizado y luego se suman estos puntajes, a menudo con pesos diferentes, para obtener un ranking final. En [13] se propone un ajuste dinámico de pesos entre el filtrado colaborativo y el basado en contenido, iniciando con una distribución asignada, y ajustándola de acuerdo con los resultados de las recomendaciones surgidas de la distribución original.
2. **Conmutado (*Switching Hybrid*):** El sistema cambia entre diferentes técnicas de recomendación según el contexto o criterio actual. Por ejemplo, podría usar filtrado basado en contenido si el perfil del usuario es nuevo, y cambiar a filtrado colaborativo una vez que el usuario ha interactuado con un número mínimo de ítems [12].
3. **Mezclado (*Mixed Hybrid*):** Las recomendaciones de diferentes sistemas se presentan juntas en la misma lista, sin combinar sus puntajes. Por ejemplo, una página podría mostrar una fila de "Cursos Populares", basado en la popularidad general, y otra fila de "Cursos Recomendados para Ti", basado en filtrado colaborativo o de contenido, [11].
4. **Combinación de Características (*Feature Combination Hybrid*):** En lugar de combinar los resultados de los sistemas, se combinan sus datos de entrada. Las características de un enfoque se inyectan en el otro. Por ejemplo, los resultados de un sistema basado en contenido, como el "género" de un curso, podrían usarse como una característica

adicional en un modelo de filtrado colaborativo para mejorar su precisión.

5. **En Cascada (*Cascade Hybrid*):** Las técnicas se aplican en una secuencia de etapas. Un primer recomendador, generalmente menos costoso computacionalmente, genera una lista de candidatos iniciales. Luego, un segundo recomendador, más refinado, se aplica sobre esta lista reducida para generar el ranking final. Esto permite mejorar la calidad sin incurrir en el costo de aplicar el método más complejo a todo el catálogo [12].
6. **Aumento de Características (*Feature Augmentation Hybrid*):** Un sistema genera una predicción o una característica que luego es utilizada como entrada para otro sistema. Por ejemplo, un modelo de filtrado colaborativo podría predecir las calificaciones faltantes en la matriz usuario-ítem, y esta matriz "densa" se usaría luego para entrenar un modelo basado en contenido.
7. **Meta-Nivel (*Meta-level Hybrid*):** Un modelo de recomendación aprende a combinar los resultados de otros. El modelo completo de un recomendador, por ejemplo, el perfil de un usuario en un sistema basado en contenido se utiliza como entrada para otro recomendador, que podría ser un sistema de filtrado colaborativo que busca similitudes entre perfiles de usuario basados en contenido [12].

La elección de la estrategia de *hibridación* depende de los objetivos específicos del sistema, de la naturaleza de los datos disponibles y de los problemas que se buscan resolver. Otras consideraciones también deben ser tenidas en cuenta, como por ejemplo, disponibilidad de recursos computacionales, impacto esperado, cantidad de ítems disponibles, perfiles de los usuarios, esfuerzo requerido, entre muchas otras. Una combinación de esta naturaleza permitiría ofrecer recomendaciones personalizadas, relevantes y alineadas con los objetivos organizacionales, sin embargo, podría no ser tan relevante en términos de importancia para la organización.

3.1.7. Procesamiento de Lenguaje Natural

El Procesamiento del Lenguaje Natural (NLP**)** es una disciplina de la inteligencia artificial y la lingüística computacional enfocada en permitir que las computadoras comprendan, interpreten y procesen el lenguaje humano [14]. En el contexto de los sistemas de recomendación, el NLP es una herramienta fundamental, ya que gran parte de la información sobre los ítems, como productos, artículos o, en este caso, cursos, se encuentra en formato de texto no estructurado. Las técnicas de NLP permiten transformar este contenido textual, como las descripciones o etiquetas de los cursos, en representaciones numéricas que los algoritmos de machine learning pueden procesar eficazmente.

Una tarea central del NLP en este ámbito es la vectorización de texto, cuyo objetivo es convertir documentos en vectores numéricos que capturen su significado. Para esta tarea, existen diversos enfoques, entre los que destacan los modelos estadísticos y los basados en embeddings:

- **TF-IDF (Term Frequency-Inverse Document Frequency):** Es una técnica estadística que representa los documentos como vectores donde cada componente corresponde a la importancia de un término. La importancia se calcula multiplicando la frecuencia del término en el documento (TF) por la inversa de su frecuencia en todo el corpus (IDF). Aunque es eficaz para ponderar la relevancia de las palabras clave, su principal limitación es que no captura las relaciones semánticas entre los términos; trata cada palabra como una entidad independiente [15].
- **Embeddings de Palabras (Word Embeddings):** Son técnicas que aprenden representaciones vectoriales densas para las palabras a partir de grandes corpus de texto. A diferencia de TF-IDF, estos modelos, como Word2Vec o GloVe (Global Vectors for Word Representation), posicionan palabras con significados similares cerca unas de otras en el espacio vectorial [16]. GloVe, por ejemplo, lo logra analizando las estadísticas globales de co-ocurrencia de palabras en el corpus. Esta capacidad de capturar similitudes conceptuales, por ejemplo, entender que "finanzas" y "contabilidad" son temas relacionados, es crucial para generar recomendaciones temáticamente coherentes.

La literatura comparativa reciente en el campo ha demostrado [17] consistentemente las ventajas de los modelos de *word embeddings* sobre los enfoques estadísticos tradicionales. En tareas como la clasificación de texto, la recuperación de información y los sistemas de recomendación, los *embeddings* como GloVe suelen ofrecer una mayor precisión y coherencia temática, ya que su representación semántica es significativamente más rica y contextual.

Asimismo, encuestas académicas recientes confirman que los modelos de word embeddings son superiores en múltiples métricas frente a métodos estadísticos convencionales, especialmente en tareas complejas de procesamiento de texto. [18]

3.1.8. Evaluación de Sistemas de Recomendación

La evaluación de un sistema de recomendación, particularmente uno basado en un enfoque de aprendizaje no supervisado como el clustering, requiere una estrategia multifacética para validar tanto la calidad del modelo subyacente como la utilidad de sus resultados. El

rendimiento se puede medir a través de dos enfoques complementarios: la evaluación intrínseca del modelo de segmentación y la evaluación cualitativa de las recomendaciones finales.

3.1.8.1 Evaluación Intrínseca de Modelos de Clustering

Cuando un sistema de recomendación se basa en la segmentación de usuarios, la evaluación de la calidad de los clústeres generados es un paso fundamental. Esta evaluación se realiza mediante métricas cuantitativas que analizan la estructura geométrica de los grupos sin requerir etiquetas de *ground truth*. El objetivo es medir la cohesión (similitud de los puntos dentro de un clúster) y la separación (disimilitud entre clústeres). Las métricas estándar para este propósito incluyen:

- **Coeficiente de Silhouette:** Este coeficiente proporciona una medida de qué tan bien está asignado cada punto de datos a su clúster en comparación con otros clústeres. El valor se calcula para cada punto y varía entre -1 y +1. Un valor cercano a +1 indica que el punto está bien agrupado y lejos de los clústeres vecinos. Un valor cercano a 0 sugiere que el punto se encuentra en la frontera entre dos clústeres, mientras que valores negativos pueden indicar una asignación incorrecta. El promedio del coeficiente para todos los puntos da una puntuación global donde valores más altos indican una mejor calidad de clustering.
- **Índice de Calinski-Harabasz:** Conocido también como el Criterio de Ratio de Varianza, este índice se define como la relación entre la dispersión inter-clúster y la dispersión intra-clúster. Un modelo de clustering efectivo maximizará la distancia entre clústeres y minimizará la distancia dentro de ellos, por lo que para esta métrica, un valor más alto denota una mejor segmentación.
- **Índice de Davies-Bouldin:** Este índice calcula la similitud promedio de cada clúster con su clúster más similar, basándose en la relación entre las distancias intra-clúster y las distancias inter-clúster. Un valor más bajo indica que los clústeres están mejor separados, ya que su similitud es baja. Para esta métrica, un valor más bajo es preferible.

El análisis combinado de estas métricas es una herramienta para la selección de hiperparámetros, como la determinación del número óptimo de clústeres (k) en algoritmos como K-Means.

3.1.8.2 Evaluación Cualitativa de las Recomendaciones

Además de la validación matemática de la estructura del modelo, es necesario evaluar si las recomendaciones generadas son funcionalmente útiles y relevantes. La evaluación cualitativa se centra en el análisis del resultado final del sistema, es decir, en la lista de ítems recomendados, para juzgar su pertinencia.

Este tipo de evaluación se realiza mediante un análisis manual y experto de las recomendaciones generadas para un perfil de usuario determinado. El proceso busca responder a preguntas sobre la coherencia semántica, la relevancia contextual y la diversidad de las sugerencias. Permite verificar si el sistema recomienda ítems lógicamente complementarios al perfil del usuario y si evita la sobre especialización, validando así la efectividad del sistema desde una perspectiva de aplicación. La Coherencia Temática responde a la pregunta ¿Las recomendaciones se relacionan lógicamente con el historial de intereses del usuario? La Relevancia contextual, a ¿Los cursos sugeridos son apropiados para el posible rol o trayectoria profesional del usuario?, y la Diversidad y novedad a ¿El sistema ofrece una variedad de opciones o se limita a temas muy similares? Y a ¿Presenta cursos que el usuario no habría descubierto fácilmente por su cuenta (serendipia)?

3.1.9. Fases de un proceso de recomendación

De acuerdo con [3], las fases de un proceso de recomendación incluyen los siguientes componentes:

- Fase de recolección de información: Durante esta fase, el objetivo es reunir suficiente información sobre los usuarios para generar un perfil o modelo de usuario que será el utilizado en las tareas de predicción. Este perfil puede estar compuesto por atributos disponibles de los usuarios, su comportamiento y el contenido de los cursos o recursos a los que accede. La meta es disponer de la mayor cantidad de información sobre los usuarios posible. Entre los datos que se busca recolectar, pueden estar los siguientes: Retroalimentación explícita del usuario (Como los “likes”, o estrellas de puntuación), al igual que la implícita (Comentarios o comportamiento del usuario) y la retroalimentación híbrida.
- Fase de aprendizaje: Durante esta fase se aplica un algoritmo para el filtrado de las opciones a partir del perfil de usuario construido en la fase anterior.
- Fase de recomendación/predicción: Aquí se hace la recomendación o la predicción sobre los elementos que podrían ser de preferencia para el usuario. Las

recomendaciones se pueden generar a partir del conjunto de datos recolectados sobre el usuario, a partir de la actividad del usuario, o de una combinación de ambos.

3.2. ANTECEDENTES

La problemática de recomendar contenido educativo personalizado en plataformas *LMS* ha sido abordada por diversos investigadores, proporcionando ejemplos valiosos y algunas arquitecturas de referencia. Estos trabajos previos informan el diseño y las técnicas consideradas para el presente proyecto.

Un ejemplo relevante es el trabajo de Evale [19], titulado *"Learning Management System with Prediction Model and Course-content Recommendation Module"*. En este estudio, se propone una arquitectura para un sistema de recomendaciones de cursos de Java dentro de un *LMS* desarrollado en *Ruby on Rails*. Utilizando un historial de 5 años de datos de usuarios, se aplicó la técnica de minería de datos KDD (*Knowledge Discovery Database*). Los atributos clave para modelar el perfil del usuario incluyeron género, edad, curso, sección, horario y tres medidas de rendimiento académico. Este proyecto ilustra una implementación concreta de un sistema de recomendación en un *LMS*, detallando flujos para la oferta de recomendaciones y evaluando los atributos relevantes mediante la herramienta WEKA.

Otro enfoque interesante se presenta en "*E-learning Recommendation System Based on Cloud Computing"* [20]. Este trabajo propone un sistema de recomendaciones genérico, diseñado para ser incorporado en cualquier *LMS*, utilizando la infraestructura de computación en la nube de *Google Cloud Services*. Se detalla la arquitectura, los servicios y el flujo del proceso, con el objetivo de utilizar recursos bajo demanda. Aunque la plataforma *KME360* del presente proyecto está construida sobre *Amazon Web Services (AWS),* los principios de diseño y los procesos de un sistema basado en la nube son transferibles y pertinentes, dado que ambos proveedores ofrecen servicios y funcionalidades comparables.

En el artículo "Learning Content Recommendations on Personalized Learning Environment Using Collaborative Filtering Method" [21], se describe el desarrollo e implementación de un sistema de recomendaciones para un Entorno de Aprendizaje Personalizado (PLE) utilizando filtrado colaborativo. Aunque enfocado en un PLE, el documento explica el diagrama de proceso, la metodología de evaluación de las recomendaciones y el cálculo del error, conceptos que son transversales y aplicables a los sistemas *LMS*. Este trabajo sirve como un buen ejemplo de la implementación y evaluación del desempeño de un modelo de recomendación basado en una técnica ampliamente utilizada.

Finalmente, en "*Combination of machine learning algorithms for recommendation of*

courses in E-Learning System based on historical Data" [22], exploran técnicas de minado de datos como la Agrupación (*clustering*) y los algoritmos de Reglas de Asociación pueden ser efectivas para sistemas de recomendación de cursos. Su enfoque se basa en las elecciones de otros estudiantes sobre los cursos disponibles en la plataforma para generar recomendaciones. Este antecedente destaca la validez de utilizar técnicas de agrupamiento, para identificar patrones y generar sugerencias personalizadas.

Estos estudios previos ofrecen una base sólida, mostrando diversas arquitecturas, técnicas de modelado de usuarios, algoritmos de recomendación y metodologías de evaluación que son fundamentales para el desarrollo del sistema de recomendación para la plataforma *KME360*.

4. COMPRENSIÓN DEL NEGOCIO

El caso de uso para este proyecto se centra en KME360, una plataforma de Gestión del Aprendizaje (LMS/LXP) desarrollada por la empresa Nuevosmedios, y su implementación para un cliente estratégico: la Cámara de Comercio de Cali (CCC). KME360 funciona como el ecosistema de aprendizaje digital de la CCC, a través del cual se ofrece una amplia gama de contenidos formativos, como cursos virtuales, seminarios web y recursos breves, orientados a fortalecer las competencias del tejido empresarial. La importancia de optimizar la experiencia en esta plataforma es fundamental para la misión de la CCC, ya que al proporcionar formación relevante, se puede impulsar la competitividad de los empresarios y se fomenta el crecimiento económico.

El principal desafío que enfrenta la plataforma en este contexto es la sobrecarga de información, un problema magnificado por la escala de su operación. La naturaleza de la CCC define de manera fundamental las características del problema a resolver. A diferencia de un entorno de capacitación corporativo tradicional, donde los usuarios suelen ser empleados de una misma empresa con roles y objetivos de aprendizaje más homogéneos, la CCC ofrece un servicio de formación abierto al público, por lo tanto, su oferta formativa está diseñada para un ecosistema empresarial diverso, lo que se manifiesta en variedad de formatos como seminarios web, contenidos breves, llamados píldoras de conocimiento, y cursos de mayor duración. Los temas, también son variados, pues están dirigidos a un público que comprende desde emprendedores noveles hasta empresarios consolidados de diversos sectores y regiones, lo que hace que el descubrimiento de contenido relevante sea una tarea compleja e importante.

El análisis del conjunto de datos revela que la plataforma gestiona un catálogo de aproximadamente 1,200 cursos distintos para una base de más de 122,000 usuarios únicos. Esta vasta cantidad de opciones, combinada con la base de usuarios heterogénea, hace que un sistema de recomendación sea relevante para sus usuarios. Sin un mecanismo de guía inteligente, un usuario podría no encontrar un curso de alto valor para su negocio simplemente porque está oculto entre cientos de otras opciones. Esta fricción representa un riesgo directo para los objetivos de la CCC, pudiendo resultar en un bajo compromiso y en la subutilización de los recursos formativos.

Por lo tanto, la solicitud de desarrollar un sistema de recomendación surgió como una necesidad estratégica para transformar la plataforma de un repositorio estático a una experiencia de aprendizaje proactiva y personalizada. El objetivo de negocio es mejorar el compromiso del usuario y las tasas de finalización de cursos, facilitando el acceso a los

contenidos más pertinentes. En consecuencia, el éxito de este proyecto se define por la capacidad del sistema para analizar las preferencias de los usuarios y generar recomendaciones coherentes, precisas y contextualmente relevantes, aumentando así el valor percibido de la plataforma para su diversa comunidad de usuarios.

Traducidos desde el contexto del negocio, los objetivos del sistema de recomendación son:

- Aumentar la interacción de los usuarios con la plataforma, sugiriéndoles contenidos que de otro modo no habrían descubierto.
- Aumentar la tasa de finalización de cursos al presentar a los usuarios contenidos más atractivos para su formación.
- Transformar la plataforma de un catálogo estático a una experiencia de aprendizaje dinámica y adaptada a cada individuo.

5. METODOLOGÍA

La fase de Preparación de Datos es un paso crítico en el ciclo de vida de CRISP-DM, ya que la calidad y el formato del conjunto de datos de entrada impactan directamente en el rendimiento del modelo final. Para este proyecto, el proceso de transformación de los datos crudos (más de 500,000 interacciones de más de 122,000 usuarios en aproximadamente 1,200 cursos), en un dataset estructurado y listo para el modelado, se dividió en varias fases lógicas.

El siguiente diagrama (Figura X) ilustra el flujo de trabajo completo para la preparación de los datos, el cual se puede agrupar en tres grandes bloques funcionales que se detallarán en las subsecciones de este capítulo:

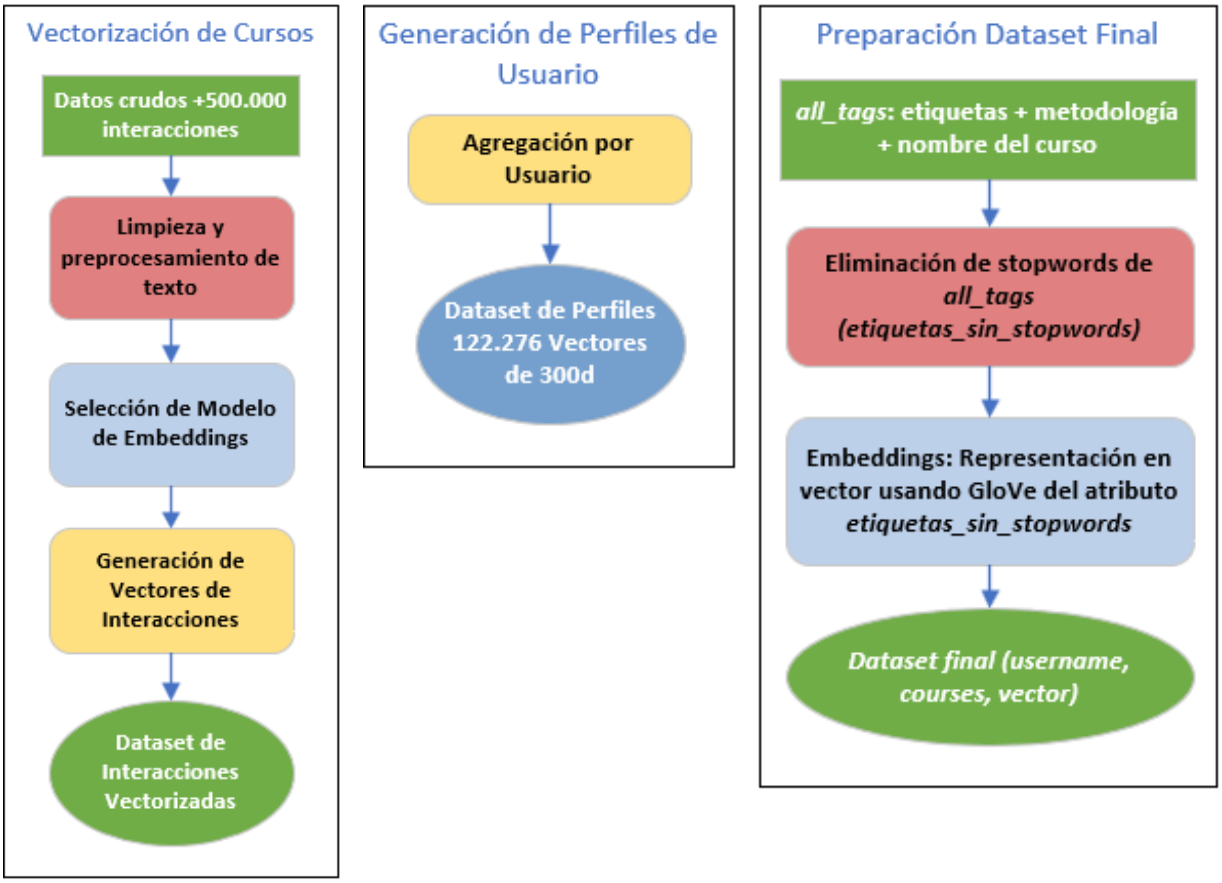


Figura 2: Flujos del Proceso de Preparación de Datos

El primer bloque, *Vectorización de Cursos*, describe el pipeline completo para transformar el contenido textual no estructurado de cada curso (sus etiquetas, nombre y metodología) en una representación numérica densa y semánticamente rica, utilizando para ello un modelo de embeddings GloVe.

El segundo bloque, *Generación de Perfiles de Usuario*, detalla cómo se utiliza el resultado anterior, en conjunto con el historial de interacciones de cada usuario, para construir un perfil vectorial único que encapsula sus intereses latentes.

Finalmente, el resultado de estas fases es un Dataset Final limpio y estructurado, donde cada uno de los 122,276 usuarios está representado por un único vector. Este conjunto de datos es la entrada fundamental para la fase de Modelado, donde se aplicarán los algoritmos de clustering para la segmentación.

5.1. PREPARACIÓN DE DATOS

5.1.1. SELECCIÓN DE DATOS

Para construir un sistema de recomendación, lo primero que hicimos fue caracterizar adecuadamente los ítems que se van a recomendar, es decir los usuarios y su interacción con los cursos. El conjunto de datos tiene información de alrededor de 1200 cursos distintos y recursos virtuales de aprendizaje, y de la interacción que los usuarios han tenido con ellos, específicamente a cuáles se han inscrito. Las características identificadas en este conjunto de datos son:

- ***Course uuid*** (Identificador Único del Curso, Variable categórica nominal): Un identificador alfanumérico único para cada curso, esencial para el seguimiento y la referenciación.
- ***Course name*** (Nombre del Curso, Variable tipo texto): El título formal del curso. Esta es una fuente rica de información textual descriptiva y puede ser fuente para mejorar el etiquetado del curso.
- ***Tags*** (Etiquetas del Curso, Variable tipo texto): Palabras clave o frases asociadas a cada curso, proporcionando metadatos semánticos sobre su contenido y temática. Esta característica es fundamental para enfoques basados en contenido.
- ***Username*** (Usuario, Variable categórica nominal): Identificador único del usuario que ha tomado el curso.
- ***inscription date*** (Fecha de Inscripción, Variable de tipo Fecha): Indica la fecha en la que se inscribió el usuario en el curso.
- ***final_datee*** (Fecha de Finalización, Variable de tipo Fecha): Si el usuario ha terminado el curso, este campo muestra la fecha en que terminó.
- ***Grade*** (Nota/Calificación, Variable numérica continua): Nota obtenida por el usuario dentro del curso.

- ***Percentage*** (Porcentaje de Avance, Variable numérica continua): Porcentaje en que el usuario ha completado los elementos del curso sobre el total.
- ***days of course*** (Duración del Curso, Variable numérica discreta): Dato que indica el tiempo de duración en días que tiene el usuario para finalizar el curso desde el momento de su inscripción.
- ***Student status*** (Estado del Estudiante en el Curso, Variable categórica): Se usa para saber si el usuario solo se inscribió, pero no avanzó en el curso, si lo terminó o si avanzó, pero no lo terminó. Puede tomar los valores de Inscrito, Aprobado, Reprobado.
- ***Course status*** **(**Estado del Curso, Variable categórica): Estado del curso en la plataforma.
- ***Methodology*** (Metodología del Curso, Variable categórica): Describe el formato o la modalidad del curso típicamente, Virtual o Presencial. Aunque presentaba valores nulos, se decidió imputarlos, considerándola relevante para la caracterización. Siendo una plataforma *LMS*, la imputación de los valores nulos se hizo con el valor "Virtual".
- ***category*** (Categoría del Curso, Variable categórica**):** Clasificación general del curso, por ejemplo: 'Gratuito' o 'Costo'.

De las características mencionadas, se escogieron los siguientes para proceder al siguiente paso de la preparación, pues al combinarse, permiten construir una representación semántica del curso, permitiendo identificar y relacionar los usuarios que se han inscrito en un curso, es decir, que han tenido interés en un tema en particular. Las características seleccionadas son:

- Course uuid
- Course name
- Tags
- Username
- Methodology

Para la construcción de una representación vectorial de los cursos, se decidió consolidar las características textuales. Se creó una nueva característica combinada, denominada *all_tags*, concatenando los campos *Tags*, *Methodology* y *Course name*. Esta representación unificada fue posteriormente sometida a un proceso de limpieza en la que se realizó una conversión a minúsculas, la eliminación de caracteres especiales y de *stopwords* en español, para obtener la columna *etiquetas_sin_stopwords*. Esta última sirvió como base para la generación de *embeddings* vectoriales de los cursos utilizando el modelo *GloVe*, una técnica común para capturar relaciones semánticas en datos textuales [23]. El vector

resultante, que es un *embedding* de 300 dimensiones, encapsula las características del curso de una manera densa y numérica, facilitando cálculos de similitud y la construcción de perfiles de usuario.

En cuanto a las características *username* y *Course uuid,* estas permiten inferir el comportamiento del usuario, específicamente, en cuales cursos se han matriculado. El dataset está conformado por registros que tienen estas dos características, indicando que un usuario (un *username)* se ha matriculado en un curso (un *Course uuid*), lo cual es fundamental para entender sus preferencias e intereses.

Si bien el historial de interacciones proporciona los datos "brutos", no constituye por sí mismo un perfil que un modelo de machine learning pueda procesar directamente. Para aplicar técnicas de *clustering*, que es el núcleo de nuestro enfoque, es necesario transformar este historial heterogéneo en una representación numérica unificada y consistente para cada usuario, es decir, en un perfil vectorial.

La decisión de priorizar las características textuales de los cursos (*Tags, Course name, Methodology*) permite aplicar técnicas de *feature engineering* basadas en contenido, como la generación de *embeddings*, para obtener una descripción semántica rica y profunda de cada ítem. A su vez, el historial de interacciones, es decir los registros de los cursos a los que se ha inscrito un usuario constituye la base para aplicar un método colaborativo. Al ser los cursos de libre inscripción, se considera que la acción de matricularse es una señal implícita y suficiente de interés por parte del usuario, lo que valida el uso de su historial para inferir sus preferencias latentes [2]. Estas dos fuentes de información, es decir, las propiedades semánticas de los cursos y el comportamiento histórico de los usuarios son, por tanto, las características fundamentales procesadas en la fase de Preparación de Datos que permiten la construcción del sistema de recomendación propuesto.

El objetivo de este proyecto es segmentar a los usuarios en grupos de interés similares, un enfoque de filtrado colaborativo basado en modelos. Para lograrlo, cada usuario debe ser representado como un punto en un espacio vectorial multidimensional, donde la distancia entre dos puntos refleje la similitud de sus intereses. Por lo tanto, el propósito fundamental de los siguientes pasos de la preparación de datos es construir este perfil vectorial para cada uno de los 122,276 usuarios, basándose en las propiedades semánticas de los cursos que ha consumido.

5.1.2. LIMPIEZA DE DATOS

El proceso de limpieza de datos se dividió en varias subtareas clave: la consolidación de datos, el manejo de valores nulos y la limpieza de características textuales.

Durante la consolidación del conjunto de datos, debido al gran volumen de registros de interacción, la extracción de datos desde la base de datos de producción se realizó en fragmentos segmentados por periodos temporales, de 3 meses cada uno, hasta abarcar todos los registros existentes comprendiendo los años 2022, 2023, 2024 y 2025. La primera tarea de preparación consistió en unir estos archivos *".csv"* en un único *DataFrame* de *Pandas*, creando así el conjunto de datos de trabajo unificado para los siguientes pasos del preprocesamiento.

Un análisis exploratorio inicial reveló la presencia de valores nulos en columnas importantes:

- ***Tags*:** Esta característica es fundamental para el análisis semántico de los cursos. Se observó que un pequeño porcentaje de los registros carecía de etiquetas. Dado que imputar estos valores con un marcador genérico no aportaría información útil al modelo, se tomó la decisión de eliminar las filas que presentaban valores nulos en este campo, asegurando así que todos los cursos en el *Data*set final tuvieran una base de contenido textual para su análisis.
- ***Methodology*:** Esta columna presentaba valores nulos en los casos en que no se especificaba la modalidad del curso. Considerando el contexto de la plataforma *KME360*, donde la mayoría de la oferta es virtual, se aplicó una estrategia de imputación, rellenando los valores faltantes con el término "Virtual". Esta decisión permitió mantener la integridad de los registros y enriquecer el corpus textual de cada curso.

En la limpieza y normalización de características textuales, debimos procesar las etiquetas, pues al ser ingresadas manualmente o provenir de diferentes fuentes, presentaban inconsistencias que podían afectar la capacidad del modelo para identificar patrones semánticos. Se observaron problemas como el uso mixto de mayúsculas y minúsculas, la presencia de caracteres especiales, el uso de palabras en inglés y la variación terminológica.

Para abordar esto, se implementó un pipeline de limpieza de texto sobre la columna construida *all_tags,* que contiene la concatenación de las columnas *Tags*, *Course name* y *Methodology*. Dicho pipeline consistió en los siguientes pasos:

Imputación sobre el campo *methodology*: Se aplicó imputación sobre los valores vacíos en este campo, para asignar el valor "Virtual" ya que la gran mayoría de las capacitaciones que realiza el cliente son con este tipo de metodología.

Conversión a Minúsculas: Se estandarizó todo el texto a minúsculas para asegurar que

palabras como "Finanzas" y "finanzas" fueran tratadas como el mismo término.

Eliminación de Caracteres Especiales: Se utilizaron expresiones regulares para remover todos los caracteres que no fueran alfanuméricos, como puntuación, símbolos y demás, dejando únicamente palabras y números.

Traducción de palabras del inglés al español: Algunas etiquetas contienen palabras en inglés. Para poder tenerlas en cuenta se usó la librería *Google Translator de Deep_translation*. Cada palabra traducida se almacenó en un diccionario de datos que guardaba su respectiva traducción. Cuando se encontraba esa palabra más adelante, se buscaba en el diccionario y se extraía su equivalencia en español. De esta manera, se redujeron mucho los tiempos de respuesta.

Eliminación de *Stopwords*: Se filtraron las palabras comunes en español que no aportan un significado semántico distintivo, como "el", "la", "de", "con", entre otras, utilizando la lista de *stopwords* proporcionada por la librería NLTK de Python.

El resultado de este proceso fue la columna *etiquetas_sin_stopwords*, un corpus de texto normalizado y limpio para cada curso, compuesto por palabras individuales o *tokens* que sirvió como base para la posterior generación de representaciones vectoriales.

5.1.3. TRANSFORMACIÓN DE DATOS

Una vez preparado el corpus textual de cada curso se trabajó sobre este corpus resultante de la limpieza para poder transformar esta información no estructurada en una representación numérica. Para esta tarea se optó por una técnica de Procesamiento del Lenguaje Natural (NLP): los *embeddings* de palabras o word *embeddings* en inglés. Estos modelos aprenden representaciones vectoriales densas para las palabras, donde los vectores de términos con significados similares se ubican cerca en el espacio vectorial [24], por ejemplo, “contabilidad” y “finanzas" serán entendidos conceptos relacionados y estarán en un espacio vectorial cercano.

5.1.3.1 Selección del Modelo de *Embeddings*: *GloVe*

Para este proyecto, se seleccionó el modelo *GloVe (Global Vectors for Word Representation)* [23]. *GloVe* es un algoritmo de aprendizaje no supervisado que genera representaciones vectoriales a partir de las estadísticas de co-ocurrencia de palabras en un corpus masivo. A diferencia de otros modelos como *Word2Vec*, *GloVe* se entrena sobre una matriz global de co-ocurrencia, lo que le permite capturar de manera eficiente tanto el significado local como las estadísticas globales del lenguaje.

Para el desarrollo del sistema de recomendaciones de cursos en la plataforma kme360, se planteó la necesidad de convertir las etiquetas asociadas a los cursos en representaciones numéricas que pudieran ser procesadas por los algoritmos de agrupamiento y recomendación. Ante esta necesidad, se evaluaron distintas técnicas de vectorización, destacándose TF-IDF y *(Global Vectors for Word Representation)* [23].

Tras analizar sus características, se optó por utilizar GloVe debido que a diferencia de TF-IDF, que se basa únicamente en la frecuencia de aparición de palabras en documentos sin considerar su significado, GloVe permite generar vectores que capturan relaciones semánticas y contextuales entre palabras. Esto resulta crucial en el contexto de un LMS, donde es deseable que cursos etiquetados con términos relacionados conceptualmente, aunque diferentes en forma, se consideren similares para efectos de recomendación. Por ejemplo, cursos etiquetados con "machine learning", "deep learning" y "IA" tendrán representaciones numéricas cercanas en el espacio vectorial de GloVe.

De igual manera, el uso de vectores semánticos densos permite que el sistema recomiende cursos no solo por coincidencia literal de etiquetas, sino también por afinidad temática. Esto enriquece la experiencia del usuario y aumenta la posibilidad de descubrir contenidos relevantes que podrían pasar desapercibidos en un modelo basado exclusivamente en coincidencias textuales.

Posteriormente se procedió a realizar un ejercicio utilizando un subconjunto del dataset de 100.000 registros para evaluar las métricas que se obtienen utilizando los vectores que genera TF-IDF y GloVe para aplicar k-means. Utilizando 100.000 registros, y una cantidad de clústeres de 600 los datos fueron los siguientes:

Técnica	**Silhouette Score**	**Davies-Bouldin**	**Calinski-Harabasz**
TF -IDF	0.5351	1.5808	533.6281
GloVe	0.6038	1.1744	989.1412

Tabla 1: Ejercicio de comparación de IF-IDF y GloVe

También podemos verlo gráficamente:

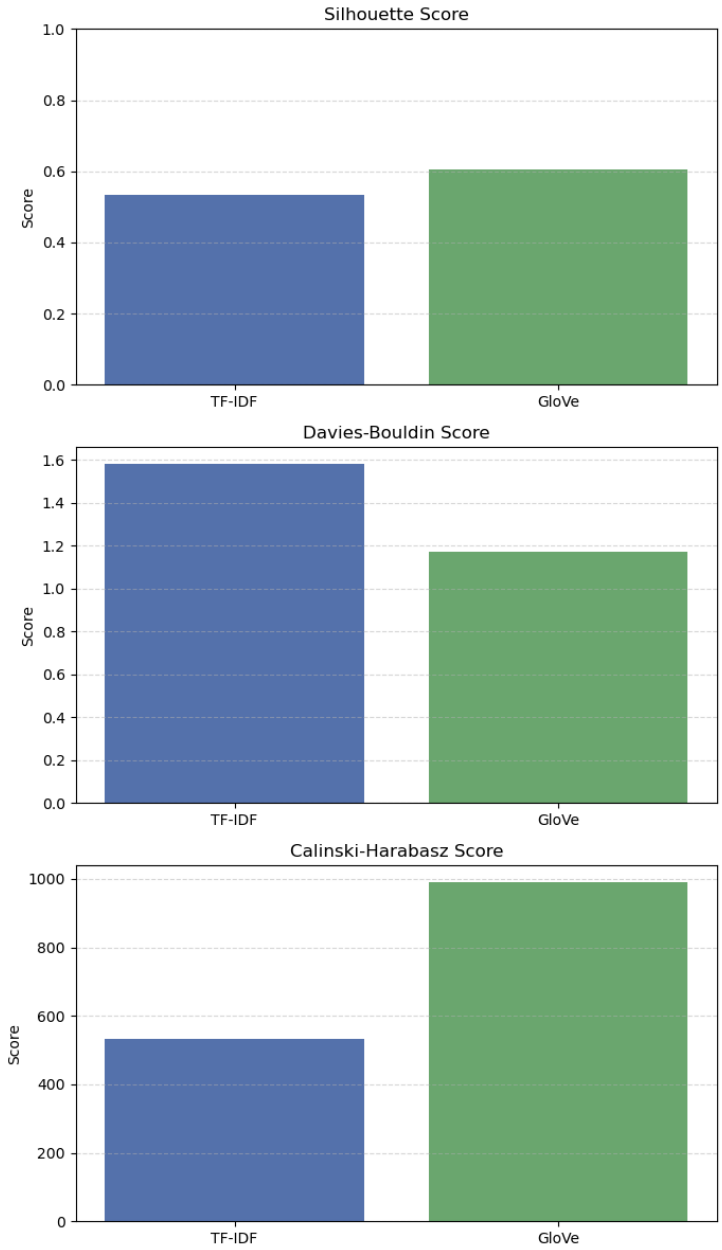


Figura 3: Comparación de métricas entre TF-IDF y GloVe

Como podemos ver, GloVe nos da un desempeño un poco mejor frente a TF-IDF en todas las métricas, ya que da más cerca de 1 en Silhouette, en Davies Bouldin el valor está más cerca de 1 y en Calinski nos da un valor bastante más alto.

Con esta información tomamos la decisión de continuar avanzando con la vectorización utilizando GloVe.

Se utilizó el corpus pre-entrenado *GloVe-sbwc.i25.vec*, que contiene vectores de 300 dimensiones para más de 850,000 palabras en español. La elección de un modelo pre-

entrenado sobre un gran corpus generalista es ventajosa, ya que asegura que los vectores tengan un conocimiento semántico amplio, incluso para términos que podrían no ser frecuentes en el *Data*set específico de los cursos de *KME360*.

5.1.3.2 Proceso de Generación de Vectores de Curso

El proceso para obtener un único vector representativo para cada curso fue el siguiente:

1. **Tokenización:** El primer paso consistió en segmentar el corpus de texto limpio de cada curso en unidades semánticas básicas, conocidas como *tokens*. Se aplicó una tokenización a nivel de palabra (*word-level tokenization*), dividiendo la cadena de texto por espacios. Esta estrategia es adecuada ya que las características semánticas de los cursos están contenidas en palabras individuales o en secuencias cortas de palabras (n-gramas). El resultado de esta etapa fue una lista de *tokens* para cada curso, que sirvió como entrada para el siguiente paso de mapeo vectorial.
2. **Mapeo de Vectores:** Cada *token* de la lista se buscó en el diccionario del modelo GloVe pre-entrenado. Para optimizar el rendimiento, se implementó un mecanismo de caché para las traducciones de términos en inglés: cada palabra traducida se almacenaba en un diccionario propio para evitar consultas repetidas al servicio de traducción, lo que significó una mejora sustancial en el tiempo de ejecución. Si una palabra se encontraba (ya sea directamente o tras su traducción), se recuperaba su vector correspondiente de 300 dimensiones.
3. **Manejo de Palabras Desconocidas (*Out-of-Vocabulary*):** Se observó que algunos *tokens*, especialmente términos técnicos muy específicos, acrónimos o errores tipográficos, no se encontraban en el corpus de GloVe. Para manejar estos casos, se les asignó un vector de ceros. Esta estrategia asegura que las palabras desconocidas no aporten al cálculo del vector promedio final, evitando así la introducción de ruido en la representación semántica.
4. **Agregación de Vectores:** Finalmente, el vector representativo para cada curso se calculó como el promedio de los vectores de todos los tokens válidos (no nulos) que lo componen.

El resultado de este pipeline es la transformación de cada curso en un único punto (un vector de 300 dimensiones) en un espacio vectorial semántico. Esta representación densa y numérica es computacionalmente más eficiente que los vectores dispersos y permite realizar cálculos de similitud significativos, como la similitud del coseno, para comparar cursos basándose en su contexto y significado, y no solo en la coincidencia exacta de

palabras clave.

5.1.4. GENERACIÓN DE PERFILES DE USUARIO

Una vez que cada curso en el catálogo fue transformado en una representación vectorial semántica, el siguiente paso fue construir un perfil numérico para cada uno de los 122,276 usuarios únicos del *data*set tal como se mencionó en la sección 5.1.1. El objetivo de esta fase es representar las preferencias e intereses de cada usuario en el mismo espacio vectorial de 300 dimensiones que los cursos, lo que permite comparaciones directas y significativas entre usuarios, y entre usuarios y cursos.

Para lograr esto, se implementó una estrategia de perfilado basada en el historial de interacciones de cada usuario. La premisa fundamental es que el perfil de un individuo puede ser modelado como una agregación de los contenidos que ha consumido voluntariamente [2]. El proceso fue el siguiente:

1. **Recuperación de Historial:** Para cada *username* en el conjunto de datos, se recuperó la lista de todos los cursos con los que ha interactuado.
2. **Agregación de Vectores de Curso:** Se accedió a los vectores *GloVe* de 300 dimensiones previamente generados para cada uno de los cursos en el historial del usuario.
3. **Cálculo del Vector Promedio:** El perfil vectorial final para cada usuario se calculó como el promedio aritmético de los vectores de todos los cursos de su historial.

Matemáticamente, si un usuario ha interactuado con un conjunto de cursos $Cu = \{c_1, c_2, \ldots, c_n\}$, y cada curso c_i está representado por un vector de *embedding* v_i, entonces el vector de perfil del usuario p_u se calcula como:

$$p_u = (1/n) * \Sigma v_i \text{ (para i desde 1 hasta n)}$$

Esta representación del perfil de usuario como el centroide de los ítems consumidos es una técnica robusta y comúnmente utilizada en sistemas de recomendación basados en contenido y modelos híbridos [25]. El vector resultante, p_u, encapsula los intereses agregados del usuario, posicionándolo en el espacio semántico. Por ejemplo, un usuario que ha tomado mayoritariamente cursos de finanzas y contabilidad tendrá un vector de perfil muy cercano a los vectores de otros cursos de esa misma área temática.

Esta transformación es el paso final de la fase de Preparación de Datos. El resultado es un conjunto de datos donde cada uno de los 122,276 usuarios está representado por un único vector de 300 dimensiones. Este conjunto de perfiles vectoriales es la entrada directa para

la fase de Modelado, donde se aplicarán algoritmos de *clustering* como *K-Means* y *DBSCAN* para segmentar a los usuarios en grupos con intereses similares, sentando así las bases para la generación de recomendaciones colaborativas.

5.2. MODELADO

Tras la fase de Preparación de Datos, que culminó con la creación de un conjunto de datos de perfiles de usuario vectorizados, se procede a la fase de Modelado de la metodología CRISP-DM. Este capítulo detalla el proceso de selección, entrenamiento y evaluación de los algoritmos de clustering. El objetivo es encontrar el modelo más efectivo para segmentar a los usuarios en grupos de interés similares, sentando así las bases para la generación de recomendaciones colaborativas.

5.2.1. SELECCIÓN DE ALGORITMOS DE APRENDIZAJE AUTOMÁTICO

La estrategia adoptada para generar recomendaciones personalizadas se articula en dos fases: la representación vectorial que se describe en el capítulo 5 y el agrupamiento de usuarios o *clustering*. El objetivo del *clustering* es segmentar a los usuarios en grupos con perfiles de intereses similares para, posteriormente, recomendarles ítems populares dentro de su propio grupo, un enfoque clásico de filtrado colaborativo basado en modelos [22].

Para esta tarea, se seleccionaron dos algoritmos de *clustering* para una evaluación comparativa, basándose en su popularidad en la literatura y sus características operativas:

- ***K-Means clustering*:** Es un algoritmo de particionamiento iterativo que asigna cada punto de datos al clúster cuyo centroide es el más cercano. Sus principales ventajas son su eficiencia computacional, su escalabilidad a grandes conjuntos de datos y la facilidad para interpretar sus resultados como grupos de usuarios. Sin embargo, requiere que el número de clústeres (K) se especifique a priori y tiende a encontrar clústeres de forma esférica, lo que puede no ser ideal para estructuras de datos complejas [2].
- ***DBSCAN (Density-Based Spatial clustering of Applications with Noise):*** Es un algoritmo basado en densidad que agrupa puntos que están densamente empaquetados, marcando como ruido los puntos que se encuentran en regiones de baja densidad. Su principal ventaja es que no requiere predefinir el número de clústeres y puede encontrar agrupaciones de formas arbitrarias. No obstante, es sensible a sus parámetros (*eps* y *min_samples*) y puede tener dificultades con *Datasets* que presentan variaciones significativas en la densidad de sus clústeres [26].

Ambos algoritmos fueron seleccionados para ser entrenados y evaluados con el conjunto de perfiles de usuario, con el fin de determinar cuál de ellos ofrecía la segmentación más efectiva y útil para el problema de recomendación de cursos en la plataforma *KME360*. Los

resultados de esta evaluación comparativa se detallan en el Capítulo EVALUACIÓN GENERAL Y SELECCIÓN DEL MODELO5.2.2.

Además de los algoritmos K-Means y DBSCAN, se consideró el uso de Modelos de Mixturas Gaussianas (*Gaussian Mixture Models - GMM*) como una alternativa más sofisticada para la tarea de *clustering*. *GMM* es un modelo probabilístico que asume que los datos son generados a partir de una mezcla de un número finito de distribuciones gaussianas, cada una con su propia media y matriz de covarianza [27].

La principal ventaja teórica de *GMM* sobre K-Means es su flexibilidad. Mientras que K-Means asume clústeres de forma esférica, *GMM* puede modelar clústeres elípticos de diferentes tamaños y orientaciones. Adicionalmente, *GMM* proporciona una asignación "suave" o probabilística, indicando la probabilidad de que un punto de datos pertenezca a cada uno de los clústeres, lo que podría capturar de manera más rica la naturaleza de los intereses de un usuario, que a menudo no pertenecen exclusivamente a una única categoría.

Sin embargo, a pesar de sus ventajas teóricas, la aplicación de *GMM* en este proyecto enfrentó una barrera computacional insuperable. La alta dimensionalidad de los perfiles de usuario (vectores de 300 dimensiones) incrementa drásticamente la complejidad del algoritmo, particularmente en la estimación de las matrices de covarianza para cada componente gaussiano. Los intentos preliminares de entrenar el modelo *GMM* sobre el conjunto de datos completo resultaron en un consumo de memoria y tiempo de procesamiento que excedía los recursos disponibles, incluso en un entorno de cómputo robusto. Por esta razón, y a pesar de su potencial, *GMM* fue descartado en una fase temprana, y el análisis se centró en los algoritmos K-Means y DBSCAN, más viables desde una perspectiva computacional para la escala de este problema.

5.2.1.1 Entrenamiento y aplicación de *K-Means clustering*

El algoritmo K-Means es un método de particionamiento iterativo cuyo objetivo es agrupar los *n* puntos de datos (en nuestro caso, los 122,276 perfiles de usuario) en *k* clústeres predefinidos, minimizando la inercia o la suma de las distancias cuadradas dentro de cada clúster [2].

Una de las principales consideraciones al usar K-Means es la selección del número de clústeres (*k*), ya que es un hiperparámetro que debe definirse a priori. Una elección inadecuada de *k* puede llevar a una segmentación deficiente que agrupe usuarios con intereses dispares, o a una sobre segmentación, creando grupos demasiado pequeños y poco útiles para hacer las recomendaciones. Para determinar un valor de *k* apropiado para nuestro conjunto de datos, se utilizó el Método del Codo *(Elbow Method)*.

Para esto, se entrenó el modelo K-Means con un rango de valores de *k,* desde 2,000 hasta 30,000, y se calculó la inercia[1] para cada valor. La gráfica de inercia frente a *k* muestra una disminución esperada a medida que *k* aumenta. Sin embargo, no se observó un "codo" pronunciado o inequívoco que sugiriera un único valor óptimo. Esta falta de un codo claro puede interpretarse de varias maneras, por ejemplo, que los perfiles de usuario se distribuyen de manera relativamente continua en el espacio vectorial, sin agrupaciones naturales marcadamente definidas, o que el rango óptimo de *k* podría ser diferente o la forma esférica de los clústeres de K-Means podría no ser la ideal para capturar toda la complejidad de la estructura subyacente de los datos.

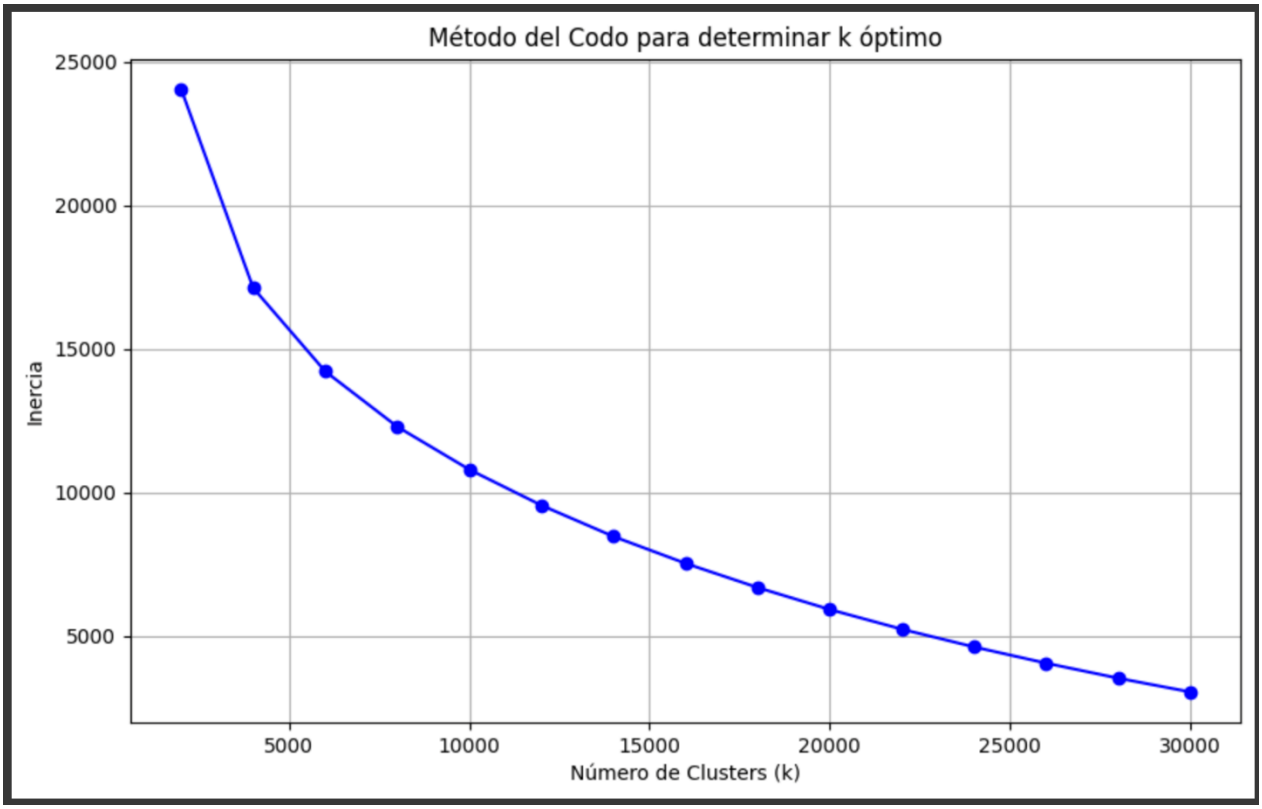


Figura 4: Método del Codo para la selección de k en K-Means

Evaluación Cualitativa y Cuantitativa de la Estructura de los Clústeres

Además del método del codo, que proporciona una guía heurística, se evaluó la calidad de la segmentación para diferentes valores de *k* utilizando un conjunto de métricas cuantitativas estándar para la evaluación de algoritmos de *clustering* no supervisado. Estas métricas evalúan la calidad de los clústeres basándose en dos criterios principales: la

[1] Suma de las distancias al cuadrado de cada punto a su centroide más cercano.

cohesión, que mide qué tan similares son los puntos dentro de un mismo clúster, y la separación, que mide qué tan distintos son los clústeres entre sí. Las métricas seleccionadas fueron:

- **Coeficiente de Silhouette (Silhouette Score):** Este Coeficiente es una métrica utilizada para evaluar la calidad de los clústeres al medir qué tan bien está asignado cada punto de datos a su propio clúster en comparación con otros clústeres. A diferencia de otras métricas que evalúan el *clustering* en su conjunto, el Silhouette proporciona una medida de la consistencia para cada punto individual, cuyo promedio da una puntuación global de la calidad de la segmentación. Esta métrica mide qué tan similar es un punto a su propio clúster, es decir, la cohesión, en comparación con otros clústeres, que sería la separación. El valor del coeficiente varía entre -1 y 1. Un valor cercano a +1 indica que el punto está bien asignado a su clúster y lejos de los clústeres vecinos. Un valor cercano a 0 indica que el punto está en el límite, o muy cerca del límite de decisión entre dos clústeres, y valores negativos indican que el punto podría haber sido asignado al clúster incorrecto. Para esta métrica, un valor más alto es mejor.
- **Índice de Calinski-Harabasz (Calinski-Harabasz Score):** También conocido como el Criterio de Ratio de Varianza, este índice se define como la relación entre la dispersión inter-clúster, que se calcula con la suma de las distancias al cuadrado entre los centroides de los clústeres y el centroide global, y la dispersión intra-clúster, que es la suma de las distancias al cuadrado de cada punto a su propio centroide. Intuitivamente, un buen *clustering* tiene clústeres compactos y bien separados. Por lo tanto, para esta métrica, un valor más alto es mejor.
- **Índice de Davies-Bouldin (Davies-Bouldin Score):** Este índice se calcula como la similitud promedio entre cada clúster y su clúster más similar, donde la similitud es la relación de las distancias intra-clúster con respecto a las distancias inter-clúster. Un valor más bajo indica que los clústeres están mejor separados, ya que la distancia entre ellos es mayor en comparación con su tamaño. Por lo tanto, para esta métrica, un valor más bajo es mejor.

A continuación, la Tabla 1 presenta los resultados de estas métricas para los diferentes valores de *k* evaluados, junto con la observación de si se generaron clústeres vacíos.

K	Silhoutte Score	Calinski-Harabasz score	Davies-Bouldin score	Número de clústeres vacíos
2.000	0,43	578,5	1,32	0
3.000	0,46	479,7	1,38	0

4.000	0,49	411,6	1,43	0
5.000	0,50	363,9	1,43	0
6.000	0,51	328,8	1,40	0
6.500	0,51	314,5	1,38	0
7.000	0,52	302,3	1,36	0
8.000	0,53	282,5	1,32	0
9.000	0,53	266,7	1,27	0
10.000	0,54	254,3	1,22	0
11.000	0,54	244,5	1,16	0
12.000	0,54	236,5	1,12	0
13.000	0,54	230,1	1,08	0
14.000	0,55	225,5	1,04	0
15.000	0,55	221,4	1,01	0
16.000	0,55	218,6	0,97	0
17.000	0,55	216,5	0,94	0
18.000	0,56	215,3	0,91	0
19.000	0,56	214,9	0,88	0
20.000	0,56	215,2	0,86	0
21.000	0,56	216,1	0,83	0
25.000	0,57	225,2	0,75	0
27.000	0,57	234,0	0,71	0
30.000	0,58	253,8	0,67	0

Tabla 2: Métricas de Evaluación de clustering para los diferentes valores de k

Para cada valor de k también se revisaron los histogramas de los clústeres resultantes. A continuación, se muestran unas gráficas relevantes para el análisis realizado, empezando por las gráficas para k = 3.000 y k= 4.000. Al examinar visualmente la distribución de los tamaños de los clústeres para diferentes valores de *k*, se observan patrones claros que informan la decisión del modelo. Las gráficas para K=3000 y K=4000 (ver Figura 5 y Figura 6) son particularmente ilustrativas de los desafíos que presentan valores de *k* más bajos.

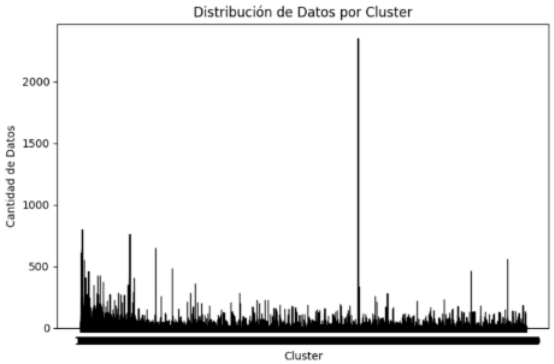


Figura 5: Histograma para k = 3.000

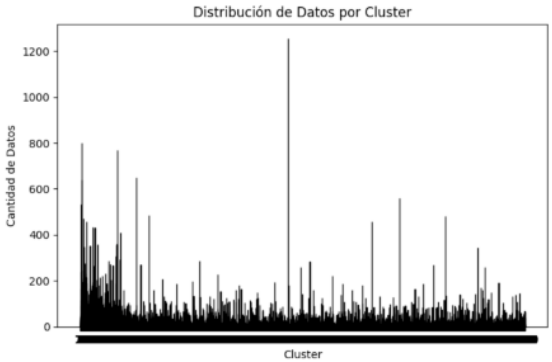


Figura 6: Histograma para k = 4.000

Para el caso del histograma para el valor de k = 3.000 se puede observar que el gráfico está

condicionado por un clúster excepcionalmente grande, que está cerca de los 2.500 usuarios. Este "mega-clúster" es tan dominante que unos pocos clústeres aparecen con valores de unos 800 usuarios, y la gran mayoría está por debajo de los 500 usuarios, muchos de ellos por los 100 usuarios. Este patrón sugiere que el modelo, con solo 3,000 centroides disponibles, no es capaz de capturar adecuadamente la diversidad de los perfiles de usuario. Como resultado, una gran cantidad de usuarios con intereses potencialmente distintos, pero sin un grupo claro al que pertenecer, son asignados a un único clúster de gran tamaño. Una recomendación basada en este valor de k sería problemática. Para los usuarios en el clúster dominante, las recomendaciones serían muy genéricas y poco personalizadas, ya que se basarían en los intereses agregados de un grupo muy heterogéneo. Esto anularía el propósito de la segmentación.

Para el otro caso, con k=4.000 la distribución es muy similar a la de K=3.000. Sigue existiendo un clúster principal que concentra una cantidad desproporcionada de usuarios, mayor a 1.200. Aunque el tamaño de este clúster dominante ha disminuido ligeramente en comparación con K=3.000, la estructura general del histograma es la misma: un pico masivo y una base de clústeres mucho más pequeños. A pesar de aumentar el número de clústeres a 4,000, el modelo todavía tiende a crear un grupo principal muy grande. Esto refuerza la hipótesis de que valores de *k* en este rango bajo son insuficientes para segmentar de manera efectiva la base de usuarios. La granularidad no es la adecuada para diferenciar los distintos nichos de interés presentes en los datos.

El análisis de estas distribuciones demuestra que, si bien las métricas cuantitativas como Calinski-Harabasz podrían favorecer valores bajos de *k*, cualitativamente estas segmentaciones son deficientes. La presencia de un clúster que agrupa a una fracción tan grande de la población de usuarios indica una pobre separación y es una clara señal de que se necesita un número mayor de clústeres para modelar de forma más fiel y útil las comunidades de interés dentro de la plataforma *KME360*.

Continuando con el análisis, las distribuciones para K=9.000 y K=10.000 observadas en la Figura 7 y en la Figura 8, muestran un movimiento hacia una segmentación más granular y repartida, resolviendo en gran medida el problema del "mega-clúster" observado con valores de *k* más bajos.

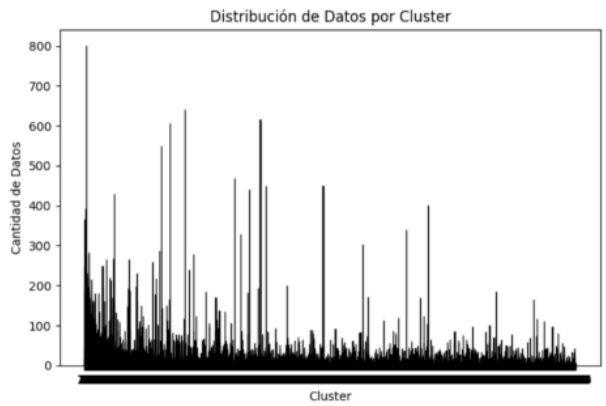


Figura 7: Histograma para k = 9.000

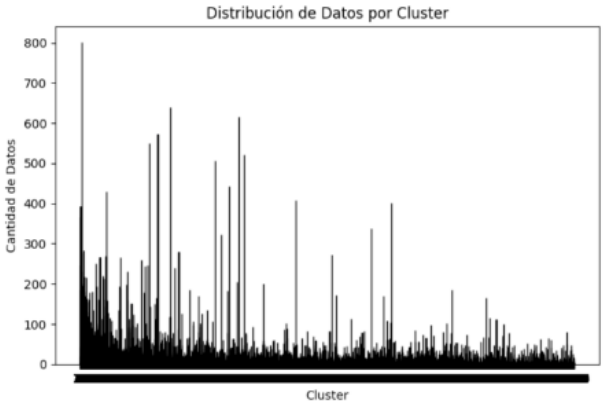


Figura 8: Histograma para k = 10.000

En el caso de k = 9.000 el cambio más notable es la ausencia de un único clúster dominante que agrupe a miles de usuarios. En su lugar, la distribución de tamaños es más equilibrada. El clúster más grande ahora contiene alrededor de 800 usuarios, seguido de otros clústeres de entre de 600 y 400 usuarios. Se observa una población saludable de clústeres de tamaño mediano (entre 100 y 300 usuarios), junto con la esperada "cola larga" de clústeres más pequeños. Con 9,000 centroides, el algoritmo K-Means tiene la flexibilidad suficiente para identificar y separar grupos de interés más específicos que antes estaban consolidados en un único clúster. La distribución se asemeja más a lo que se esperaría en un ecosistema de aprendizaje real: algunos temas o rutas de aprendizaje populares que atraen a grupos considerables de usuarios, y muchos otros temas de nicho que atraen a grupos más pequeños y especializados. Esta segmentación es mucho más prometedora. Las recomendaciones generadas a partir de clústeres de 500 usuarios serían significativamente más personalizadas que las de un clúster de 2,500. El sistema comienza a tener la capacidad de diferenciar entre, por ejemplo, "usuarios interesados en finanzas generales" y "usuarios interesados específicamente en análisis de riesgo financiero".

Para k = 10.000 la tendencia observada en K=9.000 se mantiene. La altura del pico más alto es igual, y se observa una cantidad semejante de clústeres con entre 600 y 400 usuarios, al igual que entre los de 100 a 30 usuarios. Aunque un valor de K=10.000 debería permitir una segmentación aún más fina, según lo que se observa en el histograma el comportamiento es muy similar al de k = 9.000, sugiriendo que no hay un cambio significativo que indique que al seguir subiendo el número de clústeres el comportamiento vaya a mejorar. Esto se corrobora al analizar los histogramas para k = 12.000 y k = 15.000, en donde se observa un comportamiento muy similar, como se puede ver en las siguientes figuras. Al final de este documento, se presenta el ANEXO 1 que incluye histogramas generados para distintos valores de k.

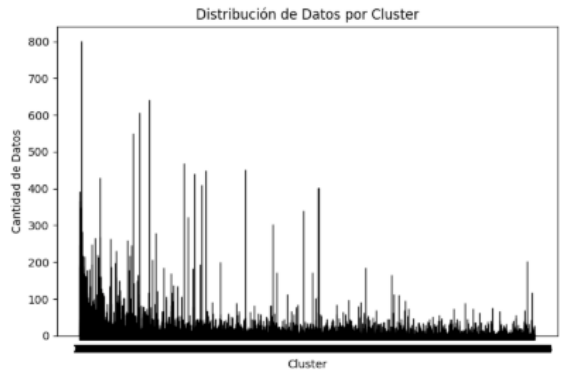


Figura 9: Histograma para k = 12.000

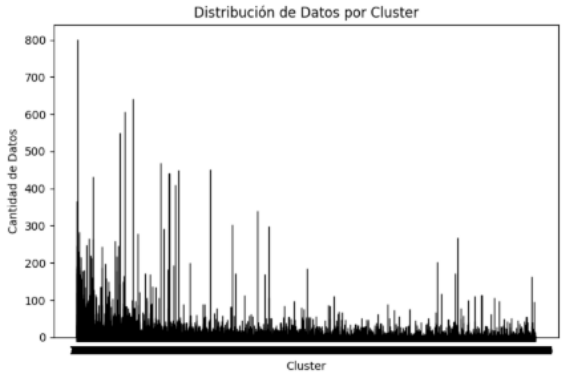


Figura 10: Histograma para k = 15.000

El modelo es capaz de capturar matices de interés más sutiles. Sin embargo, se debe empezar a considerar el riesgo de sobre-segmentación. Si los clústeres se vuelven demasiado pequeños, la cantidad de cursos disponibles para recomendar dentro de cada grupo podría disminuir, limitando la diversidad y novedad de las sugerencias. Este nivel de granularidad es potencialmente muy poderoso para la personalización. No obstante, es aquí donde el equilibrio entre la precisión de la segmentación, reflejado en la mejora de métricas como Silhouette, y la utilidad práctica de los clústeres se vuelve crítico. Un clúster debe tener un tamaño suficiente para contener un catálogo de cursos diverso del cual recomendar.

Los anteriores histogramas demuestran que, a partir de un cierto umbral, antes de k = 9.000 y mayor a k = 4.000, según se ha visto, el modelo K-Means logra una segmentación mucho más significativa y cualitativamente superior. La elección final de *k* se convierte en un compromiso para lograr una alta granularidad para la personalización, favorecido por valores de K más altos, y mantener clústeres de un tamaño robusto que puedan soportar un proceso de recomendación diverso y útil. Los resultados de estas gráficas, en conjunto con las métricas cuantitativas, parecieran indicar que una decisión de seleccionar un valor de *k* en el rango mayor a 4.000 y menor a 9.000 podría ser adecuado.

Los valores de *k* en el rango de 6.000 a 7.000 representan el "punto de inflexión" donde el comportamiento del modelo de *clustering* pasa de una segmentación gruesa a una más granular y significativa. El análisis de sus distribuciones de tamaño, en conjunto con las métricas de evaluación, es fundamental para justificar la elección final del hiperparámetro.

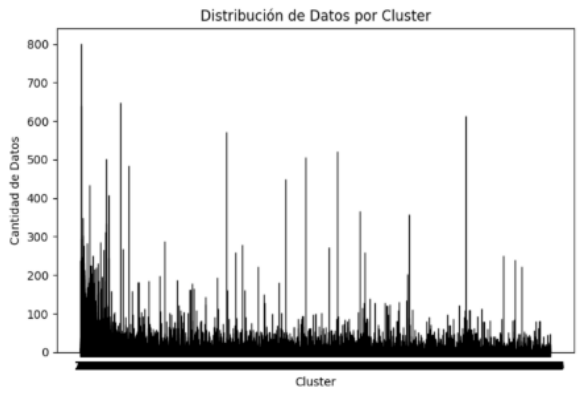


Figura 11: Histograma para k = 6.000

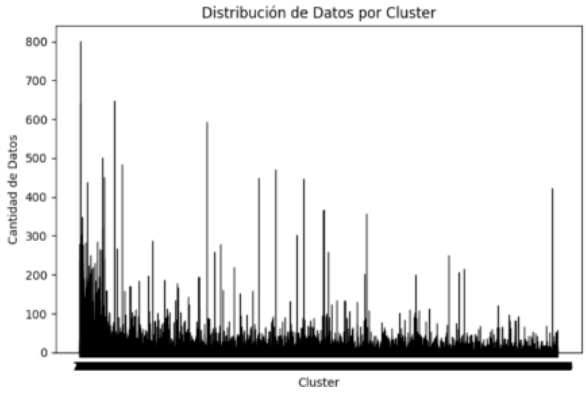


Figura 12: Histograma para k = 7.000

En el caso del histograma para K = 6.000 se observa un cambio significativo con respecto a los valores de *k* más bajos. El "mega-clúster" dominante ha desaparecido por completo. En su lugar, se observa una distribución mucho más saludable, con varios clústeres de tamaños semejantes, que ahora tienen un tamaño grande pero más razonable, con un pico máximo que se sitúa en torno a los 800 usuarios. Existe una población de clústeres de tamaño mediano, de entre 200 y 600 usuarios, lo que indica que el modelo está identificando subgrupos de interés bien definidos. Se mantiene una "cola larga" de clústeres más pequeños, que probablemente representan nichos de interés muy específicos o usuarios con historiales de interacción menos diversos. Este valor de *k* parece ser el umbral en el que el algoritmo K-Means logra romper los grandes grupos heterogéneos y empieza a encontrar una estructura de comunidad más natural dentro de los datos. La distribución de tamaños se alinea bien con la intuición del negocio: hay temas populares que agrupan a cientos de usuarios, pero también una gran diversidad de intereses más específicos.

En el histograma para k = 7.000 la gráfica continúa la tendencia observada en K=6.000. La distribución se vuelve aún más granular. El tamaño del clúster más grande se mantiene, y la cantidad de clústeres en los rangos medianos aumenta ligeramente. Un valor de K=7.000 proporciona una segmentación ligeramente más fina que K=6.000. Sin embargo, la mejora en la estructura de la distribución no es tan drástica como el salto observado al pasar de un k más bajo a 6.000. Esto sugiere que estamos entrando en una zona de rendimientos decrecientes, donde aumentar *k* produce segmentaciones cada vez más similares en su estructura cualitativa.

5.2.1.2 Entrenamiento y aplicación de DBSCAN

Como alternativa a K-Means, se exploró el algoritmo DBSCAN. Al ser un método basado en densidad, a diferencia del K-Means, no requiere especificar el número de clústeres de antemano y puede identificar grupos de formas arbitrarias, así como clasificar puntos como

ruido, lo cual podría ser útil para identificar usuarios con perfiles de interés muy atípicos [26].

El rendimiento de DBSCAN depende críticamente de dos hiperparámetros: *eps*, que es la distancia máxima para que dos puntos sean considerados vecinos, y min_samples, que determina el número mínimo de puntos en una vecindad para que un punto sea considerado un punto central.

Para estimar un valor óptimo de *eps*, se utilizó la técnica de la gráfica k-distance. Se calculó la distancia de cada punto a su vecino número 99 (k = min_samples - 1 = 99) y se graficaron estas distancias en orden ascendente. El "codo" o punto de máxima curvatura en esta gráfica indica un umbral de densidad natural en los datos. La inspección visual y la aplicación de la librería KneeLocator sobre la curva sugirieron un valor teórico óptimo de *eps* = 36.07.

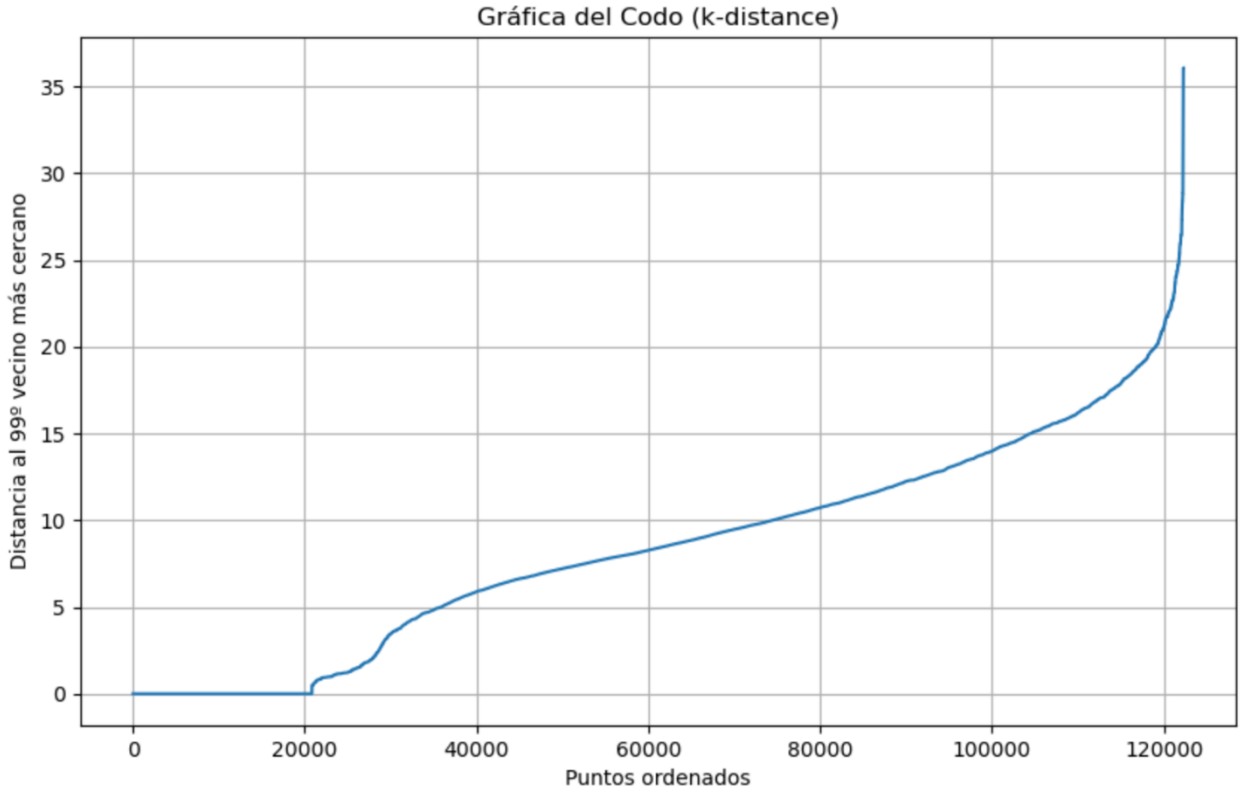


Figura 13: Gráfica del Codo para DBSCAN

Se realizaron varios intentos para ejecutar el algoritmo DBSCAN sobre el conjunto completo de 122,276 perfiles de usuario:

- **Ejecución con *eps* Teórico:** Se intentó ejecutar el algoritmo con el valor de *eps* cercano al óptimo (*eps* ≈ 36). Sin embargo, debido a la alta complejidad

computacional de DBSCAN en *Data*sets de gran tamaño y alta dimensionalidad (O(n log n) o $O(n^2)$ dependiendo de la estructura de datos), la ejecución nunca finalizó, incluso en un entorno de cómputo robusto (Google Colab Pro). Esto indica que, si bien el valor de *eps* podría ser teóricamente correcto, su aplicación práctica era inviable con los recursos disponibles.

- **Ejecución con *eps* restrictivo:** El valor de *eps* con el que se ejecutaba el algoritmo se fue bajando hasta lograr una ejecución en donde se obtuviera un resultado, al menos exploratorio. El valor mucho más pequeño y restrictivo con el que se ejecuto fue de 5. Esta ejecución sí finalizó y generó 1,020 clústeres distintos. No obstante, un hallazgo significativo fue que 36,742 usuarios (aproximadamente el 30% del total) fueron clasificados como ruido, lo que hace que este resultado no sea significativo ni apropiado.

5.2.2. EVALUACIÓN GENERAL Y SELECCIÓN DEL MODELO

Tras el entrenamiento y la evaluación exploratoria de los dos algoritmos de *clustering* candidatos, se procedió a la selección del modelo definitivo que impulsaría el motor de recomendaciones. La decisión se tomó basándose en una evaluación integral que consideró no solo las métricas cuantitativas, sino también el rendimiento computacional, la interpretabilidad de los resultados y la alineación con los objetivos de negocio del proyecto.

5.2.2.1 Comparación entre K-Means y DBSCAN

Si bien DBSCAN ofrece ventajas teóricas como la capacidad de encontrar clústeres de formas arbitrarias, su aplicación en este proyecto demostró ser inviable. La imposibilidad de ejecutar el algoritmo con el parámetro *eps* óptimo debido a limitaciones computacionales y el alto porcentaje de ruido (30%) en las ejecuciones experimentales lo descartaron como una opción práctica. Un sistema que no puede generar recomendaciones para casi un tercio de su base de usuarios sencillamente no cumple con los objetivos de un sistema de recomendación y mucho menos con los objetivos de negocio de la plataforma *KME360*.

En contraste, el algoritmo K-Means demostró ser una alternativa superior para este caso de uso por las siguientes razones:

- **Eficiencia y Escalabilidad:** K-Means es computacionalmente más eficiente y escalable para *Data*sets de gran tamaño y alta dimensionalidad como el utilizado en este proyecto.

- **Cobertura Total:** A diferencia de DBSCAN, K-Means asigna cada punto de datos a un clúster, garantizando una cobertura del 100% de la base de usuarios. Esto es crítico, ya que evita el problema de no poder generar recomendaciones para un subconjunto significativo de la población.
- **Interpretabilidad:** Las agrupaciones generadas por K-Means son relativamente fáciles de interpretar como grupos cohesivos de usuarios con intereses comunes, lo que simplifica la implementación de la lógica de recomendación posterior.

Por estos motivos, se seleccionó K-Means como el algoritmo de *clustering* para el sistema.

5.2.2.2 Justificación de la Elección de k = 6.000

Una vez seleccionado K-Means, el paso final fue determinar el número óptimo de clústeres (*k*). Como se discutió en la sección 0, el análisis cuantitativo de las métricas de evaluación (Coeficiente de Silhouette, Calinski-Harabasz y Davies-Bouldin) no arrojó un único valor de *k* como óptimo de manera concluyente. Por lo tanto, la decisión final se fundamentó en un análisis combinado de estas métricas con la evaluación cualitativa de la estructura de los clústeres.

La elección de k=6,000 se justifica como la solución más equilibrada y pragmática:

1. **Calidad Estructural de la Segmentación:** Como se observa en los histogramas de distribución, K=6,000 es el punto en el que el modelo logra superar la formación de "mega-clústeres" dominantes, generando una distribución de tamaños de grupo más natural y diversa, que se alinea con la lógica de negocio de tener tanto temas populares como nichos de interés.
2. **Equilibrio de Métricas Cuantitativas:** Este valor de *k* ofrece un buen balance entre los indicadores. Obtiene un Coeficiente de Silhouette de 0.51, un valor robusto que indica una buena cohesión y separación, y mantiene un Índice de Calinski-Harabasz (328.8) competitivo, evitando la degradación observada en valores de *k* mucho más altos.
3. **Principio de Parsimonia:** Aunque valores de *k* superiores mostraron mejoras marginales en algunas métricas, el salto cualitativo en la calidad de la segmentación no justificaba el aumento en la complejidad del modelo. K=6,000 se estableció como el "punto dulce" que ofrece la mayor ganancia en estructura con la menor complejidad necesaria.

En consecuencia, el modelo final seleccionado para el despliegue piloto fue un K-Means entrenado con k=6,000 y un random_state=42 para asegurar la reproducibilidad de los

resultados. Este modelo constituye el artefacto principal de la fase de modelado, listo para ser utilizado por el servicio de recomendaciones en tiempo real.

Una vez que el modelo K-Means ha sido entrenado y cada usuario ha sido asignado a un clúster, el sistema está preparado para generar recomendaciones personalizadas en tiempo real. La lógica implementada se basa en un enfoque de filtrado colaborativo basado en modelos y centrado en el usuario. La premisa fundamental es que a un usuario le resultarán relevantes aquellos cursos que han sido populares entre otros usuarios con perfiles de interés similares, es decir, los miembros de su mismo clúster.

El proceso de generación de recomendaciones se activa mediante una solicitud al servicio y sigue una secuencia de pasos bien definida para garantizar una respuesta rápida y pertinente:

1. **Identificación del Vecindario del Usuario:** El primer paso es determinar el grupo de pares o "vecindario" del usuario activo. Al recibir una solicitud con un *username*, el sistema consulta el artefacto del modelo *clustered_users.parquet*. Este archivo actúa como un índice que mapea eficientemente a cada usuario con el identificador del clúster al que fue asignado durante la fase de entrenamiento.
2. **Recuperación del Catálogo de Recomendaciones del Clúster:** Una vez identificado el clúster del usuario, el sistema accede al segundo artefacto, *courses_by_cluster.parquet*. Este archivo contiene, para cada clúster, un catálogo agregado de todos los cursos únicos que han sido consumidos por sus miembros. Esta lista representa el conjunto de recomendaciones potenciales para cualquier usuario perteneciente a ese grupo.
3. **Filtrado de Ítems Consumidos:** Para asegurar que las recomendaciones sean novedosas y no redundantes, es crucial eliminar los cursos que el usuario activo ya ha tomado. El sistema recupera el historial de interacciones del *username, el cual se encuentra dentro del atributo courses* y lo compara con el catálogo de recomendaciones del clúster.
4. **Generación de la Lista Final de Recomendaciones:** Se realiza una operación de diferencia de conjuntos, donde se excluyen los cursos ya consumidos del catálogo de recomendaciones del clúster. La lista resultante contiene ítems que son relevantes para el grupo de interés del usuario pero en los que él aún no ha participado. Esta lista final de identificadores de curso (*Course uuid*) es la que se devuelve a la plataforma *KME360* para ser presentada al usuario.

Este enfoque asegura que las recomendaciones sean tanto personalizadas, al estar

basadas en el clúster específico del usuario, como colaborativas, al aprovechar la "sabiduría colectiva" de los usuarios con perfiles similares presentes en el mismo clúster.

5.2.2.3 Proceso Offline: Entrenamiento del Modelo y Preparación de Datos

Este proceso se ejecuta de manera previa al despliegue en producción y se repite periódicamente para actualizar los modelos con las nuevas interacciones de los usuarios.

1. **Entrenamiento del Modelo de *clustering*:** Los perfiles vectoriales de los usuarios se utilizan como entrada para entrenar un modelo de *clustering* (K-Means, en nuestro caso). El objetivo es agrupar a los usuarios con intereses similares.
2. **Generación de Artefactos del Modelo:** El resultado del entrenamiento se guarda en dos archivos optimizados. Estos archivos se guardan en formato columnar (*Parquet)* y se almacenan en un repositorio especial para objetos, es decir, para guardar los archivos en formato *Parquet*:
    - *clustered_users.parquet*: Un archivo que mapea cada *username* a su ID de clúster correspondiente.
    - *courses_by_cluster.parquet*: Un archivo que contiene, para cada clúster, la lista de los *Course uuid* de los cursos tomados por los miembros de ese grupo.

6. DESPLIEGUE Y RESULTADOS

6.1. ARQUITECTURA DE SOFTWARE

Para la creación del sistema de recomendación contemplado en este proyecto, se diseñó una arquitectura desacoplada que separa el proceso de entrenamiento del modelo del servicio de recomendación en tiempo real. Esta decisión de diseño es fundamental, dado que el entrenamiento de los modelos de *clustering* sobre un gran volumen de perfiles de usuario es computacionalmente intensivo y se realiza con menor frecuencia, mientras que el servicio de consulta de recomendaciones debe ser altamente disponible, rápido y escalable para responder a las solicitudes de los usuarios en tiempo real.

Para la implementación de la lógica, se optó por un enfoque "sin servidor" (*serverless*) mediante microservicios en la nube de *Amazon Web Services (AWS*). Este paradigma reduce la sobrecarga de gestión de infraestructura, facilita la escalabilidad automática según la demanda y optimiza los costos, ya que solo se paga por el cómputo consumido durante la ejecución de las funciones [28]. Esto además es coherente con la definición de la arquitectura del *LMS* en el que se van a implementar las recomendaciones.

La arquitectura se compone de dos grandes bloques funcionales como se ilustra en la Figura 14, el proceso de entrenamiento del modelo (offline), y el servicio de recomendaciones (online).

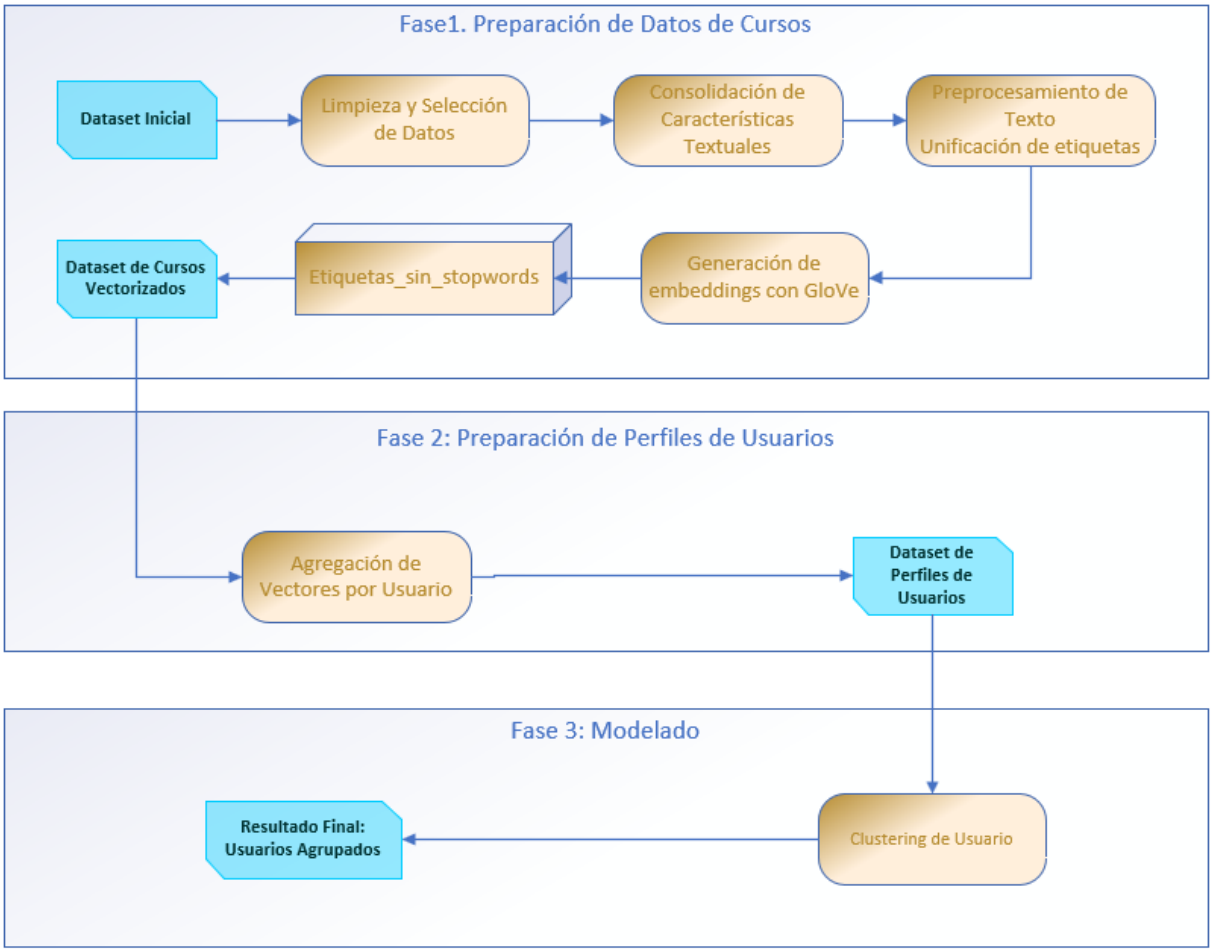


Figura 14: Bloques Funcionales del Sistema de Recomendaciones

6.1.1. Servicio de Recomendaciones en Tiempo Real

Este componente se activa cuando un usuario interactúa con la plataforma *KME360*.

1. **Solicitud:** El *LMS KME360* envía una solicitud *HTTP GET* al punto de entrada del sistema, un *API Gateway* de *AWS*, incluyendo el *username* del usuario.
2. **Lógica de Recomendación:** El *API Gateway* invoca una función *AWS Lambda* escrita en Python. Esta función ejecuta la lógica de recomendación:
    a. Lee el archivo *clustered_users.parquet* desde S3 para identificar el clúster al que pertenece el *username* solicitado.
    b. Lee el archivo *courses_by_cluster.parquet* para obtener la lista de todos los cursos asociados a ese clúster.
    c. Recupera el historial de cursos ya tomados por el usuario.

d. Filtra la lista de cursos del clúster, eliminando aquellos que el usuario ya ha consumido.

3. **Respuesta:** La función Lambda construye una respuesta en formato *JSON* con la lista de los *Course uuid* recomendados y la devuelve al *API Gateway*, que a su vez la reenvía a la plataforma *KME360* para su visualización.
4. **Monitoreo:** Cada ejecución, incluyendo el *username* y las recomendaciones generadas, se registra *en Amazon CloudWatch* para fines de monitoreo, depuración y análisis de rendimiento.

6.1.2. Estrategia para el Arranque en Frío

Un desafío crítico en los sistemas de recomendación es cómo atender a usuarios nuevos que carecen de un historial de interacciones. En nuestro sistema, un usuario sin historial no puede tener un perfil vectorial y, por lo tanto, no puede ser asignado a un clúster. Para mitigar este problema conocido como *user cold start* [2], se propone la siguiente estrategia alternativa para proporcionar valor desde la primera interacción del usuario con la plataforma.

- Cuando el sistema de recomendación recibe una solicitud para un usuario que no existe en el conjunto de datos de perfiles (*clustered_users.parquet*), se activa una lógica de respaldo.
- Esta lógica consultará el catálogo completo de cursos y seleccionará los *N cursos más recientes* que han sido añadidos a la plataforma. Esto tiene sentido dado que los cursos en la plataforma rotan de manera constante, y estarán disponibles los últimos publicados para el presente periodo. De esta forma, el usuario podrá acceder al listado activo de cursos y empezar a construir su historial.
- Adicionalmente, esta lista de cursos recientes puede ser ordenada por un criterio secundario, como la popularidad general (el número total de inscripciones), para asegurar que las primeras sugerencias sean atractivas y hayan sido validadas por la comunidad de usuarios en general.

Este enfoque asegura que todos los usuarios, incluidos los nuevos, reciban una lista de recomendaciones útil y dinámica. A medida que el nuevo usuario comience a interactuar con los cursos sugeridos, su comportamiento se registrará y su perfil vectorial se generará en el siguiente ciclo de entrenamiento del modelo, permitiendo así una transición fluida hacia recomendaciones totalmente personalizadas basadas en *clustering*.

6.2. IMPLEMENTACIÓN DEL SERVICIO DE RECOMENDACIÓN

El siguiente diagrama ilustra el flujo de interacciones entre los diferentes componentes arquitectónicos cuando la Plataforma *KME360* solicita recomendaciones para un usuario específico. El sistema está diseñado utilizando servicios de *Amazon Web Services* (*AWS*) para la lógica de backend, almacenamiento y monitoreo, que es el proveedor que utiliza la empresa dueña de *KME360*.

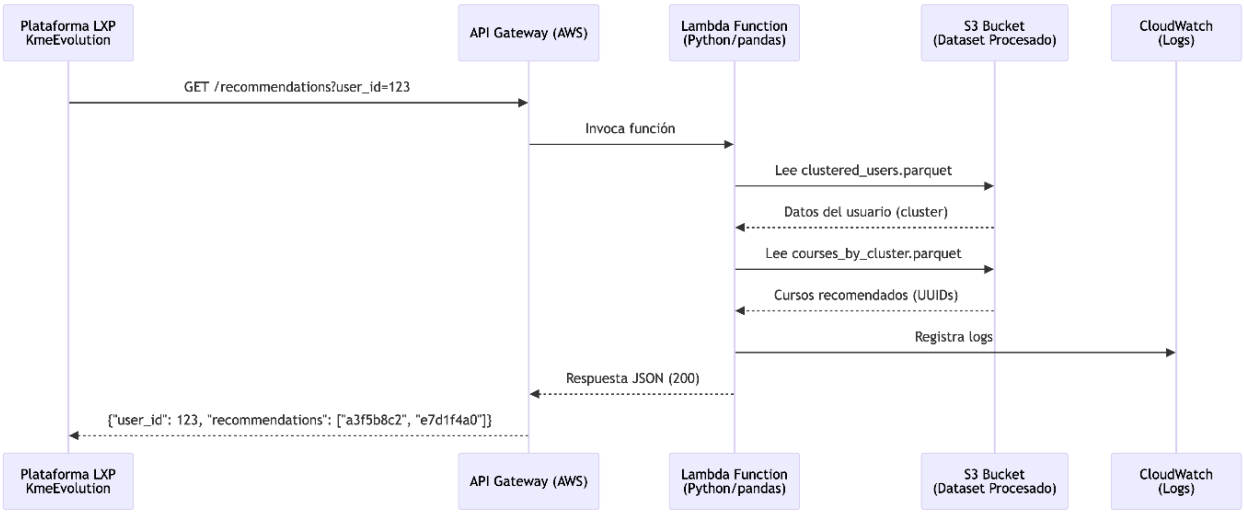


Figura 15: Diagrama de interacción de los componentes del Sistema de Recomendación

6.3. RESULTADOS DE DESPLIEGUE PILOTO

6.3.1. Caso de prueba y resultados

Para validar la implementación del servicio de recomendación, se diseñó un caso de prueba que simulaba el ciclo completo de una solicitud en un entorno controlado. Para esto, se seleccionó un usuario de prueba representativo del conjunto de datos, cuyo historial de interacciones fue utilizado para la construcción de su perfil vectorial. El proceso se dividió en dos etapas: la prueba de la *API* y la integración en una interfaz de desarrollo. El objetivo fue analizar la lista de recomendaciones generada por el sistema para este usuario y evaluar su coherencia semántica y relevancia contextual.

Inicialmente, se realizaron pruebas funcionales directas sobre el servicio REST desplegado en *AWS API Gateway*. Para este fin, se utilizó la herramienta de cliente *API Bruno,* una alternativa a otras herramientas conocidas como *Postman*. El objetivo de esta primera

prueba fue el de verificar que el servicio:

1. Recibe correctamente el usuario (*username)* como parámetro.
2. Invoca exitosamente la función *AWS Lambda*, como inicio del proceso de recomendación.
3. Devuelve una respuesta en formato *JSON* con una estructura válida.
4. Contiene una lista de identificadores de curso (*Course uuid*) como se espera, para poder hacer las recomendaciones.

6.3.2. Ejecución y análisis cualitativo de resultados

Una vez validada la funcionalidad técnica del servicio, se procedió a una evaluación cualitativa de las recomendaciones. Se seleccionó un usuario de prueba, llamado *adminpruebas*, y se ejecutó la solicitud para este usuario. Dado que la respuesta del servicio es una lista de identificadores, se realizó un análisis manual para interpretar su relevancia:

- Se obtuvo la lista de *Course uuid* recomendados para el usuario de prueba, en este caso, 33 cursos fueron recomendados. Con estos identificadores, se puede acceder a los demás datos de los cursos, como su nombre, metodología, etc.

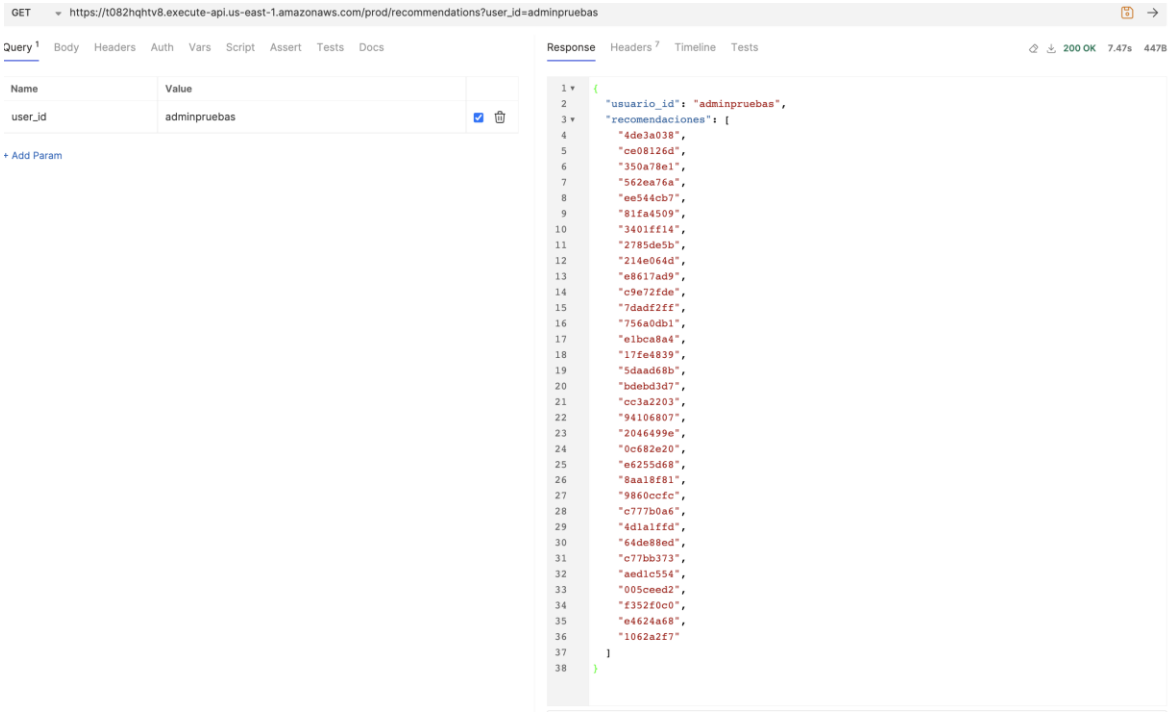


Figura 16: Listado de Cursos (Course uuid) sugeridos para el usuario adminpruebas

Las primeras 12 sugerencias corresponden a los siguientes nombres de cursos:

Course uuid	Nombre del curso
c9e72fde	Aprende a segmentar clientes e incrementa tus ganancias
81fa4509	Fotografía básica para empresarios y emprendedores
1062a2f7	Excel básico, Curso Virtual
7dadf2ff	Cómo llevar las cuentas de tu negocio
756a0db1	Renovación de la matrícula mercantil y de la inscripción de las Entidades sin Ánimo de Lucro
e4624a68	¡Activa tu creatividad para triunfar en los negocios!
562ea76a	Crea tu tienda virtual paso a paso
e8617ad9	Fotografía básica para empresarios y emprendedores
3401ff14	Guía práctica para la creación de empresa
2785de5b	LinkedIn para empresas
ee544cb7	Tus Finanzas en Acción: Claves para Comenzar el Año con Éxito
214e064d	Ventas a pruebas de crisis

Tabla 3: Listado de cursos recomendados para el usuario adminpruebas

- El resultado mostrado en el servicio ya excluye el historial de cursos de dicho usuario por lo tanto en las recomendaciones no debería salir ninguno de aquellos cursos en los que el usuario se encuentra matriculado, sin importar su estado dentro del curso.
- Se dejaron solo los cursos en los que el usuario podría matricularse, es decir, los cursos que no ha tomado y que están disponibles para matricula. Es importante aclarar que la plataforma entrega grupos 4 de cursos para recomendar, es decir, se ofrecen 4 inicialmente, y si el usuario quiere ver más, se le ofrecen otros 4 cursos, y así sucesivamente se entregan más grupos de a 4 cursos al usuario hasta acabarse los cursos que se pueden recomendar. La siguiente tabla muestra un listado de los primeros 4 cursos que se presentan al usuario.

Course uuid	Nombre del curso
7dadf2ff	01-02 Cómo llevar las cuentas de tu negocio
214e064d	Ventas a pruebas de crisis
81fa4509	Fotografía básica para empresarios y emprendedores
cc3a2203	02-03 Cómo llevar las cuentas de tu negocio

Tabla 4: Primer grupo de cursos recomendado al usuario adminpruebas

- Finalmente, se revisó la pertinencia y relación de los cursos recomendados con los cursos ya tomados por el usuario.

Este análisis cualitativo confirmó que las recomendaciones eran coherentes, sugiriendo cursos complementarios y relacionados con el perfil inferido del usuario, como "Excel

básico o "01-02 Cómo llevar las cuentas de tu negocio", y validando que el sistema no recomendaba cursos que el usuario ya había tomado. También se puede observar que se recomendaron los dos cursos de “Cómo llevar las cuentas de tu negocio” que están programados para meses diferentes, y que, a pesar de que en su nombre llevan la indicación de la temática, para el recomendador son dos cursos diferentes pues en su nombre tienen una diferenciación por los meses de ejecución, sin que esto sea un error propiamente.

Por último, se comprobó que ejecuciones posteriores para usuarios diferentes arrojaban recomendaciones diferentes, tal como se puede ver en la siguiente figura:

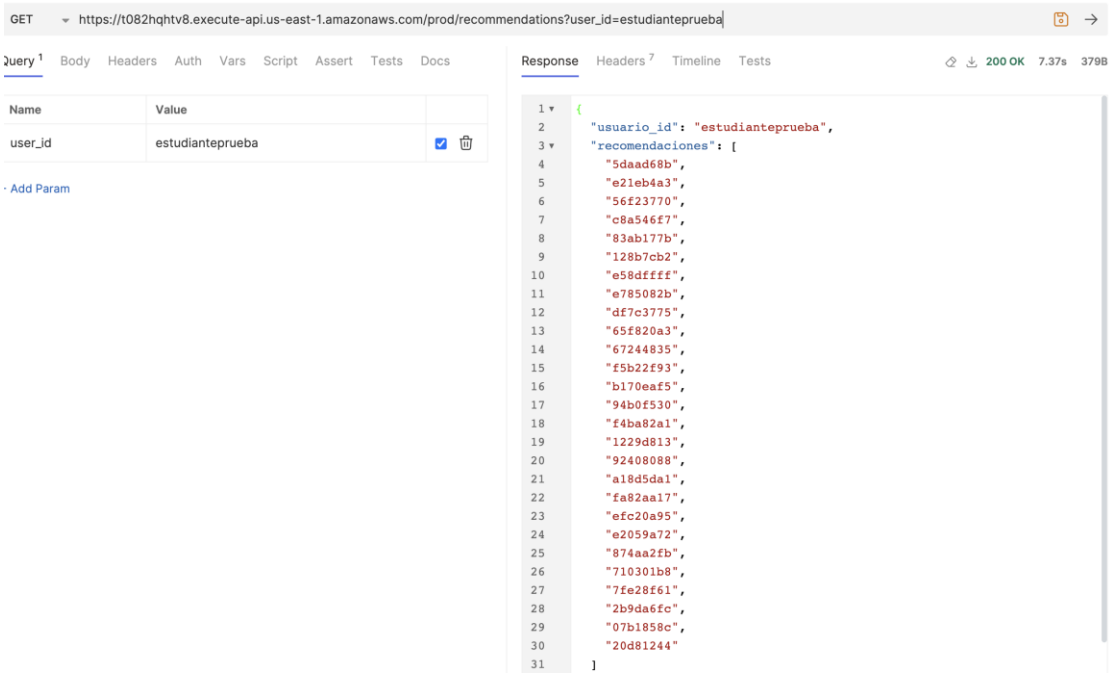


Figura 17: Listado de Cursos (Course uuid) sugeridos para el usuario estudianteprueba

6.3.3. Integración en el Entorno de Desarrollo

Como paso final del despliegue piloto, se realizó la integración del servicio en un ambiente de pruebas de la plataforma *KME360*. Se implementó una llamada desde la interfaz de la plataforma al *API Gateway* del sistema de recomendaciones. Los resultados se presentaron al usuario en la página principal o *home* de la plataforma, dentro de una nueva sección titulada “Eventos recomendados para ti”. Este nombre de “Eventos” corresponde a la forma como la Cámara de Comercio de Cali nombra a lo que en la plataforma son Cursos.

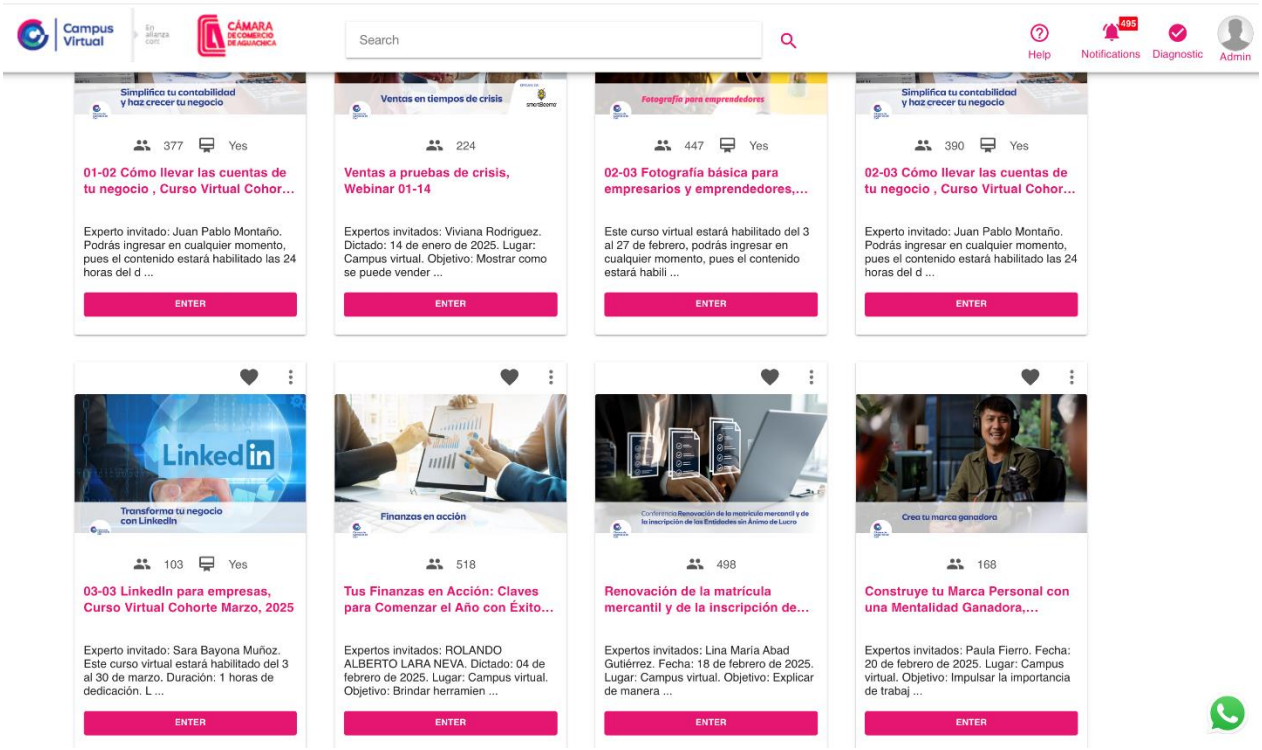


Figura 18: Implementación de la sección "Eventos recomendados para ti" en KME360

Esta implementación final no solo validó la integración técnica entre la plataforma y el servicio *serverless*, sino que también demostró la viabilidad de presentar las recomendaciones de manera fluida y no intrusiva dentro de la experiencia de usuario existente.

6.3.4. Discusión de la Efectividad

El análisis cualitativo de la muestra de recomendaciones sugiere que el sistema es efectivo para:

- Identificar intereses latentes, pues las recomendaciones no se limitaron a cursos con las mismas etiquetas que los ya tomados, sino que se recomendaron temas complementarios y relevantes para el perfil profesional inferido. Esto se puede ver, por ejemplo, con la recomendación de "Cómo llevar las cuentas de tu negocio" que es complementario al curso "Contabilidad para no especialistas" que el usuario ya había tomado.
- Proveer diversidad, dada la gran cantidad de cursos sugeridos que demuestra que el sistema puede ofrecer una amplia gama de opciones, evitando la sobre especialización.

- Validar la segmentación por *clustering*, dada la coherencia temática de las recomendaciones, lo que valida indirectamente que el proceso de *clustering* logró agrupar a usuarios con perfiles de interés genuinamente similares.

6.3.5. Limitaciones y Próximos Pasos

Es importante señalar que esta evaluación es exploratoria y cualitativa. Una validación cuantitativa y a mayor escala requeriría la implementación de pruebas con varios usuarios finales, como se discute en el capítulo 7.2, métricas como la Tasa de Clics (CTR) o encuestas de satisfacción del usuario proporcionarían una medida más directa de la efectividad y precisión percibida del sistema. No obstante, esta prueba piloto con datos reales confirma que el modelo propuesto es funcional y capaz de generar recomendaciones coherentes y contextualmente relevantes.

7. CONCLUSIONES Y TRABAJOS FUTUROS

Este capítulo final presenta las conclusiones derivadas del desarrollo e implementación del sistema de recomendación para la plataforma *KME360*. Se retoman los objetivos específicos del proyecto para evaluar su cumplimiento y se resumen los hallazgos clave. Adicionalmente, se proponen diversas líneas de investigación y desarrollo futuro que pueden extender y mejorar el trabajo aquí presentado.

7.1. CONCLUSIONES

El desarrollo de este proyecto permitió cumplir a cabalidad con los objetivos propuestos, culminando en la implementación de un sistema de recomendación funcional para la plataforma KME360. Como primera conclusión fundamental, se destaca que fue posible identificar y procesar las características clave de los cursos y las interacciones de los usuarios para construir perfiles de aprendizaje personalizados. Se demostró que, a pesar de la ausencia de datos demográficos explícitos o de calificaciones estandarizadas, el historial de interacciones constituye una fuente de datos suficiente y robusta para modelar las preferencias latentes de los aprendices. El simple acto de inscripción en cursos de libre elección validó ser un indicador de interés lo bastante fuerte como para construir perfiles de usuario coherentes. Adicionalmente, la transformación de las características textuales de los cursos en representaciones vectoriales mediante embeddings GloVe fue una estrategia exitosa para capturar las relaciones semánticas entre los contenidos, permitiendo ir más allá de la simple coincidencia de palabras clave y sentando una base sólida para el modelado posterior.

El análisis de los algoritmos de *clustering* reveló que, en un ecosistema de aprendizaje tan diverso como el de la Cámara de Comercio, la segmentación de usuarios es un problema no trivial que no siempre presenta una estructura de grupos clara y evidente. El método del codo no arrojó un valor óptimo inequívoco para *k* en K-Means, y DBSCAN clasificó a una porción significativa de la población como ruido. Esto sugiere que los perfiles de interés de los usuarios se distribuyen de manera continua y heterogénea en el espacio vectorial. La elección de K-Means con un *k* de 6.000 se validó no como la solución matemáticamente "perfecta", sino como una solución pragmática y efectiva que logra un equilibrio entre la granularidad necesaria para la personalización y la utilidad práctica de los clústeres generados.

Finalmente, el diseño de una arquitectura desacoplada y sin servidor (*serverless*) demostró ser una solución idónea para este tipo de problemas en un entorno de negocio. Separar el

entrenamiento intensivo y periódico del modelo del servicio de consulta en tiempo real, que debe ser rápido y escalable, es una decisión arquitectónica fundamental que garantiza la viabilidad y eficiencia del sistema a largo plazo.

En esencia, este trabajo no solo entrega un producto funcional, sino que también aporta un conocimiento valioso sobre la caracterización de usuarios en plataformas *LMS*/*LXP*, la complejidad de la segmentación en *Data*sets reales y la implementación de arquitecturas de machine learning eficientes en la nube. Los resultados del despliegue piloto confirman que el enfoque híbrido basado en modelos es capaz de generar recomendaciones diversas y contextualmente relevantes, sentando una base sólida para mejorar de forma tangible la experiencia de aprendizaje en *KME360*

7.2. TRABAJOS FUTUROS

El sistema desarrollado sienta una base sólida sobre la cual se pueden construir futuras mejoras y extensiones. Las siguientes propuestas se presentan como los próximos pasos lógicos para enriquecer y validar el sistema:

- Implementar un protocolo de evaluación offline riguroso para medir la precisión del ranking de las recomendaciones. Esto implicaría dividir el historial de interacciones en conjuntos de entrenamiento y prueba, ocultar algunos ítems del historial de prueba y medir qué tan bien el sistema puede predecirlos utilizando métricas como Precision@k, Recall@k, y NDCG (Normalized Discounted Cumulative Gain).
- La validación más definitiva del sistema provendría de su interacción con usuarios reales. Se podría diseñar e implementar una prueba A/B, donde un grupo de usuarios recibe las recomendaciones del sistema y un grupo de control no, para medir el impacto directo en métricas de negocio clave como la Tasa de Clics (CTR), la tasa de finalización de cursos o la satisfacción del usuario a través de encuestas.
- Realizar un análisis semántico más profundo de los clústeres generados. Se podría utilizar técnicas de modelado de tópicos o la extracción de las palabras más representativas de cada clúster para verificar si los usuarios agrupados comparten intereses genuinamente coherentes, lo que podría, a su vez, permitir la creación de "etiquetas de clúster" automáticas (ej. "expertos en finanzas", "principiantes en marketing").
- La lista de recomendaciones actual no está ordenada. Un próximo paso importante sería implementar una capa de ranking que priorice las sugerencias dentro de la lista. Este ranking podría basarse en la popularidad del curso dentro del clúster, la

pertinencia del contenido o una puntuación de similitud calculada entre el perfil del usuario y el vector de cada curso recomendado.

- Aunque se propuso una estrategia basada en la recencia, se podrían explorar enfoques más sofisticados. Por ejemplo, al registrarse un nuevo usuario, se le podría presentar una breve encuesta de intereses para generar un perfil vectorial inicial y asignarlo a un clúster de manera más precisa desde el primer momento.

La implementación de estas líneas de trabajo futuro permitiría no solo cuantificar con mayor precisión el rendimiento del sistema, sino también enriquecer su funcionalidad para ofrecer una experiencia de aprendizaje aún más personalizada, adaptativa y efectiva.

8. REFERENCIAS BIBLIOGRÁFICAS

[1] S. Assami, N. Daoudi y R. Ajhorn, «Personalization Criteria for enhancing learner engagement in MOOC platforms.,» de *2018 IEEE Global Engineering Education Conference (EDUCON)*, 2018.

[2] F. Ricci, L. Rokach, Shapira y P. B. Kantor, Recommender System Handbook, New York: Springer, 2011.

[3] F. Isinkaye, Y. Folajimi y B. Ojokoh, «Recommendation systems: Principles, methods and evaluation,» *Egyptian Informatics Journal,* vol. 16, nº 3, pp. 261-273, Noviembre 2015.

[4] J. Itmazi y M. Megias, «Using recommendation systems in course management systems to recommend learning objects,» *The International Arab Journal of Information Technology,* vol. 5, nº 3, pp. 234-240, Julio 2008.

[5] A. Klašnja-Milićević, M. Ivanović y A. Nanopoulos, «Recommender systems in e-learning environments: a survey of the state-of-the-art and possible extensions,» *Artificial Intelligence Review,* vol. 44, nº 4, pp. 571-604, 2015.

[6] R. Wirth, P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer y R. Wirth, CRISP-DM 1.0: Step-by-step data mining guide, SPSS Inc, 2000.

[7] S. Huber, H. Wiemer, D. Scheneider y S. Ihlenfeldt, «DMME: Data mining methodology for engineering applications - A holistic extension to the CRISP-DM model,» de *Procedia CIRP*, 2019.

[8] B. D. Vega Moreno, *Desarrollo de un sistema de recomendación basado en filtrado colaborativo para Big Data mediante Spark y AWS,* Cuenca: Ecuador, 2023.

[9] J. F. Lema Pérez y A. Mora Ruiz, *Desarrollo de Sistema de REcomendacion basado en conocimiento para apoyar la enseñanza del desarrolo web a traves de recursos educativos abiertos.,* Medellín: Tecnológico de Antioquia, 2018.

[10] A. K. Mahmood, «International Symposium in Information Technology. Annual IEEE Computer Conference,» de *International Symposium in Information Technology: Proceedings*, 2010.

[11] S. T. Moncada, *Desarrollo de un sistema de recomendaciones basado en frecuencia*

de consultas y filtrado colaborativo para convocatorias y becas de proyectos de investigación, Santiago de Cali: Universidad de San Buenaventura, 2016.

[12] R. Burke, «Hybrid recommender systems: Survey and experiments.,» de *User Modeling and User-Adapted Interaction 12.*, Fullerton, California: Kluwer Academic Publishers, 2002, pp. 331-370.

[13] X. V. Cadena y N. A. Carantón, *Planteamiento de un sistema recomendador basado en contenido y colaborativo para la elección de libros,* Bogota D.C., 2017.

[14] D. Jurafsky y J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, 3ra ed., 2025.

[15] J. Niyirora, A Brief Review of Mechine Learning for Text, 2024.

[16] C. Petridis, *ext Classification: Neural Networks VS Machine Learning Models VS Pre-trained Models,* Philadelphia, Pennsylvania, 2024.

[17] J. Ravi y S. Kulkarni, *Text embedding techniques for efficient clustering of twitter data,* 2023, pp. 1-11.

[18] A. Pak, A. Ziyaden, T. Saparov y I. Akhmetov, «Word Embeddings: A Comprehensive Survey,» *Computación y Sistemas,* vol. 28, 2024.

[19] S. D. Evale, «Learning Management System with Prediction Model and Course-content Recommendation Module,» *Journal of Information Technology Education: Research,* vol. 16, nº 1, pp. 437-457, 2017.

[20] M. Rahhali, L. Oughdir, Y. Jedidi, Y. Lahmadi y M. Z. El Khattabi, «E-learning Recommendation System Based on Cloud Computing,» de *Lecture Notes in Electrical Engineering*, 2021.

[21] A. F. Hidayat, D. Suwawi y K. Laksitowening, «Learning Content Recommendations on Personalized Learning Environment Using Collaborative Filtering Method,» de *8th International Conference on Information and Communication Technology*, Yogyakarta, Indonesia, 2020.

[22] S. B. Aher y L. Lobo, «Combination of machine learning algorithms for recommendation of courses in E-Learning System based on historical data,» *Knowledge-Based Systems,* pp. 1-14, Octubre 2013.

[23] J. Pennington, R. Socher y C. Manning, «GloVe: Global vectors for word representation,» de *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.

[24] T. Mikolov, K. Chen, G. Corrado y J. Dean, «Efficient Estimation of Word Representations in Vector Space,» de *roceedings of the 1st International Conference on Learning Representations (ICLR)*, 2013.

[25] M. J. Pazzani y D. Billsus, «Content-based recommender systems,» de *The adaptive web*, 2007.

[26] M. Ester, H. Kriegel, J. Sander y X. Xu, «A density-based algorithm for discovering clusters in large spatial databases with noise,» de *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, 1996.

[27] M. Bishop, Pattern Recognition and Machine Learning., Springer, 2006.

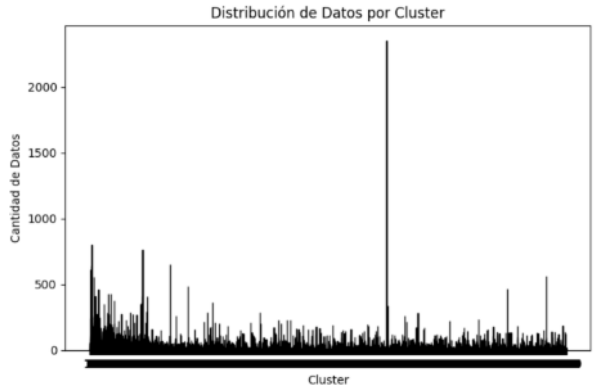
[28] M. Richards, Software Architecture Patterns, O'Reilly Media, Inc., 2015.

[29] W. Krichene y S. Rendle, «On sampled metrics for item recommendation,» *Communications of the ACM,* vol. 65, nº 7, pp. 75-83, 2022.

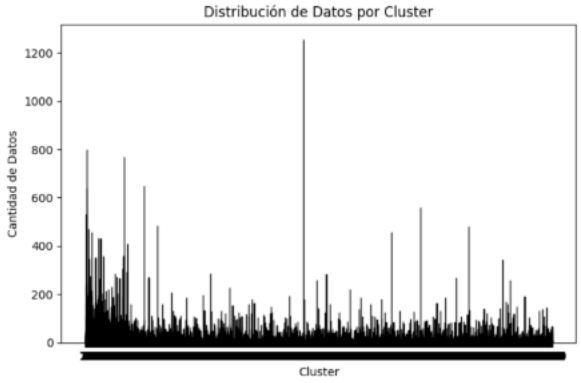
[30] A. K. Shubban, «Recommendation System, Understanding the basic concepts.,» 07 2021. [En línea]. Available: https://www.analyticsvidhya.com/blog/2021/07/recommendation-system-understanding-the-basic-concepts/.

ANEXO 1: HISTOGRAMAS PARA DIFERENTES VALORES DE k

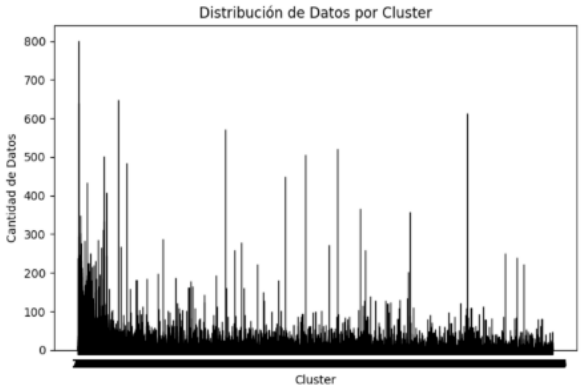
A continuación, se muestran los histogramas resultantes de la ejecución del algoritmo K-Means con diferentes valores de K. Estas gráficas sirvieron para visualizar el tamaño de los clústeres más grandes, insumo fundamental en el análisis realizado para escoger el número K utilizado en el modelo final del sistema de Recomendaciones.



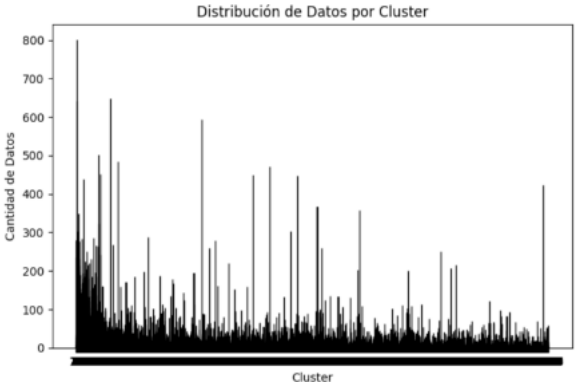
K = 3.000



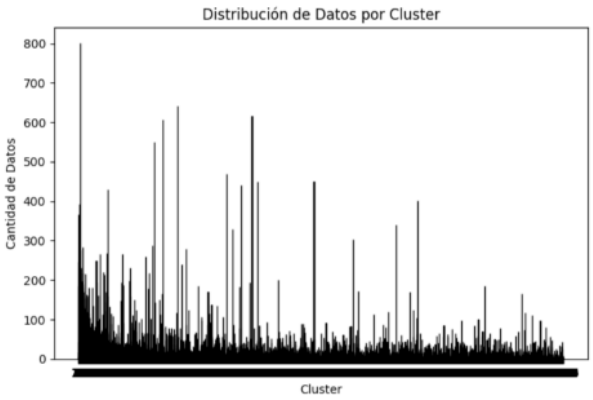
K = 4.000

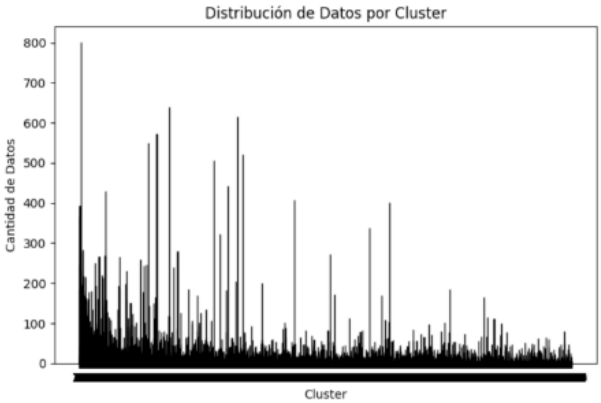


K = 6.000

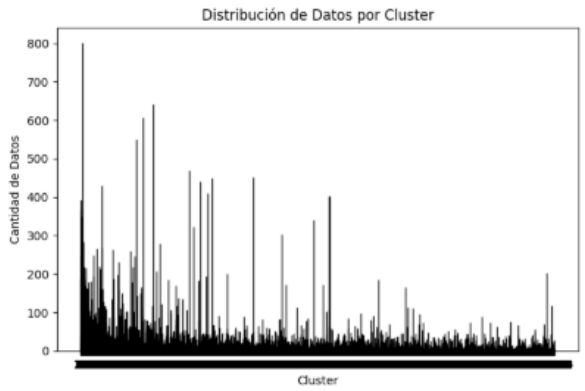


K = 7.000

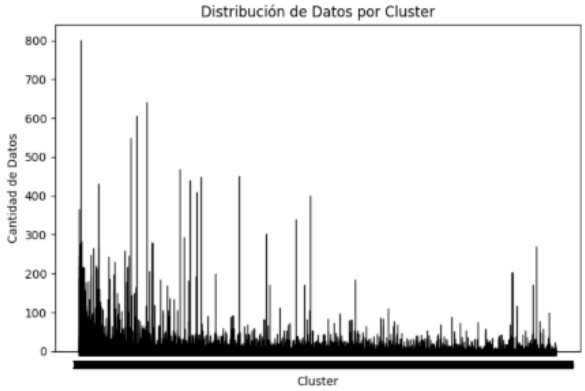




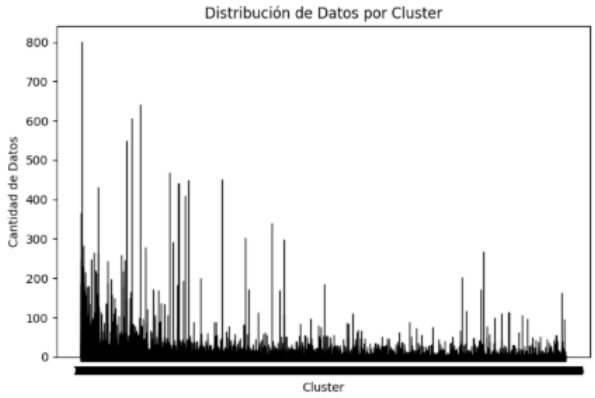
K = 9.000



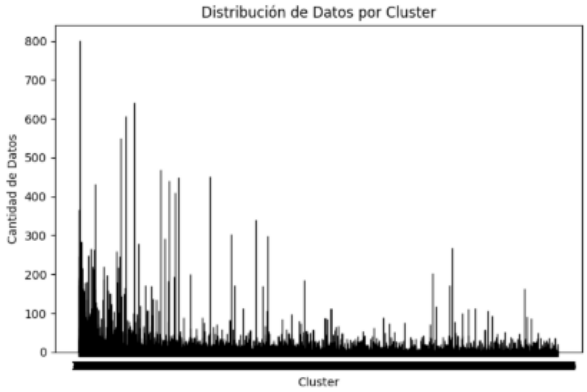
K = 10.000



K = 12.000

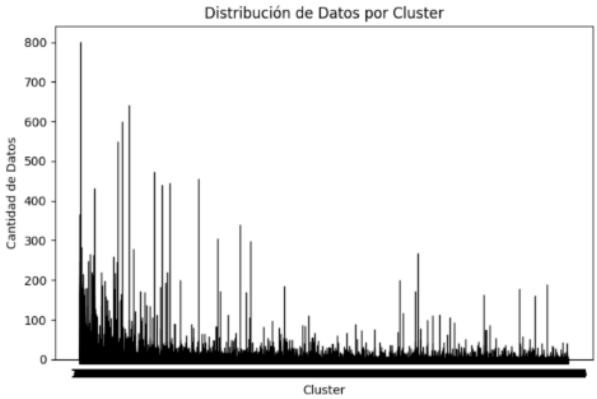


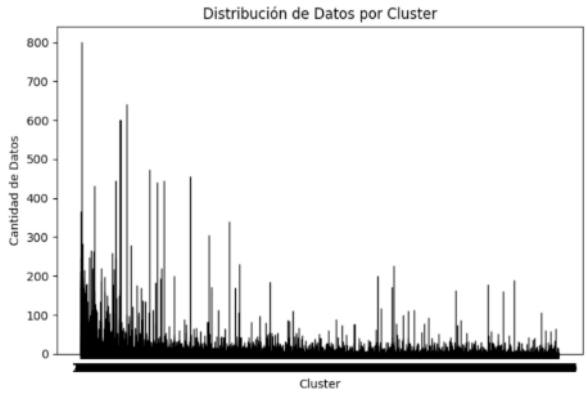
K = 13.000



K = 15.000

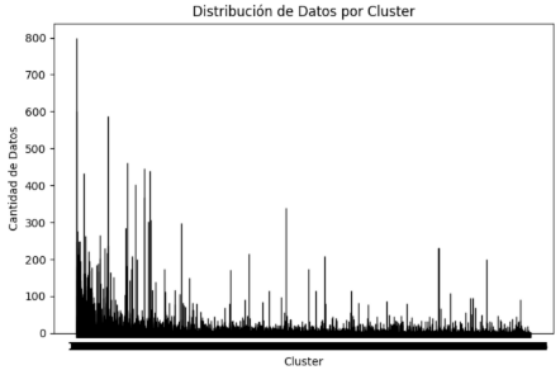
K = 16.000



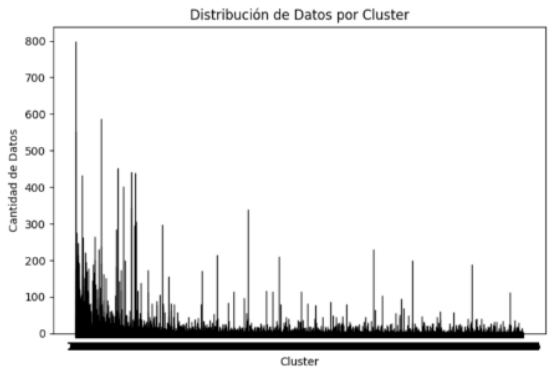


K = 18.000

K = 19.000



K = 25.000



K = 30.000

ANEXO 2: CARPETA CON CÓDIGO IMPLEMENTADO

Con el fin de garantizar la transparencia y la reproducibilidad de los resultados, se adjunta un enlace a la carpeta de Google Drive que contiene los archivos con los códigos fuente del documento de Google Colab y de la función lambda que implementa el servicio REST: https://drive.google.com/drive/folders/13VVihTJEVZp18k9kXvjOle14o41X5HOX?usp=drive_link.

Cuaderno de Google Colab

Este cuaderno muestra cada una de las fases del ciclo de vida CRISP-DM aplicadas durante el presente proyecto, incluyendo:

Preparación de Datos: La carga del dataset inicial, los procesos de limpieza, la transformación de datos textuales, la generación de embeddings de cursos con GloVe y la construcción de los perfiles vectoriales de usuario.

Modelado y Evaluación: El entrenamiento de los modelos de clustering (K-Means y DBSCAN), la ejecución de los experimentos para la selección de hiperparámetros (como la determinación de k mediante el método del codo y las métricas de clustering), y la justificación de la selección del modelo final.

El cuaderno está comentado para facilitar su comprensión y podría ser ejecutado en un entorno de Google Colab para replicar los experimentos y validar los hallazgos presentados en este documento.

Función Lambda

Como parte de la solución final, se desarrolló un servicio REST utilizando **AWS Lambda**, con el objetivo de permitir la consulta de cursos recomendados para un usuario específico, basándose en los resultados del proceso de clusterización.

La función Lambda realiza las siguientes operaciones:

- **Lectura del dataset clusterizado desde S3**: al iniciar, la función descarga el archivo desde el bucket S3 y lo carga en un DataFrame de pandas.
- **Recepción del parámetro username por querystring**: el nombre del usuario se recibe como parámetro en la URL.
- **Identificación del cluster del usuario**: se consulta el cluster al que pertenece el usuario en el dataset.

- **Selección de usuarios del mismo cluster**: se filtran los usuarios que pertenecen al mismo cluster.
- **Recolección de cursos recomendados**: se obtienen todos los cursos de los usuarios del mismo cluster, eliminando los cursos que ya ha realizado el usuario consultado.
- **Devolución de recomendaciones**: el servicio retorna un listado JSON con los cursos sugeridos.