UNIVERSIDAD PERUANA UNIÓN

ESCUELA DE POSGRADO

Unidad de Posgrado de Ingeniería y Arquitectura

Sistema de recomendación de libros basado en algoritmos de similaridad para el Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión

Tesis para obtener el Grado Académico de Maestro en Ingeniería de Sistemas con Mención en Dirección y Gestión en Tecnología de Información

Autor:

Rogelio Mamani Chile

Asesor:

M. Sc. Fredy Abel Huanca Torres

Lima, setiembre de 2022

DECLARACIÓN JURADA DE AUTORÍA DE TESIS

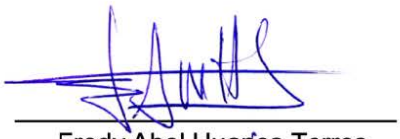
Fredy Abel Huanca Torres, de la Escuela de Posgrado, Unidad de Posgrado de Ingeniería y Arquitectura, de la Universidad Peruana Unión.

DECLARO:

Que la presente investigación titulada: **"SISTEMA DE RECOMENDACIÓN DE LIBROS BASADO EN ALGORITMOS DE SIMILARIDAD PARA EL CENTRO DE RECURSOS PARA EL APRENDIZAJE Y LA INVESTIGACIÓN DE LA UNIVERSIDAD PERUANA UNIÓN"** constituye la memoria que presenta el Ingeniero Rogelio Mamani Chile para aspirar al Grado Académico de Maestro en Ingeniería de Sistemas con Mención en Dirección y Gestión en Tecnología de Información cuya tesis ha sido realizada en la Universidad Peruana Unión bajo mi dirección.

Las opiniones y declaraciones en este informe son de entera responsabilidad del autor, sin comprometer a la institución.

Y estando de acuerdo, firmo la presente declaración en la ciudad de Lima, a los veintitrés días del mes de setiembre del año 2022.



Fredy Abel Huanca Torres

ACTA DE SUSTENTACIÓN DE TESIS DE MAESTRO(A)

En Lima, Ñaña, Villa Unión, a 05 días del mes de setiembre del año 2022, siendo las 4:30 p. m, se reunieron en la modalidad online sincrónica, bajo la dirección del Señor Presidente del Jurado: Mg. Immer Elias Cuellar Rodriguez, el secretario: Mg. Danny Lévano Rodríguez, los demás miembros: Dr. Juan Jesús Soria Quijaite y el Mg. Nemias Saboya Rios y el asesor: M.Sc. Fredy Abel Huanca Torres, con el propósito de administrar el acto académico de sustentación de Tesis de Maestro(a) titulada: "Sistema de recomendación de libros basado en algoritmos de similaridad para el Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión " del Bachiller/Licenciado(a) Rogelio Mamani Chile

Conducente a la obtención del Grado Académico de Maestro(a) en: Ingeniería de Sistemas

(Nomenclatura del Grado Académico)

con Mención en Dirección y Gestión de Tecnologías de Información. El Presidente inició el acto académico de sustentación invitando al candidato hacer uso del tiempo determinado para su exposición. Concluida la exposición, el Presidente invitó a los demás miembros del Jurado a efectuar las preguntas, cuestionamientos y aclaraciones pertinentes, los cuales fueron absueltos por el candidato. Luego se produjo un receso para las deliberaciones y la emisión del dictamen del Jurado.

Posteriormente, el Jurado procedió a dejar constancia escrita sobre la evaluación en la presente acta, con el dictamen siguiente:

Bachiller/Licenciado (a): Rogelio Mamani Chile

CALIFICACIÓN	ESCALAS			Mérito
	Vigesimal	Literal	Cualitativa	
Aprobado	15	B-	Bueno	Muy Bueno

(*) Ver parte posterior

Finalmente, el Presidente del Jurado invitó al candidato a ponerse de pie, para recibir la evaluación final. Además, el Presidente del Jurado concluyó el acto académico de sustentación, procediéndose a registrar las firmas respectivas.

Presidente

Secretario

Asesor

Miembro

Miembro

Bachiller/Licenciado(a)

Dedicatoria

A mis queridos padres Pedro e Hilaria, por brindarme su apoyo incondicional y por ser mi motivación en mi diario vivir.

A mis hermanos que me apoyan a lograr mis metas personales.

Agradecimientos

A Dios por la vida y por las facultades que nos ha concedido, además Él es la fuente de toda la sabiduría.

A mis apreciados docentes de la Universidad Peruana Unión que impartieron sus conocimientos en la Maestría en Ingeniería de Sistemas, mención en Dirección y Gestión de Tecnologías de Información.

Al M.Sc. Fredy Abel Huanca Torres por sus sabios consejos y orientaciones, asimismo por su perseverancia y paciencia para el logro de mis objetivos académicos.

Tabla de contenido

Índice de tablas

Índice de figuras

Índice de anexos

Resumen

El presente trabajo tiene el propósito de optimizar el performance del sistema de recomendación de libros basado en algoritmos de similaridad que permita sugerir recomendaciones de textos con contenidos relevantes a los usuarios del Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión. El método CRISP DM se ha aplicado a un caso de sistema de recomendación de libros para el análisis de datos y optimización del modelo. El método de filtrado colaborativo ha permitido identificar las preferencias de los usuarios y la de otros usuarios con características similares para generar las predicciones; y se ha usado el modelo K-Nearest Neighbor con el algoritmo de similitud de coseno para calcular la mayor similitud entre los usuarios y los libros para ofrecer recomendaciones a los usuarios. En la experimentación se ha obtenido un buen performance del modelo de recomendación con un promedio de 0.83 de exactitud medainte el ajuste de hiperparámetros en el entrenamiento. Sin embargo, en los problemas de clasificación de multiclase se presentan las clases desequilibradas, donde las clases minoritarias obtienen promedios muy bajos; para manejar esta situación se ha usado la técnica de sobremuestreo de las clases minoritarias logrando balancear los datos, lo que ha permitido obtener el promedio total de 0.91 de accuracy, lo que muestra el mejor performance del modelo. La métrica de evaluación que se ha aplicado es la matriz de confusión, obteniendo promedios esperados de precisión (0,91) y de sensibilidad (0.91) lo que evidencia que se puede realizar la predicción de las recomendaciones precisas. Se concluye que se puede lograr la optimización del performance del modelo de sistema de recomendación de libros basados en algoritmos de similitud sin demandar mucha capacidad de cómputo dependiendo del tamaño de la muestra del conjunto de datos.

Palabras clave: Sistema de recomendación, filtrado colaborativo, algoritmo de similitud, K-Nearest neighbor.

Abstract

The purpose of this paper is to optimize the performance of a book recommendation system based on similarity algorithms to suggest text recommendations with relevant contents to the users of the Learning and Research Resources Center of the Universidad Peruana Unión. The CRISP DM method has been applied to a book recommendation system case for data analysis and model optimization. The collaborative filtering method has allowed the identification of user preferences and that of other users with similar characteristics to generate the predictions; and the K-Nearest Neighbor model with the cosine similarity algorithm has been used to calculate the highest similarity between users and books to provide recommendations to users. In the experimentation, a good

performance of the recommendation model has been obtained with an average of 0.83 average accuracy of hyperparameter setting in training. However, in the multiclass classification problems there are unbalanced classes, where the minority classes obtain very low averages; to handle this situation, the technique of oversampling of the minority classes has been used, balancing the data, which has allowed to obtain an overall average of 0.91 accuracy, which shows the best performance of the model. The evaluation metric that has been applied is the confusion matrix, obtaining expected averages of accuracy (0.91) and sensitivity (0.91), which shows that the prediction of accurate recommendations can be performed. It is concluded that the performance optimization of the book recommendation system model based on similarity algorithms can be achieved without demanding much computational power depending on the sample size of the dataset.

Keywords: Recommender system, collaborative filtering, similarity algorithm, K-Nearest neighbor.

CAPÍTULO I

PLANTEAMIENTO DEL PROBLEMA

1.1. Identificación del problema

Los buscadores web son las herramientas para encontrar información en internet u otros sistemas de contenidos con mucha cantidad de información, pero cada vez es más complicado encontrar contenidos de manera rápida y eficiente por la inmensa cantidad de información. Por tanto, en los últimos años, los buscadores de información utilizan los sistemas de recomendación para facilitar a los usuarios para que encuentren contenidos de interés optimizando el tiempo con mínimo de esfuerzo y dedicación.

Las universidades con la finalidad de actualizar los materiales bibliográficos realizan compras de libros cada año, lo que incrementa el número de ítems en sus bibliotecas; en efecto, para manejar la complejidad de la cantidad de información, se han implementado sistemas de apoyo de búsqueda de contenidos; sin embargo, los usuarios necesitan pasar mucho tiempo para encontrar lo requerido y muchas veces con resultados insatisfactorios. Según la entrevista al responsable del TI del CRAI campus Juliaca, que más de 50% del total de usuarios que reportaron que no encontraron el contenido que buscaban, lograron obtener el material requerido con la ayuda de un profesional de la biblioteca.

Las bibliotecas del Centro de Recursos para el Aprendizaje y la Investigación (CRAI) de la Universidad Peruana Unión usan el sistema tradicional de búsqueda de información, donde el usuario interactúa con el sistema través de una caja de texto, en donde debe escribir una cadena de texto compuesto de palabras clave o metadatos utilizados en las bibliotecas para encontrar la información en una colección de contenidos; esto puede resultar problemático si el usuario no usa los términos adecuados, no tiene claridad sobre el tema que requiere buscar o no es capaz de expresar por escrito su necesidad de información, para que el sistema de búsqueda recupere documentos con contenidos pertinentes [1]. Los criterios de búsqueda no son óptimos y requiere manejo de técnicas de búsqueda y mucho tiempo para encontrar contenidos según las necesidades de información de los usuarios.

En la Figura 1, se muestra la interfaz de búsqueda tradicional de libros que utilizan las bibliotecas del CRAI. Se realizó una búsqueda básica de libros usando las palabras claves “Sistemas de información” y se obtiene 132 resultados, lo que significa que el usuario tendría que revisar los contenidos de cada título.

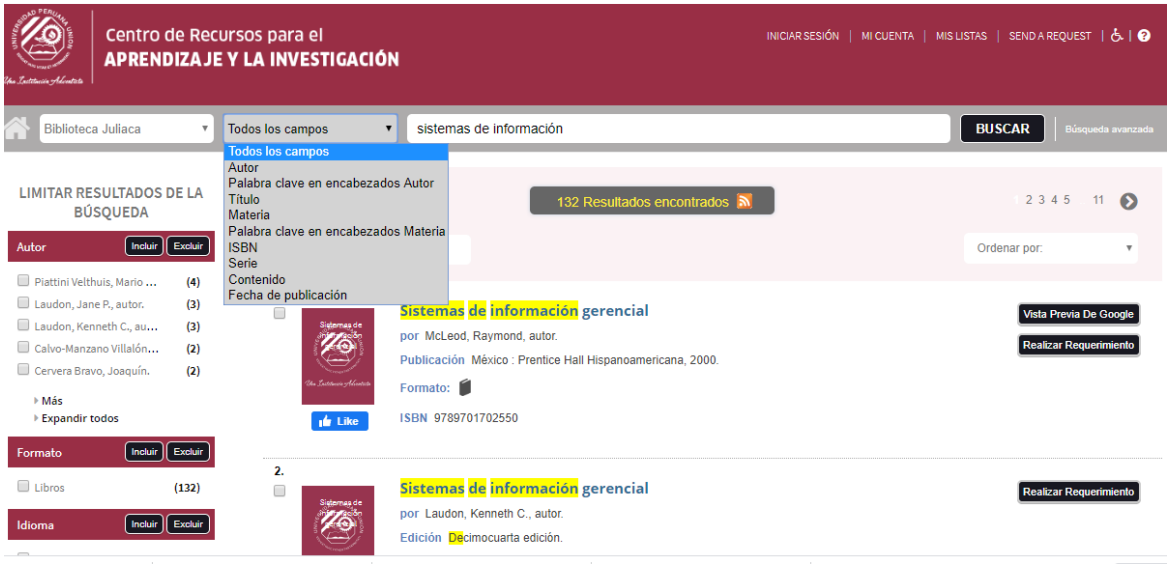


Figura 1. Interfaz de búsqueda tradicional de libros de la biblioteca del CRAI

Además, el sistema actual no cuenta con un módulo de sugerencias ni recomendaciones de libros a los usuarios según sus necesidades de información y no se conoce las preferencias de los usuarios, lo que no permite ofrecer sugerencias de contenidos personalizados. Además, los usuarios satisfechos no pueden colaborar con otros usuarios similares haciendo valoraciones de libros con contenidos relevantes.

Para enfrentar estas dificultades, los sistemas de búsqueda de contenidos optan por el uso de diferentes métodos de sistemas de recomendación como técnica para recuperar la información de manera rápida; sin embargo, dependiendo de la cantidad de información que manejan, los mecanismos de retroalimentación y la capacidad de cómputo, se pueden presentar los problemas de optimización de sistemas de recomendación, dificultad para la recopilación de la mayor cantidad de información a través de los mecanismos de retroalimentación y la optimización de algoritmos para mejorar el performance y minimizar el consumo de memoria [2].

Las bibliotecas del CRAI de la Universidad Peruana Unión también enfrentan algunos de estas dificultades por el aumento de la cantidad de ejemplares de libros que dificultan encontrar contenidos de interés de los usuarios. Según el responsable de Tecnologías de Información del CRAI, actualmente las bibliotecas cuentan con aproximadamente 64,883 ejemplares de libros y 9,000 usuarios respectivamente (CRAI, 2019). En tal sentido, el método de sistema de recomendación de filtrado colaborativo puede ayudar a encontrar la similitud de los usuarios para realizar predicciones y sugerencias de contenidos a los usuarios, pero depende de la matriz de calificaciones y la colaboración con valoraciones proporcionadas por otros usuarios. Por tanto, uno de los principales dificultades de este

método es la escasez de matriz de calificaciones [3], esto no permite predecir recomendaciones eficientes acordes con las preferencias de los usuarios. Entonces la optimización de los sistemas de recomendación es de prioridad para lograr la precisión de las predicciones, para ello se requiere algoritmos con mejores hiperparámetos y buen rendimiento del modelo.

Por consiguiente, en el presente trabajo se abordará el problema de optimización de sistemas de recomendación para sugerir libros con contenidos relevantes acorde a las preferencias de los usuarios de las bibliotecas del CRAI, además de la evaluación y selección del mejor algoritmo de similaridad para lograr dicho propósito.

1.2. Formulación del problema

1.2.1. Problema general

Considerando el problema mencionado anteriormente, se plantea la siguiente interrogante: ¿Cómo se puede optimizar el performance del sistema de recomendación de libros utilizando algoritmos de similaridad para ofrecer recomendaciones según las preferencias de los usuarios del Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión?

1.2.2. Problemas específicos

- El sistema de información actual usa criterios de búsqueda no muy óptimos y es complejo encontrar contenidos de interés para los usuarios por el aumento de la cantidad de libros en las bibliotecas de la universidad; además, no cuenta con un módulo para realizar recomendaciones de libros con contenidos relevantes según las preferencias de los usuarios. ¿Cómo el uso de algoritmos de similaridad en el sistema de recomendación de libros puede ayudar a ofrecer recomendaciones precisas a los intereses de los usuarios del CRAI?

- Para el problema de optimización de performance ¿Cómo se puede optimizar el performance del modelo logrando los mejores hiperparámetros para lograr recomendaciones de libros según las preferencias de los usuarios?

1.3. Objetivos de la investigación

1.3.1. Objetivo general

Optimizar el performance del sistema de recomendación de libros basado en algoritmos de similaridad que permita ofrecer recomendaciones de textos con contenidos

relevantes según los intereses de los usuarios del Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión.

1.3.2. Objetivos específicos

- Definir el método de sistema de recomendación que se adapte a las características del conjunto de datos disponible para la generación de recomendaciones de contenidos de libros.
- Evaluar y validar el algoritmo de similaridad que permita la optimización del sistema de recomendación de libros según las preferencias de los usuarios del CRAI con el mejor performance del modelo.
- Evaluar la precisión y la sensibilidad del modelo de sistema de recomendación de libros.
- Visualizar los resultados de recomendaciones de libros mediante una interfaz de aplicación móvil de sistema Android.

1.4. Justificación de la investigación

El trabajo se justifica en forma teórica, práctica y metodológica, lo que se detalla en seguida:

1.4.1. Justificación teórica

El aprendizaje automático o Machine Learning es aplicado en los diferentes de áreas de conocimiento para resolución de problemas complicados a través de descubrimiento de patrones en datos masivos y es una de las ramas de investigación más difundidos en ciencias de la computación. Las técnicas de machine learning se aplican en menor o mayor grado de velocidad y profundidad en diferentes sectores mejorando la eficiencia de las organizaciones. Consiste en la aplicación de herramientas tecnológicas a través de técnicas y algoritmos que usan un conjunto de datos para detectar patrones en forma automática con el objetivo de realizar predicciones y tomar decisiones en el futuro mediante el autoaprendizaje, permite personalizar servicios y lograr la ventaja competitiva.

El aprendizaje automático da origen a los sistemas de recomendación que permiten ofrecer información, productos y servicios personalizados a los usuarios y se han desarrollado para reducir parte de la sobrecarga de información en la red. Estos sistemas son ampliamente utilizados para ayudar a los usuarios a obtener la información de manera fácil y rápida.

En el presente trabajo se ha abordado el problema de optimización en los sistemas de recomendación de libros, con el propósito de lograr recomendaciones precisas según las preferencias de los usuarios. Existen diferentes tipos de algoritmos que se pueden aplicar para casos específicos, pero considerando el caso del problema mencionado se apuesta por la aplicación de algoritmos de similaridad como una alternativa para abordar el problema y profundizar investigaciones sobre la evaluación de algoritmos existentes para optimizar los sistemas de recomendaciones que se pueden aplicar en diferentes áreas del conocimiento.

1.4.2. Justificación práctica

La aplicación de Machine Learning en los servicios bibliotecarios a través de sistema de recomendación permitirá recomendar contenidos de libros a los usuarios de las bibliotecas del CRAI a través de un aplicativo de sistema Android; además los usuarios pueden interactuar con el sistema valorando los libros con contenidos relevantes, para que éstos puedan ser sugeridos a otros usuarios similares. Asimismo, permite encontrar contenidos relevantes según las necesidades de información para ofrecer las recomendaciones basadas no sólo en libros de contenidos similares, sino también en los documentos más utilizados, o en los mejor valorados por los mismos usuarios.

A su vez recomendar libros a los estudiantes que nunca han pedido prestados libros de la biblioteca y finalmente lograr el uso de los recursos de la biblioteca de manera eficiente. Para ello es necesario reunir la mayor cantidad de datos relacionados con el perfil de usuario y su comportamiento, que permita medir su interés sobre los contenidos de libros o grupo de contenidos relacionados con cursos [2].

Los sistemas de recomendación de libros en las bibliotecas de las universidades deben ser considerados como una herramienta adecuada para la interacción de los usuarios con los sistemas de búsqueda de información para obtención de contenidos para las actividades académicas.

1.4.3. Justificación metodológica

Se justifica por la aplicación del aprendizaje automático supervisado para problemas de clasificación, empleando el modelo K-Nearest Neighbors con el método de filtrado colaborativo para la optimización del sistema de recomendación de libros utilizando algoritmos de similaridad. A continuación, se mencionan los pasos para la construcción de sistemas de recomendaciones: Extracción de datos, preparación de data set, preprocesamiento de datos, procesamiento de datos, modelado, evaluación e implantación.

Los algoritmos de similaridad permiten conocer la similitud de los usuarios y los ítems para descubrir las preferencias de los usuarios, para ello es fundamental contar con el conjunto de datos bien preparado, elegir el modelo adecuado y el método de recomendación que se adapte al data set con que se cuenta; a su vez es importante para lograr la optimización de un sistema de recomendación para sugerir contenidos de libros de acuerdo a las preferencias de los usuarios de la biblioteca del CRAI, ahorrándole tiempo y tener experiencias satisfactorias. Se puede mencionar el caso de Amazon que optaron por algoritmos de similitud de coseno [4].

Cabe mencionar que la revolución industrial 4.0 tiene como una de sus características, mantener abierto los canales digitales de interacción del usuario con los servicios, esto permite responder eficientemente a las exigencias de los usuarios en la búsqueda de contenidos y mejorar los servicios del CRAI, lo que podría convertirse en una ventaja competitiva.

CAPÍTULO II

MARCO TEÓRICO

2.1. Antecedentes de investigación

El estudio realizado por Ji, Yannan, Shi, & Xiang [5] con título “Research on personalized book recommendation model for new readers” tuvo el objetivo de proponer un modelo de recomendación utilizando el algoritmo de filtrado colaborativo para estudiantes que nunca han tomado prestados libros de la biblioteca. El modelo de recomendación de libros genera las listas de recomendaciones para los usuarios objetivo utilizando sus registros de selección de cursos y los datos de préstamos existentes de usuarios conocidos. Se propuso un modelo de recomendación combinado con los datos de selección de cursos y el algoritmo de filtrado colaborativo para resolver el problema de arranque en frío que consiste en que el usuario objetivo no tiene registro de préstamos (nuevos usuarios). Además, analiza diferentes métodos de cálculo de similitud como Euclides, Coseno y Jaccard para resolver el problema de arranque en frío en los sistemas de recomendación. Para el problema de arranque en frío, en la primera etapa, se recomienda usar la similitud de Jaccard para recomendar 5 libros al usuario nuevo, porque de esta manera se puede obtener una alta precisión.

Según Liang & Wan [6] en su trabajo de investigación “The Desing and implementation of Books Recomendation System” tuvo como objetivo el diseño de un sistema de recomendación principalmente basado en filtrado de colaboración de acuerdo con las preferencias de grupos con las mismas aficiones e intereses, los usuarios pueden recomendar información que les interesa a otros usuarios. Para proveer el servicio de recomendación de libros personalizado de alta calidad para los lectores se ha usado el algoritmo de filtrado de colaboración y reunir las necesidades de información personalizado en minería de datos incluyendo los intereses potenciales de los lectores. El sistema de recomendación solo provee servicio de recomendación para usuarios registrados, eso significa que el usuario debe iniciar sesión en la página. Se implementó una página de sistema de recomendación de libros personalizado utilizando la función de recomendación de expertos para recomendar libros a los nuevos lectores y para sugerir nuevos libros a los lectores, lo que ha mejorado la tasa de utilización de libros y la calidad del servicio de información de la universidad.

El trabajo de investigación realizado por Wang et al. [3] con el título “Computing User Similarity by Combining SimRank++ and Cosine Similarities to Improve Collaborative Filtering” con el objetivo de validar la efectividad de la similitud agregada y evaluar la

efectividad de las recomendaciones en el método de filtrado colaborativo basado en el usuario; se han usado cuatro medidas de similitud en experimentos comparativos los cuales son: similitud de coseno, similitud de SimRank++, coeficiente de correlación de Pearson y similitud agregada con datos de MovieLens 100k y DBpedia. Los resultados obtenidos muestran que la similitud agregada supera a los demás en RMSE y MAE.

Tewari & Priyanka [7] han realizado el trabajo de investigación denominado “Book Recommnedation System based on collaborative filtering and association rule mining for college students” para elaborar un nuevo enfoque de un sistema de recomendación de libros en la web, para estudiantes universitarios y académicos . Se ha utilizado las técnicas de clasificación de minería de datos para generar un conjunto de reglas de agrupamiento con la finalidad de clasificar los registros de datos en un conjunto de tres clases de editores, análisis de comportamiento de compra del cliente. Además, se ha utilizado el método filtrado colaborativo basado en usuarios para recomendaciones personalizados basándose en la similitud de perfil de interés de los usuarios. Se ha logrado un sistema que recomienda libros a los usuarios en función a los precios y editores preferidos.

En otra investigación realizado por Tewari, Kumar & Barman [8] “Book Recommendation System based on combine features of content based filtering, collaborative filtering and association rule mining” el trabajo presenta un sistema de recomendación basada en una combinación de características de filtrado de contenidos, filtrado colaborativo y reglas de asociación de minería de datos con la finalidad de lograr recomendaciones eficientes y eficaces a los comprados de libros. El sistema de contenido ofrece información relevante a los compradores basado en la historia de la data de compra de libros, dando el contenido de la visión general de los libros en la que el comprador está interesado utilizando el filtrado de contenido. El filtrado colaborativo se ha usado porque se basa en la opinión de otros usuarios y se tomado el promedio ponderado de la calificación del usuario objetivo. El sistema de recomendación funciona en offline y almacena recomendaciones en el perfil web del comprador, y cuando se conecte en la siguiente vez las recomendaciones se generan automáticamente.

Según Aygün & Yildiz [9] en su trabajo de investigación denominado “Development of contenido based book recommendation system using genetic algorithm” con el objetivo de evaluar, interpretar y predecir las próximas elecciones basado en el interés. El sistema de recomendación híbrido se desarrolló con algoritmo genético basado en notas de libros, como un modelo de red social de libros en el cual se encuentran lectores y libros, cada usuario tiene una biblioteca virtual para leer, que puede actualizar de acuerdo a las revisiones y evaluaciones del libro; se han obtenido resultados altamente satisfactorios al

usar el nuevo sistema de recomendación de libros basados en lecturas de libros y revistas, comentarios y nuevos libros que pueden atraer el interés del lector.

El trabajo de investigación realizado por Tian, Zheng, Wang, Zhang & Wu [10] titulado "College library personalized recommendation system based on hybrid recommendation algorithm" tuvo el objetivo de diseñar un sistema de recomendación personalizado para bibliotecas universitarias basado en un algoritmo de recomendación híbrido. Se ha aplicado el filtrado colaborativo y algoritmo de recomendación basado en contenidos, realizando la clasificación de lectores, el establecimiento de matriz de ranking de usuarios por items, la construcción de un modelo de espacio vectorial y el cálculo de similitud entre usuarios, considerando las características de los libros y lectores universitarios. Además, se utilizó la plataforma de Big Data de Spark para lograr el sistema de recomendación de libros personalizado y mejorar la tasa de utilización de los libros. Se ha realizado el experimento con datos de la Biblioteca de Universidad de Tecnología de Mongolia Interior, obteniendo resultados que explican que los métodos híbridos proporcionan recomendaciones con mayor precisión que los enfoques nativos.

La investigación realizada por Zhu [11] con el título "A book recommendation algorithm based on collaborative filtering" tuvo el objetivo de investigar el algoritmo de recomendación colaborativa y proponer un método sobre el agrupamiento de K-means y K-nearest neighbor para dividir los grupos de usuarios. Para el experimento se ha utilizado la data histórica del proveedor de libros en audio de Douban. Después del análisis, según las características de la estructura de datos, se propone el método de recomendación híbrido. Para los usuarios con datos históricos, este método puede mejorar la precisión de la recomendación. En cuanto al método de agrupamiento el de mayor precisión es K-nearest neighbor.

Según el trabajo de investigación de Priyanka, Tewari, & Barman [12] titulado "Personalised book recommendation system based on opinion mining technique" presenta un sistema de recomendación de libros online para usuarios que buscan libros, considerando las características específicas de un libro; el objetivo principal fue desarrollar una técnica con recomendaciones de muchas opciones de libros para usuarios de acuerdo a las características específicas de los libros que son del interés de los usuarios. Las recomendaciones están basadas en la combinación de características de clasificación y técnica de minería de opinión. El sistema trabaja en modo offline sin problemas.

En Colombia, en la Universidad de Córdoba, se ha realizado una investigación titulado "Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje

basado en la percepción del usuario: caso Rodas" cuyo objetivo fue diseñar un sistema de recomendación de objetos de aprendizaje en repositorios basado en el filtrado colaborativo adaptando el algoritmo K-vecinos (k-nearest neighbors) fundamentado en la percepción que los usuarios tienen de usabilidad y utilidad sobre los objetos de aprendizaje. Se desarrolló el sistema con la finalidad de mejorar la búsqueda de objetos de aprendizaje; además se implementó un sistema de valoración de objetos de aprendizaje para que los usuarios participen calificando dichos objetos. Se ha utilizado las técnicas estadísticas sobre el reporte de las valoraciones que dio un usuario sobre los ítems. Se validó la eficacia de las recomendaciones utilizando data mining aplicado a un grupo de usuarios seleccionados aleatoriamente. Las recomendaciones fue de utilidad y ayudaron en el ahorro de tiempo en las búsquedas [13].

En la Universidad Nacional de Colombia se ha realizado una investigación "Propuesta de un sistema de recomendación hibrido semántico de objetos de aprendizaje en el área de educación ambiental basado en linked data" con el objetivo de proponer una arquitectura para realizar recomendaciones de objetos de aprendizaje. Se utilizó las tecnologías de datos asociados con ontologías para recuperar datos desde la Web. Como resultado se desarrolló un sistema de recomendación híbrido cuya arquitectura está basada en un modelo de capas, que contiene objetos que se relacionan semánticamente a un tema que desea consultar, los cuales son clasificados por relevancia. El sistema usa las técnicas de filtrado colaborativo y basado en contenido, para descubrir las preferencias y conocer las características de un usuario en particular. Para la extracción de datos de los metadatos de los repositorios de objetos de aprendizaje se ha usado Linked data [14].

En Perú, en la UNMSM se ha realizado una tesis titulada "Sistema de recomendación de libros basado en ontologías asociadas a tesauros: el caso de la Biblioteca de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional Mayor de San Marcos" con el objetivo de desarrollar y lograr la implementación de un sistema de recomendación de libros que ayude a seleccionar un material adecuado. Se ha utilizado las técnicas de recomendación apoyado en ontologías asociadas a tesauros. Los resultados muestran que el sistema recomienda al seleccionar los campos de estudio y permite generar una retroalimentación por medio de las calificaciones y comentarios [15].

2.2. Bases teóricas

2.2.1. Sistemas de recomendación

Según Robillard, Maalej, Walker & Zimmermann [16] el sistema de recomendación se define como un software de aplicación que provee información de ítems estimados para

ser evaluado para un trabajo de ingeniería de software en un entorno dado. El potencial interés para los usuarios es que los sistemas de recomendación proveen sugerencias de ítems respondiendo a las preguntas para realizar una actividad.

Los sistemas de recomendación permiten el filtrado de toda la información que se puede encontrar mostrando los resultados que se consideran de interés para los usuarios, y que faciliten la toma de decisiones sin invertir mucho tiempo ni esfuerzo extra (Kabassi, 2010, citado por Álvarez [17], p. 11).

También se define como un sistema que tiene la capacidad de analizar y procesar la data histórica de los usuarios, de los productos o contenidos para transformar en conocimiento práctico, es decir, determina qué producto puede ser interesante para un usuario [4]. Los objetivos de sistemas de recomendación pueden ser variados desde principales como incrementar las ventas hasta operativos y técnicos como lograr la relevancia, la novedad, la serenidad e incremento de diversidad de recomendaciones. Sin embargo, los sistemas de recomendación tienen como meta principal operativo a la relevancia [18].

Ventajas de sistemas de recomendación

Para Núñez [19] ofrecen varias ventajas tanto para los usuarios como para las instituciones que ofrecen servicios. A continuación, se mencionan algunas de ellas:

- Mejora la satisfacción del usuario, ya que le ayuda a encontrar nuevos contenidos de su preferencia con facilidad y mínimo de esfuerzo.
- Ahorro de tiempo, se puede encontrar contenidos u objetos que necesita con menos tiempo.
- Ofrecer recomendaciones personalizadas de acuerdo al perfil de los usuarios.
- Colaboración de los usuarios actuales con la valoración de los ítems relevantes.

2.2.1.1. Clasificación de los sistemas de recomendación

Muchas de las recomendaciones se basan en las estimaciones de valoraciones (opiniones) que los usuarios hacen de determinados ítems, según estas estimaciones, estos pueden ser recomendados a los usuarios o a un usuario de manera más personalizada. Los sistemas de recomendación logran la personalización basándose en una arquitectura de tres elementos principales: datos históricos, datos de entrada y un algoritmo. Según Oliveira (2012) citado por Tarazona, Chávez y Ferro [20] los métodos de

recomendación se clasifican en: basado en filtrado colaborativo, basado en contenido, basado en conocimiento y sistemas híbridos.

2.2.1.1.1. Sistemas de recomendación basados en contenidos

Son los primeros métodos propuestos para los sistemas de recomendación, según Lee et al. (1997) citado por Phino [21] permite recomendar documentos de textos mediante la comparación de sus contenidos y el perfil de cada usuario. Se requiere analizar las propiedades de los ítems preferidos que el usuario ha visto o ha utilizado y la información sobre el comportamiento de cada usuario, el mismo que sirve para construir sus perfiles.

Esta técnica basada en contenidos se fundamenta en la idea de recomendación de nuevos ítems con categorías similar a estos preferidos por los usuarios actuales. Las categorías preferidas por un miembro del grupo son almacenadas en el perfil de ese usuario; estas categorías son derivados de descripciones de ítems existentes consumidos por el usuario [22].

En este enfoque los ítems a recomendar han sido previamente valorados por los usuarios, esto puede presentar algunos inconvenientes como la sobre especialización, subjetividad del contenido y problema del nuevo usuario [19].

Este tipo de recomendación usa distintos enfoques para integrar el conocimiento como: razonamiento basado en casos, uso de ontologías, sistema de recomendaciones basados en restricciones y relaciones de preferencia [23].

2.2.1.1.2. Sistemas de recomendación de filtrado colaborativo

Este método de recomendación se considera de mayor fiabilidad, ya que utiliza la información relativa a las preferencias de los usuarios; además se basa en las preferencias de otros usuarios con características similares. El concepto de “vecinos más cercanos” es utilizado para descubrir un historial similar de la base de datos con la historial de un usuario nuevo. Consiste en comparar a un usuario activo con cualquier otro usuario para encontrar las coincidencias o similitudes más cercanas, y utilizar las calificaciones del usuario actual para hacer recomendaciones.

El filtrado colaborativo está basado en la idea de recomendación de ítems que son derivados desde las preferencias de vecinos más próximos (nearest neighbors); es decir, usuarios con preferencias similares a los usuarios actuales [22]. Sin embargo tiene limitaciones cuando los elementos tienen pocos atributos que permitan describirlos y cuando las preferencias de los usuarios están basados en sensaciones [24].

Algoritmo K-Nearest Neighbors

Según Ridwang et al. [25] los algoritmos se definen como estructuras o armazones que se aplican al lenguaje computacional o programación con el propósito de ayudar en la resolución de problemas donde hay datos de entrada y salida como resultado de un proceso que se ha ejecutado.

El algoritmo de K-Nearest Neighbors (KNN) es un método utilizado por aprendizaje supervisado, cuyo propósito es clasificar nuevos elementos según los atributos y datos de entrenamiento. Tiene varias ventajas, es de fácil implementación, no es paramétrico o sea no se necesita que los datos estén distribuidos teóricamente [26] y es más efectivo especialmente cuando los datos de entrenamiento tienen mucho ruido y son masivos. Sin embargo este algoritmo tiene un alto costo de computación debido al cálculo de distancia desde cada instancia de consulta en todo el entrenamiento [27], debido a que la clasificación está basado en la comparación de la similitud de los datos de observación con cada uno de los datos del conjunto de entrenamiento[26].

La clasificación es el problema para predecir etiquetas de clase discretas para patrones sin etiqueta basados en observaciones; por lo que tiene como objetivo aprender un modelo funcional F que permita una predicción razonable de la etiqueta de clase Y para un patrón desconocido X.

La clasificación KNN se basa en que los patrones más cercanos a un patrón destino, para el cual buscamos la etiqueta, ofrecen información útil sobre la etiqueta. Los patrones más cercanos a K en el espacio de datos se asigna la etiqueta de clase a que pertenece, para ello requiere una medida de similitud. También se aplica a problemas de clasificación multiclase, donde para un patrón desconocido predice la etiqueta de clase de la mayoría de los patrones más cercanos a K en el espacio de datos.

Las técnicas de este modelo son fuertes en el caso de conjuntos de datos grandes y dimensiones bajas [28]. Además, es muy sensible o depende de dos elementos: primeramente, la variable K, con distintos valores de K se puede obtener diferentes resultados, por lo que este valor se elige con pruebas de varias instancias. El segundo es la métrica de similitud que se utiliza, establece la relación de cercanía entre los puntos de datos de entrenamiento.

El algoritmo más usado en filtrado colaborativo es K-Nearest Neighbor (KNN) que significa vecinos más cercanos y tiene dos perspectivas definidas: KNN por usuario y KNN por ítem, que habitualmente se conocen como filtrados colaborativos por usuario y por ítem.

Filtrado colaborativo basado en usuario

En filtrados colaborativos basados en usuario, se realizan recomendaciones a éste los ítems que han sido de preferencia a usuarios similares.

Un ejemplo que puede ilustrar es la promoción de boca a boca en la cual las opiniones de los amigos y familiares son muy importantes para tomar una decisión. En escenarios online, estos son reemplazados por Nearest Neighbors (vecinos más cercanos) que son usuarios que tienen preferencias similares a las de un usuario actual. Por lo tanto, la calificación de un ítem por parte de un usuario especifica en qué medida un ítem le puede gustar a otro usuario; con ello, las predicciones se pueden estimar en qué grado un usuario preferirá un ítem que no consumió hasta ahora [22].

Filtrado colaborativo basado en ítems

En cambio, en filtrado colaborativo basados en ítems, al usuario se le recomienda los ítems que son similares a ítems que han gustado a otros usuarios similares.

El funcionamiento de sistemas de recomendación está basado en la colaboración de otros usuarios con intereses similares al usuario actual, proceso que permitiría ofrecer recomendaciones personalizados a los usuarios de la biblioteca del CRAI.

2.2.1.1.3. Sistemas de recomendación basado en conocimiento

En el enfoque basado en conocimiento, las recomendaciones de ítems están basadas en deducciones de necesidades e intereses particulares de los usuarios, para ello se utiliza el conocimiento sobre cómo un ítem determinado responde a la necesidad de un usuario específico [19].

Estos sistemas de recomendación se centran en un conocimiento profundo de los elementos o ítems ofrecidos; además datos como el conjunto de reglas o métricas de similitud y conjunto de ítems. Dependiendo de los requisitos de usuario (preferencias) que se tiene, las reglas o restricciones describen qué ítems deben recomendarse [22].

2.2.1.1.4. Sistemas de recomendación híbridos

Las recomendaciones híbridas ayudan a subsanar limitaciones específicas de un método de recomendación con la complementación de otro método [22]. En los sistemas de recomendación híbridos se puede aprovechar las ventajas de los algoritmos basados en contenido y filtrado colaborativo para combinarlos y luego realizar recomendaciones más efectivas.

El objetivo del desarrollo de los sistemas de recomendación híbridos es lograr la mejor precisión en la re comendación. El peso de recomendación híbrida se basa en derivar recomendaciones realizando una combinación de los resultados calculados por recomendadores individuales [22].

En la Tabla I se muestra los enfoques de sistemas de recomendación híbrida con sus métodos y la forma de calcular los pesos, donde:

RECS: Conjunto de recomendadores.

S: Predicción de recomendador individual.

Score: Puntaje de ítem.

TABLA I
ENFOQUES DE SISTEMAS DE RECOMENDACIÓN HÍBRIDA.

Método	Descripción	Cálculo
Ponderado	Predicciones (S) calculados de recomendadores individuales.	Score(item)=$\sum_{rec \in RECS} S(item, rec)$
Mezclado	Predicciones calculados de recomendadores individuales (S) combinado en un resultado de recomendación.	Score(item)=zipper-function(item, RECS)
Cascada	La predicción de un recomendador (n-1) es usado como entrada para el siguiente recomendador (n).	Score(ítem)=S(ítem,rec_n) ← S(ítem,rec_{n-1}) ← …

Nota. Adaptado de Group Recommender Systems por Ferfernig, Boratto y Tkalcic; p. 12.

2.2.1.2. Dimensiones y métricas de evaluación de sistemas de recomendación

En los últimos años ha aumentado el desarrollo de diferentes sistemas de recomendación, por tanto, para la evaluación de recomendaciones, primeramente, se necesita identificar las dimensiones y las diferentes métricas para cuantificar cada dimensión.

2.2.1.2.1. Dimensiones

Para Robillard et al [16] existen una variedad de dimensiones para desempeñar una evaluación eficiente de un sistema de recomendación; algunas de éstas describen las características cualitativas y otras las cuantitativas. En la Tabla II, se presentan las dimensiones categorizadas:

TABLA II
CATEGORIZACIÓN DE DIMENSIONES

Centrado en recomendaciones	Centrado en usuario	Centrado en sistema	Centrado en entrega
Exactitud	Confiabilidad	Robustez	Usabilidad
Cobertura	Novedad	Tasa de aprendizaje	Preferencia de usuario
Diversidad	Cambio beneficioso	Escalabilidad	
Confianza de recomendador	Utilidad	Estabilidad	
	Riesgo	Privacidad	

En la Tabla III, se presenta un breve resumen de las dimensiones con sus métricas y el tipo de variable:

TABLA III
RESUMEN DE MÉTRICAS

Dimensión	Métrica / técnica	Tipo
Exactitud	*Calificación*: Error cuadrático medio, normalizado RMSE, error absoluto medio, normalizado MAE. *Ranking*: Medida de rendimiento normalizada basada en la distancia, Spearman's, Kendall's, normalizado con descuento con ganancia acomulada. *Clasificación*: Precision, Recall, tasa de falsos positivos, especifidad, F-medida, curva característica de funcionamiento del receptor.	Cuantitativa.
Cobertura	Cobertura de catálogo, cobertura de catálogo ponderada, predicción cobertura, cobertura de predicción ponderada	Cuantitativa.
Diversidad	Medida de diversidad, diversidad relativa, curva de precisión de diversidad, estadísticas Q, establecer diferencias teóricas de listas de recomendaciones	Cuantitativa.
Integridad	Estudios de usuarios.	Cualitativa.
Confianza	Modelo de similitud de vecinos consciente, indicadores de similitud.	Cualitativa/ cuantitativa.
Novedad	Comparación de listas de recomendaciones y perfiles de usuario, ítems populares contados.	Cualitativa/ cuantitativa.
Serenidad	Comparación de listas de recomendaciones y perfiles de usuario, clasificabilidad	Cualitativa/ cuantitativa.
Utilidad	Función de utilidad basada en ganancias, estudio de intención del usuario, estudios de usuario.	Cualitativa/ cuantitativa.
Riesgo	Depende de la aplicación y la preferencia del usuario.	Cualitativa
Robustez	Cambio de predicción, índice de aciertos promedio, rango promedio.	Cuantitativa.
Tasa de aprendizaje	Corrección con el tiempo.	Cuantitativa
Usabilidad	Estudios de usuarios (encuesta, observación, monitoreo).	Cualitativa/ cuantitativa.
Escalabilidad	Tiempo de entrenamiento, rendimiento de recomendación.	Cuantitativa

Estabilidad	Cambio de predicción.	Cuantitativa
Privacidad	Privacidad diferencial, RMSE vs. curva de privacidad diferencial.	Cualitativa/ cuantitativa.
Preferencia de usuario.	Estudios de usuarios.	Cualitativa/ cuantitativa.

Nota. Resumen adaptado de libro Recommendatión System in Software Engineering por Robillard et al., 2014.

Para esta investigación se elegirán algunas dimensiones que están relacionados con los usuarios, la entrega y el mismo de sistema de recomendación. Estas dimensiones se detallan brevemente:

A. Dimensión exactitud

Los sistemas de recomendación para que tengan valor deben proporcionar resultados útiles que se acerquen a los intereses e intenciones de los usuarios. Por lo tanto, la medida clave es la corrección de las recomendaciones dadas a un usuario. La exactitud de una recomendación se refiere a la alineación con un punto de referencia o qué tan cerca está a las cualidades deseadas.

Existen diferentes métodos para medir la corrección que dependen del tipo de recomendaciones, se podría predecir cómo los usuarios califican un ítem, clasificación de los ítems más interesantes a los menos interesantes para un usuario en una lista.

Existen diferentes enfoques para la evaluación de precisión de sistemas de recomendación. Estos se clasifican en: métricas de precisión de predicción como Mean Absolute Error (MAE), métrica de precisión de clasificación como Precision and Recall, y métrica de precisión de rango como Kendall's Tau [22].

1) Métricas de clasificación

La métrica de clasificación más utilizadas en los sistemas de recomendación son la precisión y recuperación. Esta métrica se aplica en escenarios de evaluación que no están en línea, donde el entrenamiento de algoritmos de recomendación se realizan con una parte de las muestras disponibles y posteriormente se avalúan comparando las predicciones con el grupo de prueba, que son parte de los datos retenidos [22].

Unas de las medidas de dimensión exactitud es la Precision y Recall para comparar las recomendaciones en cada contexto de usuario. Las métricas de clasificación más usados se detallan a continuación:

Precisión

Es la división del número de ítems recomendados que son relevantes en relación al número total de ítems recomendados, lo que representa el porcentaje de recomendaciones correctas. Está dado por la fórmula:

$$precision\ k(g) = \frac{|recomendados_k(g) \cap relevantes(g)|}{k}$$

Se tiene un conjunto de ítems recomendados al usuario y otro conjunto de ítems relevantes para dicho usuario, la métrica permite saber cuántos de los recomendados son relevantes para el usuario. La *k* representa el total de ítems recomendados.

Recall

Es la división del número de ítems recomendados que son relevantes en relación al número de todos los ítems relevantes, o sea en qué medida las recomendaciones realizadas responden a las necesidades de recomendación. Está dado por la siguiente fórmula:

$$recall\ k(g) = \frac{|recomendados_k(g) \cap relevantes(g)|}{|relevantes(g)|}$$

Estas métricas tienen una compensación importante para medir la exactitud de las recomendaciones, cuando la recomendación es una lista larga mejora la recuperación (Recall) pero es probable que reduzca la precisión (Precision). Cuando se mejora la precisión a menudo empeora la recuperación. Por tanto, la medida F es el equilibrio entre precisión y recuperación, cuya fórmula es:

$$F = 2\ x\ \frac{precision\ x\ recall}{precision + recall}$$

2) Métricas de predicción

Cuando las recomendaciones tienen la intención de predecir cómo los usuarios califican los ítems de interés, a menudo se utilizan dos métricas de error, que se detallan a continuación:

Error Cuadrático Medio (Mean Square Error)

El Mean Square Error (MSE) es una función de riesgo, mide el promedio del cuadrado de la cantidad de elementos que se diferencia de la cantidad que se estima. Esta diferencia puede producirse debido a los datos aleatorios o porque no se ha tomado en cuenta alguna

información que podría dar un cálculo más preciso [24]. Esta métrica penaliza demasiado las desviaciones grandes.

El MSE se define:

$$MSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicido_i - Actual_i)^2}{N}}$$

Predicido es la calificación estimada por el modelo y el Actual es la calificación que tenía originalmente. Por ejemplo, si un usuario califica 5 a un libro y nuestro pronóstico es 4, entonces el MSE es 1. Las recomendaciones serán mejor cuando el valor de MSE sea menor.

Root Mean Squared Error (RMSE)

El RMSE puede determinar la diferencia entre las calificaciones reales de los usuarios y las predicciones llamados también residuos. Los sistemas de recomendación pueden sobreestimar o subestimar la calificación, los residuos pueden ser positivos y negativos. El RMSE se calcula con la siguiente fórmula:

$$RMSE(T) = \sqrt{\frac{\sum_{(u,i)\in T}(r\hat{}_{ui} - r_{ui})^2}{N}}$$

Donde: r_{ui} es la calificación real de usuario *u* para el ítem *i*, $r\hat{}_{ui}$ es el valor predicho y *N* es el número de todas las calificaciones.

Mean Absolute Error (MAE)

La medida básica de medición de predicción de error es Mean Absolute Error (MAE). La función está dada: usuarios estimados por calificación de ítems [22].

$$MAE(T) = \frac{\sum_{(u,i)\in T} |r\hat{}_{ui} - r_{ui}|}{N}$$

3) Métricas de ranking

Esta métrica es usada no solo en contar los ítems relevantes sino también la posición del ítem en una lista de recomendación. Uno de estas métricas es Discounted Cumulative Gain (DCG) que es basado en la idea de qué ítems aparecen más abajo en un resultado de recomendación que debería ser penalizado bajando los valores de relevancia en forma logarítmica [22].

B. Dimensión utilidad

La utilidad es el valor que el sistema o el usuario obtienen de una recomendación. Por consiguiente, el valor de una recomendación correcta se basa en la utilidad de un ítem o elemento. Una posible evaluación es considerar la utilidad de análisis de costo-beneficio; sin embargo, la evaluación debe considerar la variación de utilidad junto con las medidas de utilidad y parametrizar el grado de riesgo que los usuarios pueden tolerar en la evaluación, ya que cada uno puede abordar el riesgo de diferente manera.

C. Dimensión usabilidad

La efectividad de los sistemas de recomendación se valida con que los usuarios pueden usarlos sin dificultad de manera apropiada y ajustarse a los principios generales de usabilidad. Esto significa que deben ser eficaces, eficientes y proporcionar cierto grado de satisfacción a sus usuarios finales. La aceptación de recomendación depende de los contenidos que se presentan en el interfaz del usuario, algunos pueden ser solo sugerencias, otros proporcionan una lista de recomendaciones clasificadas y muchos sistemas de recomendación requieren configuración, preferencias del usuario y perfil de usuario. Todas estas interfaces tienen un gran impacto en la usabilidad del sistema de recomendación [16].

Los factores de usabilidad de los sistemas de recomendación generalmente se evalúan a través de estudios de usuarios.

2.2.1.3. Técnicas de retroalimentación de información

Para generar mejores recomendaciones se debe tener mecanismos de recopilación de una cantidad suficiente de información de los usuarios con la finalidad de descubrir sus preferencias y gustos por determinados ítems. La información se debe obtener a través de la interacción de los usuarios con la aplicación[19]. Este proceso se denomina retroalimentación y se clasifica en dos tipos: la retroalimentación implícita y la explícita.

2.2.1.3.1. Retroalimentación implícita

El proceso de retroalimentación implícita consiste en evaluar los comportamientos y acciones ejecutadas por los usuarios en el sistema de recomendación para obtener información y comprender las preferencias e intereses de los usuarios.

2.2.1.3.2. Retroalimentación explícita

Es un mecanismo del sistema de recomendación en que el usuario puede valorar los contenidos a través de una puntuación como uso de estrellas o likes para calificar los ítems u objetos. Los usuarios pueden valorar los contenidos de los libros según el interés que tienen de cada contenido.

2.2.1.4. Métricas de similaridad

El concepto de similaridad se ha formalizado mediante el uso de métricas o medidas de distancia; para conocer la similitud de dos individuos o instancias se requiere usar una función para calcular la distancia que existe entre las instancias o ítems [29].

La función de similaridad debe definirse en función de los atributos de los objetos. La definición de similaridad y el método en el que los puntos son agrupados varían según el algoritmo de agrupación que se aplique. La variedad de algoritmos de clustering son adecuados para diferentes conjuntos de datos y diferentes propósitos [30].

La extracción de conocimientos en las bases de datos, se inició con el desarrollo de bases de datos relacionales, luego las bases de datos multidimensionales y OLAP (OnLine Analytical Processing) que permiten la consulta a las bases de datos a varios niveles. En la última década, surge el desarrollo de minería de datos, algoritmos avanzados de análisis que facilitan el descubrimiento de nuevos conocimientos para lograr la competitividad de las organizaciones modernas.

La minería de datos contiene conjunto de técnicas para encontrar patrones (similitudes de atributos) automáticamente, denominado agrupamiento o clustering. Estas técnicas intentan crear grupos de objetos, con tal que los elementos que lo componen los grupos sean muy parecidos, mientras que los grupos sean los más distintos posibles.

Para realizar predicciones los sistemas de recomendación requieren definir y cuantificar la similitud entre ítems o usuarios, entonces la distancia será la medida de la similitud (cuantificación) o la diferencia entre observaciones. Además, las observaciones tienen variables asociadas, cuando más se asemejen dos observaciones, estarán más cercanos.

2.2.1.4.1. Medidas de distancia o similitud

Las medidas de distancia o de similitud permiten calcular el grado de relación entre dos elementos basado en la semejanza o diferencia de los atributos. Las medidas de

similitud más usados en los sistemas de recomendación son: distancia de Euclides, distancia de coseno, distancia de Manhathan, correlación de Pearson e índice Jaccard.

a) Distancia del coseno

La distancia de dos puntos de un vector, sería el coseno del ángulo que forman:

$$d(x, y) = arccos\left(\frac{x^T y}{\|x\| * \|y\|}\right)$$

Los dos vectores que forman el ángulo de coseno se pueden interpretar como la medida de similitud de sus orientaciones. El valor de coseno puede ser 0, 1, -1 dependiendo de sus orientaciones y serán más similares cuando los vectores tienen la misma orientación.

El coseno del ángulo entre dos vectores a comparar es el resultado, la similitud es mayor cuando el ángulo es menor [31].

Métrica de similitud de coseno

La métrica de similitud de coseno mide la distancia existente entre dos usuarios o ítems en función del ángulo que forman los vectores.

$$sim(x, y) = \frac{\sum_{i \in B_{x,y}} r_{x,i} . r_{y,i}}{\sqrt{\sum_{i \in B_{x,y}} r_{x,i}^2} \cdot \sqrt{\sum_{i \in B_{x.y}} r_{y,i}^2}} \in [0,1]$$

b) Distancia Euclídea

La geometría euclidiana es la geometría en la vida diaria, es la distancia clásica, como la longitud de una distancia entre dos puntos, definiendo un triángulo o una escuadra. Hay otros caminos para medir la distancia entre dos puntos (o el largo de un arco conectando dos puntos p y q). La distancia de Minkowski mide dm las generalidades de la distancia euclidiana [32].

De acuerdo con Steinbach & Tan [33], la elección de medida de la distancia es otra consideración importante. La medida de la distancia usada es Euclidean o Manhattan. En caso de Euclides, para dos puntos, X y Y, con n atributos, esa distancia es dado por la siguiente fórmula:

$$d(x,y) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}$$

c) Distancia de Manhattan

Para Steinbach & Tan [33], la medida de la distancia de Manhattan para dos puntos X y Y con n atributos, es dado por la siguiente fórmula:

$$d(x,y) = \sqrt{\sum_{k=1}^{n}|x_k - y_k|}$$

Esta función denominada también como rectilinear distance que calcula la distancia entre dos puntos cualesquiera que sean como la sumatoria de las diferencias absolutas entre cada dimensión.

También se denomina como distancia por cuadras, que hace referencia a recorrer un camino circunvalante y no el más corto (diagonal), tal como sería el desplazamiento en Manhattan [29].

d) Coeficiente de correlación de Pearson

Es una medida de similitud que es usado en sistemas de recomendación de enfoque colaborativo. El coeficiente de correlación entre dos usuarios o ítems depende de los votos que ha recibido; cuanto más votos compartidos es mejor [24].

El coeficiente de correlación Pearson permite calcular el coeficiente lineal entre los vectores con la siguiente fórmula:

$$sim(x,y) = \frac{\sum_{u\in U}(R_{u,x}-\bar{R}_x)(R_{u,y}-\bar{R}_y)}{\sqrt{\sum_{u\in U}(R_{u,x}-\bar{R}_x)^2}\sqrt{\sum_{u\in U}(R_{u,y}-\bar{R}_y)^2}}.$$

Donde, R_{ux} representa la puntuación del usuario *u* para el ítem *x* y R_{uy} representa la puntuación de usuario *u* para el ítem *y*. $\bar{R}_x$ representa el porcentaje de puntaje de ítem *x* y $\bar{R}_y$ representa el porcentaje de puntaje de ítem *y*.

e) Similitud de Jaccard

La similitud de Jaccard se enfoca mayormente en calificaciones globales, se obtiene de la proporción entre los ítems co-calificados y los ítems calificados por ambos usuarios. Se calcula con la siguiente fórmula:

$$Sim(x,y)^{Jaccard} = \frac{(I_x \cap I_y)}{(I_y \cup I_x)}$$

Donde I_x y I_y son conjunto de ítems calificados por los usuarios *x* y *y* respectivamente.

2.3. Marco conceptual

2.3.1. Machine Learning

El aprendizaje automático (Machine Learning en inglés) es una de las disciplinas de Inteligencia artificial que están basados en algoritmos avanzados compuestos de herramientas tecnológicas (redes neuronales, aprendizaje profundo y el procesamiento de lenguaje natural) que son capaces de descubrir patrones de comportamiento en datos masivos para realizar predicciones; son utilizadas en el aprendizaje supervisado y no supervisado.

Machine Learning fundamentalmente busca desarrollar sistemas informáticos que mejoren en forma automática través de la experiencia y conocer las leyes fundamentales que rigen del proceso de aprendizaje [34].

En nuestra sociedad de información se generan patrones de alta dimensión, por lo tanto, la recopilación y comprensión de los datos nos permite mejorar la eficiencia de los procesos, pero se requieren algoritmos que sean capaces de procesar los conjuntos de datos de manera eficaz [28].

Las dos clases de problemas más importantes de Machine Learning son el aprendizaje supervisado y el no supervisado.

2.3.1.1. Aprendizaje supervisado

Según Kramer [28] el objetivo del método de aprendizaje automático supervisado es aprender un modelo de función dado, a partir de las observaciones para establecer relaciones. Este método está basado en las características observadas y las etiquetas correspondientes. En el aprendizaje automático es muy importante conocer cómo se define un patrón, y básicamente un patrón consiste en características que se han registrado y recopilado desde diferentes fuentes de datos.

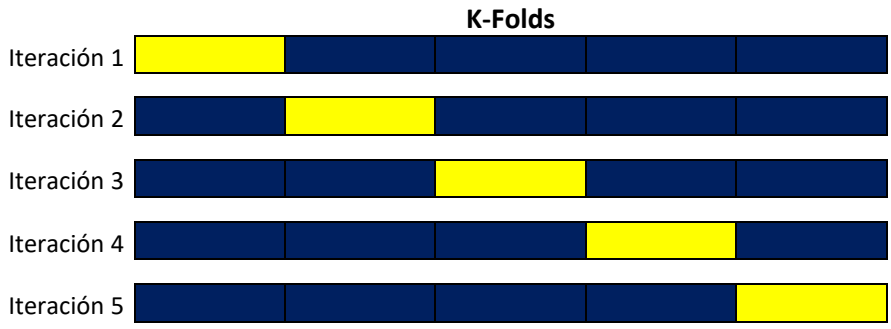
Para el trabajo se usará el modelo de aprendizaje automático supervisado, siendo que el sistema de recomendación está basado en algoritmos de similitud.

2.3.2. Validación cruzada

En inglés cross validation, es una técnica estadística que permite evaluar el rendimiento de un modelo de machine learning. La validación cruzada K-fold permite dividir los datos en K-segmentos iguales, de modo que el entrenamiento y la prueba se realizan en K iteraciones para entrenar el modelo, obteniendo la precisión en cada iteración la que se promedia como precisión del modelo [35]. Se divide la muestra en dos conjuntos de datos: entrenamiento y validación que ayuda la comprobación del performance y comportamiento del modelo.

Según Microsoft Analysis Services se recomienda los siguientes pasos para realizar una validación cruzada:

- Elegir el conjunto de datos a usar para validar el modelo.
- Seleccionar el modelo o modelos que quiere probar.
- Definir los parámetros para evaluar el modelo entrenado. Se define el número de iteraciones, número de Split y el indicador de precisión entre 0 y 1, cuando se aproxima a 1, indica que las predicciones son verdaderas, en caso contrario menor probabilidad de precisión.



El color azul representa los datos de entrenamiento y color amarillo los de validación, con tal que todos los datos entran en entrenamiento.

Sin embargo, dependiendo de la muestra de datos que se puede usar, en la mayoría se necesita realizar ajustes para que el modelo se adapte mejor al conjunto de datos.

2.3.3. Balanceo de datos

Según Cárdenas [36] las clases desequilibradas se presentan cuando se trabaja con datos reales, donde una clase tiene una gran cantidad de muestras, mientras que otras con

unos pocos [37], estos son notorios en las primeras observaciones. Por otro lado, los algoritmos convencionales solo consideran la tasa de error de la clase mayoritaria sin tener en cuenta la clase minoritaria.

Cuando se tiene un conjunto de datos desequilibrados se debe de intentar balancear las muestras de las clases minoritarias con tal de eliminar el problema de desbalanceo que incide en el conjunto de entrenamiento.

Las técnicas de remuestreo más utilizadas para abordar el problema de desbalanceo de datos son: sobremuestreo (oversampling) y submuestreo (undersampling) [36].

Oversampling. Es la técnica que permite crear nuevas muestras sintéticas aleatoriamente de la clase minoritaria para equilibrar con las muestras de la clase mayoritaria.

Figura 2 . Sobremuestreo del conjunto de datos

Undersamplig. Esta técnica reduce las muestras aleatoriamente de la clase mayoritaria para balancear con las muestras de la clase minoritaria.

Figura 3-. Submuestreo del conjunto de datos

Existen estrategias de remuestreo aleatorio como Random Oversampling que duplica aleatoriamente el conjunto de entrenamiento de la clase minoritaria y Random Undersampling que reduce en forma aleatoria el conjunto de entrenamiento de la clase

mayoritaria. Una de ellas es la técnica de sobremuestreo de minorías sintéticas (SMOTE) que consiste en crear nuevas muestras sintéticas de la clase minoritaria, eligiendo al azar un punto para calcular los k-vecinos más cercanos de ese punto [36].

2.3.4. Herramientas de desarrollo

Los científicos de datos usan herramientas de programación como Python para extraer conocimiento desde datos preparados. La extracción de esta información emplea varios modelos con propósitos basados en machine learning, algoritmos, estadística y métodos matemáticos [38]. A continuación, se mencionan los lenguajes de programación en la construcción del sistema de recomendación.

Python

Python es un lenguaje de programación dinámico y permite integrar sistemas de manera más efectiva, de propósito general que es usado en varios campos; además es usado para todo desde script descartables a largos servicios web que provee servicio ininterrumpido de 24/7. Python es usado para GUI (Grafic User Interface) y programación de base de dato, ´programación web de cliente y servidor y prueba de aplicación; asimismo es usado por científicos para escribir aplicaciones de computadores [38].

El mismo autor menciona algunas características importantes de Python son:

- Fácil de aprender y usar.
- Lenguaje expresivo.
- Lenguaje interpretado.
- Plataforma cruzada.
- Es libre y open source (código abierto).
- Extensible.
- Orientado a objeto.
- Librería estándar grande.
- Integrado.

Según Nagar [39], Python viene con miles de módulos para realizar varias tareas, pero para los científicos de computación se usan los módulos básicos necesarios que se mencionan en la tabla IV.

TABLA IV
MÓDULOS BÁSICOS DE PYTHON

Nombre de paquete	Medida	Propósito
Numpy	Python numérico	Computación numérica.
Scipy	Python científico	Científicos de computación
Sympy	Python simbólico	Computación simbólica
Matplotlib	Biblioteca matemática de ploteo	Para plotear gráficos

Recientemente Python es el lenguaje más popular para Machine Learning y Data Science; además es el quinto lenguaje más usado del mundo y uno de los tres lenguajes más usados por los científicos de datos [40].

Anaconda Suite

Anaconda es una distribución multiplataforma (Linux, MacOS y Windows) de código abierto con convenientes y completos ambientes de trabajo para desarrolladores en ciencia de datos y Machine Learning. Tiene diversos IDE (Integrated Development Environment) y presenta un interpretador de línea de comandos de Python, características avanzadas de su editor de textos como finalización de sintaxis, resaltado de palabras clave y explorador de archivos [39]. Anaconda tiene más de mil paquetes y un administrador de entorno virtual que elimina la necesidad de instalar en forma independiente las bibliotecas.

El Spyder IDE provee un uso fácil de un IDE de ambiente de desarrollo de archivos de Python como se muestra en la figura.

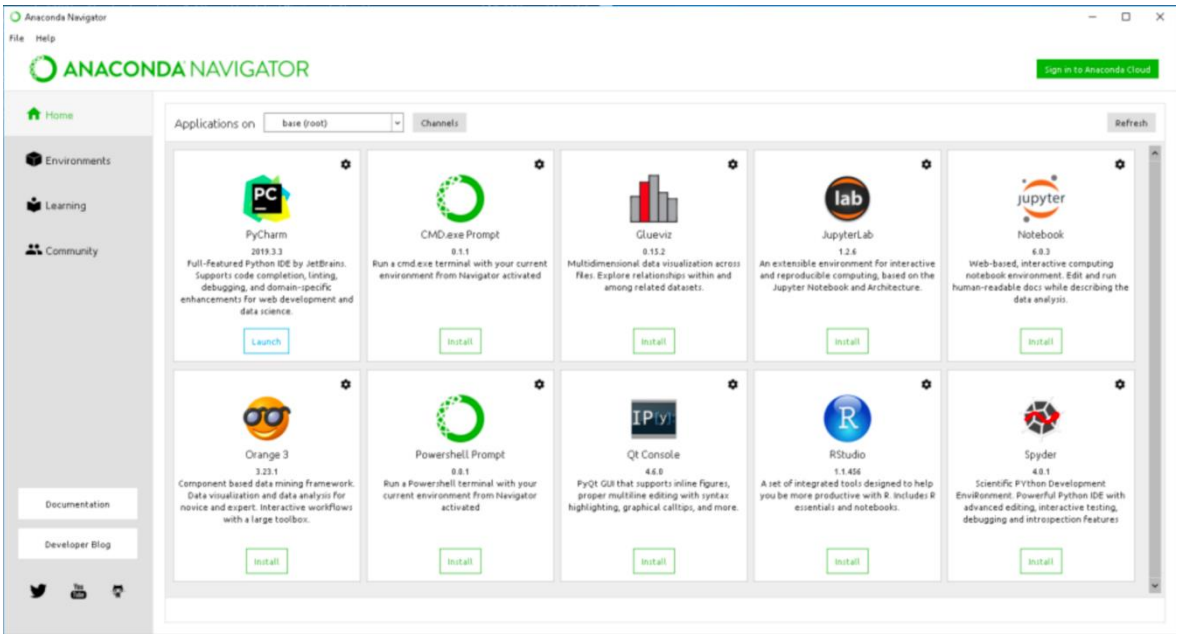


Figura 4. Interfaz de Anaconda Navigator

Anaconda Navigator tiene una interfaz gráfica que incluye enlaces a todas las aplicaciones de la distribución, incluidos RStudio, IPython, JupyterLab, Jupyter Notebook, Spyder, Glue y Orange.

Spyder

Uno de los mejores IDE de Python para aplicaciones científicas, es ligero y capaz de ejecutar complejos comandos de Python; en caso de usar Anaconda Navigator ya viene por defecto. Su característica es de código abierto, resaltado de sintaxis, tiene un visor de documentación y un explorador de variable; es interactiva lo que facilita la prueba y desarrollo de aplicaciones científicas [41].

Postgresql

Es un sistema de administración de base de datos relacional open source muy potente con la mejor reputación por su fiabilidad, arquitectura probada, solidez de funciones y rendimiento. Trabaja con diferentes plataformas y maneja eficientemente la atomicidad, seguridad y la integridad.

Django

Django es un framework web de Python para el desarrollo web y aplicaciones fomentando la rapidez, diseño limpio y pragmático, que permite concentrarse en la lógica de la aplicación y reutilización del código; asimismo es muy seguro y escalable.

Aplicación móvil

Las aplicaciones móviles o App son sistemas informáticos diseñados para ser ejecutados en dispositivos móviles como smartphome, tablets y otros, disponibles en las plataformas de distribución de Android, iOS entre otros [42].

Usabilidad

Es un atributo relacionado a la facilidad de uso, se refiere a aprender a utilizar algo de manera rápida y eficiente con menos propensión a error. Si una característica de un sistema no es utilizada por el usuario es como si no existiera [19].

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1. Tipo de investigación

El presente trabajo corresponde a la investigación aplicada y tecnológica: Se describen los hechos, se valida el algoritmo de similaridad con la data histórica de préstamo de libros de los usuarios y busca solucionar el problema de demora en las búsquedas de contenidos mediante la aplicación de soluciones tecnológicas.

Se considera aplicada porque plantea resolver problemas prácticos y normalmente se realiza la intervención en el desarrollo de la variable dependiente, y está enmarcado en la innovación técnica, industrial e científica [43]; además de algún descubrimiento que se puede aplicar en forma inmediata a procesos de una industria o negocio que permita generar alguna ganancia [44]. Se trata de plantear soluciones a las dificultades y demoras en la búsqueda de contenidos de libros.

Es tecnológica porque aborda una situación o fenómenos para transformar o crear nuevas realidades mediante la aplicación de soluciones tecnológicas y aprovechando el conocimiento científico. En efecto es el conocimiento aplicado que modifica una situación actual con finalidad práctica de uso inmediato como los inventos, innovaciones, diseños y otros. Los sistemas deben elevar la productividad de los recursos utilizados mediante la creatividad [45]. Asimismo, este tipo de investigación busca demostrar la efectividad de la tecnología para solucionar problemas técnicos a través de un proceso planificado, metódico y sistemático [46] En el trabajo se ha utilizado las herramientas tecnológicas como las técnicas de machine learning para la optimización de un sistema de recomendaciones para facilitar la búsqueda de contenidos de libros transformando la situación actual.

3.2. Diseño de la investigación

El diseño de la investigación es no experimental porque no se realiza la manipulación de las variables y es de corte descriptivo puesto que permite describir las variables en un determinado tiempo[47].

El trabajo de investigación se ha realizado en dos etapas:

Etapa 1. Se ha realizado las pruebas y validación de algoritmos de similitud con la muestra de conjunto de datos obtenidos del registro de préstamos de libros para la optimización de sistema de recomendación de contenidos de libros.

Etapa 2. La optimización del sistema de recomendación de libros basado en algoritmos de similaridad para sugerir recomendaciones de contenidos a los usuarios de la biblioteca del CRAI de la UPeU, a través de un aplicativo móvil Android.

3.2.1. Recolección de datos

Los datos son recolectados del sistema transaccional del CRAI ofrecido por terceros, sin descartar otras fuentes como archivos planos de reportes del registro de préstamos de libros de los usuarios en el período 2016 al 2019.

La muestra de datos que se ha tomado consiste en 989 libros y 661 usuarios que contiene 1466 calificaciones, obtenidos de los reportes de préstamos de libros que se han realizado durante estos años por el responsable de TI del CRAI, ya que no se tiene acceso a toda la base de datos de préstamos de libros.

Los datos que se han utilizado son de la data histórica de préstamo de libros de los usuarios de las bibliotecas del CRAI, con el propósito de la validación de los algoritmos y la construcción del sistema de recomendación de libros

3.2.2. Propuesta del sistema de recomendación de libros

En la etapa de optimización del sistema de recomendación de libros conforme a la arquitectura propuesta para lograr la calidad y precisión de las recomendaciones, además de amenorar los efectos de las limitaciones, se mencionan un conjunto de procedimientos y algoritmos aplicables al sistema de recomendación. En la Figura 5 se revela la arquitectura propuesta:

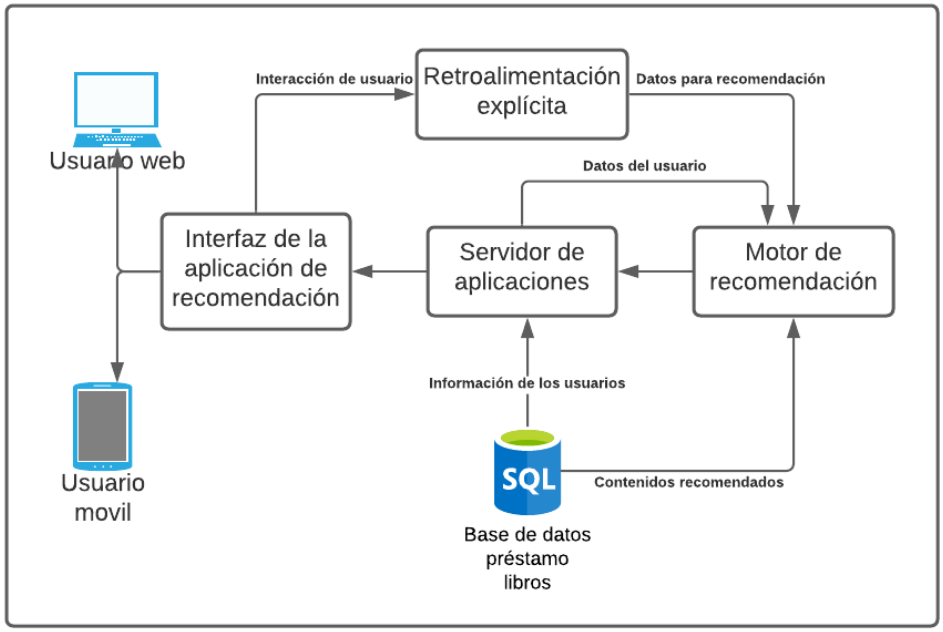


Figura 5. Arquitectura propuesta para la optimización del sistema de recomendación

La arquitectura propuesta se ha diseñado considerando los elementos claves de un sistema de recomendación, que a continuación se detallan:

Aplicación usuarios

El aplicativo de recomendación de libros cuenta con un interfaz de usuario amigable que permite interactuar con el sistema, el usuario puede visualizar la lista de recomendaciones de libros, ver los contenidos de los libros de su interés, solicitar reserva y préstamo de libros.

Retroalimentación explícita

La retroalimentación explícita permite obtener información de los usuarios a través de algunas acciones solicitadas a ellos en el aplicativo de recomendación de libros; para ello se usa el sistema de estrellas que permite dar puntuación a cada libro marcando las estrellas dependiendo de la relevancia del contenido del libro para el usuario.

Motor de recomendación

Los motores de recomendación son algoritmos que intentar predecir libros u otros productos de forma personalizada para cada usuario específico. Se ha aplicado el

algoritmo de filtrado colaborativo que proporciona contenidos de libros similares a los que se ha prestado anteriormente.

Base de datos

La base de datos almacena la data histórica de préstamo de libros que han realizado los usuarios de la biblioteca, además de las recomendaciones que se han generado por el motor de recomendación. La base de datos se actualiza con el registro y actualización de datos de los usuarios para iniciar la sesión y los registros de libros se actualizan a través de aplicación web del administrador.

Servidor de aplicaciones

El servidor se encarga de gestionar las aplicaciones del sistema de recomendación y procesos determinados en la arquitectura propuesta.

3.3. Método CRISP-DM

El método CRISP-DM es muy usado en la analítica y obtención de modelos basados en datos como un estándar para realizar proyectos de ingeniería de datos. Esta metodología es flexible y fácil de personalizar para diferentes trabajos, consta de seis fases, algunas de ellas son bidireccionales como se visualiza en la Figura 6, lo que permite volver a revisar las fases anteriores si es que es necesario [48].

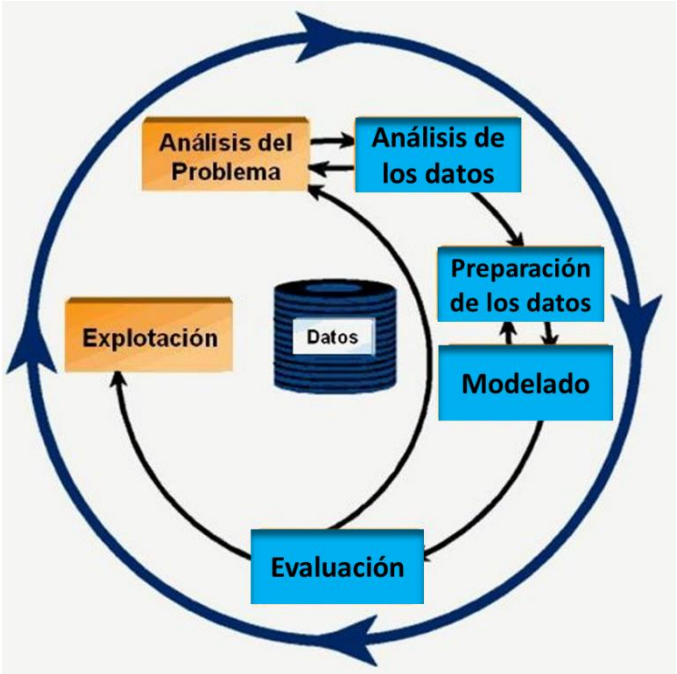


Figura 6. Fases del Método CRISP DM

3.3.1. Comprensión del negocio (análisis del problema)

En esta primera fase se determinan los objetivos y los criterios de éxito de la optimización del sistema de recomendación de libros del CRAI, es fundamental el conocimiento de datos para definir el problema de recomendaciones.

Para la determinación de los objetivos del negocio se mencionan primeramente la misión y visión del CRAI que aparece en sitio web:

Visión: “Ser líderes en la gestión de información documental en el ámbito de la Iglesia Adventista del Séptimo Día de Sudamérica y de la educación universitaria en el Perú”.

Misión: “Facilitar el acceso a la información documental que contribuya en el proceso educativo y la labor de investigación en búsqueda de la verdad y adquisición del conocimiento científico, tecnológico cultural y espiritual de los estudiantes, la comunidad universitaria y el entorno de la UPeU”.

Actualmente el CRAI no cuenta con un sistema de recomendación que facilite el acceso a la información de libros de las bibliotecas; sin embargo, el aprendizaje automático es la solución para realizar predicciones o sugerencias de productos a usuarios objetivo con la finalidad de que usen el servicio o predecir sus preferencias. La recolección del conjunto de datos para este proceso tiene poco coste pero se puede lograr beneficios importantes en la satisfacción de los usuarios.

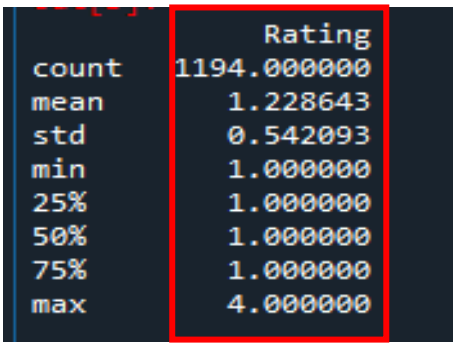
Considerando que el objetivo del CRAI es facilitar el acceso a la información bibliográfica que contribuya en el proceso de aprendizaje, de investigación y adquisición de conocimientos, se alinea con uno de los objetivos de aprendizaje automático: predecir y sugerir ítems a los usuarios según sus preferencias de información.

De acuerdo con la visión y misión del CRAI se destacan dos aspectos importantes: Facilitar el acceso a la información y gestión de información que están relacionados con los objetivos de aprendizaje automático, las cuales pueden ser posibles a través de las predicciones y recomendaciones.

Se considera como un criterio de éxito la predicción de calificación de ítems por los usuarios con un grado de precisión para ofrecer recomendaciones de libros. Asimismo el grado de precisión de las recomendaciones de libros por el sistema.

3.3.2. Comprensión de los datos

Esta fase comienza con los datos iniciales recolectados, la descripción de los datos para familiarizarse con ellos, la exploración de datos con la aplicación de pruebas estadísticas básicas para encontrar dificultades de calidad de datos y obtener conocimientos de los datos con información oculta.



	Rating
count	1194.000000
mean	1.228643
std	0.542093
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	4.000000

Figura 7.. Exploración de datos con estadísticas básicas

3.3.3. Preparación de datos

En esta etapa se ha realizado el preprocesamiento de la data histórica de préstamos de libros del CRAI, cuando se extrae del sistema normalmente este fichero está en formato XML (Extensible Markup Language) que requiere la transformación inicial al formato CSV (Comma Separated Value) para ser usado por el sistema.

El preprocesamiento de datos es de mucha importancia para lograr la calidad de datos, para ello se seguirá los siguientes procedimientos:

- **Extracción de la data histórica del sistema de biblioteca**. La data histórica de préstamo de libros se ha extraído del sistema workflow del CRAI proveído por SirsiDynix, que permite obtener reportes por usuarios y por ítems, pero solo en formato XML.

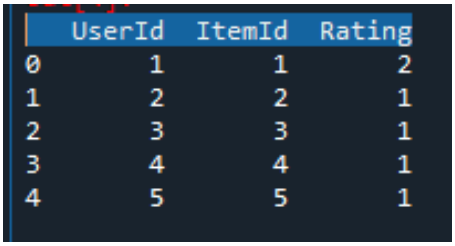
- **Limpieza de datos**. Consiste en lograr datos completos y ordenados. Los datos en formato XML no es manejable por lo que se requiere convertir al formato XLS de una hoja cálculo para usar las funciones que trae con la finalidad de filtrar los datos incompletos, datos duplicados, verificar los datos erróneos y otros. Después de tener el archivo en formato XLS requiere ser convertido al formato CSV para realizar las pruebas con los algoritmos que se utilizaran en el sistema de recomendación de libros. Además, el archivo debe contener un código de

identificación del usuario y un código de identificación de su ítem de preferencia emparejados en cada fila y cada campo ordenado por columnas.

Preprocesamiento de datos

Después de la carga de datos al dataframe se ha realizado los siguientes procedimientos:

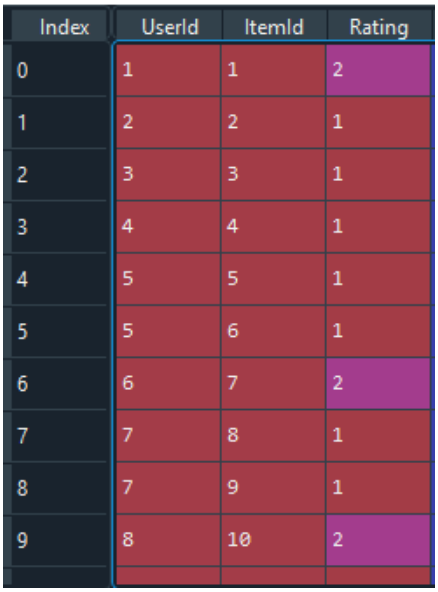
Visualización de datos. Antes de dividir el conjunto de datos para el entrenamiento y las pruebas, es necesario visualizar los datos para ver si presenta algún dato faltante e identificar las columnas numéricas que se utilizará y las columnas que no serán utilizados para el entrenamiento.



	UserId	ItemId	Rating
0	1	1	2
1	2	2	1
2	3	3	1
3	4	4	1
4	5	5	1

Figura 8. Visualización de datos

Estandarización de datos. Conocido también como normalización de los datos. Los datos visualizados pueden estar correctos y completos, sin embargo, suelen presentarse el problema de datos NAN durante el entrenamiento, por tanto, se recomienda estandarizar los datos antes del procesamiento de datos. En la figura 9 se visualizan los datos estandarizados que serán usados en el entrenamiento del modelo.



Index	UserId	ItemId	Rating
0	1	1	2
1	2	2	1
2	3	3	1
3	4	4	1
4	5	5	1
5	5	6	1
6	6	7	2
7	7	8	1
8	7	9	1
9	8	10	2

Figura 9. Datos estandarizados para el entrenamiento

3.3.4. Modelado

Para el modelado se ha elegido el K-Nearest Neighbours considerando que se aborda un problema de clasificación con el propósito de hallar similitudes entre los usuarios para ofrecer recomendaciones de libros.

Se han realizado las pruebas con las técnicas o algoritmos de similaridad para elegir el modelo que cumple con los objetivos de la analítica de datos. La técnica de modelado que mejor se adapta a los objetivos del proyecto es el que aborda los problemas de clasificación, ya que se ha abordado casos de predicciones con datos discretos.

En la Figura 10 se visualizan las pruebas que se han realizado con los modelos de Neive Bayes clasificación, regresión logística clasificación, máquinas de vectores de soporte y K-Nearest Neighbours, utilizando el conjunto de datos de préstamos de libros, resultando con la mejor precisión el modelo KNN (0,83), esto depende mucho de la muestra de datos.

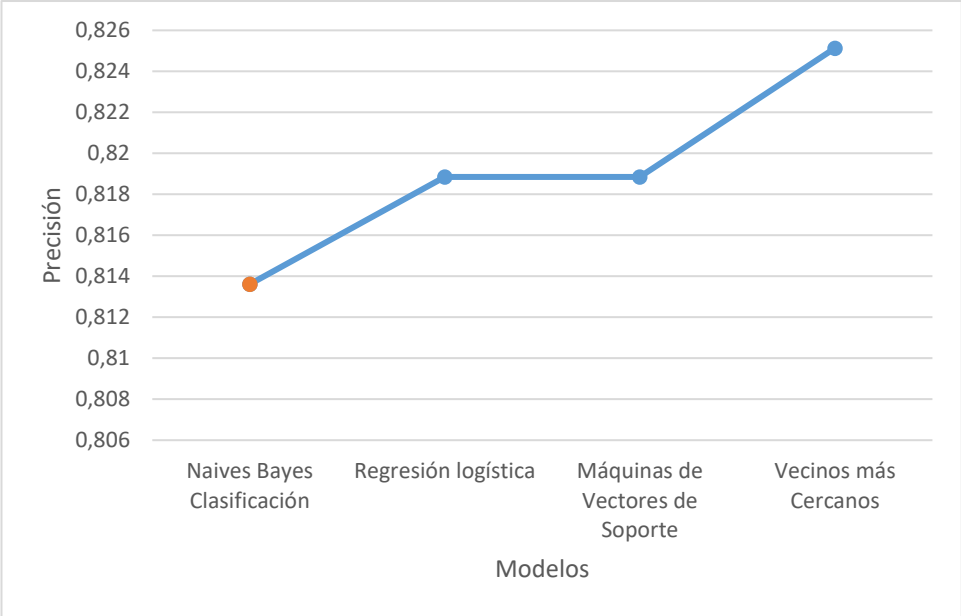


Figura 10 . Elección del modelo KNN con mejor precisión

Los procedimientos de plan de pruebas para la comprobación de la validez y precisión de predicciones de calificación de ítems por parte de los usuarios utilizando las medidas de error cuadrático medio (RMSE). El error se debe entender como la diferencia entre las predicciones de un estimador y los valores reales.

Una parte del modelado de KNN se muestra en el siguiente pseudocódigo:

```
#Dividir el conjunto de entrenamiento y de prueba
x_entrena, x_prueba, y_entrena, y_prueba → función train_test_split(x, y, tamaño_prueba=0.3, random_state=42)
Mostrar(x_entrena)
#Entrenamiento modelo KNN
modelo → función KNeighborsClassifier (n_vecinos=7, metrica="coseno")
modelo → elige rango np.range (1, 20)
exactitud_entrena →np.empty (mostrar (modelo))
exactitud_prueba →np.empty (mostrar (modelo))
Para i, k en numerado (modelo):
        knn → KNeighborsClassifier(n_vecinos = k)
        entrenar con  knn.fit(x_entrena, y_entrena)
        exactitud_entrena[i] → puntaje con knn.score(x_entrena, y_entrena)
        exactitud_prueba[i] → puntaje con knn.score(x_prueba, y_prueba)
K=7
knn → KNeighborsClassifier (n_vecinos=K, metrica="coseno")
entrenar con funcion knn.fit (x_entrena, y_entrena)
```

```
imprimir (knn.score (x_prueba, y_prueba))
pred1 → knn.predict(x_prueba)
imprimir (pred1[0])
```

Luego se ejecuta el modelo elegido con los datos de entrenamiento utilizando diferentes algoritmos de similitud con la finalidad de lograr el mejor performance del modelo.

3.3.4.1. Elección del método de recomendación

Existen varios métodos de sistemas de recomendación, la elección depende del conjunto de datos con la que se dispone para el análisis y considerando las características de los datos, se ha optado por el método de filtrado colaborativo, dejando de lado algoritmos que no aporten información relevante y que requieran mucho tiempo de computación. Existen varios algoritmos que pueden diferenciarse es su aproximación de análisis dependiendo de las características del problema a abordar. A continuación, se detalla la aplicación del método seleccionado para el sistema de recomendación de libros:

3.3.4.1.1. Método de filtrado colaborativo

Uno de los métodos más usados para sistemas de recomendación de libros es el filtrado colaborativo [6] [3] [7]. Una de las ventajas importantes es que puede realizar recomendaciones novedosas y no requiere información personal de los usuarios del sistema [49]. En este método las recomendaciones se realizan usando las preferencias de los usuarios por el contenido de los libros, normalmente las calificaciones de los usuarios se tienen en una matriz de rating, donde las filas contienen a los usuarios y en las columnas a los libros, y en cada celda contiene el número de calificación que indica el nivel de preferencia.

TABLA V
MATRIZ DE CALIFICACIÓN DEL USUARIO PARA EL ÍTEM

	$Item_1$	$Item_2$	$Item_3$	….	$Item_n$
$User_1$	R_{11}	R_{12}	R_{13}	….	R_{1n}
$User_2$	R_{21}	R_{22}	R_{23}	….	R_{2n}
$User_3$	R_{31}	R_{32}	R_{33}	….	R_{3n}
….	….	….	….	….	….
$User_m$	R_{m1}	R_{m2}	R_{m3}	….	R_{mn}

En la Figura 11 se presenta el proceso del método de filtrado colaborativo a partir de la matriz de las calificaciones para realizar las predicciones de libros que deben ser recomendados al usuario activo.

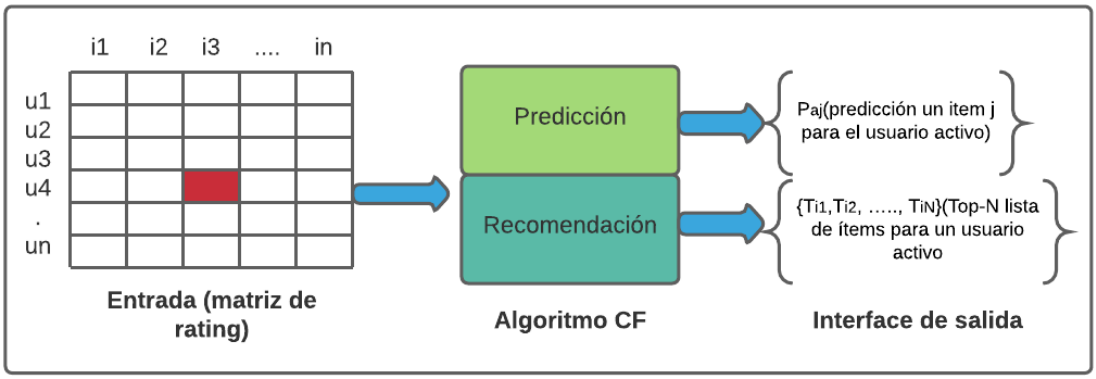


Figura 11. Proceso de filtrado colaborativo.
Adaptado de Item-based collaborative filtering recommendation algorithms por Badrul, George Karypis, Joseph Konstan y John Riedl, 2001.

Existen dos técnicas en este método para usar el algoritmo, los que se detallan a continuación:

Filtrado colaborativo basado en usuarios

El enfoque basado en usuario, en donde las calificaciones que los usuarios hacen a los libros se consideran como la puntuación de un libro. La similitud de los usuarios es el factor de ponderación para la valoración de un libro, considerando el interés que tiene de

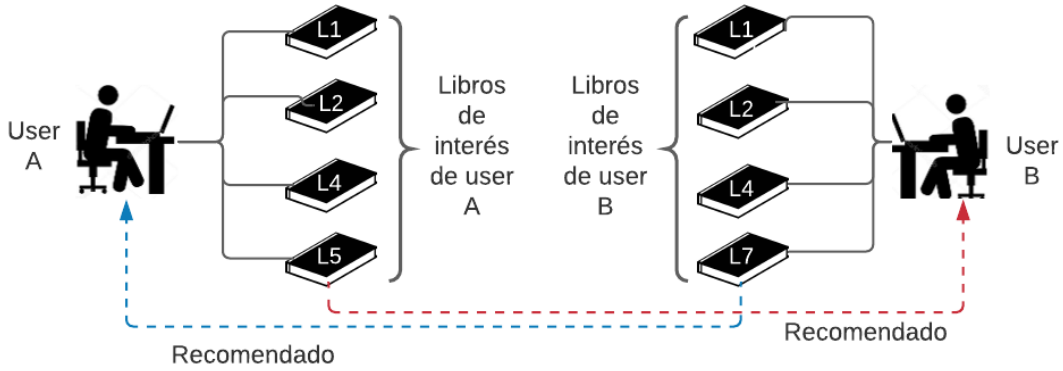


Figura 12. Método de filtrado colaborativo basado en usuarios.

un libro un usuario similar al que se va a recomendar, sirve para realizar la recomendación al usuario objetivo.

La técnica de filtrado colaborativo basado en usuario permite identificar a los vecinos k más cercanos de un usuario activo para un libro específico, y en función de estos vecinos, se calcula la predicción de la calificación del usuario activo. La similitud entre dos usuarios se puede determinarse con un algoritmo de similitud, por lo tanto, se elegirá la métrica de similitud más apropiado de acuerdo al conjunto de datos con la que se cuenta.

Para producir recomendaciones relevantes a partir de los datos extraídos, la clave es elegir el algoritmo o técnica adecuada. Además, se debe considerar que las técnicas de recomendación se basan en dos conjuntos: los libros a recomendar y los usuarios de esos libros; por tanto, es necesario identificar a los usuarios y los libros relevantes para dichos usuarios, para ello se tiene la técnica de filtrado colaborativo basado en usuarios y filtrado colaborativo basado en libros.

En esta investigación se utiliza datos recopilados implícitamente, lo que significa que el perfil del usuario es basado en todos los datos proporcionados durante el registro para la creación del perfil.

Filtrado colaborativo basado en ítems

El enfoque basado en ítem, trata de identificar las relaciones entre los ítems a través del análisis de la matriz de puntuaciones, se usan estas relaciones para predecir las puntuaciones de libros que no han sido vistos. Con cantidades grandes de datos puede trabajar mejor que basado en usuario, y puede tener mayor precisión [50].

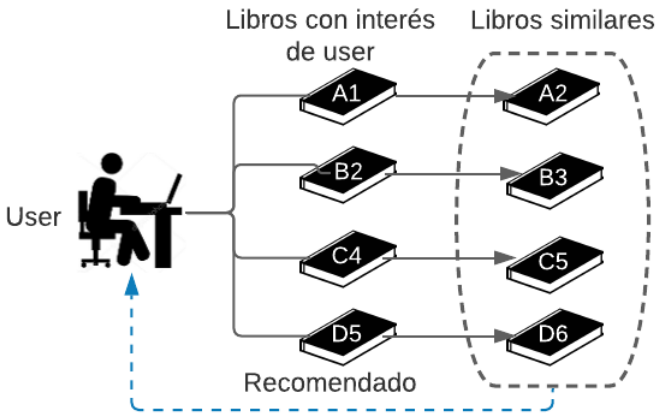


Figura 13. Método de filtrado colaborativo basado en ítems.

Una vez encontrado los ítems más similares se combinará con las calificaciones de libros por un usuario para generar una recomendación eficaz. Por tanto, la similitud entre los elementos se calcula en función de las calificaciones.

Para compensar la inflación de las calificaciones (calificaciones más altas de lo esperado) se aplicará la fórmula se similitud de coseno, al igual que en la similitud de usuarios.

3.3.4.2. Modelo K-Nearest Neighbors

Existen varios modelos para el aprendizaje supervisado, y la elección del modelo adecuado depende mucho de las características de la muestra de datos. En el trabajo se ha optado por el modelo K-Nearest Neighbors considerando las siguientes ventajas [51]:

- El modelo permite manejar problemas de clasificación múltiple con éxito, es sencillo y al mismo tiempo muy potente, además de las predicciones.
- En casos de clasificación en el que el límite de decisión es bastante irregular también tiene éxito en las predicciones.
- No existe relación entre las características del conjunto de datos.
- El modelo es relevante cuando en el conjunto de datos se considera importante la localización de datos.
- No se realiza el entrenamiento para que el modelo aprenda.
- Es muy bueno para la búsqueda de similitudes semánticas entre las principales aplicaciones de algoritmo de KNN.

Las fases generales de este método para calcular las recomendaciones son: Cálculo de vecinos, evaluación de predicciones,

Cálculo de vecinos. El algoritmo K-Nearest Neighbors (KNN) o vecinos próximos es el más utilizado para realizar recomendaciones colaborativas. Se calcula la cercanía de los usuarios con preferencias y necesidades más similares a las del usuario objetivo y asigna objetos a K centroides más próximos.

Para aplicar el algoritmo se ha seguido los siguientes pasos:

1) Definir el parámetro K, en nuestro caso K=5.
2) Forma los clústers con K vecinos más próximos al elemento nuevo según la distancia de similitud.

3) Recalcular el centroide de k clústers.

4) Repetir el paso 2 y 3 hasta que los centroides sean estables.

```
59
60  print("similaridad de usuarios con k distancias")
61  from sklearn.neighbors import NearestNeighbors
62  k=5
63  neighbords=NearestNeighbors(k,'cosine')
64  neighbords.fit(ratings_train)
65  top_k_distances,top_k_users=neighbords.kneighbors(ratings_train,return_distance=
66
```

Figura 14. Uso de KNN en código Python.

Hallar el valor de K que se ajuste para conseguir la precisión de las predicciones es fundamental. En la figura se muestra los valores K vecinos y los porcentajes de exactitud para la predicción:

Al realizar pruebas con más valores de k-vecinos se puede observar la tendencia de estabilizarse la variación de la exactitud del conjunto de entrenamiento y de prueba.

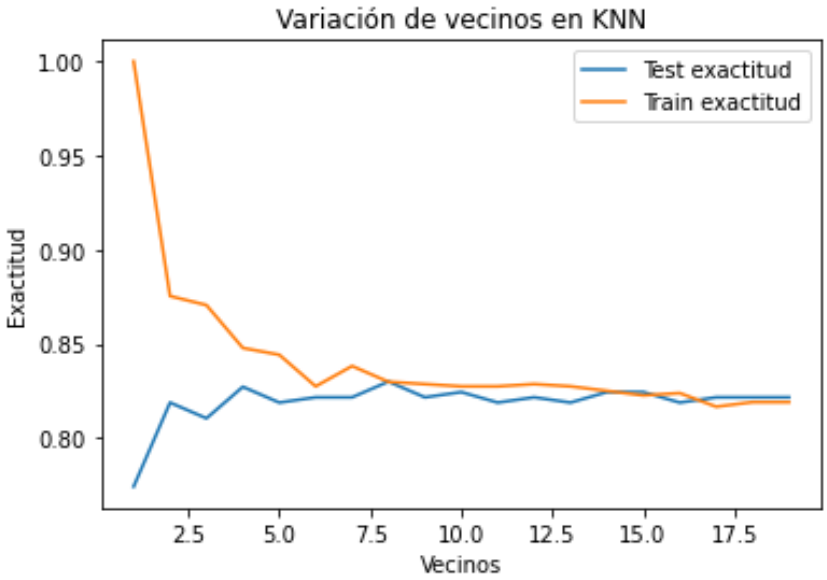


Figura 15. Exactitud de train y test con la variación de K de 1 a 20

El algoritmo KNN permite generar los perfiles de usuarios basado en la similitud entre el vector de cada usuario.

3.3.4.2.1. Métricas de similitud

Cuando se realizan procesamiento de datos es fundamental usar métricas, es decir medidas de distancia, apropiadas para el tipo de datos o sea la muestra con la que está trabajando, para obtener buenos resultados [52]. Para estimar los vecinos más cercanos

se han utilizado las principales métricas de cálculo de similitud, con el propósito de evaluar el modelo con la muestra de datos para lograr el mejor performance de las medidas de similitud para el modelo del sistema de recomendación de libros. A continuación, se mencionan las medidas de similitud:

Similaridad de Jaccard. Mide la similitud de conjuntos de datos. La similitud de dos usuarios o libros se encuentra cuando la intersección de ambos es más parecida, o sea cuando más calificaciones tengan en común es mejor, sin depender del valor de calificaciones.

$$Sim(x,y)^{Jaccard} = \frac{(I_x \cap I_y)}{(I_y \cup I_x)} = \frac{(I_x \cap I_y)}{(I_y) + (I_x) - (I_x \cap I_y)}$$

Donde I_x y I_y son del conjunto de libros calificados por los usuarios *X* y *Y* respectivamente.

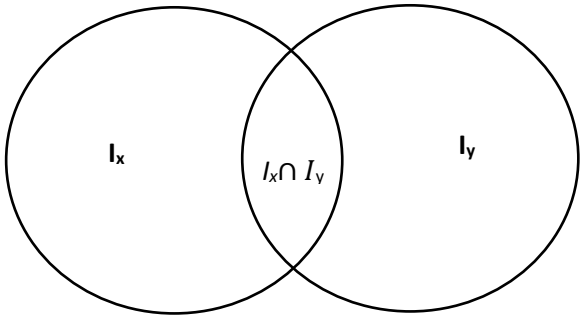


Figura 16. Similitud de dos conjuntos de libros o usuarios

La similaridad de dos usuarios es la relación de número de calificaciones comunes que se encuentran en la intersección y el número total de calificaciones es la unión [49].

Coeficiente de correlación de Pearson. La medida de similitud de Pearson es útil porque no depende de los desplazamientos y escalas de valores. Su uso en filtrado colaborativo, tiene un inconveniente, es que se necesita que exista al menos dos valores comunes para sugerir predicciones significativas. En esta métrica la puntuación promedio del usuario o libro es el centro de cada calificación. Se calcula en dos versiones: el módulo *v*, por intersección, sumando las calificaciones de los libros que tienen en común con *u*; y total donde también se añaden las calificaciones de aquellos que no conoce *u*. Aplicando la fórmula a la muestra de la data obteniendo los siguientes resultados.

$$sim(x,y) = \frac{\sum_{u \in U}(R_{u,x} - \bar{R}_x)(R_{u,y} - \bar{R}_y)}{\sqrt{\sum_{u \in U}(R_{u,x} - \bar{R}_x)^2}\sqrt{\sum_{u \in U}(R_{u,y} - \bar{R}_y)^2}}$$

Similitud de coseno

La similitud de coseno se ha utilizado para calcular la similitud entre los libros mediante los vectores de coseno y la similitud de entre los usuarios. La similitud de vector *X* y el vector *Y* se calcula con la siguiente fórmula:

$$d(x,y) = arccos\left(\frac{x^T y}{\|x\| * \|y\|}\right)$$

Para calcular la similitud entre los usuarios se ha usado la similitud de coseno, considerando que se presentan algunos desafíos como el problema de escasez de datos y dispersión de calificaciones que dificulta la correlación y otro problema es datos desequilibrados, provocando la caída del rendimiento de cálculo. Para abordar este problema se tenido en cuenta el número de elementos comúnmente calificados, cuanto mayor sea el número de libros comúnmente calificados, mayor será la similitud. Considerando la base teórica se propone usar la similitud de coseno, que se calcula con la siguiente fórmula:

$$sim(x,y) = \frac{\sum_{i \in B_{x,y}} r_{x,i} . r_{y,i}}{\sqrt{\sum_{i \in B_{x,y}} r_{x,i}^2} \cdot \sqrt{\sum_{i \in B_{x,y}} r_{y,i}^2}} \in [0,1]$$

Donde, se calcula la similitud del usuario ***X*** con el usuario ***Y*** mediante el ángulo que forman los vectores **X** y **Y,** si el ángulo es pequeño significa que hay mayor similitud entre los usuarios.

3.3.5. Evaluación

Evaluación de predicciones o valoración de preferencia. Luego de calcular la vecindad, se puede predecir la estimación del valor de preferencia que asignaría el usuario objetivo a cada libro que no ha sido valorado.

Por lo tanto, se debe evaluar la precisión de las predicciones de las calificaciones que los usuarios realizan a los libros de su interés, para ello se ha utilizado la métrica de predicciones de MSE y RMSE que son las métricas más adecuadas para ponderar errores de distribución desequilibrada; lo que permite diferenciar las calificaciones reales del

usuario y las predicciones, éstas pueden ser positivos o negativos. Esto se ha calculado con las siguientes fórmulas:

$$MSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicido_i - Actual_i)^2}{N}}$$

$$RMSE(T) = \sqrt{\frac{\sum_{(u,i)\in T}(r\hat{}_{ui} - r_{ui})^2}{N}}$$

Donde, *T* es el conjunto de pruebas, *N* es el número de pruebas y $r\hat{}_{ui}$ es el valor predicho y r_{ui} es la calificación real del usuario *u* para el libro *i*.

Dependiendo de los resultados que se obtengan, las predicciones pueden ser falsos positivos o verdaderos positivos.

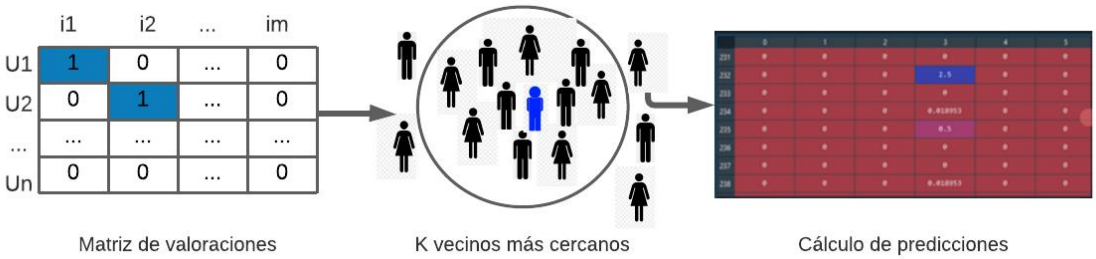


Figura 17. Proceso de filtrado colaborativo.

Recomendación de los mejor valorados Top-N. Se recomiendan los Top-N primeros libros de la lista de ítems recomendados.

Evaluación de la exactitud de clasificaciones

Para la evaluación y verificación de la precisión y exhaustividad de las clasificaciones se ha utilizado las métricas de Precision y Recall para manejar los casos de falsos positivos y verdaderos positivos. Las fórmulas de estas métricas son:

$$precision\ k(g) = \frac{|recomendados_k(g) \cap relevantes(g)|}{k}$$

$$recall\ k(g) = \frac{|recomendados_k(g) \cap relevantes(g)|}{|relevantes(g)|}$$

Estas métricas son complementarias, si la recomendación de libros es una lista larga, puede mejorar la recuperación (Recall) pero empeorar la precisión (Precision) y al revés. Para ello, la medida F1 es el equilibrio entre estas dos métricas para lograr la efectividad de las recomendaciones. Se calcula de la siguiente manera:

$$F1 = 2\,x\,\frac{precision\,x\,recall}{precision + recall}$$

Matriz de confusión

Existen varias medidas de performance de un modelo que cuantifique la calidad del clasificador [53], una de las más usados es la matriz de confusión.

La matriz de confusión es una herramienta que ayuda a visualizar los resultados del algoritmo de un modelo. En la matriz, se presentan el número de predicciones de las clases en cada columna y en las filas se presenta el número total de calificaciones reales de cada clase.

		Predicciones		
	Clase	1	2	
Real o actual	1	TP	FN	P
	2	FP	TN	N

Figura 18. Estructura de matriz de confusión

La matriz permite calcular la exactitud, precisión y sensibilidad también conocido como exhaustividad que son diferentes, pero se suele usar en forma conjunta. Para calcular se han usado las siguientes fórmulas:

$$Exactitud = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Sensibilidad = \frac{TP}{TP + FN} = \frac{TP}{P}$$

Estas métricas se han aplicado para evaluar la exactitud, precisión y sensibilidad de las predicciones de las recomendaciones.

3.4. Optimización de recomendación de filtrado colaborativo

En un sistema de recomendación real al inicio siempre se presenta el problema de escases de data y los resultados no son razonables, al mismo tiempo pocos datos del comportamiento explícito e implícito de los usuarios [54]. Por tanto, se ha decidido optar por el algoritmo de filtrado colaborativo basado en datos recopilados de atributos personales y las preferencias de los usuarios, para subsanar los problemas de escasez de calificaciones y realizar recomendaciones a nuevos usuarios.

En cuanto al modelo KNN que se ha utilizado es fundamental encontrar el mejor valor de K vecinos más cercanos y la mejor métrica de similitud para obtener mejores resultados del modelo. Luego para la optimización del modelo se ha aplicado la matriz cruzada para la evaluación y validación del modelo con mejores hiperparámetros. A su vez se ha realizado el balanceo de datos para equilibrar las clases logrando optimizar el modelo y los resultados se visualizan en una matriz de confusión en el capítulo de resultados.

3.4.1. Retroalimentación del sistema

La retroalimentación de información del sistema de recomendación de libros se realizará utilizando la técnica de retroalimentación explícita que se efectuará a través de un sistema de valoración o puntuación de cinco estrellas, que permite calificar los contenidos de los libros que son de interés del usuario. Según Núñez [19] se requiere asignar el significado del valor de cada estrella. La puntuación estará definida tal como se muestra en la Tabla VI:

TABLA VI
SISTEMA DE PUNTUACIÓN DE CONTENIDOS

Grado de interés	Puntuación
El contenido no es interesante	★
El contenido es poco interesante	★ ★
El contenido es interesante	★ ★ ★
El contenido es muy interesante	★ ★ ★ ★
El contenido es imprescindible	★ ★ ★ ★ ★

La retroalimentación o feedback del sistema según la literatura existente se realiza a través de puntaciones o calificaciones explícitas de los usuarios a un libro. Sin embargo, los sistemas de recomendación actuales en aplicaciones reales obtienen información implícitamente, o sea sin que el usuario responda preguntas o encuestas sino observando

su comportamiento e interacción con el sistema. Esto permite identificar los intereses y preferencias del usuario objetivo por un tema específico en un determinado tiempo.

Para la retroalimentación implícita del sistema de recomendación se considera la frecuencia de Likes que dan los usuarios a cada libro y la frecuencia de préstamos de libros por un usuario. Además, los likes que recibe un libro aparece en la lista de recomendaciones para el usuario, con el propósito de mostrar que el libro gusta a muchos usuarios. En la figura se muestra el uso de likes y unlikes.

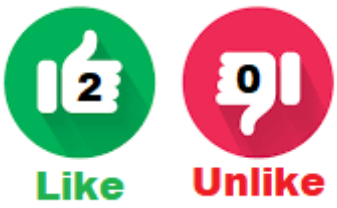


Figura 19. Retroalimentación explícita con valoración de likes en los libros prestados

La conversión a rating se ha realizado por cada usuario, si un libro fue prestado por el usuario objetivo se asigna la puntuación máxima. En cambio, en la frecuencia de likes, la máxima puntuación obtiene el peso máximo, la mínima el menor peso, y las puntuaciones intermedias se distribuyen en forma equitativa el peso dentro del máximo y mínimo.

3.4.2. Explotación (Implantación)

3.4.2.1. Interfaz de sistema de recomendación

El aplicativo de recomendación de libros se ha diseñado y se ha desarrollado considerando los requerimientos recabados a través de entrevistas al responsable de Tecnologías de Información (TI) del CRAI, quien colaborará activamente proporcionando información relevante durante el desarrollo e implementación del sistema.

Se considerará los siguientes parámetros para el diseño:

- Interfaz de usuario de fácil acceso.
- Accesibilidad mediante dispositivos móviles con sistema Android.

- Módulo de visualización de recomendaciones en el aplicativo, que permite ver la información básica del libro, con opciones de sistema de puntuación de libros relevantes y realizar comentarios de satisfacción de los contenidos de libros.

- Uso de herramientas de software libre para el desarrollo de aplicación móvil.

- Servidor local de dase de datos de sistema de recomendación.

Para el desarrollo del App de recomendación de libros se considerará el uso de herramientas Open Source, para abaratar los costos y que permita la usabilidad y escalabilidad del sistema de información. A continuación, se mencionan algunas herramientas de desarrollo:

- Python es el lenguaje de programación que se ha usado, ya que es open source con una variedad de librerías que son de mucha utilidad que se adaptan al desarrollo de aplicaciones web. Además de la plataforma Anaconda Navigator que incluye varias aplicaciones de la distribución.

- Postgress, para gestor de base de datos por la estabilidad y confiabilidad en manejo de grandes volúmenes de datos.

- Servidor HTTP Apache como plataforma de servicio web.

El sistema de recomendación de libros tiene una interfaz gráfica de usuario sencillo y amigable. El usuario primeramente debe registrarse con los datos básicos personales y académicos, luego puede iniciar la sesión para ver la lista de libros recomendados para el usuario activo. Al elegir uno de los libros puede ver su contenido e inmediatamente solicitar el préstamo o reservar dicho libro.

.

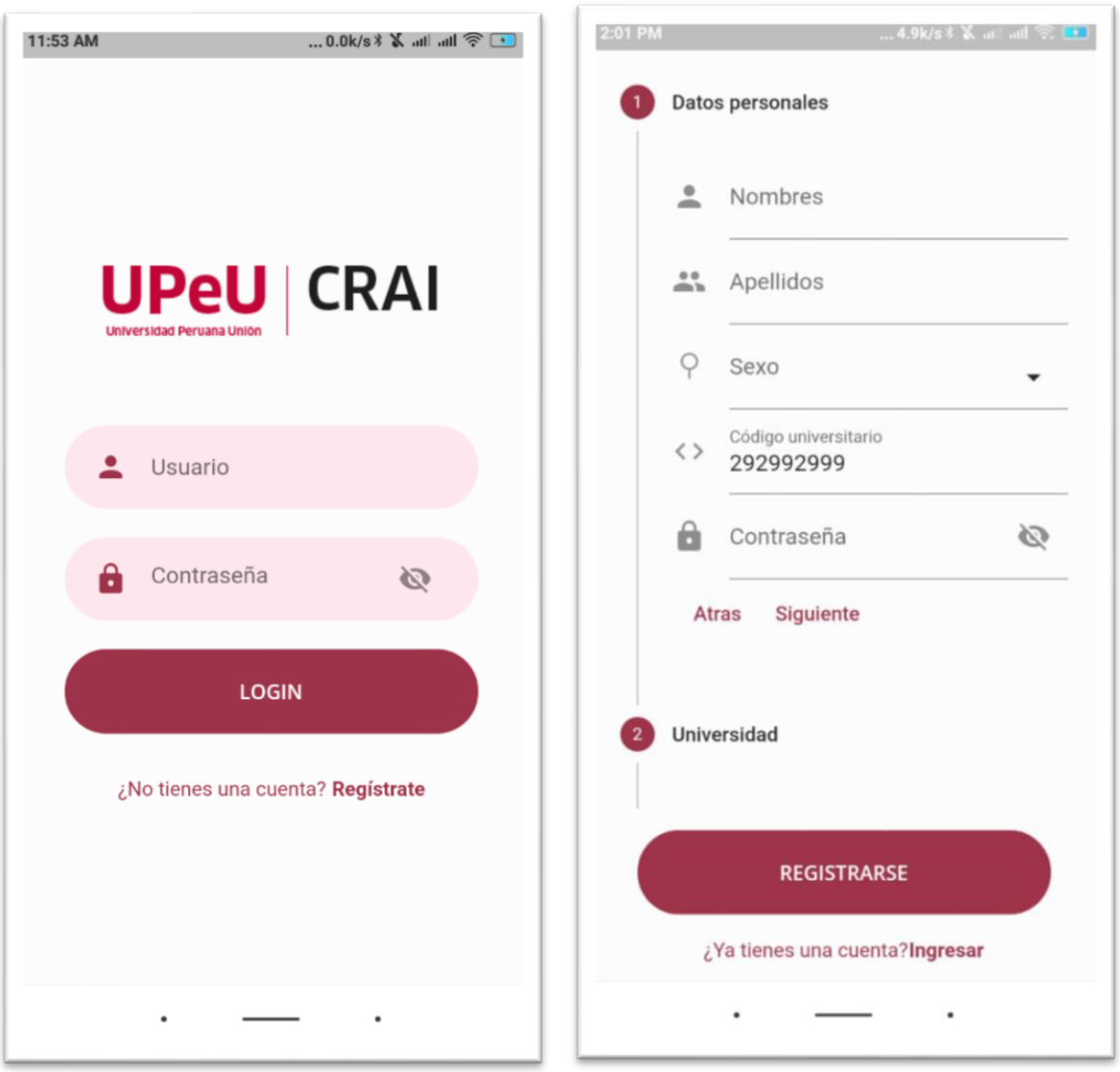


Figura 20. Interfaz gráfica de usuario del aplicativo de recomendación

En la figura se observa el formulario de registro del usuario con los datos solicitados que identifican que son estudiantes de la universidad para acceder al aplicativo de recomendación de libros. Después del registro el usuario puede iniciar la sesión con su usuario y contraseña.

.

Figura 21. Visualización de la lista de libros recomendados

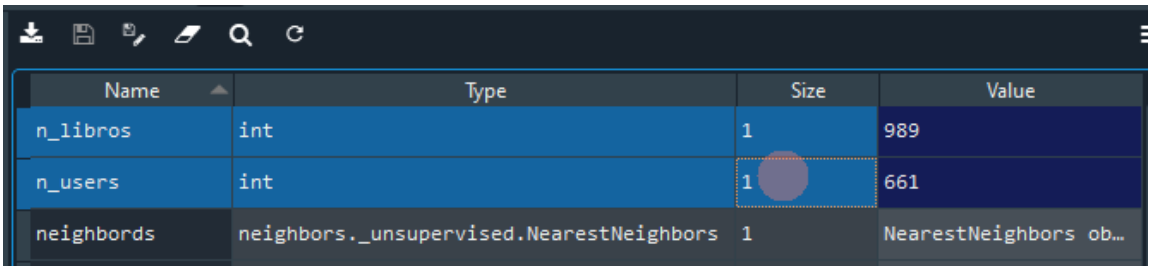
En la figura se visualiza la lista de libros recomendados para el usuario activo según las calificaciones de ítems y el perfil de usuario. La retroalimentación explícita del sistema se realiza a través de la puntuación con el sistema de estrellas según la relevancia del contenido del libro para el usuario.

CAPÍTULO IV
RESULTADOS EXPERIMENTALES Y ANÁLISIS

4.1. Dataset de experimentación

En la experimentación se ha utilizado la data set de préstamos de libros de las bibliotecas del CRAI de la Universidad Peruana Unión, el conjunto de datos denominado “Rating.csv” que contiene 1466 calificaciones de 989 libros por 661 usuarios, también los conjuntos de datos “libros.csv” que contiene los datos de los libros y “usuarios.csv” con los atributos de los usuarios.

Se ha calculado la puntuación de la dispersión (score sparsity) del conjunto de datos de préstamos de libros de la biblioteca obteniendo el 0.18, esto muestra que existe dispersión de datos y el problema de escasez de calificaciones.



Name	Type	Size	Value
n_libros	int	1	989
n_users	int	1	661
neighbords	neighbors._unsupervised.NearestNeighbors	1	NearestNeighbors ob…

Figura 22. Número de usuarios y libros de data set.

4.2. Configuración de experimentación (train y test)

En la etapa de pre-procesamiento de conjunto de datos se ha convertido la frecuencia de préstamos de libros por los usuarios en ratings, las puntuaciones que dan a los libros que son únicos para cada usuario-libro. Todos los datos que se han usado están convertidos al formato rating.

En la figura se muestra los datos en formato rating:

Index	UserId	ItemId	Rating
0	1	1	2
1	2	2	1
2	3	3	1
3	4	4	1
4	5	5	1
5	5	6	1
6	6	7	2
7	7	8	1
8	7	9	1
9	8	10	2

Figura 23. Datos en formato rating.

El conjunto de datos original se ha dividido en dos: conjunto de datos de entrenamiento (train set) y conjunto de datos de prueba (test set) en forma aleatoria. El test set es 30% (199 datos) y el train set es 70% (462 datos) del conjunto de datos.

Figura 24 . Plan de pruebas de evaluación y validadción

El propósito es maximizar el conjunto de entrenamiento para aumentar más datos y mejorar el modelo, al mismo tiempo maximizar el conjunto de test para lograr mejores resultados con el fin de medir el sistema.

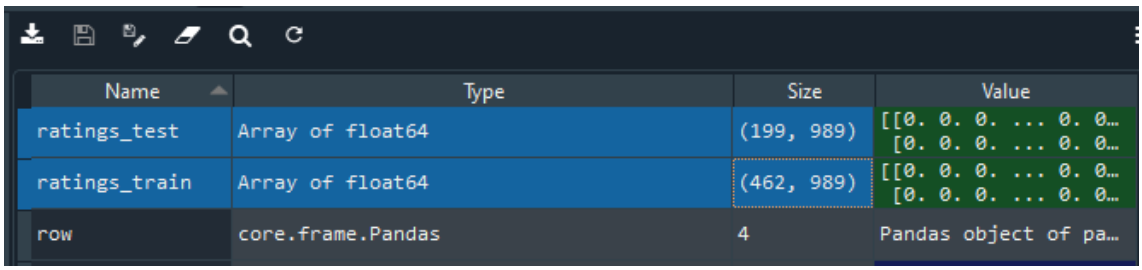


Figura 25. Ratings de entrenamiento y de prueba.

El performance del modelo de las predicciones de calificaciones se ha evaluado con la métrica de error MSE y la precisión de las predicciones de las recomendaciones con las métricas de Precision, Recall y F1 score. El entrenamiento se ha realizado en un equipo estándar con un procesador i5, velocidad 2,3 GHz y 8 GB de RAM.

4.3. Comparación de diferentes métricas de similitud

Existen diferentes métricas de similitud, en el trabajo se ha realizado pruebas con coeficiente de Jaccard, coeficiente de correlación de Pearson, similitud de Coseno y similitud de Euclídea, luego al comparar los resultados, la métrica de similitud de coseno permite lograr la mayor similitud, es decir un promedio que se acerca al 1 que es el óptimo.

Para la evaluación se ha utilizado las métricas de predicción de MSE y RMSE, y las métricas de clasificación de Precision, Recall y F1 score.

Como se puede observar en la tabla VII, la comparación de los resultados obtenidos en el entrenamiento del modelo con diferentes métricas se similitud. En las columnas 2 y 3 se presentan los errores de predicciones de calificación y en las columnas 4 al 6 la precisión y sensibilidad de las clasificaciones obtenidos en la evaluación del modelo KNN.

TABLA VII
COMPARACIÓN DE ALGORITMOS DE SIMILITUD CON DIFERENTES MÉTRICAS UTILIZANDO K=8

Medidas de distancia	MSE	RMSE	Precisión	Recall	F1
Similitud de Coseno	**0.36**	**0.60**	**0.79**	**0.83**	**0.76**
Coeficiente de Pearson	0.37	0.61	0.68	0.82	0.74
Coeficiente de Jaccard	0.37	0.61	0.68	0.82	0.74
Euclidean	0.36	0.60	0.78	0.83	0.77

En la tabla anterior se muestra que la similitud de coseno tiene mejores promedios en la precisión (0.79), sensibilidad (recall) (0.83) y F1 (0.76) esto significa que la predicción de las recomendaciones es aceptable, en comparación con otros algoritmos de similitud. Además, los promedios de error son bajos: MSE (0.36) y RMSE (0.60) en las predicciones de calificaciones de ítems.

Por tanto, se ha considerado que el algoritmo de similitud de coseno es el adecuado para el entrenamiento del modelo con la muestra del conjunto de datos.

4.4. Evaluación del modelo de sistema de recomendación

Para la evaluación de resultados de predicciones del modelo se ha utilizado las métricas de predicción de Error Cuadrático Medio (en inglés Mean Squared Error, MSE) y Raíz cuadrada de Error Cuadrático Medio (en inglés Root Mean Squared Error, RMSE), estas métricas permiten comparar el error de predicciones de calificación de ítems con la calificación real. El promedio menor de error representa la precisión y el mejor performance o rendimiento del modelo de sistema de recomendación.

4.5. Performace relativo de diferentes medidas de similitud

Para conocer el performance de los diferentes algoritmos de similaridad usando las métricas de error con el MSE y RMSE, lo que se requiere es obtener el promedio de error más bajo en las predicciones.

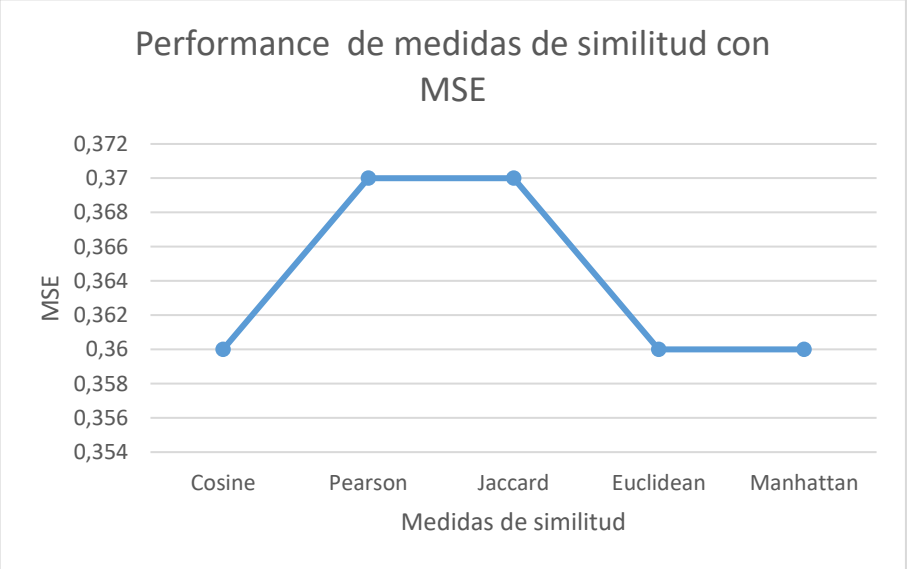


Figura 26 Performance de algoritmos según la métrica MSE

El MSE se usa cuando la preocupación es los valores inesperados, ya que la métrica amplifica los errores mayores, eso hace que no sea muy intuitivo.

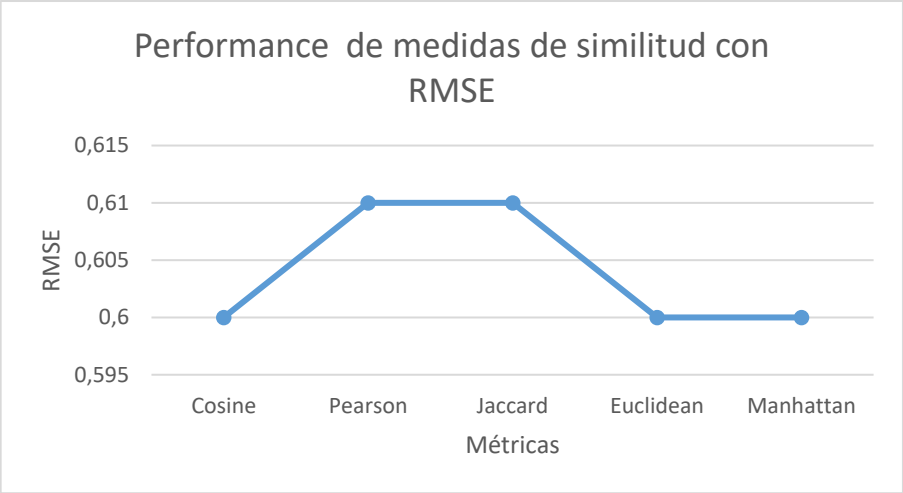


Figura 27 Performance de algoritmos según la métrica de RMSE

El RMSE es más intuitivo y se usa al final del entrenamiento como estimación de la calidad de las predicciones [55].

En las figuras 26 y 27 se muestran los resultados del performance de algoritmos de similitud en la predicción de calificaciones, los menores promedios de error (MSE 0.36 y RMSE 0.6) se obtienen con el algoritmo de similitud de coseno al igual que euclidean y manhattan. Esto puede deberse a los varios factores que influyen en las medidas de similitud como escasez de datos, pero existen mecanismos de confianza que ayuden a mejorar la precisión de las recomendaciones. Para el entrenamiento del modelo KNN se ha elegido la similitud de coseno tiene mejores resultados tal como se visualiza en la tabla comparativa.

4.6. Análisis y resultados de experimentación

El performance del modelo KNN depende de dos factores: la mejor métrica de similitud y hallar el mejor valor K para el entrenamiento. Por consiguiente. se ha realizado las pruebas con diferentes valores de K usando el lenguaje de programación Python. El valor de K=8 tiene el menor error (0.17) es el óptimo para el entrenamiento, los resultados se visualizan en la figura 28.

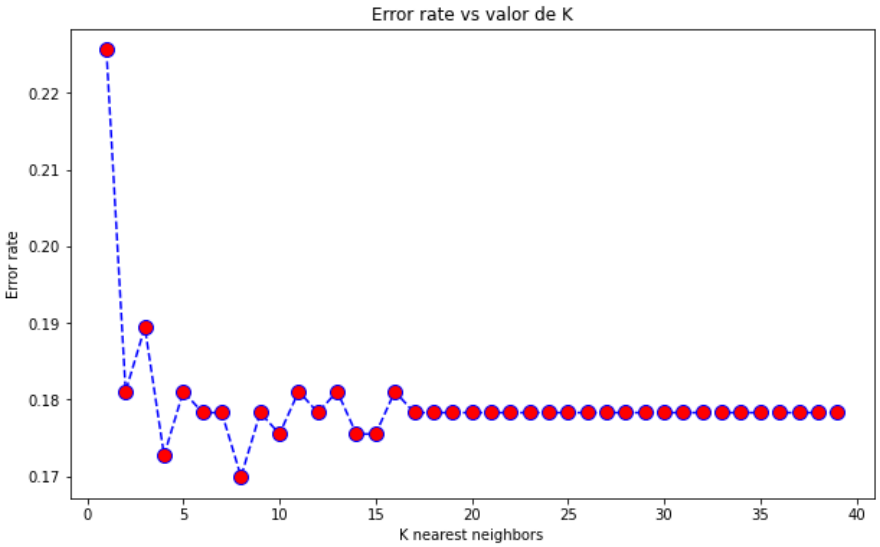


Figura 28. Tasa de error respecto a valores de K vecinos

.

Por otro lado, las métricas de clasificación que se han utilizado para la evaluación de la precisión de las predicciones del modelo de recomendación son: Recall, Precisión y F1 score.

La Precisión mide el porcentaje de libros recomendados al usuario objetivo que realmente eran relevantes para él.

Por consiguiente, las pruebas se han realizado con el valor de K=8 con diferentes algoritmos de similitud para lograr el mejor performance del algoritmo. En la figura 29 se muestran los resultados obtenidos.

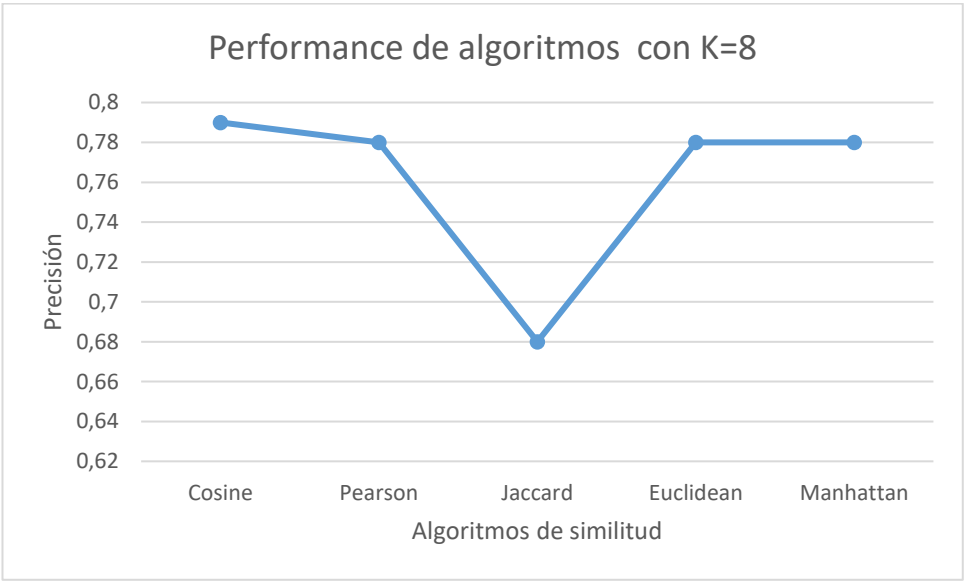


Figura 29. Performance de algoritmos con Precisión y con K=8

En la figura los resultados muestran que el algoritmo de coseno obtiene el promedio de 0.79, un valor que se acerca más al 1, lo que significa que la precisión es buena y el performance del algoritmo es mejor para las predicciones de recomendación.

Por otro lado, es recomendable aplicar la métrica de Recall que mide el promedio de libros relevantes para el usuario objetivo que efectivamente fueron recomendados. Para conocer el mejor performance de los algoritmos se ha entrenado con el valor de K=8.

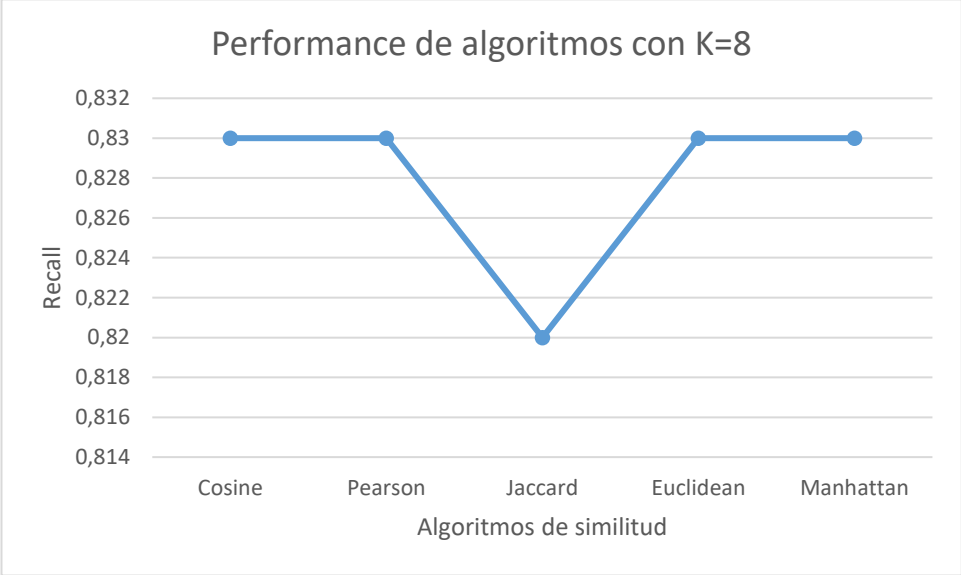


Figura 30. Performance de algoritmos con Recall y con K=8

En la figura se visualiza que la mayoría de los algoritmos logran un buen performance para la predicción de recomendación de libros relevantes para el usuario objetivo, pero esto se puede mejorar con otras técnicas de evaluación y validación del modelo.

Los valores obtenidos con las métricas de Precisión y Recall deben tener equilibrio, estos valores se pueden acercar a 1 en un caso ideal, para ello se utiliza el F1 score.

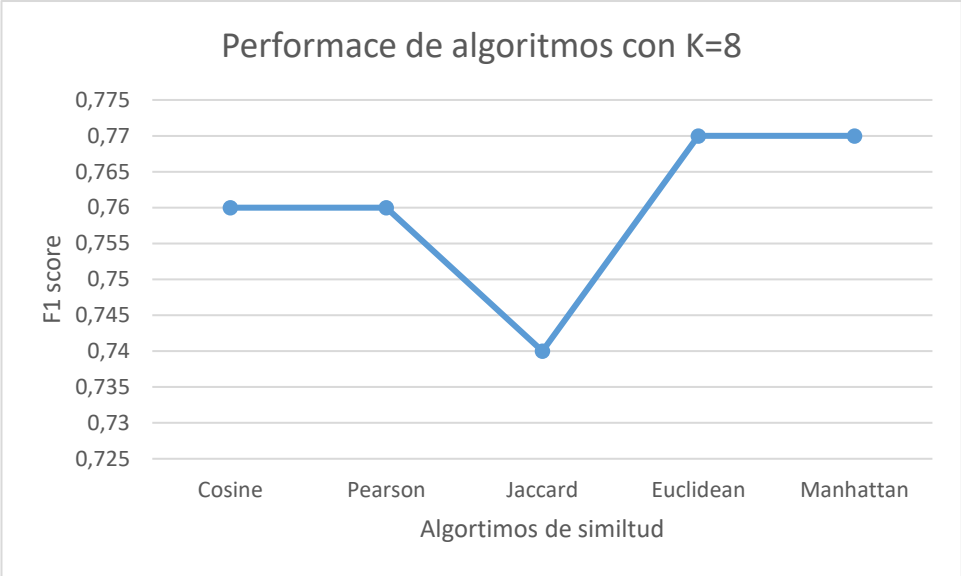


Figura 31. Performace de algoritmos con F1 score y valor de K=8

Cuando hay mucha variación en los promedios de Precision y Recall, la métrica F1 score calcula el promedio armónico de las dos métricas, obteniendo un valor más objetivo o el mejor resultado cuando se tiene interés en conocer la exactitud y la integridad. Pero en los resultados se muestran que no hay mucha variación, por tanto, no es relevante para el caso.

4.6.1. Performance de algoritmos de similitud con diferentes valores de K

La medida de performance de un modelo de predicción es fundamental para conocer la variación de puntajes de Precisión, Recall y F1 score con los algoritmos de similitud de coseno, coeficiente de Pearson, coeficiente de Jaccard, Euclidean y Manhattan.

En el modelo K-Nerarest neighbors es fundamental hallar el valor de K óptimo para obtener los mejores resultados de predicción. El cálculo de puntaje de precisión se ha realizado con tres instancias de k-vecinos más cercanos (K=8, K=12 y K=16) para demostrar la variación dependiendo del valor de K.

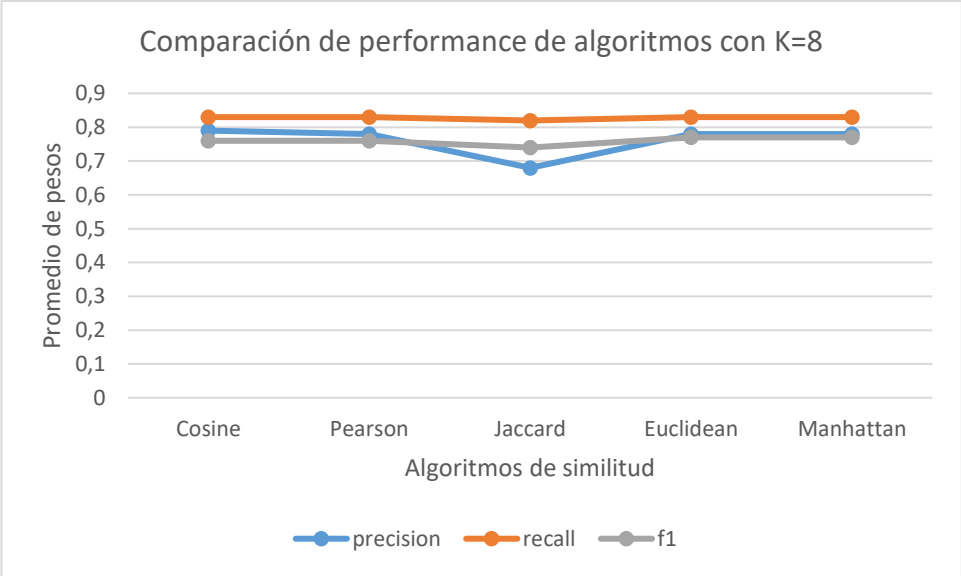


Figura 32. Comparación de algoritmos de similitud usando valor de K=8

En la figura 32 se muestran que los promedios de precisión (0.79) y recall (0.83) se aproximan a 1 con el algoritmo de coseno y se demuestra que el valor de K=8 es el óptimo para realizar las predicciones de recomendación.

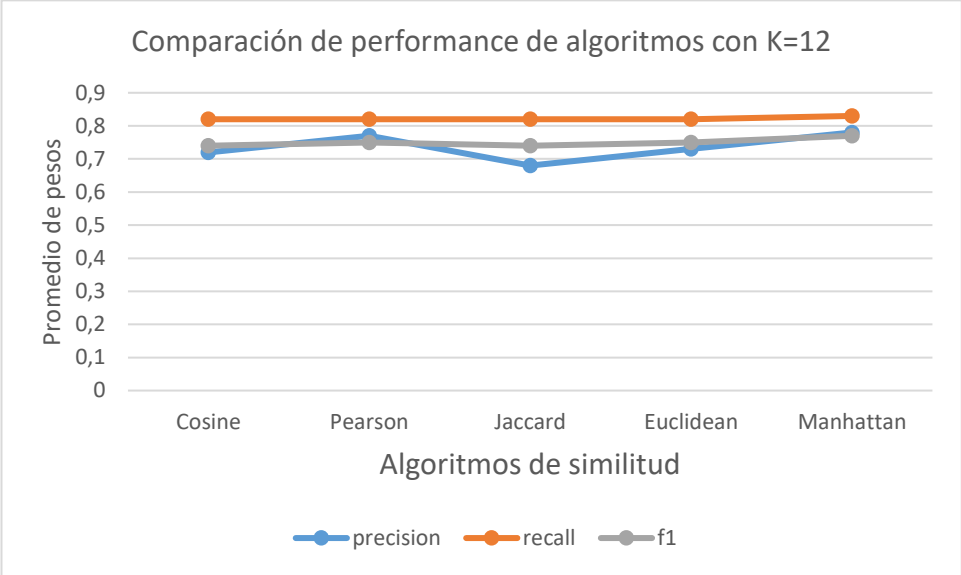


Figura 33. Comparación de algoritmos de similitud usando valor de K=12

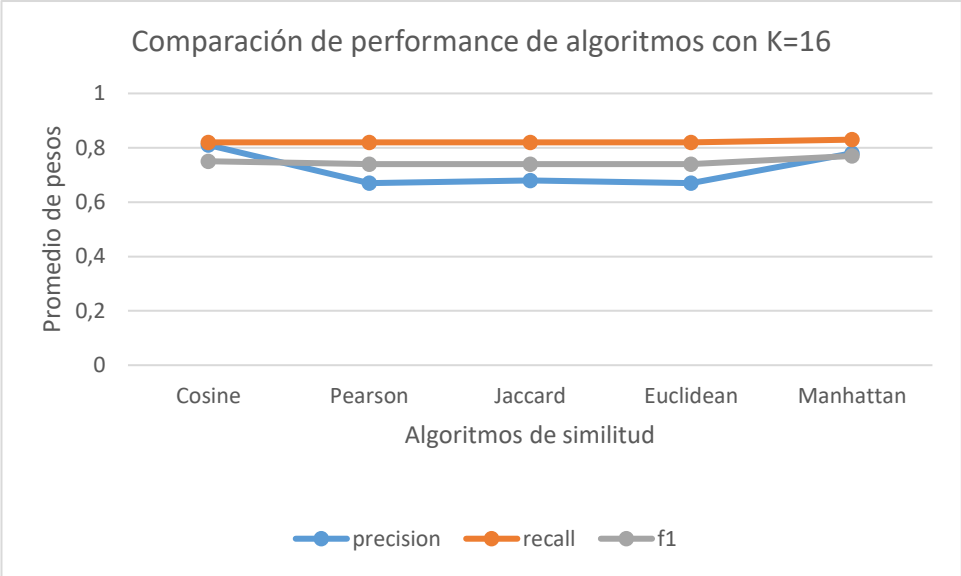


Figura 34. Comparación de algoritmos de similitud usando valor de K=16

Como se observa en las figuras 33 y 34 las pruebas del modelo realizadas con diferentes valores de K, los promedios obtenidos con K=12 son: precisión (0.72) y recall (0.82) y con K=16 la precisión (0.81) y recall (0.82) respectivamente; en cambio con K=8 se logra mejores resultados.

Después de la evaluación de los resultados, el mejor performance del modelo de recomendación se logra utilizando la métrica de similitud de coseno y el valor de K=8, para realizar las predicciones de recomendación de libros a los usuarios según sus preferencias.

4.6.2. Optimización del modelo con mejores hiperparámetros

Para la optimización del modelo de recomendación se ha usado la técnica de validación cruzada de cinco iteraciones para evitar los sesgos en el entrenamiento, y permite obtener mejores resultados en la evaluación y validación del modelo.

En un modelo, la optimización de los hiperparámentos es crucial para lograr el mejor rendimiento del algoritmo. La optimización del modelo se ha realizado con la búsqueda de hiperparámetros para encontrar mejores predicciones y con el menor error MSE y RMSE, para ello se ha utilizado la técnica de RandomSearch con la función RandomizedSearchCV de Scikit Learn que permite realizar una búsqueda aleatoria, encontrando los mejores parámetros para entrenar el modelo, teniendo control del número de combinaciones de parámetros y el número de iteraciones de búsqueda con validación cruzada (cross validation). El conjunto de datos se ha dividido en cinco pliegues, en la primera iteración,

el conjunto de pruebas utiliza el primer pliegue repitiendo el proceso hasta utilizar todos los pliegues, y se estiman los errores cuyo promedio es la estimación final (0,83).

El siguiente pseucódigo muestra la aplicación de validación cruzada para la evaluación y validación de la optimización del modelo:

```
# Combinación de preprocesado y el modelado en un pipeline
pipe → Pipeline ([('preprocesamiento', preprocesor), ('modelo', NearestNeighbors (n_vecinos = 8,
metrica = "coseno"))])

# optimización de hiperparámetros
parámetro_distribuciones → ['modelo K vecinos más cercanos': np.linspace (1, 100, 500, dtype =
entero)]

grid → funcion RandomizedSearchCV (
            estimador  = pipe,
            parámetro_distribuciones = parámetro_distribuciones,
            n_iteraciones = 20,
            puntaje = 'neg_root_mean_squared_error',
            n_jobs  = multiprocessing.cpu_count () - 1,
            validación cruzada = función RepeatedKFold (n_pliegues = 5, n_repiticion = 3),
            reentrenar  = si,
            verbose = 0,
            random_state = 123,
            devolver_puntaje_entrena = si
        )
Entrenar grid.fit (x = x_entrena, y= y_entrena)

# Resultados obtenidos con grid
resultados → funcion pd.dataframe (grid.cv_result_)
filtrar resultados.filter (regex = '(param.*|mean_t|std_t)')\
          .sort_values('promedio puntaje de prueba', ascendiente = Falso)\
          .head(1)
```

Al ejecutar la función se obtienen los valores de hiperparámetros de mean train score para el posterior entrenamiento del modelo con los valores de K con errores mínimos [56].

En la figura 35 se puede observar que los mejores parámetros se obtienen con los valores de k-vecinos desde K=7 en adelante en forma ascendente en mean test score, luego se estabiliza los valores de errores mínimos desde K=49 en los 100 primeros k-vecinos.

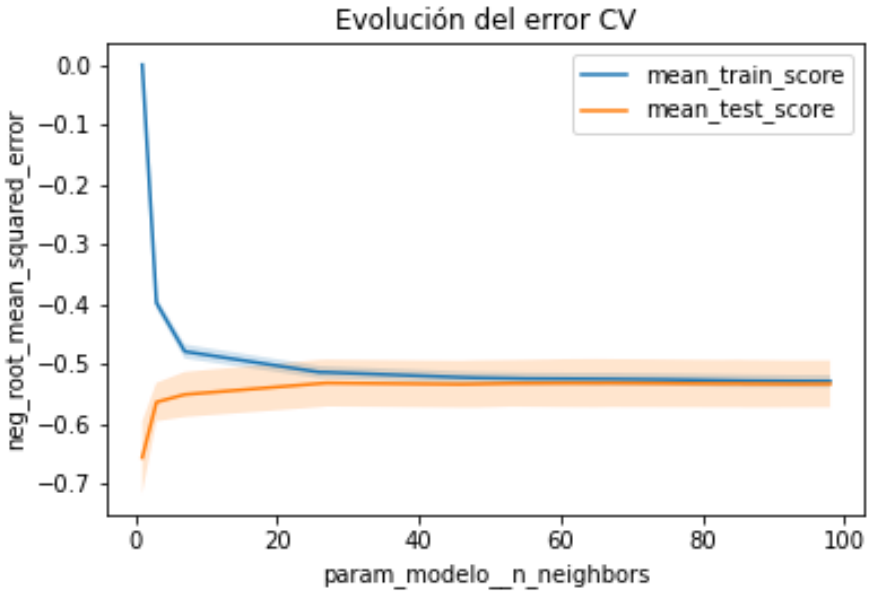


Figura 35. Promedio de RMSE con validación cruzada

En el experimento se ha definido el número de K vecinos más cercanos un rango de 1 a 100 para hallar el valor de K óptimo. Los resultados del experimento muestran que el RMSE decrece con el incremento de K vecinos más cercanos, luego se estabiliza desde el valor de K=20 en adelante.

4.6.3. Entrenamiento del modelo con mejores parámetros

Después de encontrar los mejores parámetros se ha entrenado nuevamente el modelo con la finalidad de lograr la optimización. A continuación, se muestra parte del seudocódigo del entrenamiento:

```
Buscar_parámetro= función randomizedsearchCV(modelo, parámetros, nro_iteraciones=5,
        Validación cruzada=5…)
Entrenar= buscar_parámetro.función_entrenar_fit(x_entrena, y_entrena)
Predicción=entrenar.función predict(x_prueba)
Imprimir (puntaje con nuevo parámetro (x_entrena, y_entrena))
Imprimir (puntaje con nuevo parámetro (x_prueba, y_prueba))
```

En la figura 36 se presentan los resultados obtenidos del entrenamiento del train score y test score, además el mejor parámetro de valor de K=8 que permite lograr el mejor performance del modelo. Los resultados de entrenamiento y la prueba son muy similares.

```
Train score =0.8251497005988024
Test score =0.83008356545961
{'n_neighbors': 8}
```

Figura 36. Resultados del entrenamiento con mejores parámetros

4.6.4. Métrica de evaluación del modelo

Matriz de confusión

También se conoce como matriz de error, es un método utilizado para la evaluación del rendimiento de un modelo de clasificación; se resume en una tabla la cantidad de predicciones correctas e incorrectas comparando los valores reales con los predichos.

Se ha realizado el entrenamiento del modelo con los nuevos parámetros, luego se ha realizado la evaluación de su rendimiento con la matriz de confusión, obteniendo los resultados que se visualizan en la figura 37:

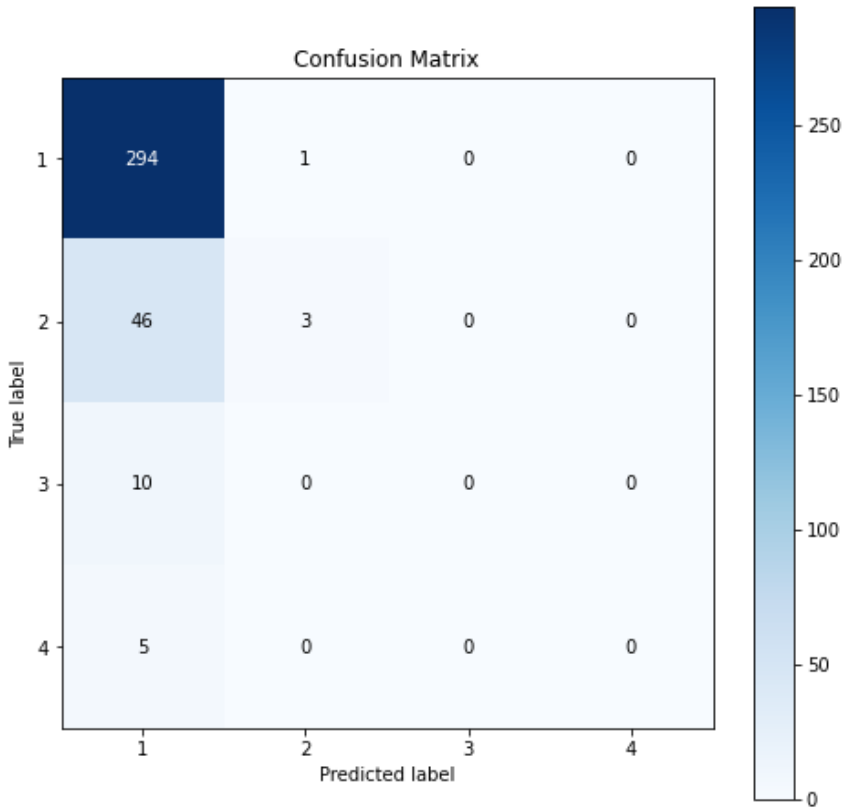


Figura 37. Evaluación del modelo con matriz de confusión

Para la matriz de confusión se ha usado el conjunto de prueba que consta de 359 datos, de los cuales 294 son evaluados como verdaderos positivos, lo que significa que son ítems relevantes para los usuarios y se han predicho como relevantes.

El número de falsos positivos es 1, lo que indica que las predicciones son óptimas. En las predicciones lo que se desea es que haya menos falsos positivos. En cambio, en las predicciones se ha tenido 46 falsos negativos, lo que significa que 46 ítems debían ser predichos como relevantes para los usuarios, pero se predijo como no relevantes.

La métrica de precisión es útil cuando los falsos positivos son muy preocupantes que los falsos negativos. Tal como se visualiza en la figura 34, el número de falsos positivos es 1; por lo tanto, no es la mejor métrica.

En contraste la utilidad de la métrica de recall es mejor cuando los falsos negativos son mayores o sea más preocupantes que los falsos positivos. En la figura anterior se muestra que el número de falsos negativos es 46 mayor que los falsos positivos. Por consiguiente, es convenientes considerar la métrica recall que en el porcentaje general es 83% lo que se visualiza en la figura siguiente.

	precision	recall	f1-score	support
1	0.83	1.00	0.90	295
2	0.75	0.06	0.11	49
3	0.00	0.00	0.00	10
4	0.00	0.00	0.00	5
accuracy			0.83	359
macro avg	0.39	0.26	0.25	359
weighted avg	0.78	0.83	0.76	359

Figura 38. Reporte de métricas de clasificación

Se ha optimizado el modelo usando mejores hiperparámetros logrando una exactitud de 83% de las predicciones correctas con respecto al total o sea los casos clasificados correctamente por el modelo, con la evaluación de accuracy (0.83) como se visualiza en la figura 38.

El porcentaje obtenido es significativo, pero las predicciones solo consideran la clase mayoritaria (clase 1); sin embargo, no se muestra las predicciones de las clases minoritarias (clases 2, 3, 4). Esto sucede cuando las clases no tienen muestras uniformes conocido como clases desequilibradas.

Clases desbalanceadas

Las clases desbalanceadas son un problema de clasificación y se considera como una tarea difícil [37], estos casos se presentan cuando una de las clases es minoritaria, es decir tiene muy pocas muestras. Los datos desbalanceados afectan al algoritmo en el proceso de generalización de la información. La medición de la efectividad del modelo solo se realiza teniendo en cuenta la clase mayoritaria con un porcentaje alto de aciertos dando

entender que el modelo funciona bien; sin embargo, no se ha tenido aciertos en la clase minoritaria.

Los problemas de clasificación multiclase que deben predecir dos o más etiquetas son bastante desafiantes cuando las clases son desequilibradas, porque requiere una muestra representativa de cada clase para que el modelo realice la predicción. Además, requiere de un diseño cuidadoso de una métrica de evaluación y las pruebas para la elección del modelo.

En la figura 39 se observa la cantidad de clases que tiene la muestra con sus respectivas calificaciones:

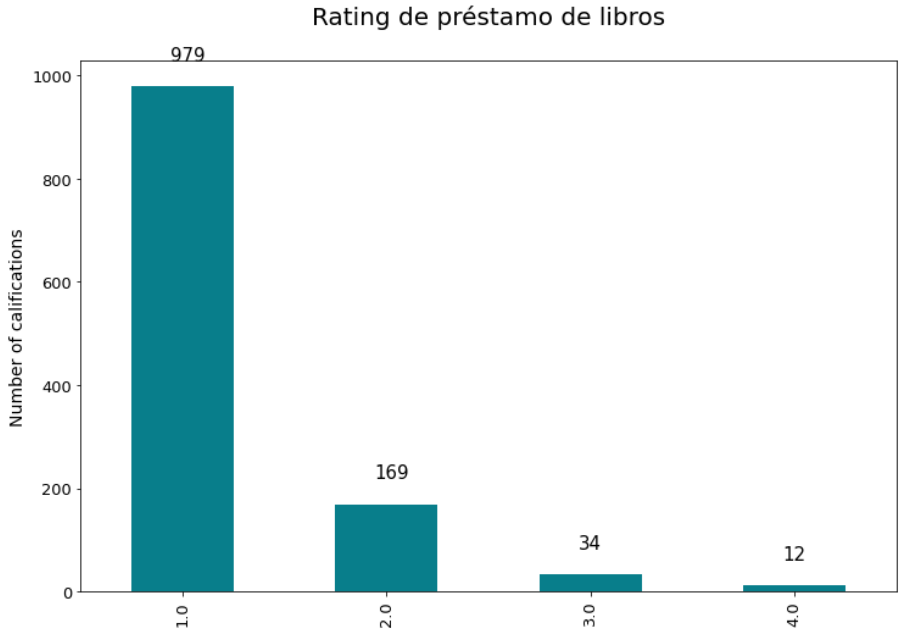


Figura 39. Las clases de data set con las calificaciones

Balanceo de datos de la muestra

La nueva muestra se generará considerando los k-vecinos más cercanos para ello se debe seleccionar uno de estos vecinos más cercanos, con la siguiente ecuación:

$$x_{new} = x_i + \lambda * (x_{zi} - x_i)$$

Donde: λ es un número aleatorio en el rango [0,1]. Esta interpolación creará una muestra en la línea entre X_i y X_{zi}. Las categorías de una nueva muestra se elige la categoría más frecuente de los vecinos más cercanos.

Existen varias técnicas para mejorar el conjunto de datos desequilibrados. En la experimentación se ha usado la función sobremuestreo aleatorio (RandomOverSampler) de las librerías de imblearn.over_sampling de Python como se visualiza en la figura 40.

```
os =  RandomOverSampler(sampling_strategy="auto", random_state=12)
x_os, y_os = os.fit_resample(x, y)
x_train, x_test, y_train, y_test=train_test_split(x_os, y_os,test_size=0.3,random_state=42)

model = KNeighborsClassifier(n_neighbors=8, metric="cosine")
model.fit(x_train, y_train)
preds = model.predict(x_test)
```

Figura 40. Código de la función para balanceo de datos de las clases

La función RandomOverSampler permite generar aleatoriamente nuevos datos sintéticos considerando los datos de las muestras de clases minoritarias para uniformizar las muestras de todas las clases. Después de ejecutar la función, las muestras de las clases minoritarias (clases 2, 3, 4) se han homogenizado con la muestra de la clase mayoritaria (clase 1), tal como se muestra en la figura.

```
In [41]: df_balanceado_smT.Rating.value_counts()
Out[41]:
4.0    979
3.0    979
1.0    979
2.0    979
Name: Rating, dtype: int64
```

Figura 41 Resultados de balanceo de datos

Después del balanceo de las clases, se ha entrenado el modelo con las nuevas muestras de las clases para lograr el mejor rendimiento del modelo KNN, utilizando el algoritmo de similitud de coseno y el mismo número de vecinos más cercanos (K=8).

Luego se ha realizado la evaluación del modelo con la matriz de confusión, obteniendo los resultados esperados que se presentan en la figura 42.

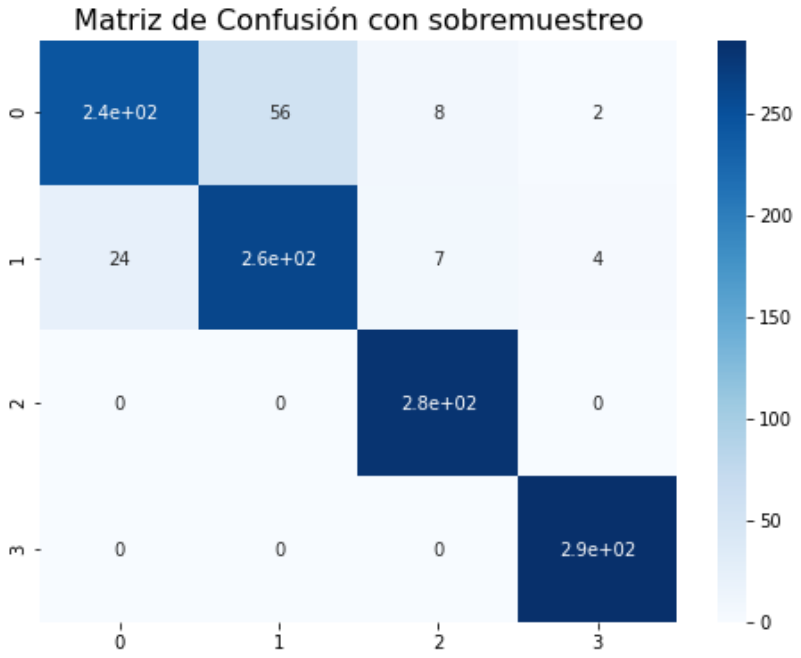


Figura 42. Evaluación del modelo con matriz de confusión con datos balanceados.

La evaluación de los resultados de matriz de confusión se realiza con las métricas de clasificación de Precición, Recall y F1 score, que permiten obtener mejores resultados. Los resultados se han obtenidos con las siguientes fórmulas:

$$\text{Precision} = \frac{TP}{(TP+FP)} \qquad \text{Recall} = \frac{TP}{(TP+FN)}$$

$$\text{Precision} = \frac{286}{(286+6)} \qquad \text{Recall} = \frac{286}{(286+0)}$$

$$\text{Precision} = 0{,}98 \qquad \text{Recall} = 1{,}0$$

$$\text{F1 score} = 2 * \frac{Precision*Recall}{Precision+Recall}$$

$$\text{F1 score} = 2 * \frac{0{,}98}{1{,}08}$$

$$\text{F1 score} = 0{,}99$$

Los resultados obtenidos muestran que los promedios de precisión y sensibilidad de las clases minoritarias han aumentado por encima de 90% y un 91% de exactitud del modelo, que son resultados óptimos esperados.

	precision	recall	f1-score	support
1.0	0.91	0.79	0.84	311
2.0	0.82	0.88	0.85	297
3.0	0.95	1.00	0.97	281
4.0	0.98	1.00	0.99	286
accuracy			0.91	1175
macro avg	0.92	0.92	0.92	1175
weighted avg	0.91	0.91	0.91	1175

Figura 43. Reporte de métricas de clasificación después del balanceo de datos.

Se ha usado el mismo conjunto de datos de prueba, pero con sobremuestreo que consta de un total de 1175 datos. En la clase 1 (clase 0 en la matriz de confusión), se tiene un total de 311 muestras, de los cuales 245 son evaluados como verdaderos positivos, lo que significa que la mayoría de ítems relevantes para los usuarios y se han predicho como relevantes. Sin embrago, se ha obtenido 66 falsos positivos, lo que significa que 66 ítems debían ser predichos como relevantes para los usuarios, pero se predijo como no relevantes; siempre se desea que haya menos falsos positivos. En cambio, se ha obtenido 24 falsos negativos, lo que significa que 24 ítems debían ser predichos como no relevantes para los usuarios, pero se predijo como relevantes.

En la clase 1 se ha obtenido una precisión de 91% de las predicciones que son correctas, y una integridad o sensibilidad (recall) de 79% de los ítems relevantes para los usuarios se predijo correctamente; lo que demuestra que el modelo tiene un buen performance y las predicciones son óptimas.

En la clase 2, el total de la muestra es 297, de los cuales 262 son evaluados como verdaderos positivos, lo que significa que los ítems relevantes se han predicho correctamente. En esta clase, se ha obtenido una precisión de 0.82 lo que significa que el 82% de las predicciones son correctas, y una sensibilidad de 0.88, lo que explica que el 88% de los ítems relevantes para los usuarios se predijo correctamente como relevantes; por tanto, se ha logrado una precisión e integridad altas lo que indica que los resultados son óptimos.

En la clase 3, el total de la muestra es 281, de los cuales 281 es evaluado como verdadero positivo, lo que significa que casi todos los ítems relevantes se han predicho correctamente. Las predicciones tienen una precisión de 0.95 lo que significa que el 95% de las predicciones son correctas, y una sensibilidad de 1.0, lo que explica que el 100% de

los ítems relevantes para los usuarios se han predicho correctamente en su totalidad como relevantes.

Asimismo, en la clase 4, los resultados obtenidos son muy similares a la clase 3 ya que las predicciones alcanzan una precisión de 98%, y una sensibilidad de 100% de los ítems relevantes para los usuarios se han predicho correctamente en su totalidad como relevantes.

En el trabajo de Tian et al. [57] sobre el sistema de recomendación personalizado de biblioteca universitaria hace una comparación de filtrado colaborativo, de contenido y algoritmo híbrido, logrando obtener un porcentaje de precisión de 40% en el híbrido, aunque se muestra que hay incremento a medida que aumenta el total de conjunto de entrenamiento. La medida de la precisión solo no es tan confiable sino también la sensibilidad que ayuda conocer si las predicciones obtenidas del modelo son relevantes y confiables.

Para las pruebas del modelo de recomendación de libros se ha aplicado el método de filtrado colaborativo, ya que es el más recomendado por mejores resultados para este tipo de ítems según los trabajos que se han realizado anteriormente [5] [6] [7] [11]. Este método usa patrones de comportamiento similares entre los usuarios, con preferencias y gustos comunes con el usuario actual.

Según uno de los objetivos del trabajo, se ha evaluado y validado el modelo de recomendación basado en el algoritmo de similaridad de coseno y con los mejores hiperparámetros con la matriz de confusión, obteniendo el mejor performance de 91% (0.91) logrando la optimización del modelo para el sistema de recomendación de libros del CRAI.

CAPÍTULO V
CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

Los sistemas de recomendación son muy usados en la organizaciones y empresas en la prestación de servicios digitales para ayudar a los usuarios a encontrar información de productos y servicios según las preferencias de los usuarios.

En el trabajo, según el objetivo principal que se ha planteado, se ha optimizado el performance del sistema de recomendación de libros usando el algoritmo de similaridad de coseno que ha permitido lograr el mejor performance del modelo de clasificación K-Nearest Neighbors. Para alcanzar tal objetivo se ha formulado los objetivos específicos, los cuales se han cumplido para lograr los resultados del trabajo. A continuación, se enuncian el cumplimiento de cada uno de los objetivos:

- El método CRISP-DM es muy usado en el análisis de datos y ha permitido ordenar los pasos a seguir para la experimentación y optimización del performance del sistema de recomendación de libros; asimismo el método de filtrado colaborativo por su enorme funcionalidad en casos de problemas de sobreinformación que enfrentan los usuarios en sus búsquedas, permite lograr el filtrado de libros con contenidos según sus preferencias y ha permitido realizar predicciones automáticas de intereses de los usuarios sobre los contenidos de libros, a través de la recopilación de información de preferencias de los mismos usuarios y de otros con intereses comunes.
- La aplicación del modelo Nearest Neighbor en el entrenamiento y optimización del modelo ha permitido manejar problemas de clasificación múltiple y predicciones. Cuando la muestra de datos de entrenamiento es multiclase, se debe realizar predicciones para cada clase; por lo tanto, el manejo de clases desequilibradas es un desafío en los sistemas de recomendación, por esta razón, se ha realizado el balaceo de la muestra de datos de entrenamiento para equilibrar las clases minoritarias a la clase mayoritaria, esto ha permitido obtener un promedio de exactitud (accuracy) de predicciones de 0.91 muy cercano al 1 que es el óptimo, demostrando el mejor performance del modelo de recomendación.
- En el modelo KNN es fundamental la elección de la métrica de similitud para lograr los resultados esperados; en el trabajo, la similitud de coseno ha tenido mejores resultados de promedio de precisión (0.79) en comparación con otras métricas de similitud como Jaccard, Euclidean, Pearson y otros. El rendimiento del modelo

depende particularmente de la muestra de datos que se ha usado en el entrenamiento; en la experimentación se ha trabajado con el dataset de préstamos de libros con calificaciones variados. La distancia de coseno usando el valor de ángulo de dos vectores ha permitido hallar la similitud entre las preferencias de los usuarios y también los ítems preferidos por usuarios similares al actual para la predicción de recomendaciones de libros. Después del balanceo de datos se llegó a obtener el promedio de precisión de 0.92 y un promedio de sensibilidad de 0.92, que muestran el mejor performance del modelo de predicciones.

- El desarrollo de una interfaz del usuario amigable de la aplicación móvil del sistema Android permite visualizar los resultados del modelo de predicciones e incentiva la interacción del usuario con el sistema de recomendación de libros de manera sencilla y efectiva, con tan solo registrarse y luego iniciar la sesión puede visualizar la lista de recomendaciones de libros con contenidos de su interés.

5.2. Recomendaciones

Considerando la importancia actual de los sistemas de recomendación se tiene las siguientes recomendaciones:

- El uso de los sistemas de recomendaciones de libros en las bibliotecas de las universidades puede ayudar a los usuarios a encontrar libros con contenidos de su preferencia, sin demorar mucho tiempo en las búsquedas y mejorar la satisfacción del usuario.

- La implementación del sistema de recomendación de libros en la biblioteca del CRAI de la universidad en la plataforma de aplicativo móvil o aplicación web ayudaría a mejorar el servicio y la experiencia del usuario.

Aunque se han cumplido los objetivos del trabajo, el modelo de sistema de recomendación de libros puede ser refinado y extendido para mejorar en los siguientes aspectos:

- Realizar pruebas del modelo con muestras de grandes cantidades de datos para lograr el mejor rendimiento y optimización del modelo, lo que supone que demandará recursos de mayor capacidad y velocidad de procesamiento de datos.

- La optimización del sistema de retroalimentación explícita e implícita que permita recopilar más información sobre las preferencias de los usuarios para el modelo

de sistema de recomendación de libros para realizar recomendaciones más eficientes y personalizados a los usuarios.

- Usar los servicios de Amazon Web Services (AWS) para realizar pruebas de funcionamiento del sistema de recomendación de libros en la nube.

REFERENCIAS

[1] M. D. Dussán, “Sistema de recomendación web basado en la actividad de los usuarios de la Universidad Nacional de Colombia,” Universidad Nacional de Colombia, 2012.

[2] E. R. Núnez, V. García, J. Pascual, C. E. Montenegro, J. M. Cueva, and O. Sanjuán, “Sistema de recomendación de contenidos para libros electrónicos inteligentes,” *Vent. Informática*, no. 26, pp. 111–127, 2012, [Online]. Available: http://revistasum.umanizales.edu.co/ojs/index.php/ventanainformatica/article/view/138/193.

[3] X. Wang, Z. Xu, X. Xia, and C. Mao, “Computing user similarity by combining SimRank++ and cosine similarities to improve collaborative filtering,” *Proc. - 2017 14th Web Inf. Syst. Appl. Conf. WISA 2017*, vol. 2018-Janua, pp. 205–210, 2018, doi: 10.1109/WISA.2017.22.

[4] E. Walid, “Un sistema de recomendación basado en perfiles generados por agrupamiento y asociaciones,” Universitat Politècnica de València, 2017.

[5] Q. Ji, S. Yannan, L. Shi, and L. Xiang, “Research on Personalized Book Recommendation Model for New Readers,” pp. 3–6, 2018, doi: 10.1109/ICISE.2018.00022.

[6] Y. Liang and S. Wan, “The Design and Implementation of Books Recommendation System,” pp. 5–8, 2018.

[7] A. S. Tewari and K. Priyanka, “Book Recommendation System Based on Collaborative Filtering and Association Rule Mining for College Students,” pp. 135–138, 2014.

[8] A. S. Tewari, A. Kumar, and A. G. Barman, “Book Recommendation System Based on Combine Features of Content Based Filtering , Collaborative Filtering and Association Rule Mining,” pp. 500–503, 2014.

[9] C. Aygün and O. Yildiz, “Genetik Algoritma Kullanılarak İçerik Tabanlı Kitap Tavsiye Sistemi Geliştirilmesi Development Of Content Based Book Recommendation System Using Genetic Algorithm,” pp. 2–5, 2016.

[10] Y. Tian, B. Zheng, Y. Wang, Y. Zhang, and Q. Wu, “College Library personalized Recommendation System Based on Hybrid Recommendation Algorithm,” 2019, doi:

10.1016/j.procir.2019.04.126.

[11] Y. Zhu, “A book recommendation algorithm based on collaborative filtering,” in *2016 5th International Conference on Computer Science and Network Technology (ICCSNT)*, Dec. 2016, pp. 286–289, doi: 10.1109/ICCSNT.2016.8070165.

[12] K. Priyanka, A. S. Tewari, and A. G. Barman, “Personalised Book Recommendation System based on Opinion Mining Technique,” no. Gcct, pp. 285–289, 2015.

[13] M. F. Caro, J. Hernández, and J. A. Jiménez, “Diseño de un sistema de recomendación en repositorios de objetos de aprendizaje basado en la percepción del usuario: caso Rodas,” *Cienc. e Ing. Neogranadina*, vol. 21 (1), pp. 51–72, 2011.

[14] J. Guzmán-Luna, J. Cartagena, C. A. Pérez, and A. C. Alonso, “Propuesta de un sistema de recomendación híbrido semántico de objetos de aprendizaje en el área de educación ambiental basado en Linked Data,” *Rev. Colomb. Tecnol. Av.*, vol. 2 (30), pp. 56–63, 2017.

[15] D. G. Huamán, “Sistema de recomendación de libros basado en ontologías asociadas a tesauros : el caso de la Biblioteca de la Facultad de Ingeniería de Sistemas e Informática de la Universidad Nacional Mayor de San Marcos,” Universidad Nacional Mayor de San Marcos, 2019.

[16] M. P. Robillard, W. Maalej, R. J. Walker, and T. Zimmermann, *Recommendation Systems in Software Engineering*. Berlin: Springer-Verlag, 2014.

[17] J. B. Álvarez, “Sistemas de Recomendación de Servicios Turísticos Personalizados en el Destino Turístico Inteligente,” Universidad de Málaga, 2014.

[18] C. C. Aggarwal, *Recommender Systems*, vol. 40, no. 3. New York, USA: Springer International Publishing Switzerland, 2016.

[19] E. R. Núñez, “Sistema de recomendación de contenidos para libros inteligentes,” Universidad de Oviedo, 2012.

[20] G. M. Tarazona, J. S. Chávez, and R. Ferro, “Modelación de sistemas de recomendación aplicando redes neuronales artificiales Recommendation systems modeling applied artificial neural networks,” 2013.

[21] J. Pinho, “MÉTODOS DE CLASIFICACIÓN BASADOS EN ASOCIACIÓN APLICADOS A SISTEMAS DE RECOMENDACIÓN,” Universidad de Salamanca,

2010.

[22] A. Felfernig, L. Boratto, M. Stettinger, and M. Tkalcic, *Group Recommender Systems*. Springer, 2018.

[23] J. Castro, “Un nuevo modelo ponderado para Sistemas de recomendación basados en contenido con medidas de contingencia y entropía,” Universidad de Jaén, 2012.

[24] F. Ibáñez, “Métodos de agrupamiento en el cálculo de distancias entre usuarios en un sistema de recomendación de filtrado colaborativo,” Universidad Carlos III de Madrid, 2011.

[25] Ridwang, A. A. Ilham, I. Nurtanio, and Syafaruddin, “Image search optimization with web scraping, text processing and cosine similarity algorithms,” 2020.

[26] J. Zamorano Ruíz, “Comparativa y Análisis de Algoritmos de Aprendizaje Automático para la Predicción del Tipo Predominante de Cubierta Arbórea,” Universidad Complutense de Madrid, 2018.

[27] Y. A. Gerhana, A. R. Atmadja, W. B. Zulfikar, and N. Ashanti, “The implementation of K-nearest neighbor algorithm in case-based reasoning model for forming automatic answer identity and searching answer similarity of algorithm case,” in *5th International Conference on Cyber and IT Service Management, CITSM 2017*, 2017, pp. 3–7, doi: 10.1109/CITSM.2017.8089233.

[28] O. Kramer, *Dimensionality Reduction with Unsupervised Nearest Neighbors*, vol. 51. Springer-Verlag, 2013.

[29] J. Hernández, M. J. Ramírez, and C. Ferri, *Introducción a la minería de datos*. Madrid: Pearson Educación, 2004.

[30] J. Ghosh and A. Liu, “K-Means,” Taylor & Francis Group, LLC, 2009.

[31] M. Á. Álvarez Carmona, “Detección de similitud semántica en textos cortos,” Instituto Nacional de Astrofísica, Óptica y Electrónica, 2014.

[32] F. Li and R. Klette, “Approximate shortest paths in simple polyhedra,” 2011, doi: 10.1007/978-3-642-19867-0_43.

[33] M. Steinbach and P.-N. Tan, “kNN: k-Nearest Neighbors,” in *The Top Ten Algorithms in Data Mining*, 2009.

[34] K. Ramasubramanian and A. Singh, *Machine Learning Using R*, Second Edi. Apress, 2019.

[35] S. Yadav and S. Shukla, “Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification,” *Proc. - 6th Int. Adv. Comput. Conf. IACC 2016*, no. Cv, pp. 78–83, 2016, doi: 10.1109/IACC.2016.25.

[36] J. A. Cárdenas, “Clasificación de aceptación de campañas para una entidad financiera, usando random forest con datos balanceados y datos no balanceados,” Universidad Ricardo Palma, 2019.

[37] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, “A study of the behavior of several methods for balancing machine learning training data,” *ACM SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 20–29, 2004, doi: 10.1145/1007730.1007735.

[38] O. Embarak, *Data Analysis and Visualization Using Python Data Analysis and Python*. Abu Dhabi: Apress, 2018.

[39] S. Nagar, *Introduction to Python for Engineers and Scientists: Open Source Solutions for Numerical Computation*, Apress. New York, USA, 2018.

[40] D. Sarkar, R. Bali, and T. Sharma, *Practical Machine Learning with Python*. 2018.

[41] L. Gonzales, “6 IDEs para Machine Learning con Python - Ligdi González,” 2018. https://ligdigonzalez.com/ide-para-machine-learning-con-python/ (accessed Apr. 09, 2020).

[42] E. M. Cabanillas and R. Mori, “Nuevo modelo de aceptación tecnológica (TAM) y su relación con el grado de aceptación del App USMP Mobile,” Universidad de San Martín de Porres, 2018.

[43] J. B. Domínguez, *Manual de Metodología de la Investigación Científica*, 3rd ed. Chimbote: Imprenta Editora Gráfica Real S, 2015.

[44] R. S. Waslawick, *Metodologia de Pesquisa para Ciência da Computação*. Elsevier Editora Ltda., 2014.

[45] C. Espinoza Montes, *Metodología de la Investigación Tecnológica*. 2010.

[46] H. Sánchez, C. Reyes, and K. Mejía, *Manual de términos en investigación científica, tecnológica y humanística*, 1st ed. Lima: Bussiness Support Aneth

S.R.L., 2018.

[47] C. M. Arispe, J. S. Yangali, M. A. Guerrero, O. R. Lozada, L. A. Acuña, and C. Arellano, *La investigación científica. Una aproximación para los estudios de posgrado*, 1ra ed. Guayaquil: Universidad Internacional del Ecuador, 2020.

[48] I. Corporation, “IBM SPSS Modeler CRISP-DM Guide.” 2012, Accessed: Dec. 30, 2021. [Online]. Available: http://www.ibm.com/spss.

[49] R. Hurtado Ortiz, “Recomendación a grupos de usuarios usando el concepto de singularidades,” Universidad Politécnica de Madrid, 2020.

[50] F. Gil, “Tecnologías que mejoran la privacidad en los sistemas de recomendación.,” Universitat Oberta de Catalunya, 2017.

[51] F. Pedregosa, “Scikit-learn: Machine Learning in Python,” *JMLR*, vol. 12, pp. 2825–2830, 2011, [Online]. Available: https://scikit-learn.org/stable/modules/neighbors.html.

[52] R. Benítes, G. Escudero, S. Kanaan, and D. M. Rodó, *Inteligencia artificial avanzada*, 1st ed. Barcelona: Editoria UOC, 2014.

[53] R. Espinar, “Modelos de Clasificación con datos no balanceados,” Universidad de Sevilla, 2018.

[54] P. Yi, C. Li, C. Yang, and M. Chen, “An optimization method for recommendation system based on user implicit behavior,” *Proc. - 5th Int. Conf. Instrum. Meas. Comput. Commun. Control. IMCCC 2015*, pp. 1537–1540, 2016, doi: 10.1109/IMCCC.2015.326.

[55] J. Martinez, “Error Cuadrático Medio para Regresión,” 2020. https://www.iartificial.net/error-cuadratico-medio-para-regresion/ (accessed Sep. 05, 2021).

[56] R. Vasco, “Optimización Automática de Hiperparámetros en Modelos de Aprendizaje Automático mediante PBIL,” Universidad de Sevilla, 2019.

[57] Y. Tian, B. Zheng, Y. Wang, Y. Zhang, and Q. Wu, “College Library Personalized Recommendation System Based on Hybrid Recommendation Algorithm,” *Procedia CIRP*, vol. 83, pp. 490–494, 2019, doi: 10.1016/j.procir.2017.04.009.

ANEXOS

ANEXO A
Matriz de consistencia

<table>
<tr><th>Problema general</th><th>Objetivo general</th><th>Metodología</th></tr>
<tr><td>¿Cómo se puede optimizar el performance del sistema de recomendación de libros utilizando algoritmos de similaridad para ofrecer recomendaciones de libros según las preferencias de los usuarios del Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión?</td><td>Optimizar el performance del sistema de recomendación de libros basado en algoritmos de similaridad que permita ofrecer recomendaciones de textos con contenidos relevantes a los intereses de los usuarios del Centro de Recursos para el Aprendizaje y la Investigación de la Universidad Peruana Unión.</td><td rowspan="3">El diseño es no experimental porque no se realizará la manipulación de las variables y es descriptivo puesto que permite describir las variables.</td></tr>
<tr><th>Problemas específicos</th><th>Objetivos específicos</th></tr>
<tr><td>El sistema de información actual usa criterios de búsqueda no muy óptimos y es complejo encontrar contenidos de interés para los usuarios por el aumento de la cantidad de libros en las bibliotecas de la universidad; además, no cuenta con un módulo para realizar recomendaciones de libros con contenidos relevantes según las preferencias de los usuarios. ¿Cómo el uso de algoritmos de similaridad en el sistema de recomendación puede ayudar a ofrecer recomendaciones de libros precisas a los intereses de los usuarios del CRAI?</td><td>• Definir el método de sistema de recomendación que se adapte a las características del conjunto de datos disponible para la generación de recomendaciones de contenidos de libros.
• Evaluar y validar el algoritmo de similaridad que permita la optimización del sistema de recomendación de libros según las preferencias de los usuarios del CRAI con el mejor performance del modelo.</td></tr>
<tr><td>Para el problema de optimización de performance ¿Cómo se puede optimizar el performance del modelo logrando los mejores hiperparámetros para lograr recomendaciones de libros según las preferencias de los usuarios?</td><td>• Evaluar la precisión y la sensibilidad del modelo de sistema de recomendación de libros.
• Visualizar los resultados mediante una interfaz de aplicación móvil de sistema Android de las recomendaciones de contenidos de libros.</td></tr>
</table>

ANEXO B
Código de balanceo de datos de las clases en Python

*Codigo Balanceo de datos: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
os =  RandomOverSampler(sampling_strategy="auto", random_state=42)
x_os, y_os = os.fit_resample(x, y)
#nm = NearMiss(sampling_strategy="auto", n_neighbors=8, version=1)
#x_nm, y_nm = nm.fit_resample(x, y)
print(f'''Shape of X before SMOTE: {x.shape}
Shape of X after SMOTE: {x_os.shape}''')

print('\nBalance of positive and negative classes (%):')
y_os.value_counts(normalize=True)*100

#x_train, x_test, y_train, y_test=train_test_split(x_sm, y_sm,test_size=0.3,random_state=42)
#x_train, x_test, y_train, y_test=train_test_split(x_smt, y_smt,test_size=0.3,random_state=42)
x_train, x_test, y_train, y_test=train_test_split(x_os, y_os,test_size=0.3,random_state=42)
#x_train, x_test, y_train, y_test=train_test_split(x_nm, y_nm,test_size=0.3,random_state=42)
len(x_train)
model = KNeighborsClassifier(n_neighbors=8, metric="cosine")
model.fit(x_train, y_train)
preds = model.predict(x_test)

# Evaluate
#print(f'Accuracy = {accuracy_score(y_test, preds):.2f}\nRecall = {recall_score(y_test, preds):.2f}\n')
cm = confusion_matrix(y_test, preds)
plt.figure(figsize=(8, 6))
plt.title('Matriz de Confusión con sobremuestreo', size=16)
sns.heatmap(cm, annot=True, cmap='Blues');

#Matriz de confusión
conf_mat = confusion_matrix(y_test, preds)
print(conf_mat)
```

ANEXO C
Manual de usuario de la aplicación

MANUAL DE USUARIO APLICACIÓN UPeU|CRAI

1. REQUERIMIENTOS

La aplicación UPEU|Crai tiene requerimientos mínimos para que funcione correctamente, los cuales se detallan:

- Sistema operativo Android.
- Conectividad (Internet).

2. INSTALACIÓN

Después de la descarga de la aplicación en e móvil, la instalación se realiza en forma automática. Luego de instalar se debe ubicar el ícono ejecutable de la aplicación, para ingresar a la aplicación UPEU|Crai, sólo es necesario tocar el icono de la aplicación.

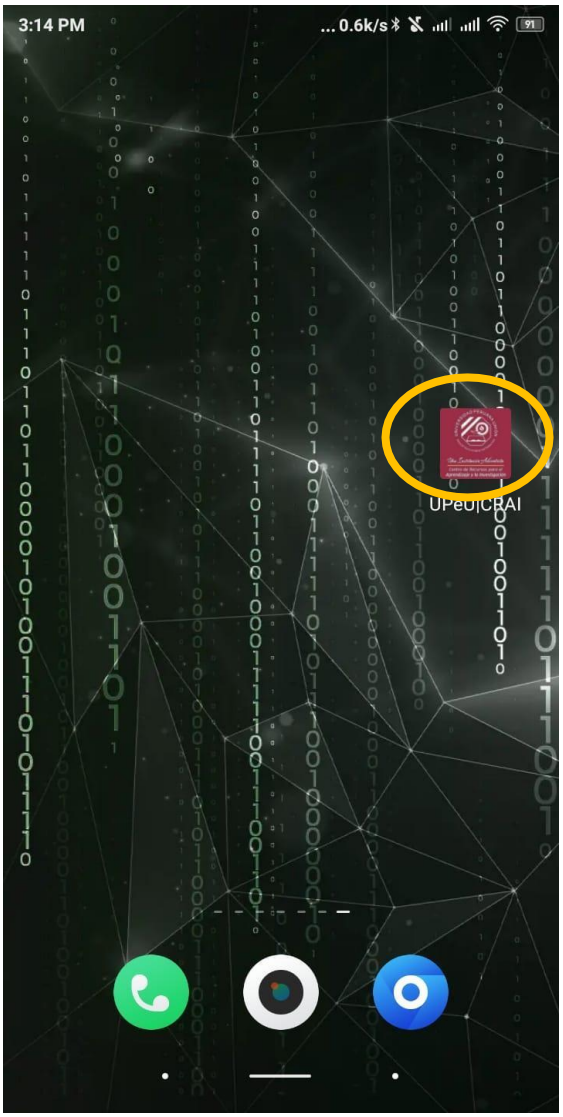


Fig. 1. Pantalla en el teléfono que muestra la aplicación UPEU|Crai

3. USO DE APLICACIÓN

La aplicación UPEU|Crai se ubica en la pantalla del dispositivo móvil y luego presionar sobre el ícono para que se inicie de manera automática. Si no existe conexión al internet, la pantalla de inicio(espere) se verá tal y como se muestra en la figura 2. Por lo tanto, la aplicación no se abrirá, pero no es un problema de la instalacióno. ¡No te preocupes! Para hacer el uso de aplicación tendrás que conectarte a internet.

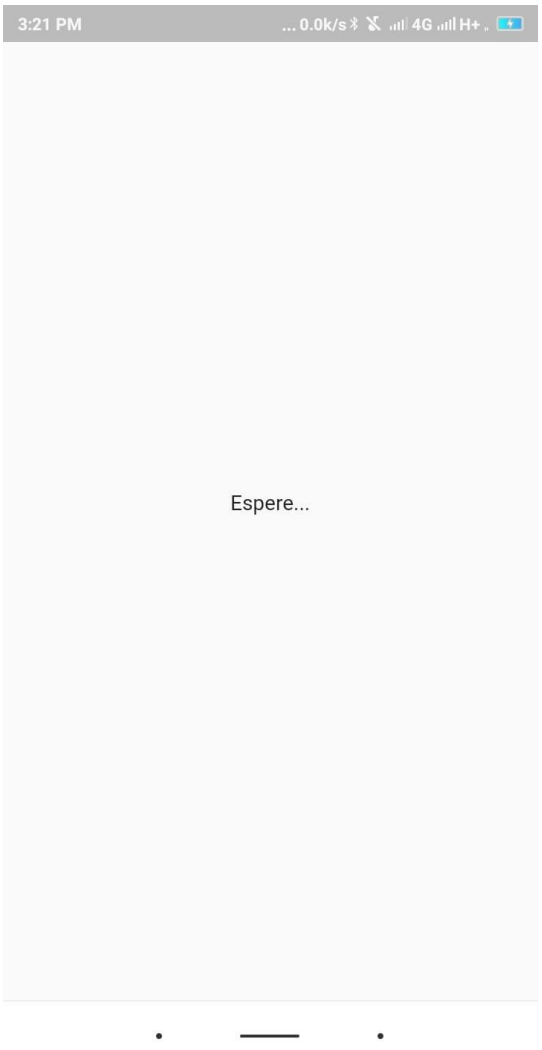


Fig. 2. Pantalla de espere de UPEU|Crai (sin internet).

Registrarse: Para el registro como usuario nuevo se debe ingresar a la opción “Regístrate” que aparece en la figura 3.

Fig. 3. Pantalla de login de UPEU|Crai.

Para registrarse debe de ingresar su código universitario, luego ingresar el código presionar el botón consultar tal como se muestra en la figura 4, si el código se encuentra la aplicación mostrara un mensaje de que el usuario ya se encuentra registrado, esto redireccionara a la pantalla login figura 3, ahora si el usuario está disponible redireccionara a la pantalla de registro figura 5.

Fig. 4. Pantalla de consulta de código universitario UPEU|Crai.

Llenado de datos, primer paso “Datos personales” se ingresa datos solicitados con son los nombres, apellidos, sexo, código universitario que ya viene llenado por la consulta realizado como se muestra en la figura 4 y la contraseña que tiene un icono de ojo para ver la contraseña, una vez completado los datos presionamos siguiente para llenar datos de la universidad.

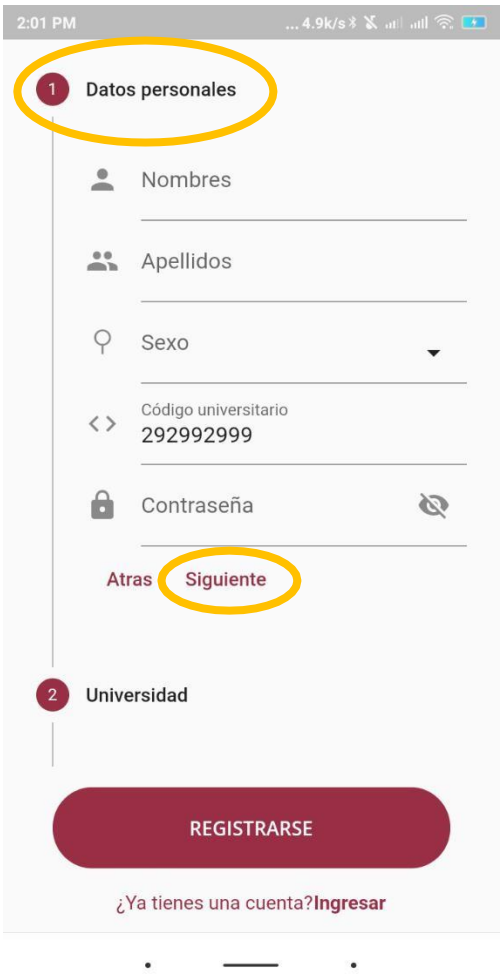


Fig. 5. Pantalla de registro de datos (Datos personales) UPeU|Crai.

Llenado de datos, segundo paso “Universidad” se ingresan los datos solicitados como son campus, facultad, escuela y ciclo. Para continuar presionar “Siguiente “o para retroceder al anterior formulario presionar “Atrás” y para retroceder a login presionar en “Ingresar”, tal como se muestra en la figura 6.

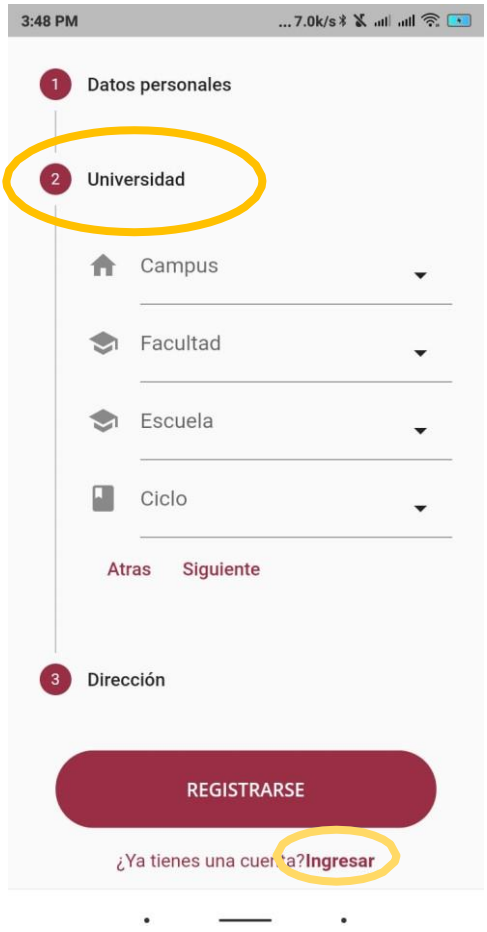


Fig. 6. Pantalla de registro de datos (Universidad) UPeU|Crai.

Llenado de datos, tercer paso “Dirección” se ingresan los datos solicitados como son el correo, celular y dirección, tal como se muestra en figura 7.

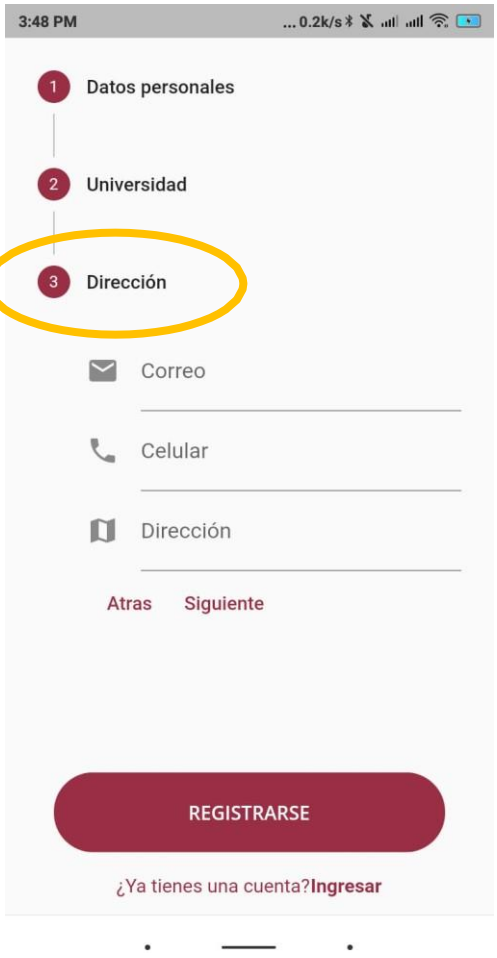


Fig. 7. Pantalla registro de datos (Dirección) UPeU|Crai.

Llenado de datos, finalmente ya llenado los datos presionamos el botón “REGISTRARSE”, eso nos redijera a la pantalla de login figura 3.

Fig. 8. Pantalla registro de datos (Botón registrarse) UPeU|Crai.

Ingresar a la pantalla de libros se debe logearse con condigo universitario y contraseña creada por el usuario tal como se muestra en la figura 9.

Fig. 9. Pantalla de login UPeU|Crai.

Las opciones que se muestran dentro del aplicativo al ingresar son las siguientes: menú con opciones como inicio, editar perfil, cambiar contraseña, mis reservaciones, libros prestados y salir. Al presionar el botón lupa esto nos redirecciona a una pantalla donde permite buscar libros en general y otra opción para filtrar libros por facultad, como se muestra en la figura 10.

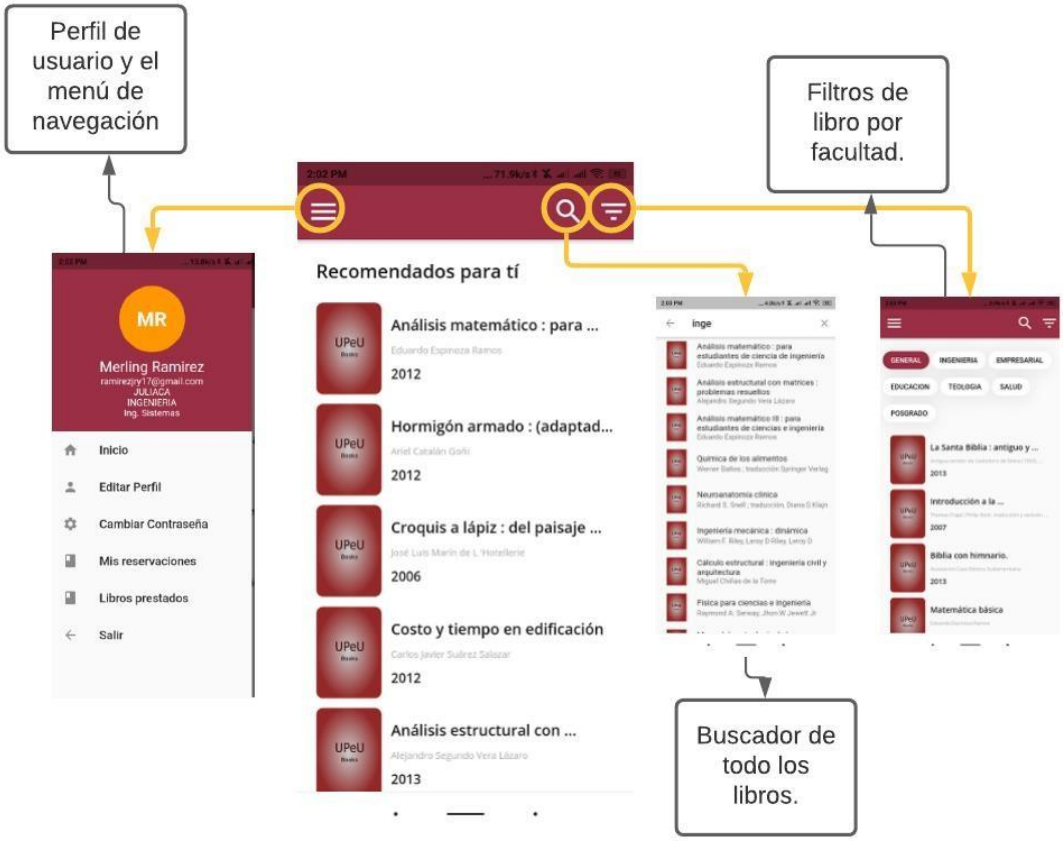


Fig. 10. Pantalla principal UPeU|Crai.

Libros recomendados: Presionar en el libro para navegar a detalles de libro, donde se puede reservar el libro presionando el botón de “Reservar el libro” y esto manda un modal con el mensaje de el título del libro que ya esta reservado, como se muestra en la figura 11.

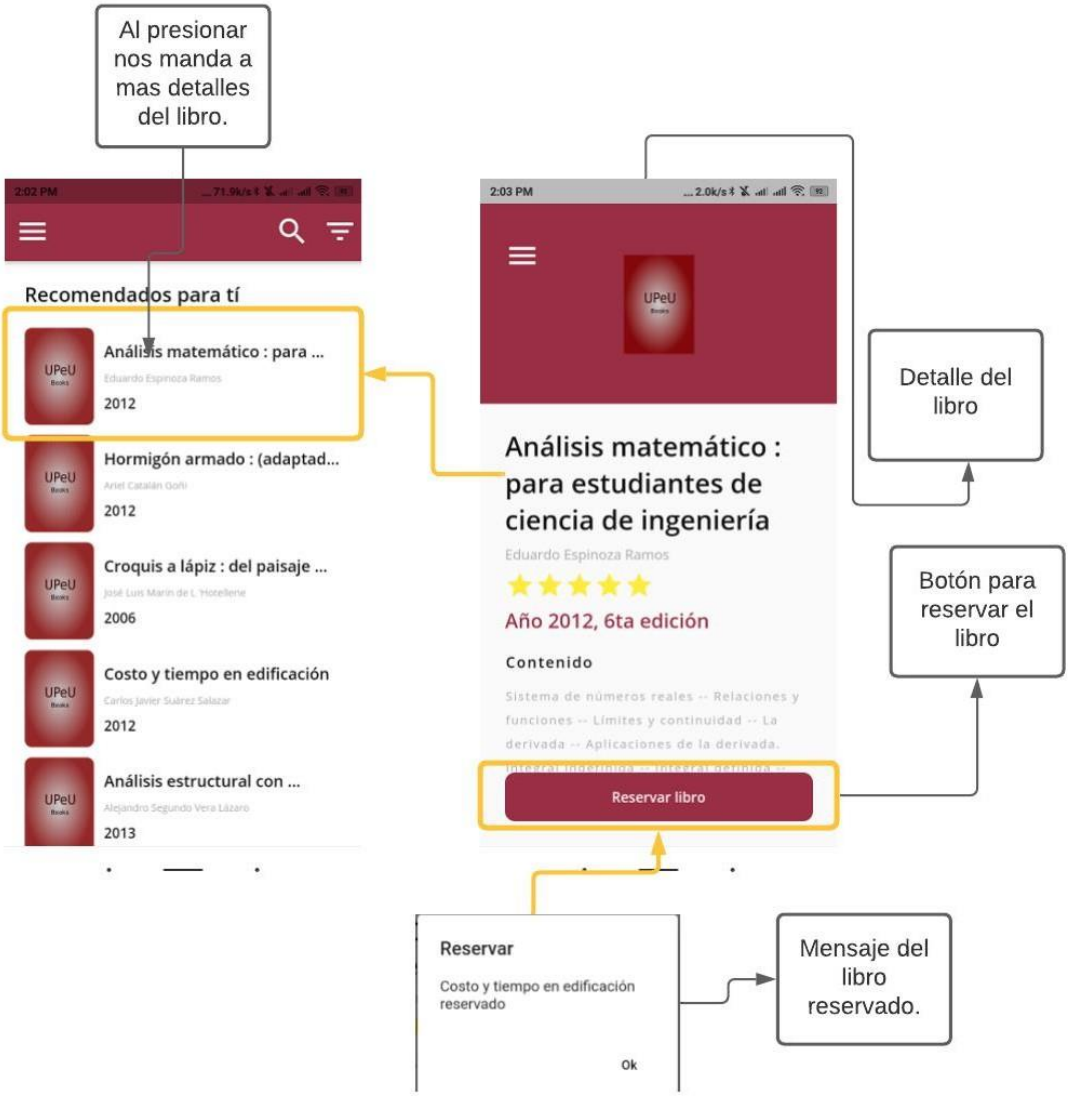


Fig. 11. Pantalla para seleccionar el libro y reservar UPeU|Crai.

Editar perfil: Es la opción para modificar los datos del ingresados en tu perfil, como se muestra en la figura 12.

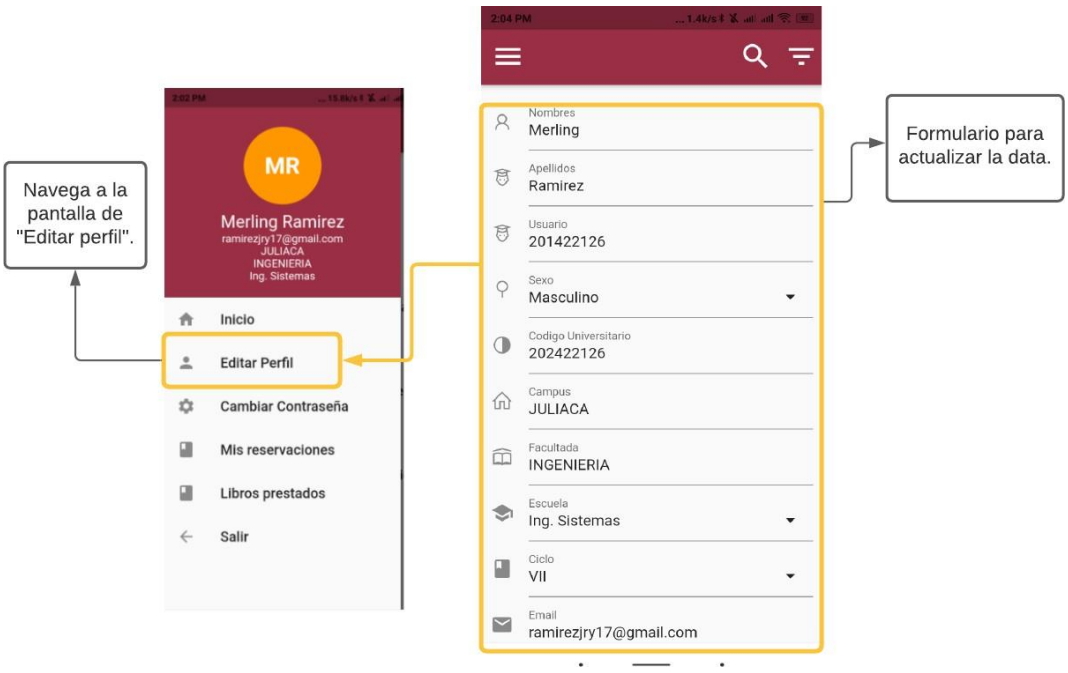


Fig. 12. Pantalla para modificar datos ingresados UPEU|Crai

Cambiar contraseña: Es la opción para modificar la contraseña, como se muestra en la figura 13.

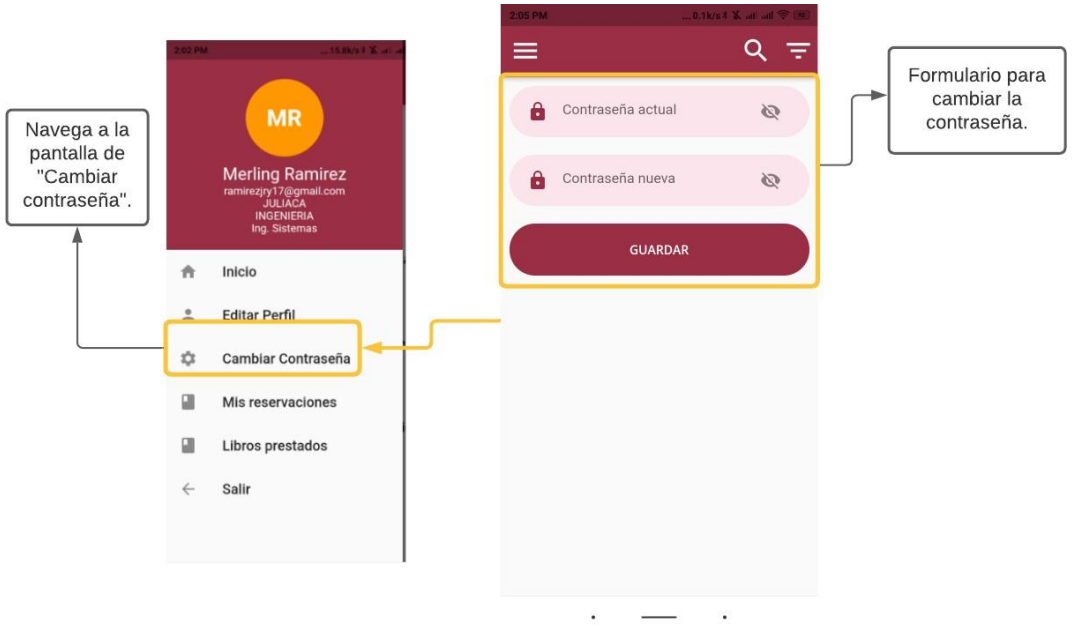


Fig. 13. Pantalla para cambiar contraseña UPeU|Crai

Mis reservaciones: Es la opción para administrar los libros reservados, como se muestra en la figura 14.

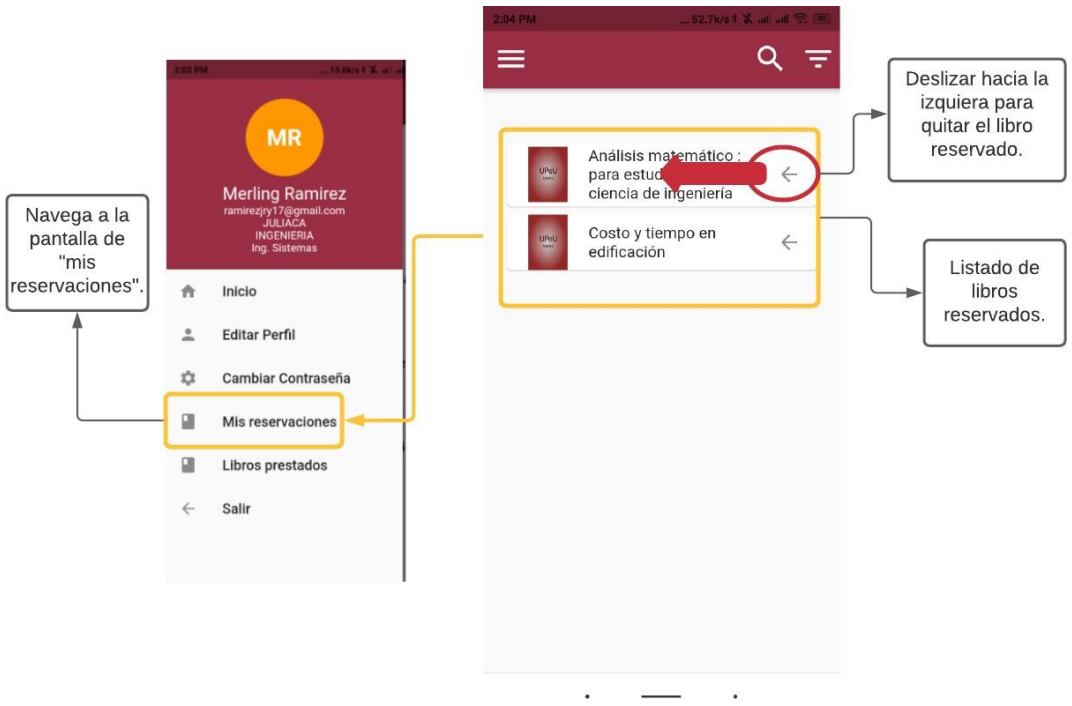


Fig. 14. Pantalla para administrar libros reservados UPeU|Crai

Libros prestados: Es la opción para ver y calificar los libros prestados por la biblioteca, como se muestra en la figura 15.

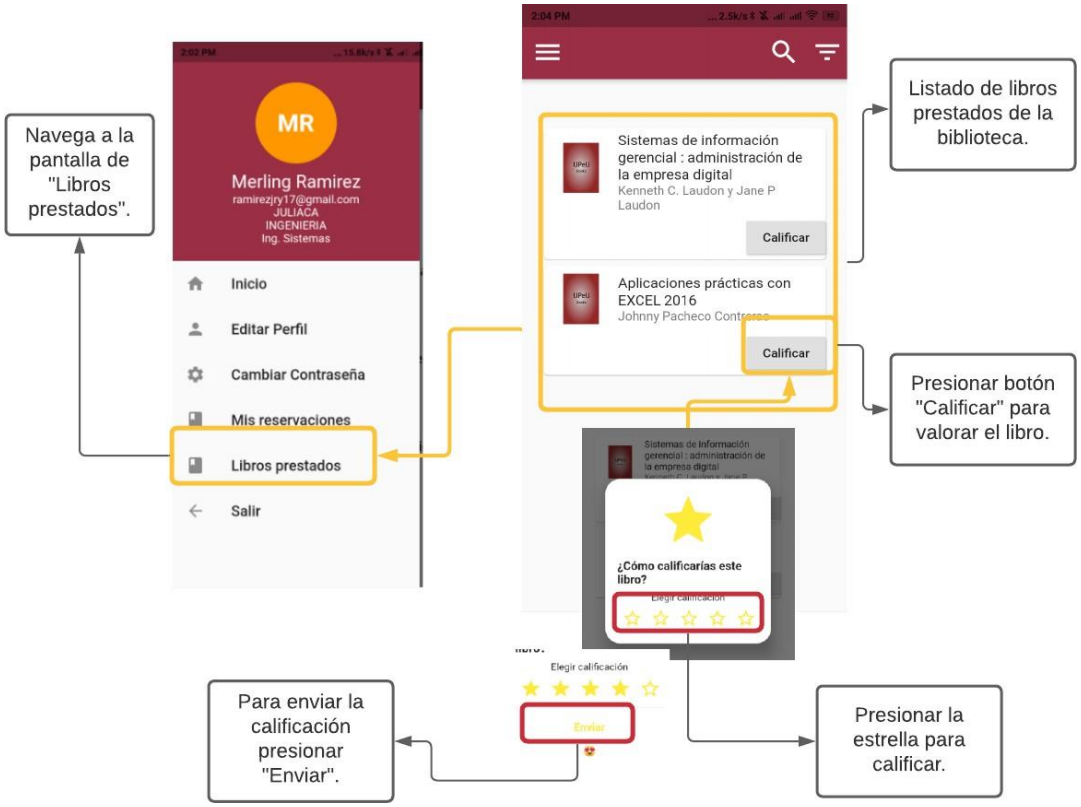


Fig. 15. Pantalla para ver y calificar libros prestados UPEU|Crai

Salir: Es la opción para cerra la sesión de la aplicación y esto manda a la pantalla login, como se muestra en la figura 16.

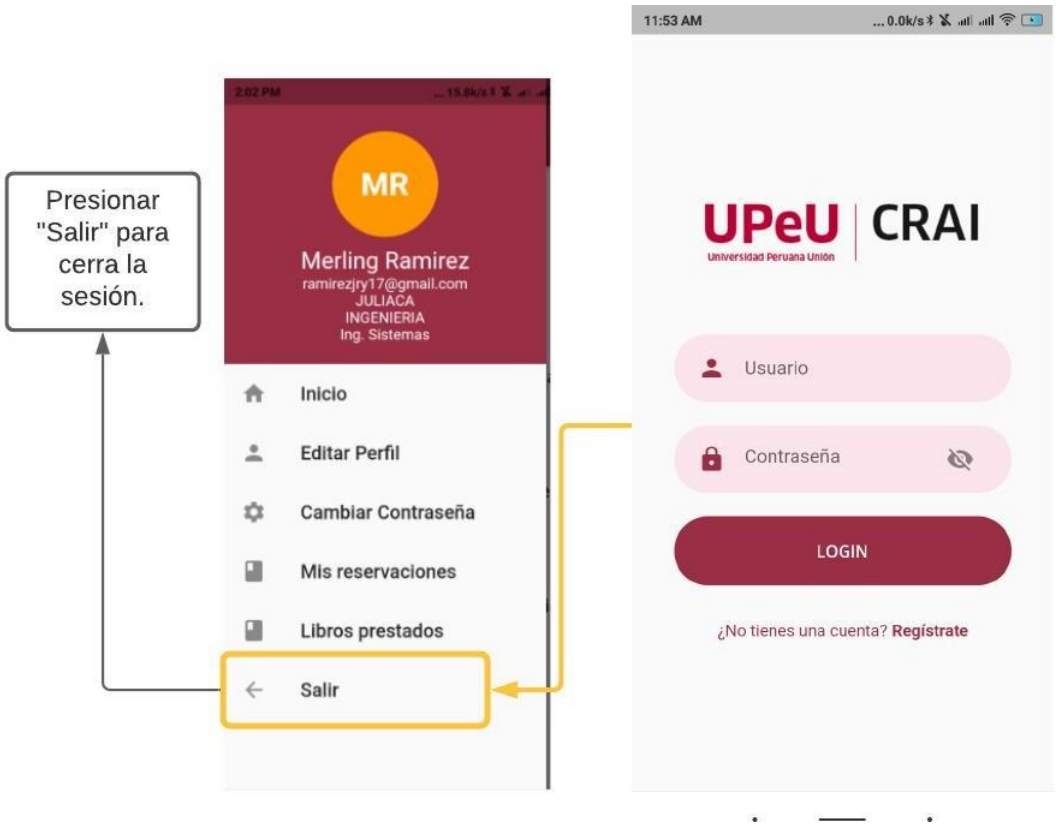


Fig. 16. Pantalla para salir de la aplicación UPeU|Crai

ANEXO D
Visualización de datos del dataset de libros y usuarios

	ItemId	signatura	...	coleccion	biblioteca
0	1	190 W27 2005	...	SALUD	BCJ
1	2	0	...	SALUD	BC
2	3	286.7325 HR 2014	...	TEOLOGIA	BC
3	4	220.5 R36 2013	...	BIBLIOTECA	BCJ
4	5	547 H22 2007	...	INGENIERIA	BC
5	6	547 C26	...	INGENIERIA	BC
6	7	W248.4 HCRIST 1988	...	BIBLIOTECA	BCJ
7	8	868.32 C45	...	EDUCACION	BC
8	9	510 M42	...	EDUCACION	BC
9	10	220.5 R36 2013	...	BIBLIOTECA	BCJ

	UserId	codigo	apellidos	...	carrera	ciclo	campus
0	1	201820142	Acarapi Muñoz	...	PSICOLOGIA	2.0	Juliaca
1	2	201611734	Adriano Yahuarcani	...	NUTRICION	0.0	Lima
2	3	201811783	Aguilar Bueno	...	PSICOLOGIA	0.0	Juliaca
3	4	201612103	Ahumada Chambi	...	ADMIN	7.0	Juliaca
4	5	201613021	Aica Llacma	...	NUTRICION	0.0	Lima
5	6	200920551	Alanoca Esteban	...	UNKNOWN	8.0	Juliaca
6	7	201810206	Alarcón Chipana	...	SISTEMAS	0.0	Juliaca
7	8	201811440	Alata Chusi	...	EDPRIM	3.0	Juliaca
8	9	201712181	Alata Mamani	...	PSICOLOGIA	5.0	Juliaca
9	10	201420536	Alemán Egusquiza	...	PSICOLOGIA	0.0	Lima

ANEXO E
Código python Estandarización de datos con la función StandardScaler()

```
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.3,random_state=42)
len(x_train)
#estandarizar características
estandarizar = StandardScaler()
x_train = estandarizar.fit_transform(x_train)
x_test = estandarizar.transform(x_test)
```

ANEXO F
Código python Precisión del modelo con similitud de Jaccard

```
print("precisión de test: ", metrics.accuracy_score(y_test, pred1))
print("precisión de modelo en Jaccard: ", jaccard_score(y_test, pred1, average="weighted"))

matrix_conf = confusion_matrix(y_test, pred1)
print(matrix_conf)

print(classification_report(y_test, pred1))
```

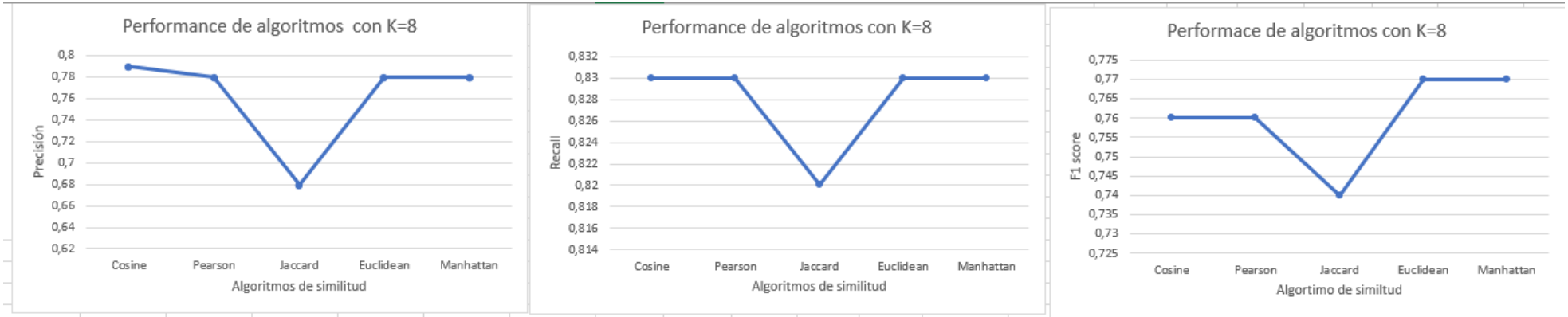
```
 MSE: 0.364903
RMSE: 0.604072
precisión de test:  0.83008356545961
precisión de modelo en Jaccard:  0.6954777410678679
```

```
[[292   3   0   0]
 [ 43   6   0   0]
 [ 10   0   0   0]
 [  5   0   0   0]]
0.7765486138745192
0.83008356545961
0.772252663588527
              precision    recall  f1-score   support

           1       0.83      0.99      0.91       295
           2       0.67      0.12      0.21        49
           3       0.00      0.00      0.00        10
           4       0.00      0.00      0.00         5

    accuracy                           0.83       359
   macro avg       0.38      0.28      0.28       359
weighted avg       0.78      0.83      0.77       359
```

ANEXO G
Performance del modelo KNN con Precisión, Recall y F1 score



ANEXO H
Parte del código Python de validación cruzada del modelo

```
# Pasos de preprocesado y del modelo en un mismo pipeline.
pipe = Pipeline([('preprocessing', preprocessor),
                 ('modelo', NearestNeighbors(n_neighbors=8, metric="cosine"))])

# Optimización de hiperparámetros
# ======================================================================
# Búsqueda de mejor hiperparámetro
param_distributions = {'modelo__n_neighbors': np.linspace(1, 100, 500, dtype=int)}

# Búsqueda random grid
grid = RandomizedSearchCV(
        estimator  = pipe,
        param_distributions = param_distributions,
        n_iter     = 20,
        scoring    = 'neg_root_mean_squared_error',
        n_jobs     = multiprocessing.cpu_count() - 1,
        cv         = RepeatedKFold(n_splits = 5, n_repeats = 3),
        refit      = True,
        verbose    = 0,
        random_state = 123,
        return_train_score = True
       )

grid.fit(X = x_train, y = y_train)

# Error de test del modelo final
# ======================================================================
modelo_final = grid.best_estimator_
predicciones = modelo_final.predict(X = x_test)
rmse_knn = mean_squared_error(
            y_true  = y_test,
            y_pred  = predicciones,
            squared = False
           )
print(f"El error (rmse) de test es: {rmse_knn}")
```

ANEXO I

Parte del código Python de entrenamiento del modelo KNN

```
#Separación de conjunto de entrenamiento y de prueba
x_train, x_test, y_train, y_test=train_test_split(x_os, y_os,test_size=0.3,random_sta
len(x_train)
#Enetrenamiento con el modelo KNN
model = KNeighborsClassifier(n_neighbors=8, metric="cosine")
model.fit(x_train, y_train)
preds = model.predict(x_test)

# Gráfico de matriz de confusión
plt.figure(figsize=(8, 6))
plt.title('Matriz de Confusión con sobremuestreo', size=16)
sns.heatmap(cm, annot=True, cmap='Blues');

#Evaluación del modelo con matriz de confusión
conf_mat = confusion_matrix(y_test, preds)
print(conf_mat)

print(precision_score(y_test, preds, average="weighted"))
print(recall_score(y_test, preds, average="weighted"))
print(f1_score(y_test, preds, average="weighted"))

print(classification_report(y_test, preds))
#Evaluación de predicciones
mse = mean_squared_error(y_test, preds)
print('MSE:% f'% mse)
rmse = math.sqrt(mse)
print('RMSE:% f'% rmse)
```

ANEXO J

Matriz de confusión y reporte de clasificación después del sobreajuste de datos

```
[[245  56   8   2]
 [ 24 262   7   4]
 [  0   0 281   0]
 [  0   0   0 286]]
0.9147526162706422
0.9140425531914894
0.912783960611837
              precision    recall  f1-score   support

         1.0       0.91      0.79      0.84       311
         2.0       0.82      0.88      0.85       297
         3.0       0.95      1.00      0.97       281
         4.0       0.98      1.00      0.99       286

    accuracy                           0.91      1175
   macro avg       0.92      0.92      0.92      1175
weighted avg       0.91      0.91      0.91      1175
```