UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Implementación y análisis de métricas para la evaluación de resultados de algoritmos de aprendizaje automático.

TRABAJO DE TITULACIÓN.

AUTOR: Jaramillo Carrión, Fulvio Fernando

DIRECTOR: Valdivieso Díaz, Priscila Marisela, Mgs

LOJA – ECUADOR

2017

2017

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

Magister.
Valdiviezo Díaz Priscila Marisela.
DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación: Implementación y análisis de métricas para la evaluación de resultados de algoritmos de Aprendizaje Automático. Realizado por Jaramillo Carrión Fulvio Fernando, ha sido orientado y revisado durante su ejecución, por cuanto se aprueba la presentación del mismo.

Loja, septiembre de 2017

f) ……..

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

Yo, Jaramillo Carrión Fulvio Fernando declaro ser autor del presente trabajo de titulación: Implementación y análisis de métricas para la evaluación de resultados de algoritmos de Aprendizaje Automático. De la Titulación Ingeniería Sistemas Informáticos, siendo Priscila Valdiviezo directora del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además, certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad.

f……

Autor. Fulvio Fernando Jaramillo Carrión

Cedula. 0705867182

DEDICATORIA

La presente Tesis está dedicada a todos quienes me apoyaron durante el trascurso de todo este proceso, sacrificando una confianza infinita y pudiéndome permitir concluir con esta etapa de mi vida.

Agradezco principalmente a mis padres Fulvio y Amparito, quien me ayudo incondicionalmente, a mi hermana Valeria por darme el ánimo necesario, y poder cumplir esta ansiada meta.

Jaramillo Carrión Fulvio Fernando

AGRADECIMIENTO

Quiero expresar mi sincero agradecimiento a la Universidad Técnica Particular de Loja, Institución que ha sabido labrarse un espacio de gloria en nuestra ciudad, así como a nivel nacional e internacional.

A los catedráticos, quienes a través de las enseñanzas impartidas supieron formarme académicamente.

Así mismo dejo constancia de nuestra especial gratitud a la titulación de Sistemas Informáticos. Y al Director de Tesis, por su apoyo relevante a la cristalización de mi aspiración académica y al desarrollo de un pensamiento investigativo.

Muchas gracias a todos, por ser parte de mi formación profesional.

INDICE DE CONTENIDOS

INDICE DE TABLAS

INDICE DE FIGURAS

INDICE DE ECUACIONES

RESUMEN

El presente trabajo de titulación presenta la investigación sobre el análisis y la implementación de métricas para la evaluación de resultados obtenidos con técnicas de aprendizaje automático. Para esto se analizaron métricas como: Precisión, Recall, F-Measure, Novedad, Confiabilidad y Mae. Posteriormente se procedió a implementar las métricas Precisión, Recall, F-Measure y Mae en el lenguaje de programación Python, utilizando el Framework de Django.

Para el análisis de los resultados se utilizó un sistema recomendador basado en Filtrado Colaborativo usando un algoritmo de k-vecinos cercanos.
Finalmente se desarrolló una página Web, que permita ingresar los resultados de algún sistema recomendador y poder visualizar los valores de las métricas en graficas estadísticas.

PALABRAS CLAVES: Filtrado Colaborativo, Collaborative Topic Regression, Precisión, Recall, F-Measure, Mae, Ptyhon.

ABSTRACT

This work order degree presents research on the analysis and implementation of Metrics for the evaluation of results obtained with automatic learning techniques. For this we will analyze Metrics such as: Precision, Recall, F-Measure, Novelty, Reliability and Mae. Subsequently the metrics Precision, Recall, F-Measure and Mae were implemented in the Python programming language, using the Django Framework.

For the analysis of the results we used a referential system based on Collaborative Filtering using an algorithm of k-nearest neighbor.
Finally, a Web page was developed, which allows to enter the results of some system of recommendation and to be able to visualize the values of the Metrics in statistical graphs.

KEY WORDS: Collaborative Filtering, Collaborative Topic Regression, Precision, Recall, F-Measure, Mae, Ptyhon.

INTRODUCCIÓN

El presente trabajo tiene como objetivo analizar e implementar métricas para la evaluación de resultados obtenidos con técnicas de aprendizaje automático. Para esto se analizarán métricas como: Precisión, Recall, F-Measure, Novedad, Confiabilidad y Mae.

El aprendizaje automático ha tenido una gran evolución en los últimos años, es posible distinguir algunos estudios previos, como son las técnicas de modelado neuronal y de decisión, a través del uso de esta técnica se creó sistemas llamados redes neuronales, debido a la poca tecnología en esos tiempos estos sistemas se quedaron en investigaciones teóricas o sistemas para un solo propósito específico.

En los años sesenta comenzó una nueva técnica de aprendizaje orientado a conceptos simbólicos el cual utilizaba estructuras lógicas o grafos en vez de métodos numéricos o estadísticos esto permitía un mayor grado de conocimiento por parte de los sistemas.

El tercer paradigma menciona el uso de técnicas orientados a los sistemas basados en conocimientos que permitía enlazar datos para poder crear sistemas más complejos.

El aprendizaje automático representa una evolución importante en los campos de la informática, análisis de datos ingeniería de software y en la inteligencia artificial, Según (Nilsson, 2005) el aprendizaje automático tiene como objetivo desarrollar técnicas que le permitan al computador aprender y recomendar información relevante al usuario.

En este campo se encuentran los Sistemas Recomendadores que utilizan las opiniones de los usuarios para conocer e identificar de manera eficiente cuales contenidos tienen mayor relevancia, Según (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013) asegura que los usuarios generar un mayor grado de confianza en los Sistemas Recomendadores cuando el usuario está de acuerdo con las sugerencias que le brinda el sistema.

Para los Sistemas Recomendadores un aspecto muy importante es el análisis de métricas ya que permiten medir el rendimiento del sistema a través de los datos suministrados, para esta evaluación se analizará la métrica como:

1. **Precisión.** representar la probabilidad de que un ítem recuperado en un buscador sea relevante para el usuario.
2. **Recall.** permite evaluar los ítems más apropiados que se le puede mostrar al usuario del total de los ítems que contiene el sistema.
3. **F-Measure.** permite establecer una medida armónica entre la Precisión y el Recall.
4. **Novedad.** indica la diferencia que existe entre los ítems recomendados por el sistema y los conocidos por el usuario, con esto se puede medir que tan novedosos son los ítems que se ofrece a los usuarios por parte del sistema recomendador.
5. **Confiabilidad.** esta métrica permite considerar que tan confiable son los ítems recomendados por el sistema para el usuario.

6. **Mae.** Permite medir el rendimiento de modelos recomendadores, calculando la magnitud promedio de los errores en un conjunto de predicciones realizadas por el sistema recomendador.

Con la implementación de las métricas Precisión, Recall, F-Measure y Mae. Se conocerá la calidad de las técnicas, métodos y algoritmos aplicados para las predicciones de las recomendaciones que brinda el sistema, facilitando la comparación de varias soluciones que se pueden aplicar para los problemas que surjan en la evaluación de estos resultados.

La importancia de esta investigación es de gran utilidad ya que permitirá contar con una implementación de las métricas que se las pueda utilizar en cualquier trabajo o investigación de experimentaciones con algoritmos de aprendizaje automático que se desarrolle en el futuro y así poder validar los resultados obtenidos.

1. PROBLEMÁTICA

1.1 Planteamiento del Problema

El uso de algoritmos de aprendizaje automático nos ayuda a enseñar a las máquinas a completar una tarea determinada o hacer predicciones sobre alguna recomendación de un tema en específico.

Para conocer si los algoritmos de aprendizaje automático aplicados al proceso de recomendación de ítems son aplicables o no, es muy importante su evaluación de resultados. Esta evaluación determinará si es o no aplicable el sistema a partir de los resultados obtenidos.

Para poder minimizar el problema de determinar si un sistema es apto o no se evaluará los resultados aplicando métricas como: Precisión, Recall, F-Measure y Mae.

Esquema del problema de evaluación de resultados.

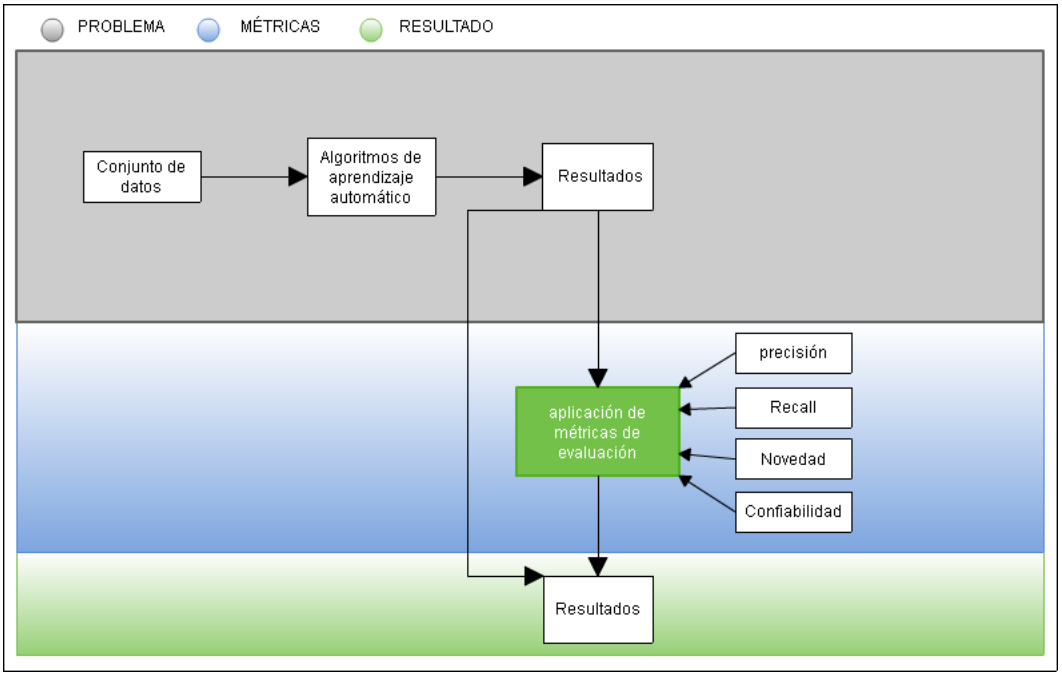


Figura 1. *Problemática de evaluación de algoritmos de Aprendizaje Automático*
Fuente. *EL autor.*
Elaborado por. *El autor.*

1.2 Objetivos

Para el siguiente trabajo de titulación se determinan los siguientes objetivos que se desean alcanzar con la investigación.

1.2.1 Objetivo General.

Analizar e implementar métricas para la evaluación de resultados obtenidos con técnicas de Aprendizaje Automático

1.2.2 Objetivos Específicos.

- ✓ Investigar las métricas utilizadas para la evaluación de resultados de algoritmos de Aprendizaje Automáticos utilizados en la recomendación de ítems.
- ✓ Comparar métricas de Precisión, Recall, Novedad y Confiabilidad para la evaluación de resultados.
- ✓ Implementar dos métricas de evaluación de resultados en un lenguaje de programación.
- ✓ Experimentar las métricas evaluadas con un conjunto de datos reales.
- ✓ Documentar los resultados obtenidos a partir de la experimentación de los datos evaluados.

1.3 Metodología

La estrategia para el presente trabajo empieza por la revisión de conceptos teóricos para el conocimiento previo que el estudiante posea en temas relacionados al análisis de métricas de evaluación de resultados obtenidos con algoritmos de Aprendizaje Automático aplicados al proceso de recomendación de ítems.

En el proceso de análisis de métricas se deben revisar sus características, en qué tipo de sistemas se las puede aplicar, conocer sus ventajas y desventajas y realizar una comparación minuciosa con el fin de elegir la métrica más apropiada para su implementación.

En la fase de implementación se utilizará paquetes de Python que es un lenguaje de programación de código abierto orientado a objetos muy versátil y rápido o R que al igual que Python es un lenguaje de programación de software libre enfocado al análisis estadístico, también se podrá utilizar otros lenguajes de programación en los cuales sea mucho más factible y fácil la implementación de estas Métricas, en esta etapa se elegirá uno lenguajes para la implementación de dos Métricas, serán seleccionadas de acuerdo a la comparación que se realizó anteriormente.

Para el proceso de experimentación se tomará un conjunto de datos disponibles en la Web usados para este tipo de experimentaciones, que luego de la aplicación de algunos algoritmos utilizado en otra tesis, se procederá a validar estos resultados con las métricas antes mencionadas, obteniendo gráficas estadísticas para una mayor comprensión de estos resultados.
Finalmente se obtendrán las conclusiones del trabajo realizado para una futura investigación.

1.4 Estructura de Trabajo de Titulación

Para el desarrollo del trabajo de titulación se va a seguir la siguiente estructura:
El capítulo I, se presenta un marco teórico, en el que se realiza una descripción de las definiciones esenciales del Aprendizaje Automático y su relación con los Sistemas Recomendadores y un análisis de métricas de evaluación de resultados obtenidas con algoritmos de Aprendizaje Automático.

El capítulo II se realizará una comparación de las métricas de Precisión, Recall, Novedad y Confiabilidad para elegir dos que serán implementadas posteriormente.

En el capítulo III se ejecutará la implementación de las métricas seleccionadas usando paquetes de Python que es un lenguaje de programación de código abierto orientado a objetos muy versátil y rápido o R que al igual que Python es un lenguaje de programación de software libre enfocado al análisis estadístico, también se podrá utilizar otros lenguajes de programación en los cuales sea mucho más factible y fácil la implementación de estas Métricas.

En el capítulo IV se realizará una experimentación con un conjunto de datos reales.

En el capítulo V se analizarán los resultados obtenidos y se presentarán conclusiones y recomendación.

CAPITULO I:
MARCO TEÓRICO

El presente capítulo se abarcará conceptos de aprendizaje Automático, algoritmos que usa el aprendizaje automático, aplicaciones que utiliza, Sistemas de recomendación, tipos de sistemas recomendadores, métricas de evaluación de resultados de aprendizaje automático como: Precisión, Recall, F-Measure, Novedad, Confiabilidad y Mae.

1.1 Aprendizaje Automático

El Aprendizaje Automático representa una evolución muy grande en los últimos años en diferentes campos de la informática como: Análisis de Datos, Ingeniería de Software e Inteligencia Artificial. Según (Nilsson, 2005) el Aprendizaje Automático permite desarrollar técnicas para que la computadora puedan aprender por sí misma a través de las experiencias que obtiene realizando tareas específicas, esforzándose en aprender cosas del pasado para realizar de forma eficiente en el futuro. (Schapire, 2013) refiere que el Aprendizaje Automático trata de crear algoritmos capaces de reconocer patrones a partir de información suministrada por el usuario. El uso de algoritmos de Aprendizaje Automático es más confiable de los de la Programación tradicional, ya que estos algoritmos son capaces de examinar grandes cantidades de información en poco tiempo y de forma eficiente.

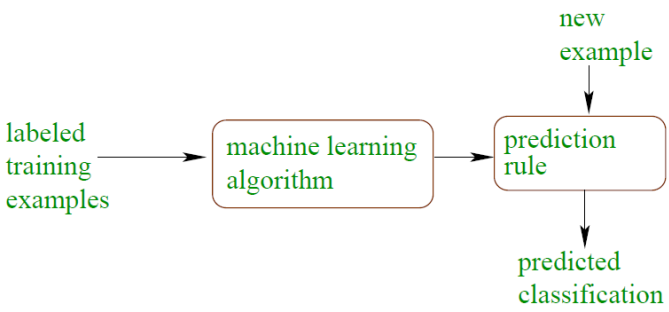


Figura 2. Problema común de Aprendizaje Automático
Fuente. Recuperado de abril del 2013, de *https://goo.gl/irRFgk*
Elaborado por. (Schapire, 2013)

En la Figura 2 se puede observar que se inicia reuniendo una gran cantidad de información de entrenamiento (training examples) para luego ser procesado por unos algoritmos de Aprendizaje Automático que a su vez produce una regla de predicción y clasificar la información obteniendo los datos más relevantes.

1.2 Algoritmos de Aprendizaje Automático

Para (Smola & Vishwanathan, 2014) Los algoritmos se agrupan en una taxonomía en función a sus salidas, algunos de ellos son:

- ✓ Aprendizaje Automático.
- ✓ Aprendizaje no supervisado.
- ✓ Aprendizaje Semisupervisado.
- ✓ Aprendizaje por refuerzo.
- ✓ Aprendizaje Multi-Tarea.

1.3 Aplicaciones que utiliza el Aprendizaje Automático

Según (Rodriguez, 2013) una gran cantidad de aplicaciones usan agentes basados en Aprendizaje Automático dentro de numerosas ramas de la industria y la ciencia como son:

- ✓ Procesamiento de Lenguaje Natural
- ✓ Sistemas de Recuperación de Información.
- ✓ Diagnósticos Médicos.

A continuación, se explicará cada una de ellas con sus respectivas aplicaciones desarrolladas.

1.3.1 Procesamiento de Lenguaje Natural.

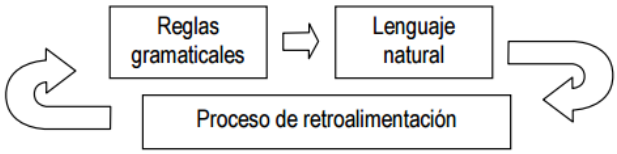


Figura 3. Lenguaje natural.
Fuente. Recuperado del 2009, de *https://goo.gl/6go6e2*
Elaborado por. Augusto et al. (2009)

Según (Augusto et al., 2009) el Lenguaje Natural permite el procesamiento del lenguaje humano a través de reglas gramaticales para poder ser interpretadas por los ordenadores. Esto contribuye al análisis sintáctico y morfológico de textos, facilitando la extracción, clasificación, agrupamiento de palabras claves de información que se encuentran en los textos o documentos a través de técnicas de Aprendizaje Automático. Dentro de estas aplicaciones se encuentran los modelos de análisis de sentimientos, usando técnicas de Aprendizaje Automático que permiten reconocer la voz si se está hablando bien o mal sobre un tema en específico.

El uso del procesamiento del lenguaje natural entre humano-máquina representa ventajas y desventajas como:

Ventajas:

- ✓ El locutor no tiene que esforzarse para aprender el medio de comunicación

Desventajas:

- ✓ El computador tiene una limitada comprensión del lenguaje, y se establece un lenguaje nueve se le debe enseñar al computador a poderlo reconocer a través de algoritmos de procesamiento de lenguaje natural.

1.3.2 Aplicaciones.

- ✓ Traducciones automáticas

Permite tomar un texto escrito en un lenguaje cualquiera y traducirlo a otro, manteniendo el mismo significado del texto original.

- ✓ Análisis de sentimientos

Identifica y extrae la información de un texto donde se observa emociones de todo tipo.

- ✓ Análisis de imágenes

Para reconocer escritura manuscrita, identificar direcciones y remitentes de un envío postal, facturas o recibos u otro tipo de documentos legales.

1.3.3 Sistemas de recuperación de información.

De acuerdo con (Ordóñez & González, 2004) La cantidad de información almacenada tiene la necesidad a crear técnicas de Aprendizaje Automático, con el fin de recuperar la información de forma rápida y precisa.

Estos sistemas se encuentran dentro de PLN (procesamiento de lenguaje natural) utilizan técnicas de Aprendizaje Automático para mejorar los buscadores de internet, y crear búsquedas más dinámicas para guardar un Ranking personalizados de las experiencias de los usuarios con respecto a sus búsquedas realizadas.

1.3.3.1 Aplicaciones.

Como lo hace notar (Kuusisto, 2015) existe una gran cantidad de aplicaciones que recuperan información para luego ser visualizada por los usuarios, entre algunas aplicaciones están:

- ✓ **Google** utiliza técnicas de Aprendizaje Automático en su motor de búsqueda para predecir páginas que son más relevantes para los usuarios.
- ✓ **Amazon** usa técnicas de Aprendizaje Automático para decir que productos son más demandados en el mercado y así poder sugerir a sus usuarios en base a cuáles son más propensos a comprar.

Para descubrir las respuestas que el usuario quiere obtener se debe determinar el tipo de respuesta que se quiere dar, generar una búsqueda relevante y calificar los resultados obtenidos.

Para esto existen 3 pasos: entrenamiento, fragmentación y el tipo de respuesta.

1.3.4 Diagnósticos médicos.

Hoy en día el Aprendizaje Automático es muy utilizado en el campo de la medicina ya que permite analizar una gran cantidad de datos y múltiples variables.

Como plantea (Zwinderman, 2013) el Aprendizaje Automático permite asistir a los médicos al diagnóstico del historial clínico de los pacientes utilizando técnicas de Aprendizaje Automático, esto ayuda a predecir enfermedades que pueda tener el paciente.

1.3.4.1 Aplicación.

Como expresa (Kononenko, 2001) una de las aplicaciones más utilizas en los diagnósticos médicos y que trabaja con técnicas de Aprendizaje Automático es:

- ✓ Diagnóstico en enfermedades del corazón

Utilizando técnicas de Aprendizaje Automático se puede interpretar los resultados del paciente y así aumentar la precisión diagnóstica.

1.4 Sistemas de recomendación

Con base en (Guevara & Rossi, 2014) Los Sistemas Recomendadores permite administrar sugerencias a los usuarios, esto facilita a los usuarios a la toma de decisiones de un tema en específico, para poder llevar a cabo estas recomendaciones el sistema debe analizar la información histórica de los usuarios.

Estos sistemas mantienen una iteración entre los usuarios el producto y las preferencias que tienen los usuarios hacia los productos que ofrece el sistema.

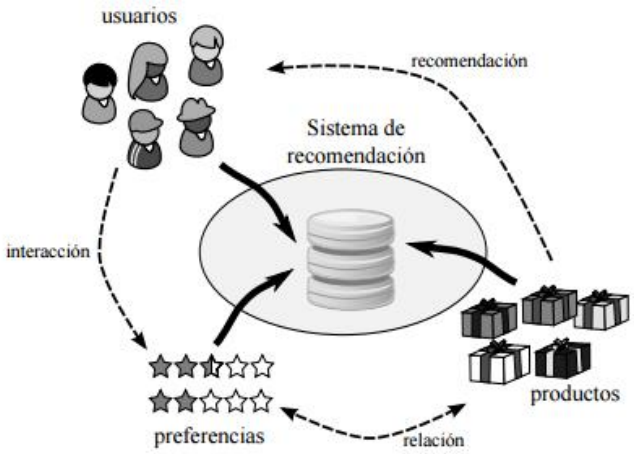


Figura 4. Esquema de un Sistema recomendador.
Fuente. Recuperado en 2013, de *https://goo.gl/ypWhua*
Elaborado por. (Formoso, 2013)

Según (Tamayo, 2012) en los Sistemas Recomendadores existe un número determinado de ítems o productos para un número de usuarios, los usuarios asignan un voto a estos ítems,

con esta información el sistema tiene que sugerir ítems a los usuarios. Para ellos debe haber ciertos parámetros como:

- ✓ Base de datos y usuarios

Se encuentra toda la información que contiene el Sistema antes del proceso de recomendación.

- ✓ Datos de entrada

Información que agrega el usuario considerando cuáles ítems son aceptables

- ✓ Algoritmo de recomendación.

Combinación de los datos previos y los datos realizados por parte del usuario con el fin de generar recomendación a través del desarrollo de un algoritmo.

Figura 5. Proceso de recomendación
Fuente. Recuperado en 2012, de *https://goo.gl/ig6qhV*
Elaborado por. (Tamayo, 2012)

En la figura 5 se muestra la interacción de existen en la información previa y los datos de entrada que administra el usuario, para que los algoritmos de recomendación inicien el proceso de dar sugerencias al usuario.

1.4.1 Tareas de un sistema recomendador.

Como menciona (Fernandez Iglesias, 2014) desde el punto de vista del usuario existe una gran cantidad de tareas o funciones que cumplen los Sistemas Recomendadores, estas funciones cumplen con el objetivo de que el usuario pueda decir si las recomendaciones son acordes a sus gustos.

- ✓ **Anotación en contexto**

El sistema predice las preferencias que tiene un usuario hacia un producto determinado basándose en el contexto, es decir, el usuario es capaz de decidir si el producto es aceptable o no.

- ✓ **Encontrar Ítems buenos**

El sistema sugiere un conjunto de productos al usuario de acuerdo con sus gustos, estos productos son recomendados al usuario de acuerdo con una puntuación para así determinar la satisfacción del usuario.

- ✓ **Encontrar todos los usuarios buenos**

El sistema recupera todos los productos de interés por parte del usuario, aunque estos productos no sean completamente del agrado del usuario

- ✓ **Recomendación en secuencia**

El sistema recomienda los productos en secuencia, de tal forma que el usuario puede tener una idea clara de que producto es. A diferencia de realizar una recomendación individual.

- ✓ **Artículos de interés**

Es usuario no puede buscar un producto en un determinado sitio Web para adquirirlo, sino solo busca artículos de interés con el simple fin de conocerlos.

- ✓ **Encontrar recomendadores creíbles**

Esta tarea ofrece al usuario la posibilidad de probar el comportamiento del propio sistema para aumentar así su fiabilidad.

- ✓ **Mejorar el perfil de usuario**

El usuario de alguna manera debe brindar al sistema recomendador los gustos que tiene hacía un producto recomendado, esta tarea se encarga de obtener esa información que brinda el usuario hacia el sistema, almacenando las puntuaciones que dio sobre un producto en específico, mejorando el perfil de usuario.

- ✓ **Expresar opiniones del usuario**

Esta tarea brinda al usuario la facilidad de expresar y aportar las sugerencias al sistema recomendador.

- ✓ **Ayudar a otros usuarios**

Esta tarea permite al usuario dar a conocer a otros tipos de usuario su información, de esta forma el usuario puede ayudar a la toma de decisiones de otras personas.

- ✓ **Influir en los demás usuarios**

Según el autor (Fernandez Iglesias, 2014) existen usuarios que influye en la toma de decisiones de otras personas e incluso ocultan información de las recomendaciones que ha realizado el sistema, esta tarea prevé este tipo de situaciones.

1.4.2 Proceso de Recomendación.

En este proceso lo primero que se debe obtener son los productos que se recomendaran a los usuarios, de acuerdo con (Fernandez Iglesias, 2014) estos productos pueden ser

obtenidos por el propio sistema o proporcionados por el usuario, para mostrar estos productos se debe tener la información acerca de producto en un formato amigable para el usuario fácil de entender.

Las predicciones de los productos que realiza el sistema recomendador se la hace de acuerdo con la información que brinda el usuario, es decir a la puntuación que realiza a un determinado producto, con esta información el usuario puede recomendar productos similares.

Para poder obtener las preferencias que brinda el usuario a un producto determinado se la puede obtener implícita o explícitamente.

Las preferencias implícitas son fáciles de obtener ya que el sistema debe ser capaz de almacenar (clics, tiempo transcurrido en una determinada página, links abiertos, entre otros).

Las preferencias explícitas son en cambio las valoraciones que da el usuario a un determinado producto. Estas preferencias explicitas se clasifican en:

- ✓ Escalas de puntuaciones. Estas escalas pueden ser numéricas o cualitativas en el que se mide el nivel de aceptación de un producto. Por ejemplo, al recomendar un producto x se da un marco de aceptación de 1 al 5 en el que el usuario debe elegir una puntuación y determinar qué beneficio considera ese producto para él.

- ✓ Puntuaciones binarias. El usuario puede elegir dos valores únicos para un producto determinado. es decir, puede escoger en una puntuación positiva o negativa.

- ✓ Puntuaciones unarias. El usuario puede decir que le pareció relevante ese producto, pero no puede elegir un rango de relevancia.

1.4.3 Aplicaciones que utilizan sistemas recomendadores.

Como lo hace notar (Adriana & Pérez, 2013) Existen una gran cantidad de aplicaciones que recomiendan ítems o productos a los usuarios, entre algunas aplicaciones están:

- ✓ **Tapesty**

Sistema experimental de correo diseñado para soportar filtros basados en contenidos y por filtrado colaborativo

- ✓ **Nerflix**

Sistema recomendador de películas

- ✓ **Amazon.com**

Utiliza algoritmos recomendadores para personalizar la tienda en línea y poder dar sugerencias a sus usuarios de productos para su venta.

- ✓ **MovieLens**

Sistema recomendador de películas gratuito, este sistema utiliza filtrado colaborativo

1.4.4 Tipos de Sistemas Recomendadores.

Según (Adriana & Pérez, 2013) Existe sistemas que ayudan a dar recomendaciones más precisas para tratar de reducir el menor error posible, algunas de ellas son.

1.4.4.1 Sistema de Filtrado Colaborativo.

Como afirma (F.O. Isinkaye, Y.O. Folajimi, 2015) el Filtrado Colaborativo permite que el sistema genere las recomendaciones utilizando la información solo para determinar qué puntuación es asignada por los usuarios a un determinado producto, esta técnica no toma en cuenta el contenido de los productos, solo necesita poderlos identificar de forma única.

Las recomendaciones se establecen únicamente por las opiniones de los usuarios que han tenido una puntuación más alta, esta técnica tiene una gran ventaja ya que los resultados obtenidos serán muy exactos ya que son los usuarios los que establecen las opiniones.

Algoritmos:

Existen algoritmos que utilizan técnicas de Filtrado Colaborativo los cuales realizan un conjunto de pasos ordenados para resolver un problema en específico, los algoritmos dependen de la información inicial que tengan para resolver el problema. De acuerdo con (Formoso, 2013) algunas de los algoritmos empleadas son:

- ✓ Basados en vecinos

El algoritmo busca otros usuarios o productos con preferencias similares, para generar recomendación acertadas. Para este algoritmo existen 3 fases principales que son:

Calcular la similitud entre los diferentes usuarios, seleccionar un vecindario según la similitud calculada entre los usuarios y calcular las predicciones o recomendación a partir de las puntuaciones que realizaron los usuarios.

- ✓ Basados en usuarios

Selecciona un grupo de usuarios con preferencias similares, comparando sus perfiles para obtener las puntuaciones que han dado a un producto en específico, con esta información el algoritmo es capaz de recomendar productos al grupo de usuario en el cual las puntuaciones de los productos fueron relativamente iguales.

- ✓ Basados en productos

Cumple la misma función del algoritmo basado en usuarios, determinar un grupo de productos similares comparando las puntuaciones que han realizado los usuarios, para ello utiliza funciones para calcular la similitud entre los productos.

- ✓ Similarity fusión

Combina los dos algoritmos, en el cual compara las similitudes de los usuarios y los productos, con el fin de obtener mejores resultados.

- ✓ Basados en regresión

Cumple con la tarea de predecir una puntuación de un producto, partiendo de la puntuación de otro.

1.4.4.2 Sistemas Basados en Contenidos.

Estos sistemas toman en cuenta el contenido que tiene el producto, es decir las características como (precio, calificación, categoría, entre otros) con esta información el sistema puede realizar recomendaciones a los usuarios según las preferencias de los mismos.

Citando a (Vera & Mamani, 2015) estos Sistemas utilizan algoritmos ”ítems a ítems” que son generados mediante reglas de correlación donde se asocian los atributos de los perfiles de los usuarios con los atributos de los ítems a recomendar.

Los sistemas Basados en Contenidos buscan información relacionada a los perfiles de los usuarios, y las características de los ítems o productos en la **figura 5** se observa algunos datos que se recogen para realizar las recomendaciones posteriormente.

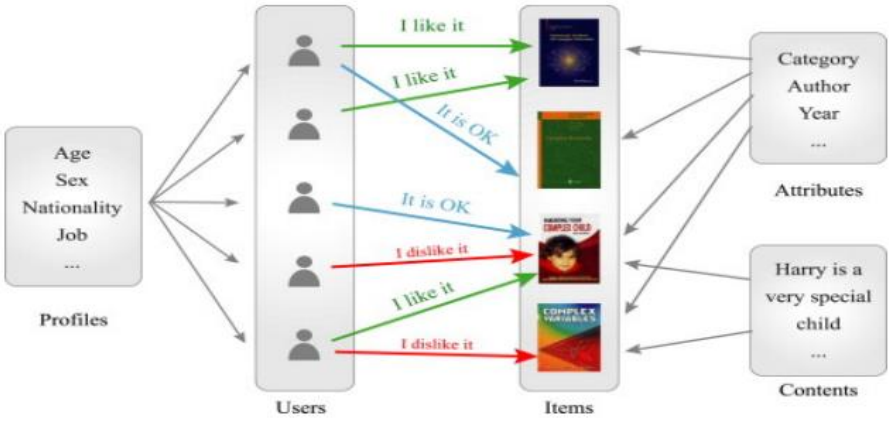


Figura 6. Contenido clave para un sistema basado en contenido.
Fuente. Recuperado en 2012, de *https://goo.gl/ig6qhV*
Elaborado por. (Tamayo, 2012)

Algoritmos:

Desde el punto de vista de (Tamayo, 2012) los algoritmos más importantes en Sistemas Basados en Contenidos son:

- ✓ Algoritmo de agrupamiento

Este algoritmo realiza agrupamiento de datos con el fin de conocer la similaridad que existe entre los ítems o productos para poder dar recomendaciones a los usuarios de manera más exacta, el agrupamiento es diferente a la clasificación ya que no existe la etiqueta que define qué clase pertenece a un determinado objeto, por lo que el aprendizaje es no supervisado.

- ✓ Algoritmos K-Medidas

Estos algoritmos son muy utilizados ya que son simples y rápidos de usar, permiten que los ítems se agrupan a un grupo determinado.

- ✓ Algoritmo Aspect Model

Permite asignar a los ítems o productos a un grupo determinado, con la diferencia que el ítem puede pertenecer a más de un grupo con diferentes grados de pertenencia.

1.4.4.3 Sistemas Híbridos.

Según (Tamayo, 2012) los Sistemas de Recomendación Híbridos son aquellos que combinan algunas técnicas de recomendación para incrementar el rendimiento del sistemas, algunos Sistemas Híbridos combinan los contenidos de los ítems o productos y las estrategias de colaboración, para ello se utilizan los perfiles de los usuarios para hacer mejor las recomendaciones.

Como destaca (Vera & Mamani, 2015) existen varios métodos para la combinación de sistemas como:

- ✓ Método ponderado

En este método se combina la puntuación que realizaron los usuarios a un ítem, con el fin de producir una única recomendación.

- ✓ Método Mixto

Presenta la recomendación de varios sistemas al usuario en el mismo al mismo tiempo.

- ✓ Método de cascada

Las recomendaciones son vistas por los usuarios en base a recomendación dadas por otras personas

1.5 Métricas de evaluación para Sistemas Recomendadores

Las métricas de evaluación para un Sistema Recomendador es un aspecto crucial para el análisis de los resultados obtenidos por estos Sistemas. Como lo hace notar (Bobadilla et al., 2013)existen una gran cantidad de métricas de evaluación las cuales se encuentran bien definidas.

Para el desarrollo de este trabajo mencionaremos 7 métricas que son: Precisión, Recall, Mae, Mesure, RMSE, Novedad y Confiabilidad.

Según (González, 2013) estas métricas se encuentran en el grupo de evaluaciones experimentales ya que permiten evaluar el rendimiento del sistema comparando los resultados obtenidos sin necesidad de la intervención del usuario para poder realizar esta evaluación se obtienen datos cuantitativos de los resultados obtenidos.

Como señala (Bobadilla et al., 2013) las métricas de evaluación tiene establecidas formulaciones estándares, lo que permite el avance progresivo de algoritmos ya que siempre estarán basadas en las mismas fórmulas previamente establecidas.

1.5.1 Precisión.

Según (Gunawardana, 2009) la métrica de Precisión ayuda a la evaluación de resultados de un Sistema Recomendador de ítems, midiendo la capacidad del sistema, esto lo realiza determinando un conjunto de ítems que sean más importantes y relevantes para el usuario que utiliza el sistema, ordenando por su importancia.

La Precisión la define (Cano, Eugenia, Enriquez, & Alexandra, 2015) como la fracción de ítems relevantes del total del conjunto de ítems devueltos por el sistema, es decir, la representación de la probabilidad de que un ítems recuperado en un buscador sea relevante:

1.5.1.1 *Formula general.*

Ecuación 1.Precisión

$$Presicion = \frac{|I \in \{Items\ Relevantes\}|}{|I|}$$

Fuente. Recuperado del 2012, de *https://goo.gl/2fyFyR*
Elaborado por. Parra (2012)

Siendo **I** el conjunto de ítems recomendados por el sistema e $I \in \{ítems\ relevantes\}$ el conjunto de ítems que son considerados relevantes en los juicios de relevancia.

Como lo hace notar (Parra & Sahebi, 2013) esta métrica proporciona al usuario una lista de recomendaciones en las cual puede evaluar los ítems relevantes y no relevantes.
En la lista puede haber un alto número de elementos recomendadores esto depende del tamaño del conjunto de datos, por lo que no es factible que el usuario sea capaz de verificar y evaluar cada uno de ellos, por lo que esta métrica solo considera a los ítems superiores que se denomina recomendación Top-N.

1.5.1.2 *Ejemplo.*

Dado un conjunto de elementos recomendados como un conjunto **S** y los elementos relevantes como el conjunto **R.**

Figura 7. Datos de un sistema recomendador Métrica Precisión.
Fuente. Recuperado del 2012, de *https://goo.gl/2fyFyR*
Elaborado por. Parra (2012)

$$Presicion = \frac{5}{10} = 0.5$$

En esta figura podemos observar que existen 5 elementos relevantes (color verde) con lo que divido por el número de elementos recomendados que es 10 nos da un total de 0.5
Esta fórmula sirve para evaluar el sistema en el contexto de un solo usuario, para poder usar esta métrica en todo un conjunto de usuarios se utiliza la **Precisión Media Promedio (MAP)** esta métrica se la obtiene calculando la medida sobre la Precisión media de la lista de recomendaciones de cada usuario. La fórmula es la siguiente:

Ecuación 2. Precisión Media Promedio

$$MAP = \sum_{n=1}^{N} \frac{AveP(n)}{N}$$

Fuente. Recuperado del 2013 de, *https://goo.gl/RfYcKB*
Elaborado por. (Parra & Sahebi, 2013)

En esta ecuación **AveP** significa la Precisión media de cada usuario **n**, es decir el promedio de los valores de Precisión obtenidos para el conjunto de recomendaciones de parte del Top-N después de recuperar cada recomendación pendiente.

1.5.2 Recall.

Como afirma (Parra & Sahebi, 2013) el Recall, o cobertura, se entiende como la fracción de los ítems relevantes recomendados por el sistema del total de ítems relevantes.

1.5.2.1 Formula General.

Ecuación 3. Recall

$$Recall = \frac{|I \in \{items\ relevantes\}|}{|\{items\ relevantes\}|}$$

Fuente. Recuperado del 2013 de, *https://goo.gl/RfYcKB*
Elaborado por. (Parra & Sahebi, 2013)

Siendo **I** el conjunto de ítems recomendados por el sistema e $I \in \{items\ relevantes\}$ el conjunto de ítems que son considerados relevantes en los juicios de relevancia.

1.5.2.2 Ejemplo.

Dado un conjunto de elementos recomendados como un conjunto **S** y los elementos relevantes como el conjunto **R**

Figura 8. Datos Sistema Recomendador Métrica Recall
Fuente. Recuperado del 2012, de *https://goo.gl/2fyFyR*
Elaborado por. Parra (2012)

$$Recall = \frac{5}{20} = 0.25$$

En esta ecuación podemos observar que existen 5 elementos relevantes con lo que divide por el número de elementos relevantes que es 20 nos da un total de 0.25

1.5.3 F-Mesure.

Como menciona (Mcnee & Konstan, 2006) esta métrica obtiene los elementos que se recomiendan con el sistema en los que el artículo es relevante para el usuario. Combinando la Precisión y Recall como se muestra en la siguiente ecuación.

Ecuación 4. F-Mesure

$$F1 = \frac{2 * Presicion * Recall}{(Presicion + Recall)}$$

Fuente. Recuperado del 2006 de, *https://goo.gl/2xXD7q*
Elaborado por. (Mcnee & Konstan, 2006)

En la evaluación de un sistema, cuando utilizamos Recall, la Precisión disminuye. Esto se puede obviar utilizado F-Measure la cual usa una medida armónica entre ambas métricas. Esta métrica se puede interpretar como el promedio ponderado entre la Precisión y Recall.

1.5.4 Novedad.

Según Castells et al.(2011) esta Métrica desempeñan un papel crucial en los Sistemas Recomendadores (RS) donde el beneficio que aportan las recomendaciones a los usuarios está relacionado a la noción del descubrimiento de nuevos ítems.

Para poder aplicar la métrica de Novedad a resultados obtenidos por Sistemas Recomendadores se debe tener experiencias previas, estas experiencias son la popularidad de los ítems recomendados, los elementos que el usuario conoce, entre otros.

La métrica de Novedad ayuda a identificar los ítems más novedosos para el usuario, como afirma Zhang (2013) esta métrica tiene dos sentidos “new-original” que permite identificar ítems nuevos que no han sido vistos por el usuario, y “refreshing-pleasantly” que brinda un contenido diferente a lo ya antes visto por el usuario. La Novedad contiene tres características principales que son:

1. Desconocido

Que el ítem que el sistema recomienda sea desconocido para el usuario.

2. Satisfactorio

Que el ítem sea satisfactorio y que aporte beneficios al usuario.

3. Desemejanza

Que el ítem sea diferente y no aparecido a lo ya antes visto por el usuario.

Según Castells et al. (2011) se consideran dos modelos para el uso de la métrica de Novedad los cuales están orientados a los ítems.

- ✓ Orientado a popularidad.
- ✓ Basado en la distancia.

1.5.4.1 Modelos de Novedad de ítems.

La Novedad basada en popularidad

Se la puede definir como la relación de una porción específica de eventos observados, dentro del conjunto globales de elementos que contiene el sistema. Esta métrica es expresada de la siguiente manera:

Usuario aleatorio

Ecuación 5. Novedad basada en usuarios aleatorios

$$\text{Novelty (i)} = \text{I (i)} = -\log_2 p(i)$$

Fuente. Recuperado del 2011 de, *https://goo.gl/PxGPxN*
Elaborado por. (Castells et al., 2011)

Donde **p(i)** representa la probabilidad de que **i** es observada y **I(i)** se denomina como auto-información, para este modelo se debe interpretar una variable aleatoria, para elegir un evento de elección de un usuario, es decir **i** es obtenida por un usuario aleatorio, con esto se refleja un factor de popularidad del ítem. En el que **novelty(i)** corresponde al logaritmo inverso de la popularidad. Con este esquema se obtiene la Novedad genérica en la medida en que es el mismo para todos los usuarios. Se puede usar una variante de Novedad relativa, esto se lo usa tomando la ecuación 1 y cambiando **p(i|u)** esto equivale a restringir nuestras observaciones al usuario objeto.

Ecuación 6.Novedad Genérica

$$\text{Novelty (i|u)} = -\log_2 p(i|u)$$

Fuente. Recuperado del 2011 de, *https://goo.gl/PxGPxN*
Elaborado por. (Castells et al., 2011)

Usuario Especifico

Para este modelo de popularidad alternativa se puede considerar la probabilidad **p(k|i)** que un elemento es conocido o está familiarizado, en lugar de que este usuario sea escogido aleatoriamente se lo puede definir usando la Novedad genérica y relativa.

Ecuación 7. Novedad Usuario Específico

$$\text{Novelty (i)} = -\log_2 p(k|i)$$

Fuente. Recuperado del 2011 de**,** *https://goo.gl/PxGPxN*
Elaborado por. (Castells et al., 2011)

La Novedad basada en la distancia

Como afirma Vargas & Castells (Vargas & Castells, 2011) la Novedad basada en distancia se define por una función de distancia entre el ítem y el contexto de experiencia, este contexto se lo representa como un conjunto de elementos es decir podemos formularlo como la distancia esperada o mínima entre el elemento y el conjunto así:

Ecuación 8. Novedad basada en distancia.

$$\text{Novelty (i|S)} = \frac{min}{i \in S} d(i,j)$$

Fuente. Recuperado del 2011 de, *https://goo.gl/PxGPxN*
Elaborado por. (Castells et al., 2011)

Donde **d** es la medida de distancia, esta distancia se la puede definir d (i,j)= 1-sim(i,j). si tomamos el conjunto de elementos en el que el usuario a interactuado es decir los elementos de su perfil obtendremos la ecuación de Novedad relativa que se visualiza anteriormente, esta ecuación sirve para **p(i|S)** solo cuando **S** sea un perfil de usuario

1.5.4.2 Modelos de navegación.

Como lo hace notar Castells et al.(Castells et al., 2011) los modelos que definen a la Novedad dependen de los tipos de datos de observación, las variables aleatorias y cualquier otra restricción, en esta sección se detallarán tres categorías principales:

- ✓ Elección

  Como un elemento se usa, se escoge, se selecciona, se accede, entre otros. Esto ayuda a tener una frecuencia asociada a este evento.

- ✓ Descubrimiento

  Cuando un ítem no ha sido visto antes por el usuario.

- ✓ Relevancia

  Se puede relacionar con la preferencia que tiene el usuario a un ítem determinado, es decir cuan útil es el producto que le brinda el sistema.

1.5.4.3 Formula General.

Un esquema general de la métrica de Novedad para la evaluación de los Sistemas Recomendadores es diferente por varios autores según Izadi (2014) brinda una ecuación simplificada que se muestra a continuación en la que está basada en la popularidad :

Ecuación 9. Novedad

$$N = \log 2(\frac{m}{d_i})$$

Fuente. Recuperado de 2014 de, *https://goo.gl/EQGDRA*
Elaborado por.(Izadi, 2014)

Donde **m** es el número de usuarios y $\boldsymbol{d_i}$ es el grado de un elemento es decir cuántas veces está bien clasificado por los usuarios.

1.5.5 Confiabilidad.

Como afirma Bobadilla et al., (2013) la métrica de Confiabilidad en los Sistemas Recomendadores evalúa la fiabilidad de las predicciones de las recomendaciones que realiza el sistema. El autor proporciona el ejemplo de que cuando el Sistema Recomendador recomienda a un usuario el artículo con una predicción de 4.5 en la escala de 5, se espera que el usuario éste completamente satisfecho. Pero para que este artículo sea confiable para el usuario se debe realizar pruebas a varios usuarios, esto quiere decir que es mucho más confiable que la predicción de 4.5 la haya echo 200 usuarios a que solo dos usuarios.
Hernando et al (como se citó en Bobadilla et al., (2013)) piensa que el uso de la métrica de Confiabilidad se limita a los RS basados en el algoritmo kNN, y puede ser usando en cualquier sistema recomendadores que esté basado en algoritmos de filtrado colaborativo.

1.5.5.1 Factores Principales.

Esta métrica se basa en dos factores principales que se muestran a continuación:

1. **Factor $\boldsymbol{S_{u,i}}$**

Este factor permite medir la fiabilidad de una predicción.
El factor $\boldsymbol{k_{u,i}}$ se lo considera más confiable cuando el número de k vecinos al momento de calificar un ítem i es mayor.
Para encontrar la información de los vecinos $\boldsymbol{k_{u,i}}$ se define con las siguientes fórmulas.

Ecuación 10. Confiabilidad Factor $S_{u,i}$

$$S_{u,i} = \sum_{v \in K_{u,i}} sim(u, v)$$

Fuente. Recuperado del 2013 de, *https://goo.gl/vyMEAs*
Elaborado por. (Bobadilla et al., 2013)

En esta fórmula el factor $\boldsymbol{S_{u,i}}$ toma en cuenta los vecinos que han calificado un ítem **i** (es decir $\boldsymbol{k_{u,i}}$)

2. Factor $\boldsymbol{V_{u,i}}$

Este factor mide el grado de desacuerdo entre los vecinos utilizados calificando el ítem **i** este es un factor negativo.

Dada la predicción $\boldsymbol{p_{u,i}}$ vamos a considerar la varianza $\boldsymbol{V_{u,i}}$ de las calificaciones que los vecinos del usuario **u** han hecho sobre un ítem **i**

Ecuación 11. Confiabilidad Factor $V_{u,i}$

$$V_{u,i=}\frac{\sum_{v\in K_{u,i}} sim(u,v).\left(r_{x,i}-\bar{r}_x-p_{u,i}+\bar{r}_u\right)^2}{\sum_{x\in K_{u,i}} sim(u,v)}$$

Fuente. Recuperado del 2013 de, *https://goo.gl/vyM_EAs*
Elaborado por. (Bobadilla et al., 2013)

El autor Hernando et al.(2013) considera una muestra de ejemplo para entender este factor, la cual contiene dos predicciones $\boldsymbol{p_{u,i1}}$ y $\boldsymbol{p_{u,i2}}$

1. En la predicción $\boldsymbol{p_{u,i1}}$ existen 10 vecinos del usuario **u** que han calificado el artículo, en los cuales todos ellos han dado un valor de 5. Es decir $\boldsymbol{p_{u,i1}=5}$ y $\boldsymbol{V_{u,i}=0}$
2. En la predicción $\boldsymbol{p_{u,i2}}$ existen 10 vecinos del usuario **u** que han calificado El ítem **i2**, la diferencia es que 5 de ellos calificaron los ítems **i2** con un valor de 5 y los otros cinco calificaron los ítems con un valor de 1. Es decir $\boldsymbol{p_{u,i1}=3}$ y $\boldsymbol{V_{u,i}=4}$

Se puede ver que es más fiable en la predicción 1 ya que los vecinos del usuario **u** estuvieron más de acuerdo con las calificaciones de la predicción 2.

1.5.5.2 Fórmula General.

Una vez establecido los valores de $\boldsymbol{S_{u,i}}$ y $\boldsymbol{V_{u,i}}$ se define la fórmula de Confiabilidad:

Ecuación 12. Confiabilidad

$$R_{u,i=}\left(f_{s(}S_{u,i)}*f_{v(}V_{u,i)}{}^{f_{s(}S_{u,i)}}\right)^{\frac{1}{1f_{s(}S_{u,i)}}}$$

Fuente Recuperado del 2013 de, *https://goo.gl/HM9r9r*.
Elaborado por. (Hernando et al., 2013)

En esta fórmula se encuentra integrado los dos factores antes mencionados, vale recalcar que esta Métrica es usada es Sistema Recomendadores basados en algoritmos de filtrado colaborativo.

1.5.6 Mean Absolute Error (MAE).

Según (Chai, Draxler, & Prediction, 2014) el error absoluto medio es utilizado para medir el rendimiento de modelos recomendadores, calculando la magnitud promedio de los errores en un conjunto de predicciones realizadas por el sistema recomendador.

Esta métrica de evaluación toma el error producido en las predicciones comparándolo con los resultados esperados para evaluar cómo se resume en la siguiente ecuación.

Ecuación 13. MAE

$$MAE = \frac{| \in \{calificacion - prediccion|}{|\{numero\ de\ casos\}|}$$

Fuente. Recuperado del 2013 de, *https://goo.gl/RfYcKB*
Elaborado por. (Parra & Sahebi, 2013)

Como afirma (Mortensen, 2007) Esta métrica realiza una comparación con las calificaciones estimadas y las calificaciones reales que son recomendadas al usuario. MAE proporciona el mismo peso a errores a todos los elementos que se recomiendan. Existen casos en que hay elementos más importantes que otros por lo que no es la mejor opción utilizar esta métrica.

Sin embargo, la evidencia sugiere que otras métricas muestran mejorías cuando el valor MAE disminuye, esto demuestra que el MAE es muy importante en el caso de evaluar un sistema. Además, su cálculo subyacente es simple y tiene propiedades estudiadas.

CAPITULO II:

COMPARACIÓN DE MÉTRICAS DE EVALUACIÓN

Los Sistemas Recomendadores están desarrollados bajo un conjunto de herramientas de Software y técnicas para brindar sugerencias acerca de los contenidos que el usuario quiere encontrar, tienen como objetivo facilitar al usuario a la toma de decisión sobre un producto a comprar. Existe una innumerable cantidad de aplicaciones que utilizar Sistemas Recomendadores como son Amazon, Netflix, entre otros.

En este capítulo se compara las 6 métricas analizadas en el CAPITULO 1 (Precisión, Recall, Novedad, Confiabilidad, Measure, Mae) para saber cuál de estas métricas tiene mayores beneficios para su implementación.

Resumen del capítulo:

- ✓ Organización de las métricas según al grupo que pertenezcan.
- ✓ Métricas de exactitud
- ✓ Métricas orientadas al descubrimiento
- ✓ Tabla comparativa

2.1 Organización de las Métricas

Según Hijikata (2014) las métricas están ubicadas en distintos grupos, los cuales se muestran a continuación:

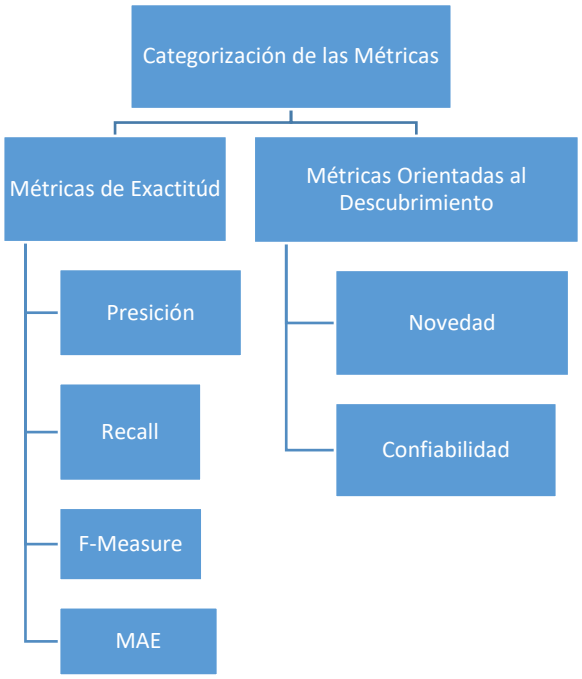


Figura 9: Organización de las Métricas de Evaluación
Fuente. Recuperado del 2014, de *https://goo.gl/sNhTis*
Elaborado por: Hijikata (2014)

2.2 Métricas de Exactitud

Son aquellas métricas que brindan un resultado preciso de un determinado conjunto de datos ayudando a filtrar los ítems o productos relevantes, de esta forma el usuario puede obtener los ítems que está buscando.

De la misma forma Linyuan (2012) da a conocer que existen otras métricas ubicadas en este grupo las cuales se las conoce como: métricas de Precisión de calificación. Las más conocidas son:

- ✓ **Mean Absolute Error** (MAE) ésta métrica calcula una cantidad estadística para ser usada en el pronóstico o predicciones de los resultados
- ✓ **Root Mean Squared Error** (RMSE) el error cuadrático medio que calcula la pérdida tomando la media de todas las diferencias entre el objeto y la predicción.

Como menciona Herlocker, Terveen, Ekstrand, Riedl, & Konstan, (2011) estas métricas son las más utilizadas y comunes en la recuperación de información, examinado la capacidad de un sistema de recuperación en el cual se identifican los recursos relevantes, midiendo la capacidad para encontrar todos los elementos pertinentes y evitar elementos irrelevantes de un conjunto determinado de datos.

A continuación, se describe la similitud que tienen las métricas de Precisión y Recall.

Tabla 1: Matriz de Confusión.

	Relevante	**Irrelevante**
Recuperados	TP	FP
No Recuperados	FN	TN

Fuente. Recuperado del 2011 de, *https://goo.gl/x6CdvF*
Elaborado por. Herlocker et al. (2011)

Estas dos métricas comparten términos similares los cuales están visibles en la **Tabla 1**, las métricas evalúan los resultados que obtuvo el sistema recomendador, conociendo cuáles son los productos relevantes y los no relevantes ya que esta tarea se encarga el sistema recomendador, sin embargo podría existir artículos que no estén disponibles en estos dos conceptos(relevante o no relevante) es decir son productos que el usuario no le gusto o no sabía que existía, estas observaciones pueden ser muy útil para conocer cuán confiable es el sistema recomendador.

Estas dos métricas utilizan la **Tabla 1** para obtener los datos que le servirán para calcular sus respectivos resultados

Precisión:

Tabla 2: Matriz de Confusión de Precisión.

	Relevante	**Irrelevante**
Recuperados	TP	FP
No Recuperados	FN	TN

Fuente. Recuperado del 2011 de *https://goo.gl/x6CdvF*
Elaborado por. Herlocker et al. (2011)

Recall:

Tabla 3: Matriz de Confusión de Recall

	Relevante	**Irrelevante**
Recuperados	TP	FP
No Recuperados	FN	TN

Fuente. Recuperado del 2011 de *https://goo.gl/x6CdvF*
Elaborado por. Herlocker et al. (2011)

La Precisión y Recall están inversamente relacionadas y dependen de **N** es decir del número de recomendaciones que ha realizado el sistema.

Estas métricas se las implementa dependiendo del tipo de información que se tiene, por ejemplo, si un usuario busca recomendaciones sobre películas, lo que necesita es encontrar una buena película (alta precisión). En cambio, si un usuario que sea abogado éste buscando procedentes legales, necesita encontrar todos los casos relevantes (alto Recall).

Para poder encontrar una relación común entre estas dos Métricas, según Herlocker et al., (2011) se la puede simplificar en una sola métrica que se la llama **F1** que es muy utilizada para evaluar los Sistemas Recomendadores.

Ecuación 14. F-Mesure

$$F1 = \frac{2 * Presicion * Recall}{(Presicion + Recall)}$$

Fuente. Recuperado del 2006 de, https://goo.gl/2xXD7q
Elaborado por. (Mcnee & Konstan, 2006)

En esta métrica se combina la Precisión y Recall dándole a las dos métricas el mismo peso.

2.2.1 Problemas comunes entre Precisión y Recall.

Como menciona Addis (2010) estas dos métricas son muy utilizadas en los Sistemas Recomendadores para recuperar información, estos sistemas deben recuperar tantos

documentos relevantes como pueda (es decir alto Recall) y se deberían recuperar pocos datos no relevantes (es decir alta Precisión).

Estos dos objetivos son difíciles de cumplir ya que las dos métricas son contradictorias con lo que al aplicar Recall se pierde la Precisión y viceversa. Esto quiere decir que no es aconsejable utilizar las dos métricas para evaluar un mismo conjunto de datos, si es necesario aplicar estas métricas el autor Herlocker et al., (2011) menciona que se las puede unir en unas sola fórmula la cual es **F1** que se menciona anteriormente.

Como expresa Cano et al., (2015) el mayor problema para aplicar estas métricas es el ruido documental y Silencio Documental.

2.2.2 Ruido Documental.

Se conoce como ruido documental a los documentos que se clasifican como relevantes, cuando en realidad no lo son, para este problema se debe filtrar los resultados.

- ✓ Empleando términos más precisos.
- ✓ Sustituyendo palabras generales por unas más específicas.

2.2.3 Silencio Documental.

Son los documentos relevantes que no se ha podido clasificar por diferentes circunstancias, por ejemplo, cuando el usuario no está enterado de que existe ese ítem, para estos casos se puede revisar.

- ✓ Si la estrategia de búsqueda de información es la correcta.
- ✓ Sustituir términos muy específicos por otros más generales.
- ✓ Si la fuente consultada es la más adecuada.

2.3 Métricas Orientadas al Descubrimiento

Son aquellas métricas que ayudan a descubrir ítems nuevos, es decir permitirle al usuario poder encontrar información que no haya visto anteriormente.

Como menciona Castells et al. (2011) estas métricas buscan mejorar las ventas de un producto que es descrito por un Sistema Recomendador brindando al usuario productos nuevos para que tenga conocimiento de ellos y que puedan ser de su agrado, dentro de este grupo también se encuentra la métrica de **Diversidad** que se aplica a un conjunto de elementos en el cual se encontrar relacionados con las diferencias del uno con el otro. Es decir, la Diversidad trata de encontrar las diferencias que tiene un producto con otro, y así poder dar recomendaciones precisas a los usuarios.Las métricas de descubrimiento se basan en mostrar artículos nuevos o que no hayan sido vistos por el usuario. Como afirma Sandoval (2012) la Métrica de Novedad debe aportar documentos diferente al usuario, y evitar la redundancia de productos.

El autor Hijikata, (2014) se refiere que la Métrica de Novedad tiene algunas distribuciones que se muestran en la Tabla 4.

Tabla 4: Tipos de Métricas de Novedad

Métrica	Objetivo	Otra Información	Inventor
Ratio de Descubrimiento	Lista	Conocida	Hijikata, IUI 2009
Precisión de la Novedad	Lista	Conocida	Hijikata, IUI 2009
Novedad del Artículo	Artículo	Ontología	Zhang, RecSys 2008
Novedad Temporal	Lista	Ninguna	Lathia, SIGIR 2010
Novedad de Cola Larga	Lista	Ninguna	Celma, RecSys 2008
Novedad Generalizada	Lista	Ninguna/Ontología	Vargas, RecSys 2011

Fuente. Recuperado del 2014, de *https://goo.gl/sNhTis*
Elaborado por. Hijikata (2014)

En esta tabla se puede observar algunos tipos de métricas que son parte de la métrica principal que es Novedad.

- ✓ **Ratio de Descubrimiento**

Calcula cuántos ítems desconocidos se recomiendan en la lista.

- ✓ **Precisión de la Novedad**

Calcula Si el elemento es desconocido para el usuario y si al usuario se le hará interesante el artículo.

- ✓ **Novedad del Articulo**

Mide la novedad del elemento recomendado.

- ✓ **Novedad Temporal**

Calcula si el sistema recomienda artículos nuevos con relación al artículo anterior

- ✓ **Novedad de Cola Larga**

Divide el ítem en tres categorías según la popularidad: Cabeza, Media y Cola.

- ✓ **Novedad Generalizada**

Mide la popularidad y la Novedad basado en la distancia.

En el grupo de las métricas de Descubrimiento también se encurta la Métrica de Confiabilidad que trata de brindar una predicción lo más confiable posible, de esta forma evitar que la recomendación sea equivocada.

A continuación, se muestra una tabla comparativa de las 4 métricas analizadas de acuerdo con diferentes criterios de comparación

2.4 Tabla comparativa

Tabla 5: Comparación de Métricas de Evaluación

Criterios Comparativos	Precisión	Recall	F-Measure	Mae	Novedad	Confiabilidad
Ruido Documental	**X**	**X**	**X**	**X**		
Silencio Documental	**X**	**X**	**X**	**X**		
Recuperar Información	**X**	**X**	**X**	**X**	**X**	
Evitar la Redundancia					**X**	
Predicciones					**X**	**X**

Fuente, El autor
Elaborado por: El autor.

En la Tabla 6 se muestra que las métricas de exactitud (Precisión y Recall, F-Measure, Mae) son utilizadas para la recuperación de Información que el usuario necesita ver, en cambio las métricas orientadas al descubrimiento (Novedad y Confiabilidad) ayudan a predecir ítems nuevos y confiables para el usuario.

CAPITULO III:
IMPLEMETACIÓN DE MÉTRICAS

En este capítulo presenta la implementación de métricas de evaluación de resultados con técnicas de aprendizaje automático, utilizando un conjunto de datos de MovieLens, a continuación, se detallan las principales actividades.

Resumen del capítulo:

- ✓ Análisis y selección de Métricas, dando a conocer la importancia y por qué se eligen estas métricas.
- ✓ Lenguajes de Programación utilizados.
- ✓ Utilización de un sistema recomendador usando Filtrado Colaborativo (CF) Y Collaborative Topic Regression (CTR).
- ✓ Uso de métodos de análisis de datos con: K-Folders y *TrainTestSplit*.
- ✓ Implementación de una página Web.
- ✓ Implementación de las métricas de Precisión y Recall, F-Measure y Mae en Python.

3.1 Análisis y selección de las Métricas

Las métricas de exactitud según (F.O. Isinkaye, Y.O. Folajimi, 2015) son las más populares usadas actualmente. Estas métricas ayudan a los usuarios a seleccionar elementos que sean de muy alta calidad disponibles en un conjunto determinado de datos, verifican el procedimiento de predicción con operaciones binarias en los que distinguen los elementos buenos y los que no son buenos para un usuario en específico.

En este capítulo se ha seleccionado un conjunto de métricas para la evaluación de resultados con algoritmos de aprendizaje automático entre ella tenemos:

- ✓ Recall
- ✓ Precisión
- ✓ F-Measure
- ✓ MAE

3.2 Lenguajes de Programación utilizados en la implementación

En esta sección se mencionarán los diferentes lenguajes de programación y los entornos de desarrollo que se utilizaron en la implementación de las métricas de evaluación de resultados

3.2.1 Programación en Java.

Según (Oracle, 2016) **Java** es un lenguaje de programación orientado a objetos, introducida por primera vez en 1995 por Sun Microsystems.

Como cita el autor (Rodríguez & Sayay, 2016) este lenguaje permite integrar características únicas que facilitar el desarrollo de aplicaciones. Algunas de ellas son:

- ✓ Escribir software en una plataforma y poder ejecutarla en otra diferente

- ✓ Creación de aplicaciones Web
- ✓ Confinación de aplicaciones o servicio
- ✓ Ejecución de programas de forma independiente
- ✓ Compatibilidad en todos los SO

3.2.1.1 Netbeans(IDE).

Es un entorno integrado de desarrollo, el cual permite programar en distintos lenguajes, es ideal para trabajar con el lenguaje de programación Java, está escrito en java y es de código abierto.

NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en control de versiones y refactoring, también facilita la escritura de código y a la navegación de clases predefinidas en la plataforma, entre sus tareas principales tenemos:

- ✓ editar código
- ✓ Compilar
- ✓ ejecutar
- ✓ depurar

3.2.2 Programación en R.

Según (Santana, 2014) **R** es un lenguaje de programación para el análisis estadístico y la representación gráfica, de distribución libre bajo la licencia **GNU**, es compatible en distintas plataformas como: Linux, Windows, MacOS, entre otros. Este lenguaje permite escribir instrucciones denominadas scripts, una de las principales novedades de **R** es que utiliza datos compuestos como son vectores, matrices, entre otros. Para usar de una forma más práctica **R** se utiliza el entorno de desarrollo **RStudio.**

3.2.2.1 RStudio.

Según (R-Tools Technology Inc, 2015) **RStudio** en un entorno de desarrollo integrado (IDE) que permite trabajo con el lenguaje de programación R, este editor incluye una consola donde se ejecuta el código creado, un editor para la creación del código fuente. También contiene un espacio de trabajo donde se cargan las funciones, objetos, entre otros. También permite instalar paquetes de forma muy sencilla.

3.2.3 Programación en Python.

Como afirma (Díaz-ricardo & Becerra-garcía, 2014) Python es un lenguaje de programación bajo licencia de software libre, el cual tiene su propia sintaxis de escritura como todos los lenguajes de programación.

Este lenguaje soporta varios tipos de programación como los orientados a objetos, programación imperativa, programación funcional, entre otros.

Alguna de las características principales se muestran a continuación:

- ✓ Orientado a varias formas de programación.
- ✓ Puede ser ejecutado en múltiples plataformas.
- ✓ Su sintaxis y semántica es sencilla.
- ✓ Utiliza tipos dinámicos.
- ✓ Se pude programar simples scripts o aplicaciones de gran tamaño.
- ✓ Es modular.
- ✓ Incluye una poderosa biblioteca de clases.

Hoy en la actualidad es uno de los lenguajes más utilizados por su fácil uso y eficiencia al momento de la creación de aplicaciones. Existen framework que permiten un manejo más óptimo y organización de los archivos que se utilizan en las aplicaciones. Uno de los framaworks más utilizados es **Django** el cual se dará una breve explicación a continuación.

3.2.3.1 Framework Django.

Según la página oficial de Django, es un framework de alto nivel para la creación de aplicaciones web que trabaja con Python, permitiendo un desarrollo rápido y un diseño limpio y pragmático. Este framework se encarga de elaborar el contenido de una página web, y poderla visualizar en alguna dirección web (URL).

Alguna de las características principales se muestran a continuación:

- ✓ Rápido
- ✓ Seguro
- ✓ Escalable
- ✓ Versátil

Tomando en cuenta estas características que ofrece Django, se lo considera como uno de los mejor frameworks para el desarrollo de aplicaciones web.

3.2.4 Cuadro de resumen de los lenguajes de programación.

Tabla 6. Tabla de resumen de lenguajes de programación

Lenguajes de Programación	Uso
Java	Utilización de un sistema recomendador basado en Filtrado colaborativo, elaborado por (Deldjoo, Elahi, & Cremonesi, 2016).
Python	Implementación de Métricas de Precisión, Recall, F-Measure y Mae. Utilizando CF.
R	Implementación de Métrica Recall utilizando CTR.

Fuente. El autor.
Elaborado por. El autor.

3.3 Sistema Recomendador utilizando CF (Collaborative filtering)

Para el funcionamiento de las métricas de evaluación de resultados usando CF se ha optado por utilizar un algoritmo elaborado por (Deldjoo, Elahi, & Cremonesi, 2016) desarrollado en **Java**. Al cual esta modificado de acuerdo con las necesidades que se necesitaron.

La técnica de filtrado colaborativo según (F.O. Isinkaye, Y.O. Folajimi, 2015) permite construir una Matriz de elementos de usuario en el que constan los rankings sobre los ítems que le gustaron y le parecieron interesantes. Con esto se puede comparar los usuarios con intereses y preferencias relevantes calculando las similitudes entre sus perfiles para hacer recomendaciones. Estos usuarios construyen un grupo llamado barrio. En el que un usuario obtiene recomendaciones de a aquellos artículos que han calificado otros usuarios anteriormente.

En la Figura 10 de puede apreciar una matriz en la que consta de columnas de usuarios y filas de ítems, en el que cada intercesión se puede dar una calificación del usuario a un ítem determinado. Con esta matriz el filtrado colaborativo puede dar predicciones en base a las calificaciones de los usuarios.

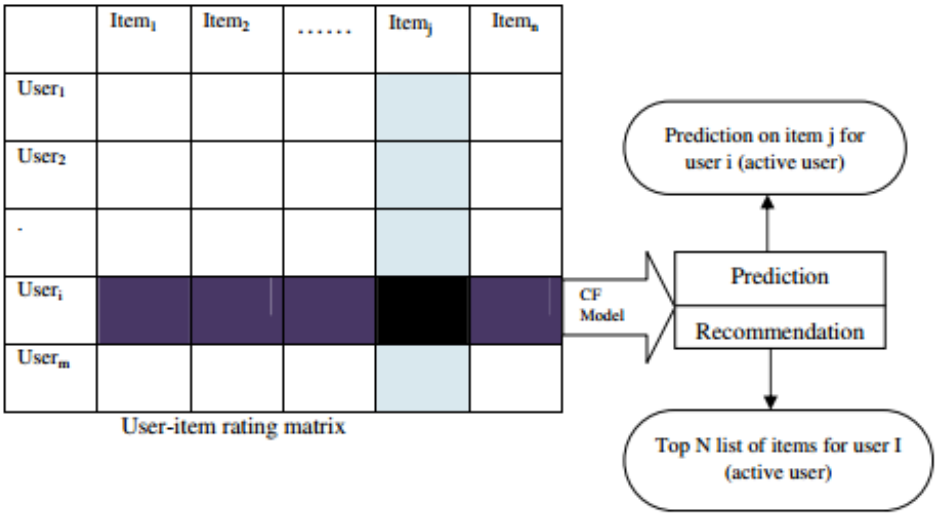


Figura 10. Proceso de Filtrado Colaborativo
Fuente. Recuperado de noviembre del 2015, de https://goo.gl/JP4oTg
Elaborado por. (F.O. Isinkaye, Y.O. Folajimi, 2015)

Según (Castells et al., 2011) al utilizar filtrado colaborativo pueden surgir algunas ventaja y desventajas, algunas de ellas son:

Ventajas:

- ✓ No tiene gran importancia el contenido de los artículos recomendados, si no que toma los rankings dados por los usuarios para dar recomendaciones
- ✓ Se pude aplicar a cualquier ítem.

Desventajas:

- ✓ Escasez (Sparsity) debido a que no todos los usuarios califican los ítems disponibles en el sistema. Dificultando calcular la similitud entre usuarios e ítems.
- ✓ El arranque en frio; esto se debe a que existen nuevos usuarios y ítems en el sistema y aún no han realizado ninguna calificación(rankings) por los que son nuevos en el sistema.
- ✓ No se puede explicar al usuario la recomendación que se le asigna por lo que no existe transparencia.

El filtrado colaborativo se puede dividir en dos grupos dependiendo de cómo se realizan las recomendaciones. Estos dos tipos de detallan a continuación:

- ✓ Basados en modelos.
- ✓ Basados en memoria.

El filtrado Colaborativo **basado en modelos** emplea las calificaciones anteriores para que un modelo aprenda, con el fin de mejorar el rendimiento del filtrado colaborativo.

El filtrado Colaborativo **basado en memoria** puede ser de dos formas: basados en usuario y basados en ítems.

La técnica de filtrado colaborativo basada en el usuario calcula la similitud entre los usuarios mediante la comparación de sus calificaciones en el mismo artículo. En cambio, las técnicas de filtrado colaborativo basado en ítems calculan las predicciones usando la similitud entre elementos (ítems) y no la similitud entre los usuarios.

3.3.1 Algoritmo Seleccionado (“k-nearest neighbor”).

El algoritmo que utiliza el sistema recomendador de CF, es el de k-vecinos cercanos basados en ítems, A continuación, se detalla su funcionamiento.

Como afirma (Milano, Deldjoo, Elahi, Cremonesi, & Bakhshandegan, 2016) uno de los algoritmos para realizar recomendaciones es k-nearest neighbor (vecinos más cercanos) el cual utiliza filtrado colaborativo. Según (Saini, 2013) es unos de los algoritmos para la clasificación de objetos, tomando en cuenta el número de vecinos más cercanos (*k*). Este algoritmo es denominado perezoso ya que no realiza ningún aprendizaje durante la fase de entrenamiento, en esta fase solo utiliza los datos de entrenamiento para poblar una muestra.

El objetivo de este algoritmo es calcular sus K vecinos más cercanos dividiendo en dos fases principales las cuales son:

1. Fase de entrenamiento
2. Fase de clasificación

Dentro de la fase de entrenamiento los datos se representan en vectores de características y las etiquetas de clase de las muestras de entrenamiento, mientras que, en la fase de clasificación, k es una constante definida por el usuario, con el fin de buscar las k muestras de entrenamiento más cercanas a ese valor. De esta forma se categoriza los puntos de consulta basándose en la distancia más cercana de una manera simple y efectiva.

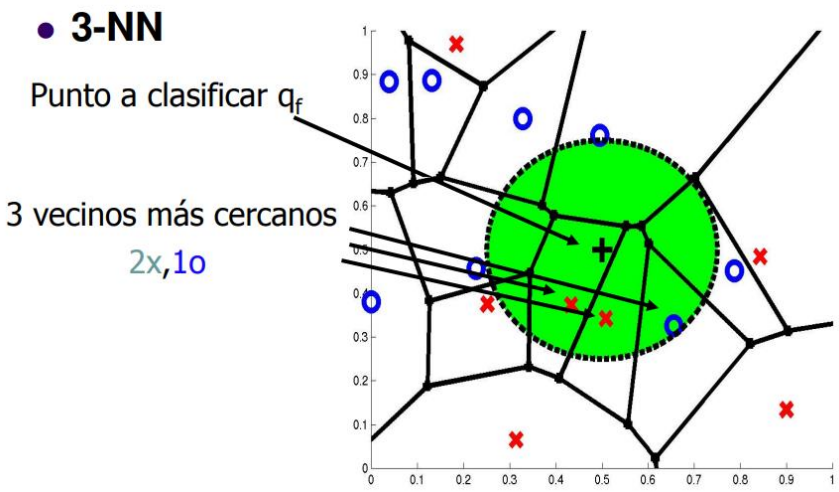


Figura 11. Diagrama de KNN
Fuente. Recuperado en 2013 de, https://goo.gl/uD9L73
Elaborado por:(Lora, 2013)

Como se muestra en la Figura 11 el algoritmo vecino cercanos establece un radio o distancia donde se encuentran el número de k vecinos más cercanos. En este ejemplo se tiene k=3 donde se seleccionan las 3 posiciones más cercanas con relación a la posición original.

Una de las ventajas de este algoritmo es que requiere pocos parámetros los cuales son k y la métrica de distancia. Generalmente cuando se elige un valor alto de K se reducen el efecto del ruido en la clasificación, pero hacen que los límites entre las clases sean menos distintos. Es recomendable elegir un número impar de k ya que evita los votos empatados.

El funcionamiento de este algoritmo parte del número de vecinos que se elijan para realizar los procedimientos posteriores. Para la generación de recomendaciones se necesita tener un conjunto de usuarias y una lista de ítems que hayan sido calificados.

Una recomendación es realizada, comparando la similitud de los ítems entre sí, usando la semejanza del coseno como se muestra en la siguiente ecuación.

Ecuación 15.Semejanza del Coseno

$$S_{i,j=}\frac{{f_i}^T f_j}{\sqrt{||f_i||}\sqrt{||f_j||}}$$

Fuente. Recuperado del 2015 de, *https://goo.gl/wmqUjk*
Elaborado por .(Bai, 2015)

Donde f_i es el ítem 1 y f_j es el **ítem 2**, para la comparación de los dos ítems el algoritmo compara el valor que tenga la calificación de los dos ítems, ya que el algoritmo de k-vecinos utilizado está basado en ítems.

```
    if (dotProduct == 0) {
        return Float.NaN;
    }
    return (float)((dotProduct
            / (Math.sqrt(normA) * Math.sqrt(normB))));
} else {
    return Float.NaN;
}
```

Figura 12. Implementación de ecuación Coseno
Fuente. Recuperado de septiembre del 2016 de, https://goo.gl/YPKWCf
Elaborado por. (Deldjoo et al., 2016)

En la figura 12 se encuentra implementada la ecuación del coseno divido el producto del ítem 1 y 2 para la raíz cuadrada de cada uno de los valores.

Una vez obtenido las similitudes se procede a recomendar ítems dependiendo del número de vecinos que haya ingresado el usuario. Obteniendo los valores del usuario y el ítem recomendado.

3.4 Métodos de Validad de Datos

Uno de los aspectos más importantes es, la validación y el análisis de los datos. Para ello se utilizan herramientas que ayudan a prevenir la superposición de los datos, cuando utilizamos modelos de sistemas recomendadores por lo general se debe ajustar el modelo a datos de entrenamiento y de pruebas para hacer predicciones.

Al utilizar un modelo de entrenamiento pueden ocurrir dos cosas: overfitting y Underfitting.

El overfitting se produce cuando se entrena el modelo demasiado bien, esto ocurre cuando el modelo es demasiado complejo. En cambio, el Underfitting es lo contrario, cuando el modelo no se ajusta a los datos de entrenamiento y por lo tanto pierde la tendencia de los datos. Para evitar estos tipos de problemas existen algunos métodos que son.

- ✓ Validación Cruzada
- ✓ Método TrainTestSplit

3.4.1 Validación Cruzada *(Cross Validation)*.

Según (Pérez-Planells & Delegido, 2015) existen muchas técnicas para validar métodos entre ellos tenemos la validación cruzada la cual tiene técnicas como *hold-out* y *k-folder*.

3.4.2 Hold-out.

Es uno de los métodos más sencillos de utilizar en la validación cruzada, este método separa los datos en dos subconjuntos; uno es usado para entrenar el modelo y el otro para realizar el test de validación.

De esta manera, se crea un modelo únicamente con los datos de entrenamiento. Con el modelo creado se generan datos de salida que se comparan con el conjunto de datos reservados para realizar la validación.

3.4.3 K-folder.

Como afirma (Pérez-Planells & Delegido, 2015) este método está basado en hold-out y es utilizado para conjuntos de datos más pequeños. El objetivo de este método es dividir los datos en k subconjuntos, con el fin de aplicar hold-out en cada subconjunto creado.

Al momento de comparar los dos métodos, el método k-fold tiene la ventaja de que todos los datos son utilizados para entrenar y validar, por lo que se obtienen resultados más representativos. Mientras que, para el método hold-out, se realiza el proceso n veces de manera aleatoria, lo que no garantiza que los casos de entrenamiento y validación no se repitan.

Una forma más representativa se muestra en la figura 13 en la que se puede observar como clasifica los datos el modelo k-folder. En este ejemplo se toma un valor de k=4 donde el método divide cada iteración en datos de prueba y datos de entrenamiento

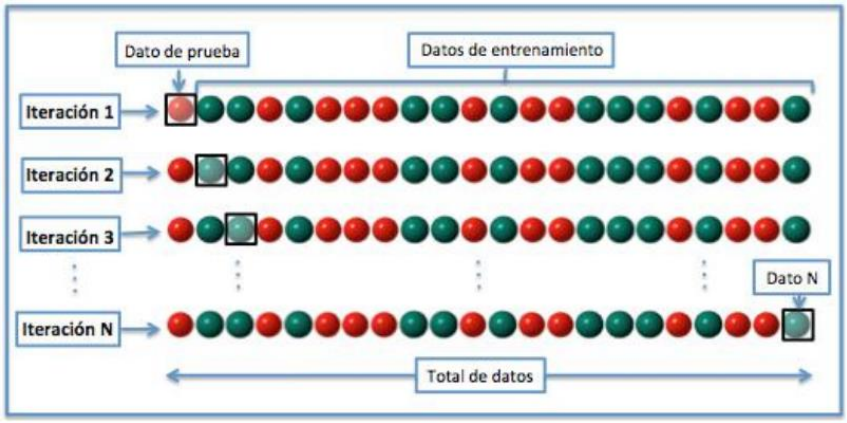


Figura 13. Cross Validation
Fuente. El autor.
Elaborado por. El autor

Este método está elaborado (Deldjoo et al., 2016) desarrollado en **Java**. El cual utiliza una clase en java, que permite la separación de los datos en dos partes:

- ✓ Test
- ✓ Entrenamiento

Estos datos se dividen dependiendo del número de carpetas.

Test

El método ***getTestData*** permite obtener los datos que serán destinos a la parte de entrenamiento del modelo.

```
public
        DataModel getTestData(
                final int foldNumber)
{
    if (foldNumber > Globals.NUMBER_OF_FOLDS) {
        throw new IllegalArgumentException(
                "Fold number is greater than max fold numner. Max fold number="
                        + Globals.NUMBER_OF_FOLDS);
    }
    final int chunk = (int)(this.dataModel.getNumberOfRatings()
            / Globals.NUMBER_OF_FOLDS);
    final int startIndex = (foldNumber - 1) * chunk;
    final int endIndex = startIndex + chunk;
    return this.dataModel.getTestData(startIndex, endIndex);

}
```

Figura 14. Obtener datos de Test k-folder
Fuente. Recuperado de septiembre del 2016 de, https://goo.gl/YPKWCf
Elaborado por. (Deldjoo et al., 2016)

En este método se obtiene como parámetro inicial el número de carpetas previamente ingresada por el usuario.

Luego se divide los datos de los rankings para el número de carpetas y los almacenamos en la variable **chunk** con el fin de obtener el inicio y el final de los datos que irán en cada carpeta.

Entrenamiento

El método ***getTrainData*** permite obtener los datos que serán destinos a la parte de entrenamiento del modelo.

```
public
        DataModel getTrainData(
                int foldNumber)
{
    if (foldNumber > Globals.NUMBER_OF_FOLDS) {
        throw new IllegalArgumentException(
                "Fold number is greater than max fold numner. Max fold number="
                        + Globals.NUMBER_OF_FOLDS);
    }
    final int chunk = (int)(dataModel.getNumberOfRatings()
            / Globals.NUMBER_OF_FOLDS);
    final int startIndex = (foldNumber - 1) * chunk;
    final int endIndex = startIndex + chunk;
    return this.dataModel.getTrainData(startIndex, endIndex);

}
```

Figura 15. Obtener datos de Entrenamiento k-folder
Fuente. Recuperado de septiembre del 2016 de, *https://goo.gl/YPKWCf*
Elaborado por. (Deldjoo et al., 2016)

En este método se obtiene como parámetro inicial el número de carpetas previamente ingresada por el usuario.

Luego se divide los datos de los rankings para el número de carpetas y los almacenamos en la variable **chunk** con el fin de obtener el inicio y el final de los datos que irán en cada carpeta.

3.4.4 Método TrainTestSplit.

Este método permite dividir todo el conjunto de datos en dos partes que son.

- ✓ Datos de entrenamiento
- ✓ Datos de prueba

El conjunto de entrenamiento permite que el modelo aprenda sobre estos datos, y posteriormente puede realizar recomendación. Mientras que el conjunto de pruebas es utilizado para validar el sistema y conocer cuan preciso es al modelo al recomendar un conjunto de ítems al usuario.

Figura 16. Método Split
Fuente. El autor.
Elaborado por. El Autor.

Como se puede ver en la figura 16 el conjunto total de datos está divido en dos subconjuntos.

El funcionamiento de este método se centra en dividir los datos dependiendo del porcentaje que se le quiera ingresar. En cada uno de estos subconjuntos se encuentran los datos de usuarios, ítems y calificaciones.

Entrenamiento.

Para obtener el subconjunto de entrenamiento, se usa el método GetTrainData, que recupera el valor que se le asigna al método anterior.

```
public
        DataModel getTrainData(
                int dataSplit)
{
    if (dataSplit > Globals.DataOriginal) {
        throw new IllegalArgumentException(
                "no se pudo obtener los datos de entrenamiento"
                        + Globals.DataOriginal);
    }

    return this.dataModel.DividirDataSplit(53);
}
```

Figura 17. Entrenamiento Split
Fuente. El autor.
Elaborado por. El Autor.

En la figura 17 muestra un ejemplo del 53% de los datos que serán utilizados para entrenar el modelo.

Test.

Para obtener el subconjunto de entrenamiento, se usa el método GetTestData, que recupera el valor que se le asigna al método anterior.

```
public
        DataModel getTestData(
                final int dataSplit)
{
    if (dataSplit > Globals.DataOriginal) {
        throw new IllegalArgumentException(
                "no se pudo obtener los datos de pruebas"
                        + Globals.DataOriginal);
    }

    return this.dataModel.DividirDataSplit(47);
}
```

Figura 18. Test Split
Fuente. El autor.
Elaborado por. El Autor.

En la figura 18 muestra un ejemplo del 47% de los datos que serán utilizados para probar el modelo.

3.5 Métricas Implementadas

En esta sección se explicará la implementación de cada una de las métricas de evaluación de resultados. Vale recalcar que estas métricas pertenecen al grupo de métricas de exactitud por lo que necesitan valores cuantitativos para poder ser calculadas.

Estas métricas están desarrolladas en el lenguaje de programación Python y para la interfaz se utilizó el framework Django como se mostrará a continuación.

Los archivos necesarios para que funciones las métricas deben estar en un formato sencillo y fácil de representar, en este caso se optó por usar archivos planos (.csv, .dat, .txt) los cuales son muy sencillos y ocupan una mínima cantidad de espacio en disco.

3.5.1 Diseño de la Aplicación.

La Aplicación Web está elaborada con el framework Django utilizado Python, la cual permite cargar los archivos necesarios para el funcionamiento de cada una de las métricas, y poder mostrar los resultados en graficas estadísticas.

El diseño de la aplicación utiliza el framework Bootstrap. El cual es un conjunto de herramientas de código abierto desarrolladas en HTML, CSS, JS.

Algunas ventajas que tiene el framework Bootstrap para la elaboración de sitios web son:

- ✓ Puedes tener una web bien organizada de forma visual rápidamente.
- ✓ El diseño es adaptable a cualquier dispositivo.

- ✓ Nos permite usar **Less**, para enriquecer aún más los estilos de la web.

Para la utilización de la página web es necesario acceder a través de un navegador, con la siguiente dirección http://127.0.0.1:8000/. Aquí se le mostrara las diferentes funcionalidades que contiene el sitio web.

- ✓ Accedo a una data set de prueba y como utilizarlo.
- ✓ Cargar Data set.
- ✓ Visualización de resultados.
- ✓ Ayuda.

3.5.2 Inicio de la Aplicación.

Figura 19.Inicio de la Aplicación
Fuente. El autor.
Elaborado por. El Autor.

En la figura 19 se observa el frontal principal de la aplicación, que contiene el objetivo general, especificado la función principal de la página web. Dentro de esta sección se presenta un enlace donde se encuentra un DataSet para poder ser descargado.

También se encuentra una sección donde se mencionan cada una de las métricas dependiendo al grupo que pertenecen. Estos grupos son:

Métricas de exactitud en clasificación.

Según (Buitrago, 2012) estas métricas se las conoce como métricas de decisión, ya que evalúan la efectividad de un sistema recomendador, ayudando a seleccionar los ítems más relevantes, es decir, con qué frecuencia el sistema efectúa recomendaciones correctas.

En este grupo se encuentran las métricas de Precisión, Recall y F-Measure.

Métricas de exactitud predictiva

Como afirma (Buitrago, 2012) estas métricas miden la cercanía de los puntajes predichos por el sistema recomendador con respecto a los puntajes reales que haya ingresado el usuario. En esta sección se encuentra la métrica Mae.

3.5.3 Cargar Data Set.

Figura 20. Cargar Data Set
Fuente. El autor.
Elaborado por. El Autor.

En la figura 20 se observa una sección donde se pueden cargar los archivos correspondientes a la utilización de las métricas de Precisión, Recall, F-Measure y Mae.

También se pude seleccionar el top n de recomendaciones, las cuales pueden ser de 5, 10, 20, 30, 40 y algún otro valor que deseé ingresar el usuario.

Los archivos utilizados son:

- ✓ Usuarios
- ✓ Lista
- ✓ Top N recomendaciones

Ratings: estos archivos están compuestos de tres campos: id del usuario, id del ítem y la calificación de un artículo.

Estos ratings son obtenidos por medio de la data de prueba que realiza el método de validación de datos que se mencionó anteriormente. es decir, por cada id de usuario que se encuentra en la data, se obtiene los ítems con sus respectivas calificaciones que haya visto.

Recomendaciones: estos archivos están compuestos por 4 campos los cuales son: id usuario, id ítems, recomendación y predicciones.

Estos archivos son generados por el sistema recomendador de k-vecinos cercanos basado en ítems en cual esta descrito anteriormente.

Top N Recomendaciones: en esta sección se seleccionan el número de recomendaciones que desea ingresar el usuario para el funcionamiento de las métricas de evaluación de resultados.

3.5.4 Resultados Precisión y Recall.

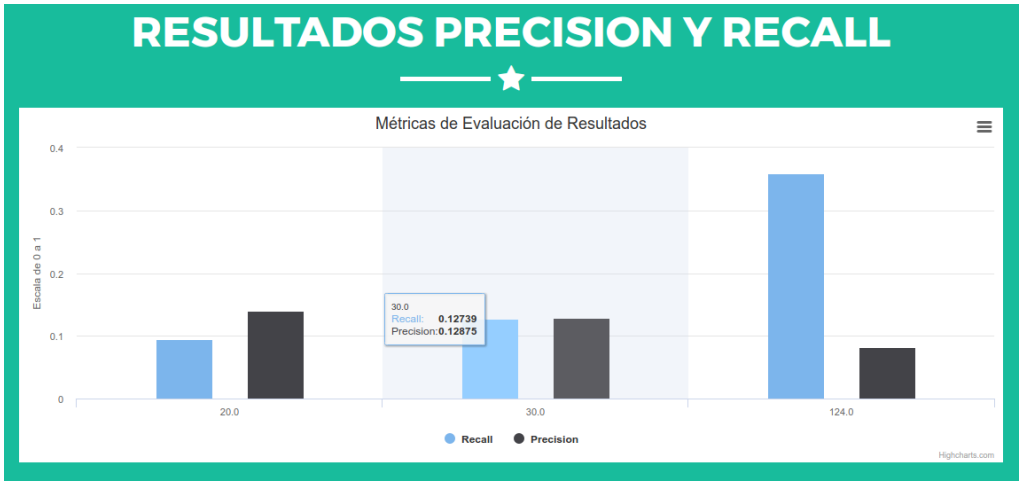


Figura 21. Analizar resultados de Precisión y Recall.
Fuente. El autor.
Elaborado por. El Autor.

En la figura 21 se observa una gráfica de barras donde se muestra los resultados obtenidos por la Métrica de Precisión y Recall.

En el eje de las X se muestran los nombres de las métricas con sus valores correspondientes. En el eje de las Y se establece una escala que se incrementa conforme las métricas aumentan sus valores.

3.5.5 Resultados F-Measure.

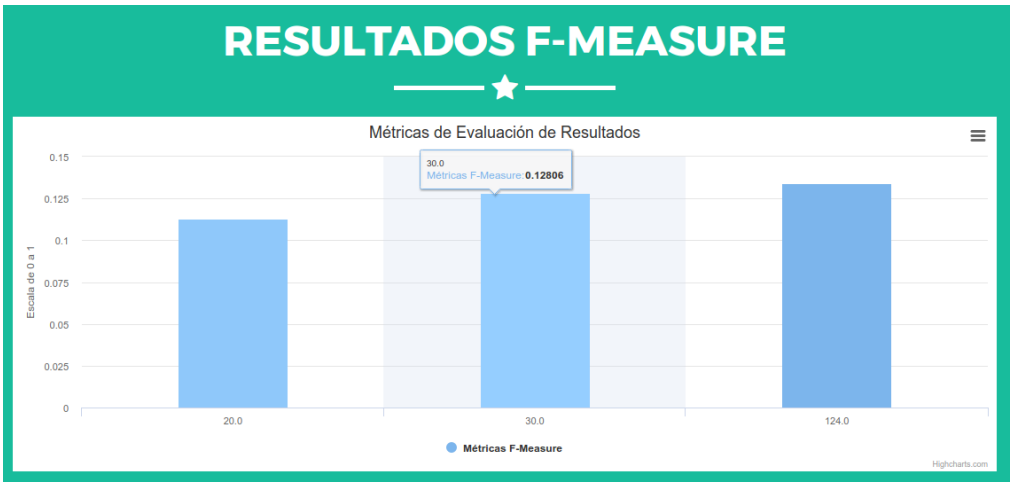


Figura 22. Analizar resultados de F-Measure
Fuente. El autor.
Elaborado por. El Autor.

En la figura 22 se observa una gráfica de barras donde se muestra los resultados obtenidos por la Métrica de F-Measure.

3.5.6 Resultados Mae.

Figura 23. Analizar resultados de F-Measure
Fuente. El autor.
Elaborado por. El Autor.

En la figura 23 se observa una gráfica de barras donde se muestra los resultados obtenidos por la Métrica de Mae.

3.5.7 Ayuda.

Figura 24. Ayuda de la aplicación
Fuente. El autor.
Elaborado por. El Autor.

En la Figura 24 se observa 3 secciones que ayudan a usar la aplicación y conocer el formato correcto que deben contener los archivos que utilizan las métricas. Estas secciones son:

Formato archivos DataSet.

En esta sección se muestra algunos ejemplos de archivos con sus características de deben contener.

Ejemplos de Resultados Métricas

En esta sección se detallan ejemplos de resultados obtenidos con diferentes archivos ingresados.

Uso y funcionamiento de la aplicación

En esta sección se detallan las características de la aplicación y todos los componentes que tiene. También se muestra una pequeña explicación de cada una de las métricas y las ecuaciones que utilizan.

A continuación, se detalla la implementación de cada una de las Métricas.

3.5.8 Controles de la Pagina Web.

Los controles realizados sobre la página Web son:

Control de ingreso de archivos

El sistema controla si se ha ingresado los dos archivos (Ratings y Recomendaciones). En caso de que no se ingrese alguno de los dos archivos, el sistema envía una alerta diciendo “Es necesario que seleccione los dos archivos.” Ya que, sin los archivos no es posible analizar los resultados con las métricas.

Control de extensión de archivos

El sistema controla si la extensión de los archivos es el correcto. En caso de que el usuario ingrese un archivo con una extensión que no sea válida. El sistema envía una alerta diciendo “Archivos incorrectos. Deben estar en formato csv, data, txt.”

Control de delimitadores de archivos

El sistema controla si los delimitadores que contienen los archivos para separar las columnas deben ser los correctos. En caso de que el usuario ingrese archivos con delimitadores inválidos, el sistema envía una alerta diciendo “Separadores incorrectos. Los separadores permitidos son: coma (,), punto y coma (;), tabulador ( ) y dos puntos (:).”

Control de contenido de archivos

El sistema controla si el número de columnas que ingresa el usuario en cada uno de los archivos es correcto. Caso contrario, el sistema envía una alerta diciendo “'El número de columnas es incorrecto. El archivo Ratings debe tener 3 columnas, y el archivo Recomendaciones debe tener 4 columnas.”

Control de Top N de recomendaciones

El sistema controla si el número de recomendaciones ingresadas por el usuario son las correctas. Caso contrario, el sistema envía una alerta diciendo “El número de recomendaciones que ha seleccionado es mayor al top N permitido de sus datos. Y el mínimo top N debe ser 5.”

3.5.9 Métrica Precisión

La Precisión la define como la fracción de ítems relevantes del total del conjunto de ítems devueltos por el sistema.

Para el funcionamiento de la métrica Precisión se necesita obtener los datos del usuario; el cual ha observado algunos ítems dándoles calificaciones en un rango de [0 a 5].

Entre más cerca este la calificación de 0; quiere decir que no le gusto ese ítem, mientras que la calificación esté más cerca de 5 significara que le gusto el ítem. por esto, Como afirma (Deldjoo et al., 2016) y (Milano et al., 2016) una calificación es relevante cuando es mayor o igual a 3.5.

Esta métrica utiliza los verdaderos positivos y el número de recomendaciones. Para obtener los verdaderos positivos se comprueba si el sistema recomendador ha dado sugerencias a los usuarios, y que esa sugerencia sea mayor a 3.5.

Con estos valores se puede aplicar la fórmula de Precisión que es:

Ecuación 16.Precisión

$$Presicion = \frac{|I \ \in \ \{Items\ Relevantes\}|}{|I|}$$

Fuente. Recuperado del 2012, de *https://goo.gl/2fyFyR*
Elaborado por. Parra (2012)

En la ecuación 1 se puede observar la implementación de la fórmula de Precisión dado como parámetros la sumatoria de los ítems positivos (relevantes) divido para el conjunto de ítems recomendados.

```
presicion =verdaderosPositivos / numRecomendaciones
print("Metrica PRESICION:",round(resultado,5))
```

Figura 25.Formula Precisión Implementada
Fuente. El autor.
Elaborado por. El Autor.

En la figura 25 podemos observar la formula implementada en el código.

3.5.10 Métrica Recall.

La Métrica de Recall tiene como objetivo mostrar la fracción de ítems relevantes devueltos por el sistema.

Al igual que la métrica de Recall se debe obtener los valores relevantes como se explicó en la métrica anterior, en resumen, estos valores son relevantes si son mayores o iguales a 3.5.

Esta métrica obtiene los usuarios considerados relevantes, es decir, son todos aquellos usuarios que hayan visto un ítem con una calificación mayor a 3.5.

Después se debe obtener los verdaderos positivos, dependiendo si el sistema recomendador ha dado sugerencias a los usuarios, y que esa sugerencia sea mayor a 3.5.

Con estos valores se puede aplicar la fórmula de Recall que es:

Ecuación 17. Recall

$$Recall = \frac{|I \ \in \ \{items\ relevantes\}|}{|\{items\ relevantes\}|}$$

Fuente. Recuperado del 2013 de, *https://goo.gl/RfYcKB*
Elaborado por. (Parra & Sahebi, 2013)

En la ecuación 3 se puede observar la implementación de la fórmula de Recall dado como parámetros la sumatoria de los ítems positivos (relevantes) divido para el número de ítems relevantes obtenidos.

```
recall=verdaderosPositivos / usuariosRelevantes
print("Metrica RECALL:",round(resultado,5))
```

Figura 26. Formula Recall Implementada
Fuente. El autor.
Elaborado por. El Autor.

3.5.11 Métrica F-Mesure.

Esta Métrica combina la Precisión y Recall dándole valores iguales a cada una.

Para el funcionamiento de esta Métrica, se necesita obtener los usuarios y una lista de recomendaciones realizadas por el sistema. Y unificar la Métrica de Precisión y Recall en una sola. Una vez obtenido estos valores se utiliza la ecuación que le métrica de F-Measure.

Ecuación 18. F-Mesure

$$F1 = \frac{2 * Presicion * Recall}{(Presicion + Recall)}$$

Fuente. Recuperado del 2006 de, https://goo.gl/2xXD7q
Elaborado por. (Mcnee & Konstan, 2006)

Esta ecuación permite obtener de una forma eficiente el problema que genera las métricas de Precisión y Recall, el cual es que al tener mayor Recall disminuye la Precisión y viceversa.

```
f_measure=2*((resultadoPresicion*resultadoRecall)/(resultadoPresicion+resultadoRecall))
print("Metrica F-Measure: ",round(measure,5))
return measure
```

Figura 27.Implementación Métrica F-Measure
Fuente. El autor.
Elaborado por. El Autor.

En la figura 27 se puede observar la implementación de la métrica, haciendo referencia a la ecuación anterior.

3.5.12 Métrica MAE.

Esta Métrica calcula la magnitud promedio de los errores en un conjunto de predicciones realizadas por el sistema recomendador. Tomando el error producido en las predicciones y comparándolo con los resultados.

Para el uso de esta métrica es necesario obtener los siguientes parámetros:

- ✓ **rating**, el cual contiene el id del usuario, el id del ítem y su calificación.
- ✓ **Predicciones** estas predicciones almacenan el valor de la calificación que ha predicho el algoritmo recomendador.

Una vez obtenidos estos parámetros que nos da el algoritmo recomendador se establecen condiciones y se utiliza la ecuación de la métrica MAE que se muestra a continuación.

Ecuación 19. MAE

$$MAE = \frac{| \in \{calificacion - prediccion|}{|\{numero\ de\ casos\}|}$$

Fuente. Recuperado del 2013 de, *https://goo.gl/RfYcKB*
Elaborado por. (Parra & Sahebi, 2013)

Esta ecuación es implementada para obtener el error absoluto

```
        error = math.fabs(ratin - predic)
        errorAcumulado = errorAcumulado + error
        numeroPredicciones = numeroPredicciones + 1
        mae = errorAcumulado / float(numeroPredicciones)
print("Metrica Mae: ", round(mae, 5))
```

Figura 28. Implementación Ecuación MAE
Fuente. El autor.
Elaborado por. El Autor.

En la figura 28 se obtiene el error, y se lo dividí para el número de predicciones (número de casos) como se visualiza en la ecuación 13.

Adicionalmente se proba la métrica de Recall, utilizando un modelo hibrido llamado CTR como se detalla a continuación.

3.6 Implementación de Métrica Recall para Modelo Hibrido (Collaborative Topic Regression)

Para la probar la métrica de Recall usados CTR el cual ha sido usado en una Tesis anterior. Se ha optado por desarrollar en el lenguaje de programación **R.**

Este modelo fue propuesto por el autor (Wang & Blei, 2011), que combina el uso de CF (Filtrado Colaborativo) y el modelo LDA (Latent Dirichlet Allocation) Esto se hace combinando tanto el vector temático latente como las calificaciones observadas para describir el elemento de vector latente en el modelo de factores.

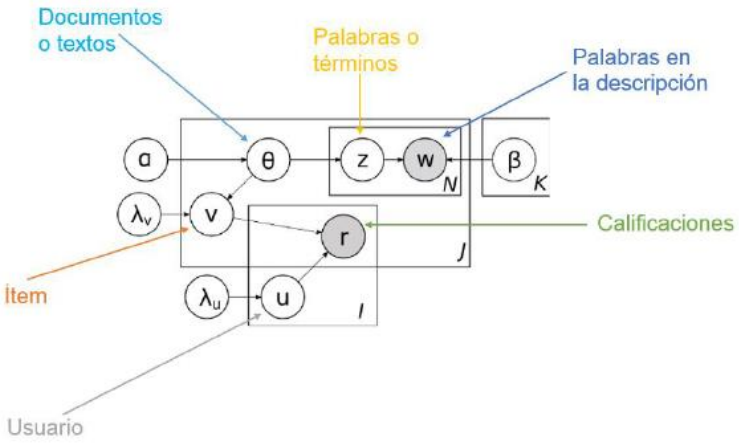


Figura 29. Modelo Grafico para CTR
Fuente. Recuperado de agosto del 2011 de, https://goo.gl/ZqzpWH
Elaborado por. (Wang & Blei, 2011)

CTR explota la información de contenido de los ítems para mejorar el filtrado colaborativo tradicional. El modelo CTR es un método que ha logrado un desempeño prometedor mediante la integración exitosa de la información de retroalimentación de los usuarios y la información de contenido que caracterizan a los ítems.

En la figura 29 se puede apreciar la incorporación del contenido en el que combina técnicas de filtrado colaborativo con técnicas de filtrado basado en contenido haciendo una técnica hibrida entre ambos.

El uso de la métrica Recall asociada a la técnica de CTR se lo puede realizar gracias a que esta métrica está dentro de la familia de métricas de predicción en el cual interactúan los usuarios y los ítems que califican.

Según (Wang & Blei, 2011) esta métrica puede ser aplicada ya que Recall considera los artículos(ítems) de valor positivo.

3.6.1 Tipos de Recomendación.

Dentro de las recomendaciones de CTR existe dos tipos los cuales se detallarán a continuación.

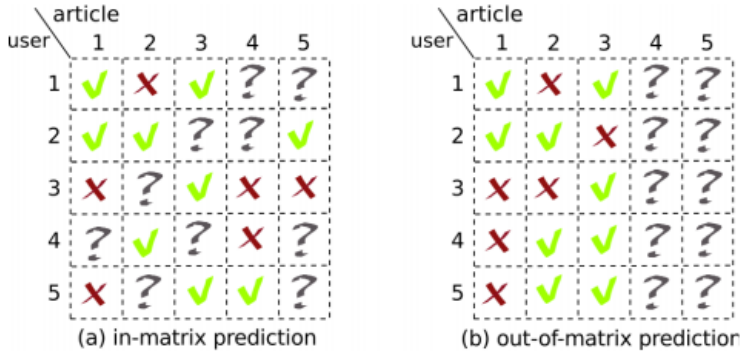


Figura 30. Tipos de Recomendaciones
Fuente. Recuperado de agosto del 2011 de, https://goo.gl/ZqzpWH
Elaborado por. (Wang & Blei, 2011)

En la Figura 30 se puede observar 3 tipos de estados: donde ✓ indica que le gusta, ✗ que no le gusta y ? es desconocido para el usuario.

3.6.1.1 In-matrix.

En la figura 30 (a) ilustra la predicción dentro de la matriz. Esto se refiere al problema de hacer recomendaciones sobre Aquellos artículos que han sido calificados por al menos un usuario en el sistema. Esta es la tarea que el filtrado colaborativo tradicional realiza dividiendo los datos en un conjunto de entrenamiento y un conjunto de pruebas.

3.6.1.2 Out-matrix.

La figura 30 (b) ilustra la matriz fuera de matriz Predicción, donde los artículos 4 y 5 nunca han sido calificados. (Esto es A veces llamada "recomendación de inicio en frío".) La colaboración tradicional Algoritmos de filtrado no pueden hacer predicciones sobre estos Porque esos algoritmos sólo utilizan información sobre otros Calificaciones de los usuarios. En un Sistema de recomendación que no puede manejar la predicción fuera de la matriz No puede recomendar trabajos recientemente publicados a sus usuarios.

3.6.2 Paquetes utilizados.

En la implementación de la Métrica de **Recall** se utilizaron paquetes como:

- ✓ library(plyr)
- ✓ Rcpp
- ✓ Matrix

3.6.2.1 Librery(plyr).

Es una librería del lenguaje de programación R que ayuda a resolver una serie de problemas comunes, por ejemplo, permite operar conjuntos de piezas por separado, para luego unirlas en una sola. Algunas de las ventajas de esta librería son:

- ✓ Encajar el mismo modelo para subconjuntos de una trama de datos.
- ✓ Nombres totalmente consistentes, argumentos y salidas.
- ✓ Aporte de la producción de data.frames, matrices y listas.
- ✓ Las barras de progreso para realizar un seguimiento de las operaciones de larga ejecución.
- ✓ Recuperación de errores y mensajes de error informativos.

3.6.2.2 Rcpp.

Es una dependencia de la librería (plyr) que permite una integración de lenguajes de programación R y C++, en el cual se pueden tener tipos de datos y objetos en R, y ser devueltas sin problemas a C++.

Esto facilita la integración de librerías compatibles en los tipos de lenguajes de programación.

3.6.3 Scripts utilizados.

En esta sección mostramos los scripts necesarios para poder usar la Métrica de Recall, utilizando el modelo CTR.

3.6.3.1 Split.r.

Este script permite dividir la data original en 5 archivos de entrenamiento y 5 de pruebas. Para luego ser procesados por el script eval.r.

Dentro de este script existen funciones principales que son:

- ✓ **cf. split.data**

Permite generar los archivos necesarios para la recomendación **in-matriz.**

Utilizamos 5 veces la validación cruzada. Para cada artículo que aparece en al menos 5 veces en las bibliotecas de usuarios, dividimos uniformemente sus pares (tanto 1 como 0) en 5 pliegues.

Consideramos iterativamente para cada conjunto de prueba y los otros datos para el conjunto de entrenamiento. Para esos Artículos que aparecen menos de 5 veces, siempre los ponemos en El conjunto de entrenamiento. Esto garantiza que todos los artículos del conjunto Aparecen en el conjunto de entrenamiento. (9% de los artículos están siempre en el conjunto de entrenamiento, ya que aparecen menos de 5 veces.) Con esto formamos valoraciones predictivas para las pruebas y se establece y genera una lista de los principales artículos recomendados por M.

- ✓ **ofm.split.data**

Permite generar los archivos necesarios para la recomendación **out-matriz.**

De nuevo utilizamos 5 veces la validación cruzada. En primer lugar, agrupamos Artículos en 5 pliegues. Para cada pliegue, ajustamos el modelo a la sub-matriz Formados por los artículos desplegados y luego probar las recomendaciones Para cada usuario en los artículos dentro de plegado. Nuevamente, formamos clasificaciones predictivas para el conjunto de pruebas, y generamos una lista de las Top M artículos recomendados.

3.6.3.2 Eval.r

Este script es el encargado de probar la métrica de Recall con los datos que se le suministre, las funciones principales con las que cuenta este script son:

- ✓ **scoring.all**

Función que proporciona un criterio de evaluación, es decir genera una matriz de predicciones por cada pliegue (archivo) generado por el script Split.r

- ✓ **summary.recall**

Esta función permite presentar el Recall **tp / (tp + fn)** donde **tp** es el número de verdaderos positivos y **fn** el número de falsos negativos, el Recall permite obtener todas las muestras positivas.

A través de esta función se obtiene el número de método usado, el número de iteración y el Recall obtenido de cada una de ellas.

3.6.3.3 Utils.so

Este archivo es generado a partir de un algoritmo hecho en C++, que permite cargar funciones en R con la ayuda de la librería (Rcpp)

CAPITULO IV:
EXPERIMENTACION

El presente capítulo presenta la experimentación de métricas de evaluación de resultados con técnicas de aprendizaje automático, utilizando el conjunto de datos de MovieLens.

Resumen del capítulo:

- ✓ Obtención de datos para la documentación de los resultados.
- ✓ Experimentación con Modelo CF
  - Análisis de resultados con número de recomendaciones.
  - Análisis de resultados con método Cross-validation.
  - Análisis de resultados con método Split.
  - Análisis de resultados con número de vecinos.
- ✓ Experimentación con Modelo CTR.
- ✓ Comparación de resultados entre CF Y CTR.

4.1 Obtención de datos

Los datos utilizados son tomados de la de la siguiente página:

https://grouplens.org/datasets/movielens/

Este conjunto de datos MovieLens fueron recopilados por el Proyecto de Investigación de GroupLens de la Universidad de Minnesota, en este sitio web se encuentra almacena datos sobre usuarios y Películas, el archivo que se utilizo es MovieLeans 100k el cual contiene lo siguiente:

- ✓ 100.000 calificaciones
- ✓ 1000 usuarios
- ✓ 1700 películas

El contenido de este DataSet tiene varios archivos con diferentes tipos de información, los archivos que fueron utilizados son los siguientes.

- ✓ U.data

Este archivo contiene un conjunto de 100.000 calificaciones realizadas por 943 usuarios sobre un conjunto de 1682 películas. Cada uno de los usuarios a calificado por lo menos 20 películas, para su utilización se ha pasado los datos en un archivo .csv que contiene los siguiente

id_Usuario	Pelicula	Calificacion
196	242	3
186	302	3
22	377	1
244	51	2
166	346	1

Figura 31. Datos de Películas.
Fuente. El autor.
Elaborado por. El Autor.

En la figura 31 se detallan los datos que son utilizados. Por ejemplo, el usuario 196 ha visto la película 242 y ha dado una calificación de 3. Vale recalcar que las calificaciones están en un rango de 0 a 5.

4.2 Experimentación con modelo CF

Los resultados presentados a continuación son obtenidos de un sistema recomendador usando filtrado colaborativo con el algoritmo de k-vecinos cercanos basados en ítems, que se explicó en el capítulo anterior. Las métricas utilizas son:

- ✓ Precisión
- ✓ Recall
- ✓ F-Measure
- ✓ Mae

4.2.1 Tabla de Resumen de resultados.

Tabla 7. Resumen de Resultados

Train/Test	Precisión	Recall	F-Measure	Mae	N_Recomendación	N_K-Vecinos
20/80	0,16484	0,02398	0,04186	0,69644	5	10
20/80	0,1304	0,04732	0,06944	0,69644	10	10
20/80	0,11058	0,08868	0,09843	0,69644	20	10
20/80	0,1014	0,12217	0,11082	0,69644	30	10
20/80	0,09372	0,15535	0,11691	0,69644	40	10
30/70	0,20083	0,02522	0,04482	0,69644	5	10
30/70	0,16605	0,05045	0,07738	0,69644	10	10
30/70	0,14222	0,09194	0,11168	0,69644	20	10
30/70	0,1288	0,12402	0,12637	0,69644	30	10
30/70	0,12066	0,15919	0,13727	0,69644	40	10
47/53	0,24196	0,02545	0,04605	0,69644	5	10
47/53	0,20186	0,0456	0,0744	0,69644	10	10
47/53	0,17011	0,08245	0,11107	0,69644	20	10
47/53	0,15534	0,11469	0,13196	0,69644	30	10
47/53	0,14571	0,1487	0,14719	0,69644	40	10
47/53	0,15502	0,08962	0,11358	0,6204	20	5
47/53	0,17011	0,08245	0,11107	0,64644	20	10
47/53	0,17254	0,05776	0,08655	0,69697	20	30

Fuente. El autor.
Elaborado por. El Autor.

En esta tabla se resumen los resultados obtenidos, con diferentes criterios de evaluación, los cuales se describirán a continuación.

4.2.2 Resultados según el Número de Recomendaciones.

Los resultados que se muestran a continuación dependen del número de recomendaciones que el sistema recomendador asigne a cada uno de los usuarios. Para la visualización de los resultados se ha extraído información con 5, 10, 20, 30 y 40 recomendaciones de películas.

4.2.2.1 *Resultados con 5 Recomendaciones*.

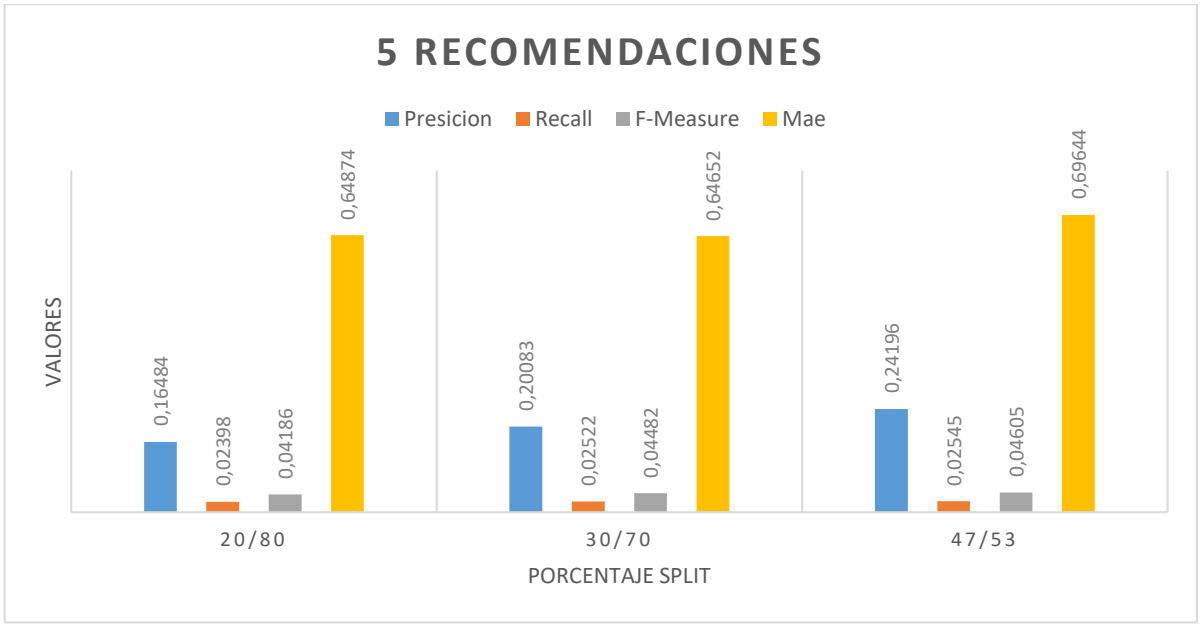


Figura 32. Resultados con 5 recomendaciones
Fuente. El autor.
Elaborado por. El Autor.

En la figura 32 se puede observar 3 grupos de datos, que son los diferentes porcentajes de que asigna el método Split para un conjunto de 5 recomendaciones. En la que se aprecia que las métricas aumentan conforme los datos de pruebas son mayores. Es decir, al tener mayores datos de prueba el sistema recomendador ha aumentado su eficiencia, al momento de dar sugerencias de películas a los usuarios.

Al dar un número de recomendaciones de 5, se observa que la métrica de Presión es mucho mayor a Recall, por lo que la fracción de documentos recuperados que son relevantes es mucho mayor. Para tener una mejor comprensión se ha usado la métrica de F-Measure que muestra un promedio ponderado entre la Precisión y Recall.

La métrica Mae no utiliza el número de recomendaciones, sino solo usa el número de calificaciones y las predicciones esperadas que realiza el sistema recomendador. Estos valores varían dependiendo del porcentaje de datos de pruebas del método Split, en la figura 37 se observa que la métrica Mae aumenta conforme los datos de pruebas son

mayores. Es decir, el promedio de los errores del conjunto de predicciones realizadas por el sistema recomendador aumenta conforme los datos de pruebas se incrementan.

4.2.2.2 Resultados con 10 Recomendaciones.

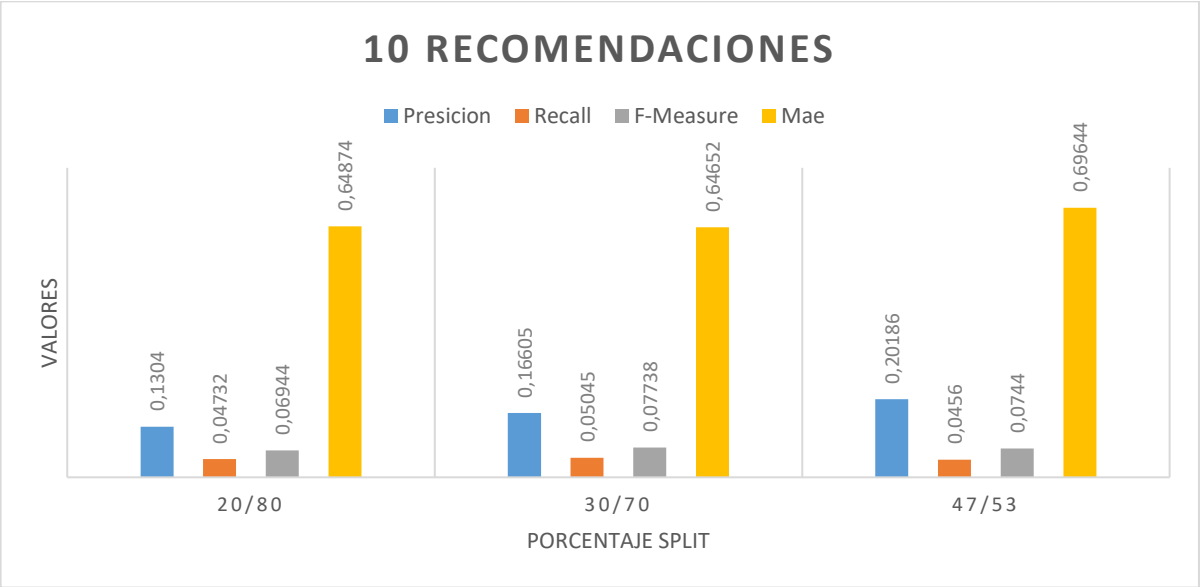


Figura 33. Resultados con 10 recomendaciones
Fuente. El autor.
Elaborado por. El Autor.

En la figura 33 se puede observar los resultados obtenidos con 10 recomendaciones. En la que se aprecia que las métricas aumentan conforme los datos de pruebas son mayores. Es decir, al tener mayores datos de prueba el sistema recomendador ha aumentado su eficiencia, al momento de dar sugerencias de películas a los usuarios.

Al dar un número de recomendaciones de 10, se observa que la métrica de Presión es mucho mayor a Recall, por lo que la fracción de documentos recuperados que son relevantes es mucho mayor.

La métrica Mae no varía a la sección anterior ya que el número de recomendaciones no afecta los valores de Mae, como se explicó anteriormente.

4.2.2.3 Resultados con 20 Recomendaciones.

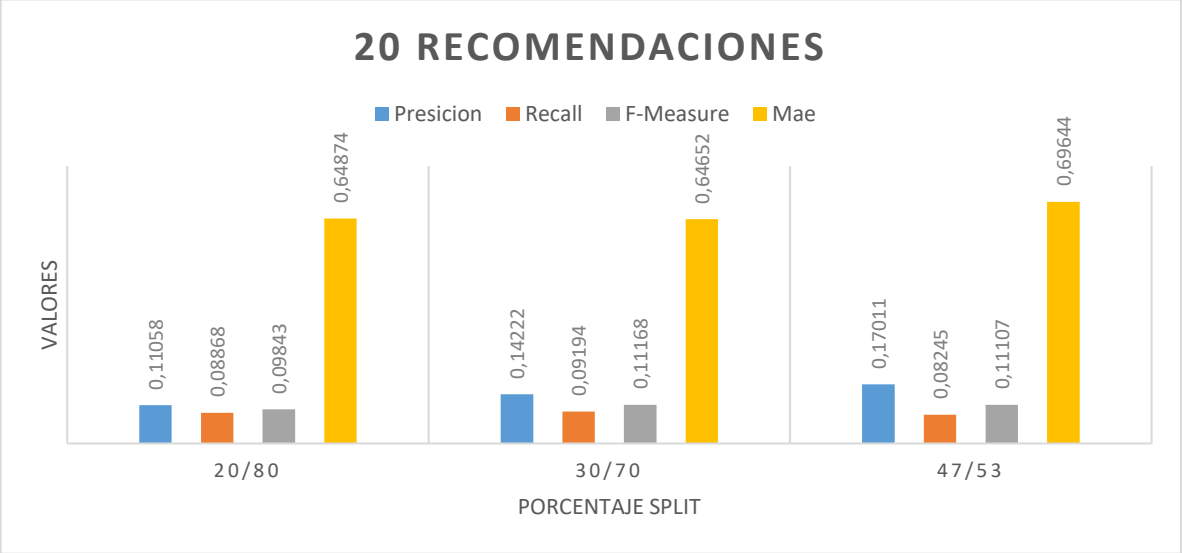


Figura 34. Resultados con 20 recomendaciones
Fuente. El autor.
Elaborado por. El Autor.

En la figura 34 se puede observar los resultados obtenidos con 20 recomendaciones. En este caso se puede ver una variación en los resultados, ya que la Métrica de Precisión aumenta conforme los datos de pruebas son mayores. Pero Recall no es así, esto se debe a que, al aumentar los datos de pruebas, la métrica Recall obtiene la fracción de los documentos que son relevantes y que recuperan con éxito mucho más baja.

La métrica Mae no varía a la sección anterior ya que el número de recomendaciones no afecta los valores de Mae, como se explicó anteriormente.

4.2.2.4 Resultados con 30 Recomendaciones.

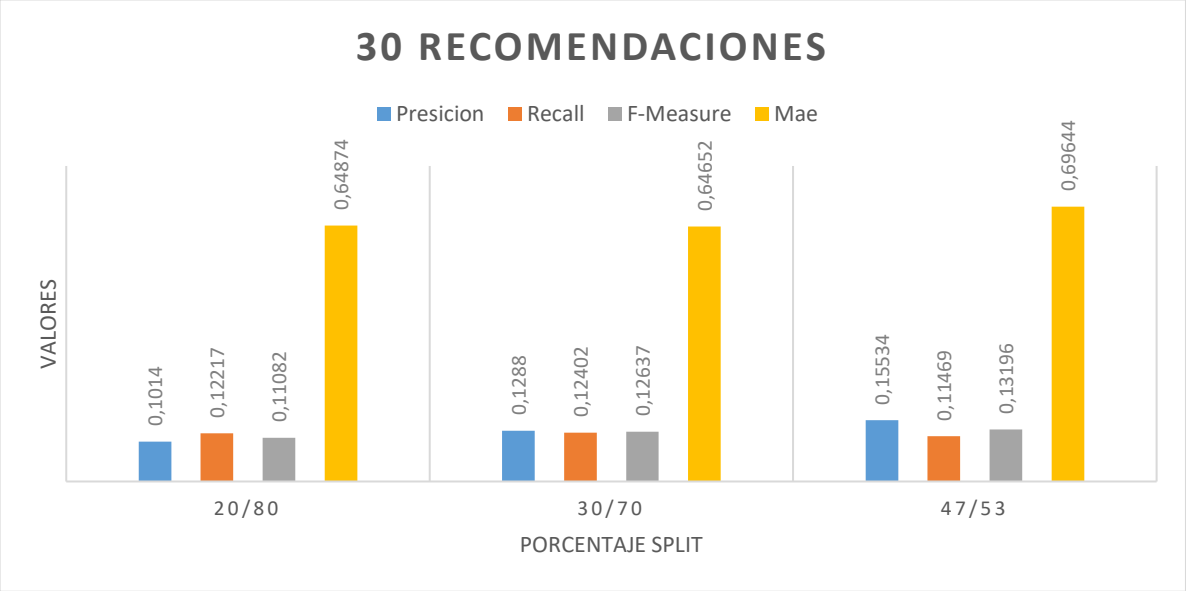


Figura 35. Resultados con 30 recomendaciones
Fuente. El autor.
Elaborado por. El Autor.

En la figura 35 se puede observar los resultados obtenidos con 30 recomendaciones. Donde se observa resultados obtenidos tienen valores un poco más similares que los descritos anteriormente. En este caso la métrica Recall tiene valores más altos que Precisión, esto se debe a que la fracción de películas que son relevantes y que fueron recuperadas con éxito aumenta con un número de recomendaciones mayor.

4.2.2.5 Resultados con 40 Recomendaciones.

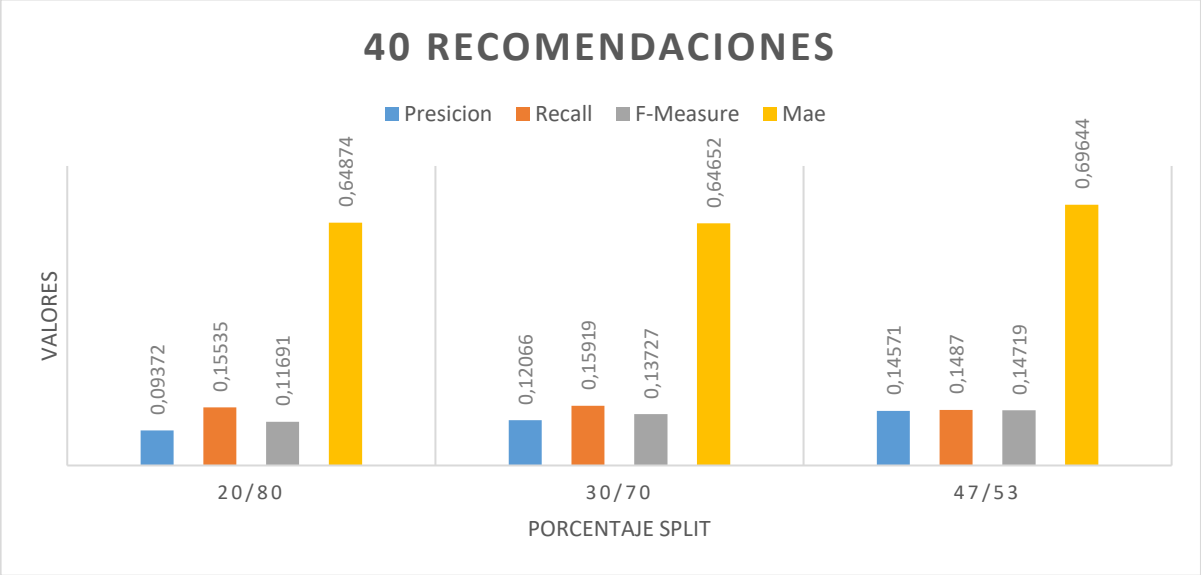


Figura 36. Resultados con 40 recomendaciones.
Fuente. El autor.
Elaborado por. El Autor.

En la figura 36 se puede observar los resultados obtenidos con 40 recomendaciones. Donde se observa que la métrica Recall es mucho mayor que en todos los casos anteriores, al igual que en el caso de 30 recomendaciones esto se debe a que la fracción de películas que son relevantes y que fueron recuperadas con éxito aumenta con un número de recomendaciones mayor.

Tomando en cuenta todos los resultados anteriores dependiendo del número de recomendaciones, el sistema recomendador es mucho más eficiente al tener un número de recomendaciones más alto, ya que las métricas de Precisión, Recall y F-Measure incrementan sus valores.

4.2.3 Resultado con Cross- Validation.

Los resultados que se muestran a continuación están obtenidos con el método de análisis de datos llamado **Cross-Validation**, el cual permite dividir los datos en carpetas para ser procedas las métricas a cada una de ellas y obtener una media de cada métrica en cada una de esas carpetas.

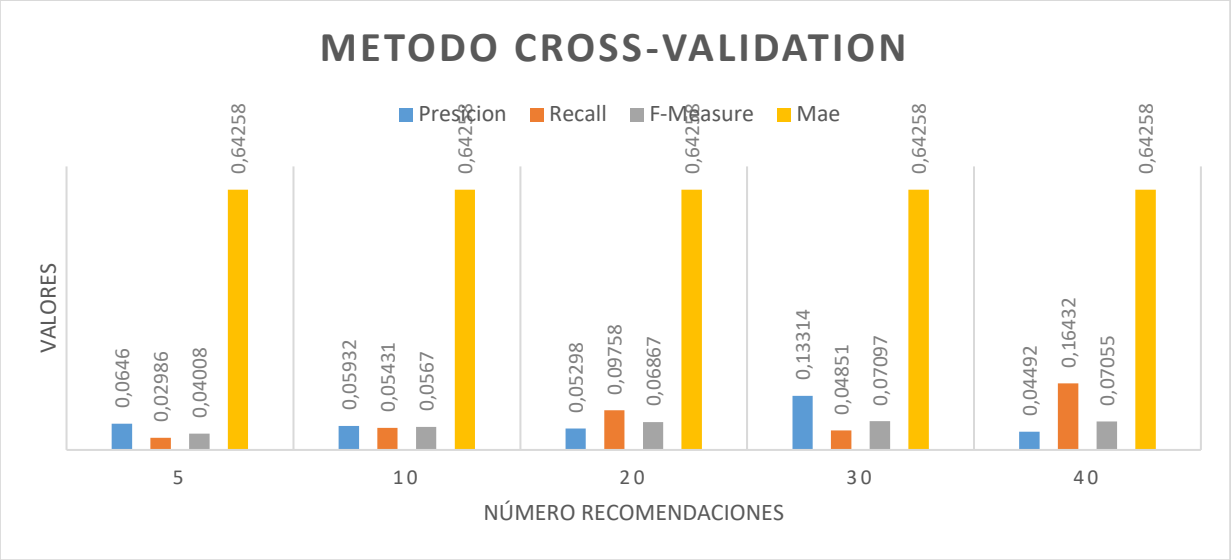


Figura 37. Resultados Cross Validation.
Fuente. El autor.
Elaborado por. El Autor.

En esta figura 37 se observa los resultados obtenidos, dividiendo los datos en 5 carpetas, y obtenido los valores de las métricas de cada una de ellas.

Al momento de aumentar el número de recomendaciones, se pude observar que los valores de las métricas aumentan.

4.2.4 Resultados con el Método Split.

Los resultados que se muestran a continuación están obtenidos con el método de análisis de datos llamado **Split**, el cual permite dividir los datos en dos partes que son. Datos de entrenamiento y datos de pruebas.

Para la visualización de los resultados se ha extraído información con:

- ✓ 20% pruebas y 80% entrenamiento.
- ✓ 30% pruebas y 80% entrenamiento.
- ✓ 47% pruebas y 53% entrenamiento.

4.2.4.1 Resultados con 20% de pruebas.

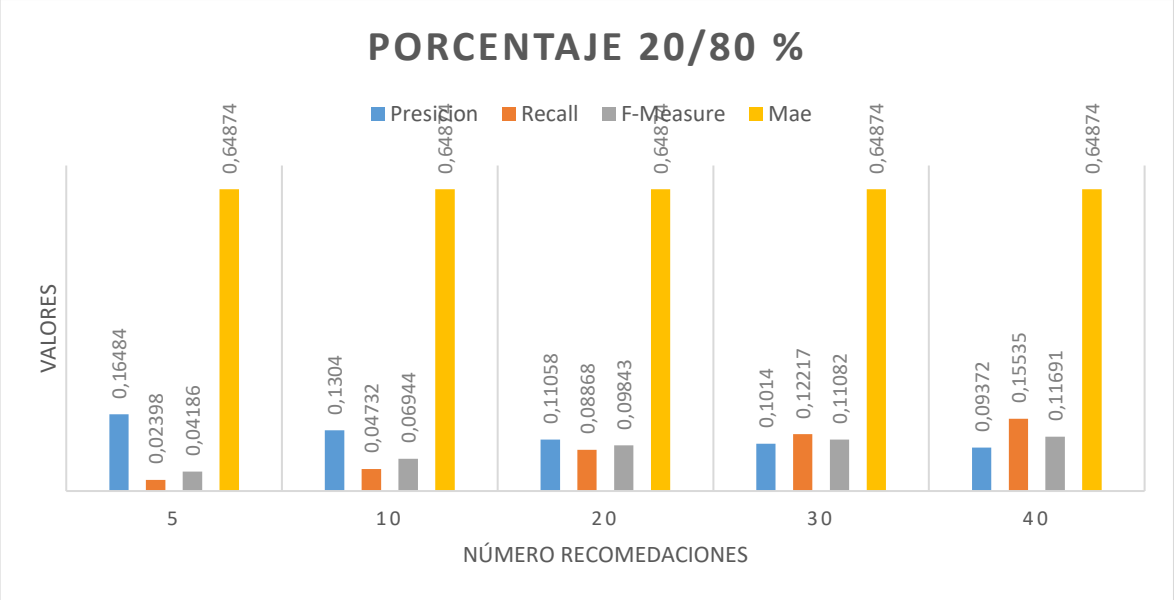


Figura 38. Resultados Split 20/80 %
Fuente. El autor.
Elaborado por. El Autor.

En la figura 38 se observa los resultados obtenidos, utilizando el 20 % de pruebas y el 80% para entrenar el modelo.

4.2.4.2 Resultados con 30% de pruebas.

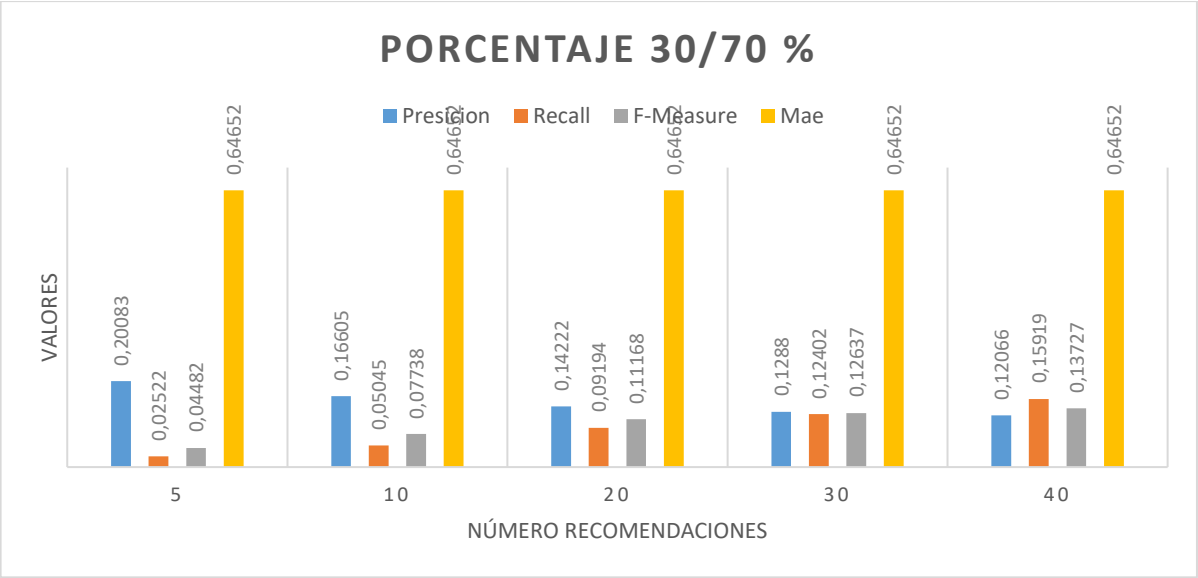


Figura 39. Resultados Split 30/80 %
Fuente. El autor.
Elaborado por. El Autor.

En la figura 39 se observa los resultados obtenidos, utilizando el 20 % de pruebas y el 80% para entrenar el modelo.

4.2.4.3 Resultados con el 47 % de Pruebas.

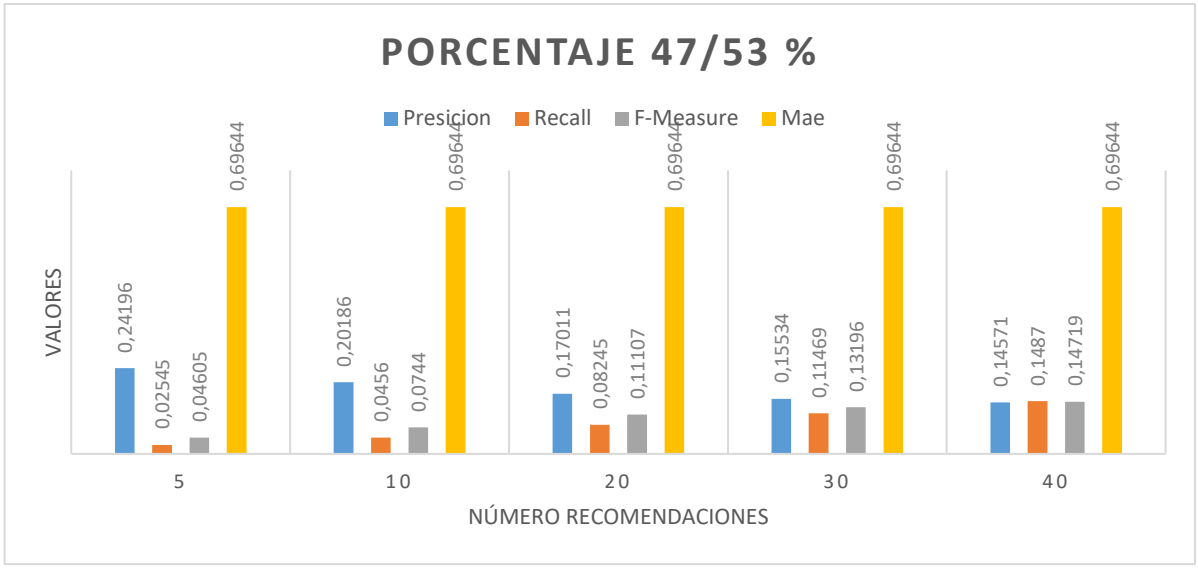


Figura 40. Resultados Split 47/53%
Fuente. El autor.
Elaborado por. El Autor.

En la figura 40 se observa los resultados obtenidos, utilizando el 47 % de pruebas y el 53% para entrenar el modelo.

En las figuras 38,39 y 40 se observa que, al incrementar los datos de pruebas, el sistema recomendador es mucho más eficiente, ya que los valores de las métricas aumentan.
Esto se debe que al tener un porcentaje en los datos de pruebas mayor. Los resultados que se presentan son más precisos.

4.2.5 Resultados según el Número de Vecinos Cercanos.

En esta sección se analizará los resultados obtenidos por el sistema recomendador según el número de k-vecinos cercanos.

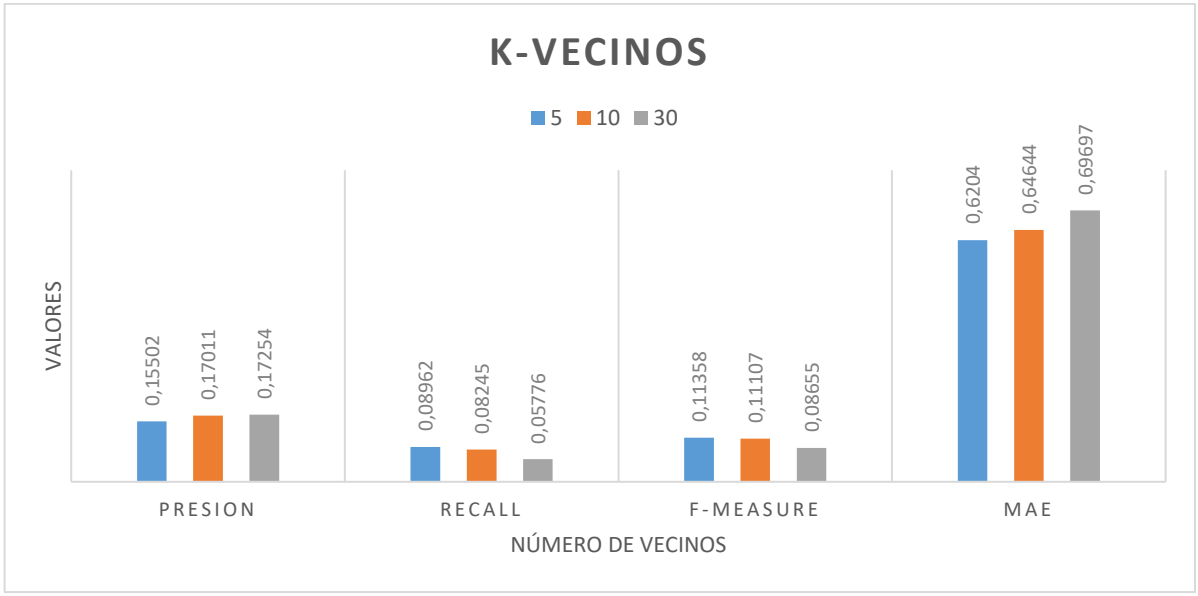


Figura 41. Resultados Vecinos Cercanos
Fuente. El autor.
Elaborado por. El Autor.

En la figura 41 se observan los resultados obtenidos con 5,10 y 30 números de k-vecinos. Dándoles a estos resultados un número de recomendaciones de 20.
El número de k-vecinos repercute en la obtención de las sugerencias que realiza el sistema recomendador por lo que, al aumentar el número de vecinos, las recomendaciones que realiza el sistema serán mucho más altas, por tanto, la Presión y Mae aumenta.
En cambio, el Recall disminuye por lo la fracción de documentos recuperados que son relevantes es mucho menor. Y en la métrica F-Mesure varía dependiendo de los valores de Precisión y Recall tengan.

4.3 Pruebas de Métrica Recall usando CTR

Los resultados obtenidos se basan en la utilización del método CTR (Collaborative Topic Regression) y el uso de dos tipos de recomendaciones:

- ✓ In-matrix prediction (predicción dentro de la matrix).
- ✓ Out-matrix prediction (predicción fuera de la matrix).

Para la experimentación de los datos se utilizó modelos generados de 25 número de tópicos.

4.3.1 Modelo de 25 Tópicos.

En esta grafica se presenta los resultados obtenidos utilizando CTR, en el cual muestra 3 parámetros que son:

- ✓ Método usado.
- ✓ Número de iteraciones.
- ✓ Métrica Recall.

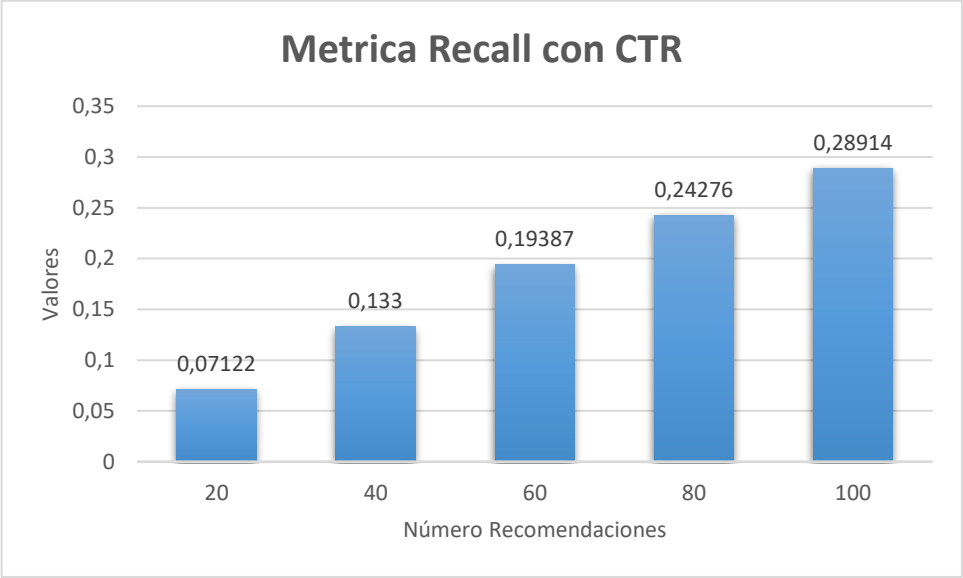


Figura 42. Resultados Recall usando in-matrix
Fuente. El autor.
Elaborado por. El Autor.

En la figura 42 se puede observar los resultados obtenidos con el método ctr-in-matrix, que al incrementar el número de iteraciones el Recall aumenta por lo la fracción de documentos recuperados que son relevantes es mucho mayor.

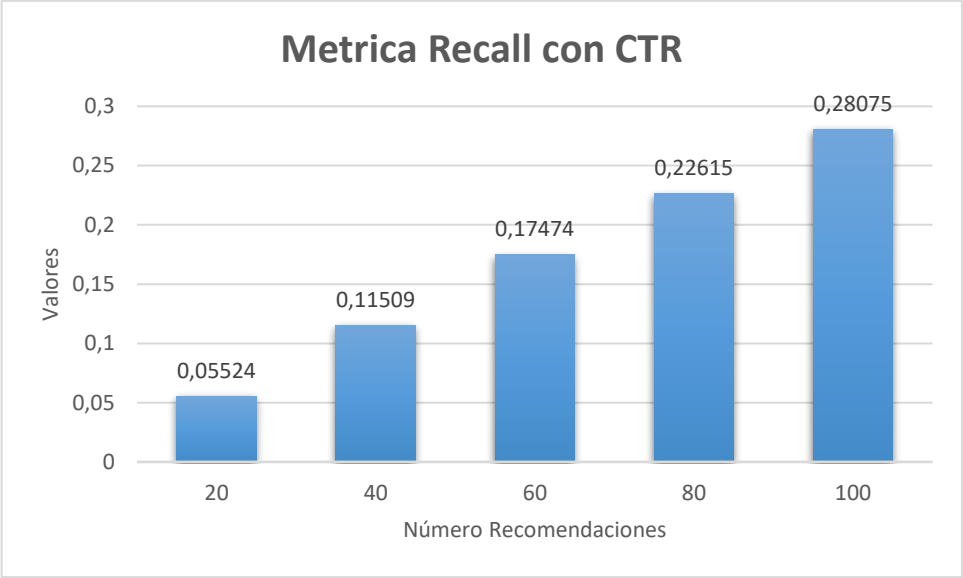


Figura 43. Resultados Recall usando out-matrix
Fuente. El autor.
Elaborado por. El Autor.

En la figura 43 se puede observar los resultados obtenidos con el método ctr-out-matrix, que al incrementar el número de iteraciones el Recall aumenta por lo la fracción de documentos recuperados que son relevantes es mucho mayor.

Tomando como referencia los dos casos anteriores se determina que el Recall aumenta conforme el número de recomendación es mayor.

Como cita (Said, 2013) cuanto más alto sea el valor de Recall más preciso será el sistema.

4.4 Comparación de resultados entre CF y CTR

Tomando como referencia a la Métrica de Recall se detalla a continuación una comparación de los resultados que se obtuvo con el modelo de filtrado colaborativo (CF) y un modelo hibrido llamado CTR.

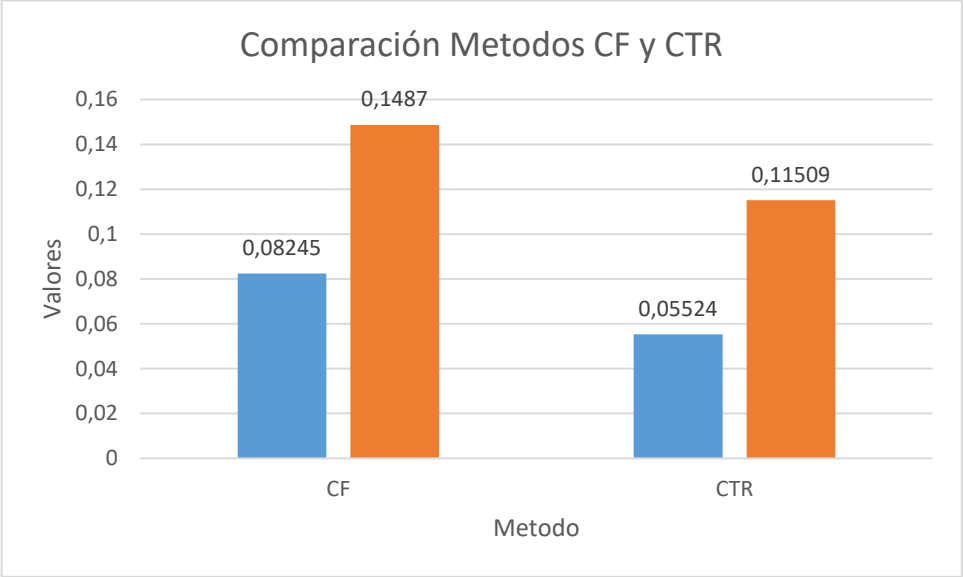


Figura 44. Resultados de CF y CTR
Fuente. El autor.
Elaborado por. El Autor.

En la figura 44 se observa los valores obtenidos con 20 y 40 número de recomendaciones. Donde los modelos tienen valores similares, es decir dependiendo del número de recomendaciones las métricas se comportan de forma similar. Por ejemplo, con 40 recomendaciones la métrica de Recall usando CF tiene un valor de 0.1487. Mientras que usando CTR tiene un valor de 0.11509.

CONCLUSIONES

Al finalizar el presente trabajo de titulación se ha podido concluir lo siguiente:

- ✓ Se implementó métricas de evaluación como: Precisión, Recall, F-Measure y Mae. Las cuales permiten analizar los resultados de algún sistema recomendador basado en filtrado colaborativo. Considerando que tan buenos son las sugerencias que brinda el sistema a los usuarios.
- ✓ Los datos desempeñan un papel muy importante en los resultados obtenidos por las Métricas, ya que las métricas de Precisión, Recall y F-Measure dependen de la capacidad que el sistema recomendador recupere ítems relevantes, es decir cuando hay muchos ratings altos (mayores de 3.5 en la escala de 1-5) el Recall sufre bajo desempeño. Caso contrario la Precisión aumentaría.
- ✓ Para que los valores de las métricas sean más equilibrados, se debe utilizar un número de recomendaciones más alto, este número depende de los datos que genere el sistema recomendador.
- ✓ Como se muestra en la sección 5.2.1 tabla 6, los valores obtenidos por las métricas son bajos, esto se debe a que el sistema recomendador utilizado, no es muy preciso al momento de dar sugerencias a los usuarios.
- ✓ Se concluye que al tener un mayor número de valores en los datos de pruebas utilizando el método Split, los resultados son más precisos. los porcentajes utilizados como más efectividad fueron:
    - o 70% para entrenamiento y 30% para pruebas.
    - o 47% para entrenamiento y 53% para pruebas.
- ✓ El método de k-vecinos es apropiado para los sistemas recomendadores utilizando técnicas de filtrado colaborativo, ya influye en la calidad de las predicciones que realiza el sistema.
- ✓ Los resultados de las métricas obtenidas a través de los métodos Cross-Validation y Split son similares.
- ✓ La métrica de precisión y Recall son inversamente proporcionales, es decir que a mayor número de recomendaciones la presión baja y el Recall aumenta, y viceversa.
- ✓ Con respecto a la Métrica Mae se puede concluir que, al incrementar el valor de la métrica el sistema realiza las predicciones defectuosamente. En caso de que la métrica Mae sea baja el sistema predice de forma correcta. Como se muestra en la sección 5.2.1 tabla 6, los valores obtenidos por Mae tienen un promedio de 0.69, esto quiere decir, que el sistema recomendador no es muy preciso al momento de efectuar las predicciones de los ratings reales.

- ✓ La implementación de la página Web está desarrollada para analizar resultados de datos que contengan ratings o calificaciones, que hayan dado los usuarios en la escala de 1-5, siendo 3.5 en valor minino para que ítems se lo considere relevante.

RECOMENDACIONES.

Al finalizar el presente trabajo de titulación de a podido recomendar lo siguiente:

- ✓ Al tener conjuntos de datos disponibles en la Web, es aconsejable poder usar modelos que permitan integrar los metadatos para poder enriquecer el sistema recomendador, de esta forma dar sugerencias a los usuarios más acertados.
- ✓ Al utilizar métricas de evaluación de resultados, es aconsejable tener un sistema recomendador eficiente, para que los valores obtenidos por las métricas sean altos, caso contrario dichos valores serán bajos por motivos que el sistema no sugiere ítems apropiados al usuario.
- ✓ El número de vecinos que utiliza el algoritmo KNN utilizado en esta Tesis, influye en la calidad de predicciones que realiza el sistema recomendador. No es recomendable utilizar muy pocos vecinos, porque no se alcanza a formar relaciones útiles, pero tampoco es bueno utilizar muchos vecinos ya que las relaciones se diluyen al ingresar mucho ruido incensario.
- ✓ Al utilizar el método de k-vecinos cercanos es aconsejable usar un número de vecinos impar, para evitar los botos empatados.
- ✓ Es recomendable usar técnicas de filtrado colaborativo donde se utiliza las valoraciones que realizan los usuarios sobre algunos ítems, de esta forma se puede dar sugerencias sobre los intereses de los usuarios, basándose en la recopilación de las preferencias o gustos de muchos usuarios, a esto se lo denomina colaboración.
- ✓ Se recomienda utilizar Dataset que contengan ratings sobre artículos o ítems, caso contrario la aplicaciones elaborada en esta tesis no podrá obtener los resultados de las métricas, como menciona (Rudloff, 2015) algunos Dataset que contienen estas características son: MovieLens, Jester, Lastfm, Bookcrossing, Netflix, Yahoo! Webscope.
- ✓ Se recomienda construir un analizador de resultados, que permita combinar técnicas de filtrado colaborativo y basado en contenido, como lo hace notar (Wang & Blei, 2011) generando un modelo hibrido(CTR), el cual permite usar el contenido de los artículos o ítems para dar recomendaciones más precisas a los usuarios.

BIBLIOGRAFÍA

Addis, A., & Addis, A. (2010). Study and Development of Novel Techniques for Hierarchical Text Categorization.

Adriana, L., & Pérez, A. (2013). Sistemas de Recomendación. Universidad Autónoma Metropolitana.

Augusto, M., Vásquez, C., Hugo, M., Huerta, V., Jaime, L., & Quispe, P. (2009). Procesamiento de lenguaje natural, 45–54.

Bai, H. V. N. and L. (2015). Cosine Similarity Metric Learning for Face Verification for Face Veri cation, (February). http://doi.org/10.1007/978-3-642-19309-5

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Knowledge-Based Systems Recommender systems survey. Knowledge-Based Systems, 46, 109–132. http://doi.org/10.1016/j.knosys.2013.03.012

Buitrago, O. R. (2012). Evaluación del Sistema de Recomendación de Patrones Pedagógicos (SRPP) en cursos de Geometría Euclidiana.

Cano, B., Eugenia, M., Enriquez, C., & Alexandra, M. (2015). Esquema de clasificación de información universitaria basado en NERC. Universidad Técnica Particular de Loja.

Castells, P., Vargas, S., & Wang, J. (2011). Novelty and Diversity Metrics for Recommender Systems : Choice , Discovery and Relevance.

Chai, T., Draxler, R. R., & Prediction, C. (2014). Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature, (2005), 1247–1250. http://doi.org/10.5194/gmd-7-1247-2014

Chapman, S. N. (2006). Planificación y control de la producción (Vol. 6).

Deldjoo, Y., Elahi, M., & Cremonesi, P. (2016). Using Visual Features and Latent Factors for Movie Recommendation, (July). http://doi.org/10.13140/RG.2.1.2375.0640

Díaz-ricardo, Y., & Becerra-garcía, R. A. (2014). El lenguaje de programación Python., 1–13.

F.O. Isinkaye, Y.O. Folajimi, B. A. O. (2015). Recommendation systems: Principles, methods and evaluation, Pages 261–273. Retrieved from https://doi.org/10.1016/j.eij.2015.06.005.

Fernandez Iglesias, D. (2014). Uso de tecnicas de recomendacion en sistemas. Universidad de Coruña.

Formoso, V. (2013). Tecnicas eficientes para la recomendacion de productos basadas en filtrado colaborativo. Universidad de Coruña.

González, Á. C. (2013). Recomendación de Contenidos Digitales basada en divergencias del lenguaje. Universidad Nacional de Educación a Distancia.

Guevara, A., & Rossi, C. (2014). Realidad Aumentada Y Sistemas De Recomendación Grupales, 23, 40–59.

Gunawardana, A. (2009). A Survey of Accuracy Evaluation Metrics of Recommendation Tasks, 10, 2935–2962.

Herlocker, J., Terveen, L. G., Ekstrand, B. M. D., Riedl, J. T., & Konstan, J. A. (2011). Evaluating collaborative filtering recommender systems, (January). http://doi.org/10.1561/1100000009

Hernando, A., Bobadilla, J., Ortega, F., & Tejedor, J. (2013). Incorporating reliability measurements into the predictions of a recommender system.

Hijikata, Y. (2014). Offline Evaluation for Recommender Systems.

Izadi, M. (2014). Unifying Inconsistent Evaluation Metrics in Recommender Systems, (Redd).

Kononenko, I. (2001). Machine Learning for Medical Diagnosis : History , State of the Art and Perspective, 1–25.

Kuusisto, F. C. (2015). Machine Learning for Medical Decision Support and Individualized Treatment Assignment.

Linyuan, L. (2012). Recommender Systems.

Lora, A. T. (2013). Predicción basada en vecinos.

Mcnee, S. M., & Konstan, J. A. (2006). Meeting User Information Needs in Recommender Systems.

Milano, P., Deldjoo, Y., Elahi, M., Cremonesi, P., & Bakhshandegan, F. (2016). How to Combine Visual Features with Tags to Improve Movie Recommendation Accuracy ?, (October). http://doi.org/10.13140/RG.2.2.21109.17122

Mortensen, M. (2007). Design and Evaluation of a Recommender System.

Nilsson, N. J. (2005). Introduction to Machine Learning. Machine Learning (Vol. 56). http://doi.org/10.1016/j.neuroimage.2010.11.004

Oracle, J. (2016). Que es Java. Retrieved from https://www.java.com/es/about/whatis_java.jsp

Ordóñez, S., & González, O. (2004). Sistemas de recuperación de información, 57–62.

Parra, D. (2012). Métricas de Evaluación. Retrieved from http://dparra.sitios.ing.uc.cl/classes/recsys-2014-2/Recsys01/clase4_metricas.html#1

Parra, D., & Sahebi, S. (2013). Recommender Systems : Sources of Knowledge and Evaluation Metrics, 149–175.

Pérez-Planells, & Delegido. (2015). Análisis de métodos de validación cruzada para la obtención robusta de parámetros biofísicos, 55–65. Retrieved from http://roderic.uv.es/handle/10550/49965

Rodriguez, D. R. (2013). Aprendizaje Automático - Machine Learning. Retrieved from https://www.dc.uba.ar/materias/aa/2013/cuat1

Rodríguez, K. C. B., & Sayay, J. B. Á. (2016). Análisis De Las Tecnologías Axis 2 Y Metro 2.0 Para El Desarrollo De Servicios Web Seguros En Java, Aplicado Al Módulo De Integración Y Reportes Del Sistema De Evaluación Docente De La Unach.

R-Tools Technology Inc. (2015). RStudio.

Said, A. (2013). Evaluating the Accuracy and Utility of Recommender Systems.

Saini, I. (2013). QRS detection using K -Nearest Neighbor algorithm (KNN) and evaluation on standard ECG databases. Journal of Advanced Research, 4(4), 331–344. http://doi.org/10.1016/j.jare.2012.05.007

Sandoval, S. V. (2012). Novelty and Diversity Enhancement and Evaluation in Recommender Systems.

Santana, J. S. (2014). El arte de programar en R: un lenguaje para la estadística.

Schapire, R. (2013). Theoretical Machine Learning, 1–7.

Smola, A., & Vishwanathan, S. V. N. (2014). Introduction to machine learning. Methods in Molecular Biology (Vol. 1107). http://doi.org/10.1007/978-1-62703-748-8-7

Tamayo, S. C. (2012). Diseño y validadción de modelos para Sistemas de Recomendación. Universidad de Granada.

Vargas, S., & Castells, P. (2011). Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems.

Vera, J., & Mamani, A. O. (2015). Modelo de sistema de recomendación de Objetos de Aprendizaje en dispositivos móviles , caso : Desarrollo del pensamiento computacional, 730–734.

Wang, C., & Blei, D. M. (2011). Collaborative Topic Modeling for Recommending Scientific Articles, pp. 448–456.

Zwinderman, T. J. C. ·. A. H. (2013). Machine Learning in Medicine. New York: Springer.

RUDLOFF, N. I. T. (2015). Sistemas De Recomendación Basados En Métodos De Filtrado Colaborativo.

ANEXOS

1 MANUAL DE USUARIO DE LA APLICACIÓN WEB ANALIZADOR DE RESULTADOS

Introducción

La aplicación de Análisis de Resultados está pensada y diseñada para facilitar a los usuarios que necesiten analizar los resultados de algún sistema recomendador utilizando Filtrado Colaborativo. Actualmente la aplicación permite ingresar archivos planos con las características necesarias, y poder visualizar los resultados obtenidos con 4 métricas de evaluación como:

- ✓ Precisión
- ✓ Recall
- ✓ F-Measure
- ✓ Mae

Interfaz Principal

El frontal principal está compuesto por el menú en la parte superior, permitiendo acceder a todas las funcionalidades de la aplicación

También se detalla el objetivo principal de la página Web, y un enlace donde se encuentra una Data Set obtenido por un sistema recomendador usando filtrado colaborativo.

En la sección de la derecha están divididas las métricas según al grupo al que pertenecen.

Cargar Data Set

En esta sección se pueden cargar los archivos necesarios para el funcionamiento de las métricas de Precisión, Recall, F-Measure y Mae.

Ratings: estos archivos están compuestos de tres campos: id del usuario, id del ítem y la calificación de un artículo.

Estos ratings son obtenidos por medio de la data de prueba que realiza el método de validación de datos que se mencionó anteriormente. Es decir, por cada id de usuario que se encuentra en la data, se obtiene los ítems con sus respectivas calificaciones que haya visto.

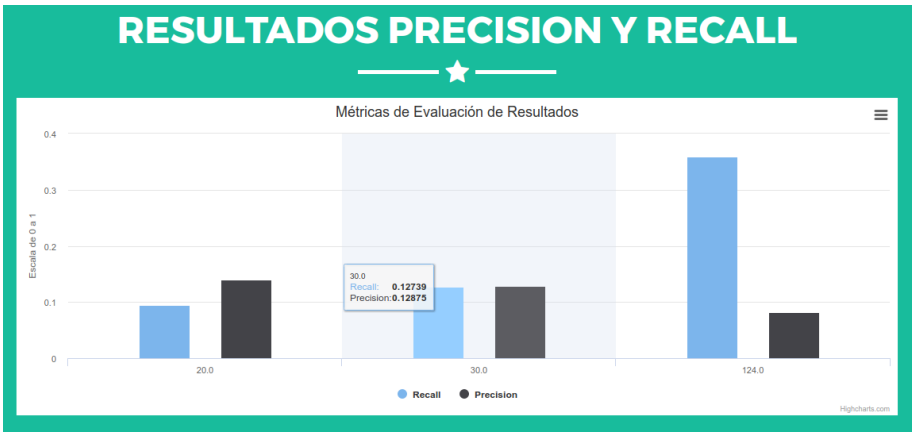
Recomendaciones: estos archivos están compuestos por 4 campos los cuales son: id usuario, id ítems, recomendación y predicciones.

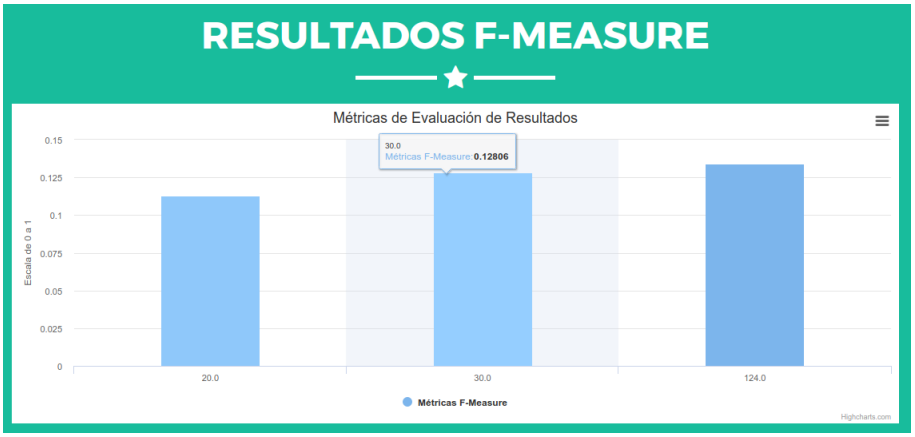
Estos archivos son generados por el sistema recomendador de k-vecinos cercanos basado en ítems en cual esta descrito anteriormente

Una vez cargados los archivos de Rating y Recomendaciones descritos anteriormente, se elegí el número de recomendaciones que se desea ingresar para el análisis de los resultados, los valores que se pueden elegir son: 5, 10, 20, 30, 40 y otro valor que quiera ingresar el usuario.

Finalmente se da clic en **analizar resultados** y listo.

Resultados





En esta sección se genera una gráfica de barras donde se muestra cada una de las métricas son sus valores obtenidos según los archivos ingresados por el usuario.

En el eje de las X se muestran los nombres de las métricas con sus valores correspondientes. En el eje de las Y se establece una escala que se incrementa conforme las métricas aumentan sus valores.

Ayuda

En esta sección observa 3 secciones que ayudan a usar la aplicación y conocer el formato correcto que deben contener los archivos que utilizan las métricas. Solo es necesario dar clic en una de las tres imágenes y listo

Cargar DataSet

En esta sección se muestra algunos ejemplos de archivos con sus características de deben contener.

Ejemplos de Resultados Métricas

En esta sección ejemplos de Resultados de las Métricas, dependiendo de los archivos ingresados y el número de recomendaciones que seleccione el usuario.

Uso y funcionamiento de la aplicación

En esta sección se detallan las características de la aplicación y todos los componentes que tiene. También se muestra una pequeña explicación de cada una de las métricas y las ecuaciones que utilizan.

2 CODIGO UTILIZADO EN LA IMPLEMETACIÓN DE LAS MÉTRICAS.

2.1 Métrica Precisión.

```
def presicion(ListaLista,usuarioLista,numeroRecomendaciones):
    #declaracion de variables
    #verdaderosPositivos=0
    usuariosRelevantes=0
    verdaderosPositivosT=0
    #longitudLista=0
    presicion=0
    contador=0
    #obtinene numero de recomedaciones
    numRecomendaciones = numeroRecomendaciones
    inicio=time.time()
    #recorre la lista de lista de usuarios y listas
    for l, u in zip(ListaLista,usuarioLista):
        #recorre cada archivo de los usuarios
        for u1 in u:
            #recorre cada archivo de lista
            verdaderosPositivos=litarecomedaciones(u1,l,numeroRecomendaciones)
            verdaderosPositivosT=verdaderosPositivosT+verdaderosPositivos
        presicion1 = (verdaderosPositivosT) / numRecomendaciones
        presicion=presicion+presicion1
        contador = contador + 1
        verdaderosPositivosT = 0
    resultado=presicion/contador
    final=time.time()
    tiempo=round(final-inicio,0)
    print("Numero de Recomendaciones: ",numRecomendaciones)
    print("Tiempo de Ejecucion es: "+str(tiempo)+" segundos")
    print("Metrica PRESICION:",round(resultado,5))
    return resultado
```

2.2 Métrica Recall.

```
def recall(ListaLista,usuarioLista,numeroRecomendaciones):
    #declaracion de variables
    verdaderosPositivosT=0
    usuariosRelevantes=0

    recall=0
    contador=0
    contad=0
    #obtinene numero de recomedaciones
    numRecomendaciones = numeroRecomendaciones
    inicio=time.time()
    #recorre la lista de lista de usuarios y listas
    for l, u in zip(ListaLista,usuarioLista):
        #recorre cada archivo de los usuarios
        for u1 in u:
            if float(u1[2]) >= 3.5:
                usuariosRelevantes = usuariosRelevantes + 1
                contad=contad+1
                verdaderosPositivos=litarecomedaciones(u1,l,numeroRecomendaciones)
                verdaderosPositivosT=verdaderosPositivosT+verdaderosPositivos
        recall1=verdaderosPositivosT / usuariosRelevantes
        #print("verdaderos",verdaderosPositivos)
        #print("rele",usuariosRelevantes)
        recall=recall+recall1
        #print("recall1",recall1)
        contador=contador+1
        verdaderosPositivosT=0
        usuariosRelevantes=0
    resultado=recall/contador
    final=time.time()
    tiempo=round(final-inicio,0)
    print("Numero de Reomendaciones: ",numRecomendaciones)
    print("Tiempo de Ejecucion es: "+str(tiempo)+" segundos")
    print("Metrica RECALL:",round(resultado,5))
    return resultado
```

2.3 Métrica F-Measure.

```
def measure(ListaLista,usuarioLista,numeroRecomendaciones):
    #declaracion de variables
    verdaderosPositivosT=0
    usuariosRelevantes=0
    #longitudLista=0
    recall=0
    presicion=0
    measure=0
    contador=0
    #obtinene numero de recomedaciones
    numRecomendaciones = numeroRecomendaciones
    inicio=time.time()
    #recorre la lista de lista de usuarios y listas
    for l, u in zip(ListaLista,usuarioLista):
        #recorre cada archivo de los usuarios
        for u1 in u:
            if float(u1[2]) >= 3.5:
                usuariosRelevantes = usuariosRelevantes + 1

            #recorre cada archivo de lista
            verdaderosPositivos=litarecomedaciones(u1,l,numeroRecomendaciones)
            verdaderosPositivosT=verdaderosPositivosT+verdaderosPositivos
        recall1=(verdaderosPositivosT) /usuariosRelevantes
        recall=recall+recall1
        presicion1 = (verdaderosPositivosT) / numRecomendaciones
        presicion = presicion + presicion1
        contador=contador+1
        verdaderosPositivosT=0
        usuariosRelevantes=0
    resultadoRecall=recall/contador
    resultadoPresicion=presicion/contador
    final=time.time()
    tiempo=round(final-inicio,0)
    print("------------Resultados Finales-----------------")
    print("Numero de Reomendaciones: ",numRecomendaciones)
    print("Tiempo de Ejecucion es: "+str(tiempo)+" segundos")
    measure=2*((resultadoPresicion*resultadoRecall)/(resultadoPresicion+resultadoRecall))
    print("Metrica F-Measure: ",round(measure,5))
    return measure
```

2.4 Métrica Mae.

```
def MetricaMae(lista):
    #inicializacion de valiables
    ListaLista=lista[1]
    usuarioLista=lista[0]
    errorAcumulado=0.0
    numeroPredicciones=0
    #implemetar ecuacion Mae
    rating=[]
    prediccion=[]
    for lis in ListaLista:
        for l in lis:
            if l[3] != '':
                prediccion.append(float(l[3]))
    for user in usuarioLista:
        for u in user:
            if u[2] != '':
                rating.append(float(u[2]))

    for predic, ratin  in zip(prediccion, rating):
        if ratin != None and ratin != "NaN":
            error = math.fabs(ratin- predic)
            errorAcumulado = errorAcumulado + error
            numeroPredicciones = numeroPredicciones + 1
            mae = errorAcumulado / float(numeroPredicciones)
    print("Metrica Mae: ", round(mae, 5))
    f=open('static/resultados/resultadosMae.csv','w')
    f.write(str(round(mae, 5)))
    f.close()
```

2.5 Control de errores JavaScrip.

```
</script>
<script type="text/javascript">
  var formatoUsuario = '{{ formatoUsuario }}';
  var separador = '{{ separador }}';
  var columnas = '{{ columnas }}';
  var archivos = '{{ archivos }}';
  var top_n = '{{ top_n }}';
  if (formatoUsuario == 'False') {
    alert('Archivos incorrectos. Deben estar en formato csv, data, txt.');
  }
  if (separador == 'False') {
    alert('Separadores incorrectos. Los separadores permitidos son: coma(,), punto y coma(;), tabulador(  ) y dos puntos(:).
      ');
  }
  if (columnas == 'False') {
    alert('El número de columnas es incorrecto. El archivo Ratings debe tener 3 columnas, y el archivo Recomedaciones debe
      tener 4 columnas.');
  }
  if (archivos == 'False') {
    alert('Es necesario que seleccione los dos archivos.');
  }
   if (top_n == 'False') {
    alert('El número de recomendaciones que ha seleccionado es mayor al top N permitido de sus datos. Y el mínimo top N
      debe ser 5.');
  }
</script>
```