UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA,

INGENIERÍA Y DISEÑO GRADO EN

INFORMÁTICA

Sistemas Inteligentes

Documentación final del proyecto

Practica Sistemas Inteligentes

Grupo 2

Marchal Mallavibarrena Miguel

Ortega Núñez Alejandro

Dirigido por

Borja Monsalve Piqueras

CURSO 2024-2025

Título Proyecto

Grupo2 / Miguel Marchal Mallavibarrena

Alejandro Ortega Núñez

TÍTULO: Practica sistemas Inteligentes

TITULACIÓN: Grado en ingenieria informatica

DIRECTOR/ES DEL PROYECTO: Marchal Mallavibarrena Miguel

y Ortega Núñez Alejandro

FECHA: Enero de 2025

RESUMEN

Este proyecto desarrolla un sistema de recomendación de películas que permite a los usuarios descubrir contenido relevante y personalizado basado en sus intereses. El enfoque combina la recomendación por similitud de sinopsis, utilizando técnicas de procesamiento de lenguaje natural como TF-IDF y similitud del coseno, y un sistema basado en el perfil del usuario generado a partir de sus valoraciones. El sistema integra una interfaz gráfica de usuario intuitiva para facilitar la interacción, incluyendo funcionalidades de búsqueda, votación y recomendaciones.

Se utilizó un dataset estructurado para capturar información sobre películas y usuarios, complementado con técnicas de Web Scraping para enriquecer los datos visuales con pósters de películas. El proyecto demostró escalabilidad y robustez al manejar grandes volúmenes de datos, gestionando entradas inválidas de manera eficiente.

Entre los principales logros destaca el diseño e implementación de algoritmos que aseguran recomendaciones precisas y relevantes, adaptadas a las preferencias del usuario. Este sistema combina métodos avanzados de análisis de texto con una experiencia de usuario fluida, abriendo la posibilidad de futuras mejoras y expansiones.

Palabras clave: Recomendación, TF-IDF, Similitud del Coseno, Procesamiento de Lenguaje Natural, Web Scraping, Perfil de Usuario.

ABSTRACT

This project develops a movie recommendation system that enables users to discover relevant and personalized content based on their interests. The approach combines recommendations by synopsis similarity, using natural language processing techniques such as TF-IDF and cosine similarity, with a user profile-based system generated from their ratings. The system integrates an intuitive graphical user interface to facilitate interaction, including search, voting, and recommendation functionalities.

A structured dataset was used to capture information about movies and users, complemented by web scraping techniques to enrich visual data with movie posters. The project demonstrated scalability and robustness by efficiently handling large volumes of data and managing invalid inputs.

Key achievements include the design and implementation of algorithms that ensure accurate and relevant recommendations tailored to user preferences. This system combines advanced text analysis methods with a seamless user experience, paving the way for future improvements and expansions.

Keywords: Recommendation, TF-IDF, Cosine Similarity, Natural Language Processing, Web Scraping, User Profile.

Índice

1.Introducción

Descripción del problema a resolver

En el mundo del cine, elegir una película para ver puede resultar complicado debido a la gran cantidad de opciones disponibles. Con nuestra aplicación buscamos desarrollar una página web que ayude a los usuarios a descubrir películas relevantes y personalizadas basándonos en sus intereses y preferencias. Para ello, proponemos un sistema de recomendación que combina la similitud de sinopsis utilizando técnicas PLN y un sistema de votaciones que permite recomendar películas similares a las calificadas altamente por los usuarios tomando en cuenta el género de la película, el autor y la sinopsis de esta.

Enfoque seguido

El proyecto aborda la recomendación de películas utilizando dos enfoques principales:

1. **Recomendación basada en sinopsis:** Se utiliza el modelo de espacio vectorial con TF-IDF (Term Frequency-Inverse Document Frequency) y el cálculo de la similitud del coseno para comparar la sinopsis de las películas.
2. **Recomendación basada en puntuaciones:** A través de las votaciones realizadas por los usuarios, se calculan recomendaciones considerando las columnas de sinopsis, autor y género. Las películas con mayor similitud a aquellas calificadas con 5 estrellas que son las de mallor valor que son las que se tendrán más encuenta, luego le seguirán las de 4 y las de 3 posteriormente(las de 1 y 2 no se tienen en cuenta ya que son muy pequeñas) .

Modelo espacio vectorial

El modelo espacio vectorial es una representación matemática que describe documentos (en este caso, sinopsis de películas) como vectores en un espacio multidimensional. Cada dimensión corresponde a un término único del vocabulario, y el peso asociado a cada término en un documento es determinado por su frecuencia (TF) y su relevancia inversa en la colección (IDF). Este enfoque permite cuantificar la similitud entre documentos mediante métodos geométricos.

TF-IDF

El método TF-IDF mide la importancia de una palabra en un documento dentro de un conjunto de documentos.

Donde:

- Utilizamos la frecuencia del término en el documento .
- Y se mide la rareza del término en el conjunto de documentos , siendo el número total de documentos(en este caso películas).

Similitud del coseno

La similitud del coseno calcula el ángulo entre dos vectores en el espacio vectorial el resultado solo puede estar entre [0,1] . Su fórmula es la siguiente:

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}| * |\vec{b}|}$$

Donde a sería el ítem 1(Ejemplo) y b sería el ítem 2(Ejemplo), se multiplican los vectores , esto sería el numerador, y el denominador lo formarían el módulo del ítem 1 multiplicado por el módulo del ítem 2.

Dominio de aplicación

El proyecto está diseñado para usuarios interesados en encontrar películas relevantes en función de su contenido y sus preferencias personales. La aplicación esta enfocada a usuarios que quieran encontrar nuevas películas relacionadas con sus gustos y no solo eso sino que también les permite poder explorar películas nuevas fuera de sus gustos con sinopsis parecidas.

Repositorios de películas

El csv de películas utilizado en este proyecto contiene las siguiente información:

- **Título de la película:** Nombre de la película.
- **Año de lanzamiento:** El año en que la película fue estrenada.
- **Sinopsis:** Una descripción narrativa de la trama de la película.
- **Puntuación de críticos y usuarios:** Valoraciones dadas por críticos profesionales y el público general.
- **Consenso:** Un resumen de las opiniones predominantes sobre la película.

- **Número total de reseñas y valoraciones:** Indicadores de la popularidad y nivel de interacción de la película.
- **Género:** Clasificación de la película (por ejemplo, acción, drama, comedia).
- **Director, productor y guionista:** Detalles sobre el equipo creativo principal.
- **Fechas de lanzamiento:** Tanto en cines como en plataformas de streaming.
- **Recaudación en taquilla:** Información financiera sobre el desempeño de la película.
- **Duración:** Tiempo total de la película.
- **Compañías de producción y mezcla de sonido:** Datos técnicos adicionales.
- **Póster e imágenes promocionales:** Enlaces a los materiales visuales asociados con cada película.

Esta base de datos proporciona los datos necesarios para implementar tanto el sistema de recomendaciones basadas en sinopsis como el sistema basado en votaciones. Los datos han sido obtenidos de una combinación de fuentes, como bases de datos públicas de cine, asegurando un equilibrio entre calidad y cantidad de información.

Repositorio de usuarios

El repositorio de usuarios es una csv que contiene información clave sobre los usuarios registrados en la plataforma. Este repositorio incluye las siguientes columnas:

- **Nombre de usuario:** Un identificador para cada usuario registrado en la plataforma. Es utilizado para autenticar y personalizar las recomendaciones.
- **Contraseña:** La clave de acceso del usuario para iniciar sesión en la plataforma. Para garantizar la seguridad de los datos, las contraseñas se almacenan de forma cifrada.
- **Votaciones:** Esta columna contiene un array list de objetos que representan las valoraciones realizadas por el usuario sobre las películas. Cada objeto en el array tiene el siguiente formato:
    - **Título de la película:** El nombre de la película que el usuario ha valorado.
    - **Rating:** La calificación dada por el usuario, que puede variar de 1 a 5 estrellas.

Este repositorio es esencial para el sistema de recomendaciones basado en votaciones, ya que permite asociar las valoraciones de cada usuario con las películas correspondientes y, a su vez, utilizar estos datos para calcular recomendaciones personalizadas en función de las puntuaciones obtenidas.

Web Scraping

Web Scraping es una técnica utilizada para extraer información de sitios web de manera automática. En este proyecto, lo utilizamos para recopilar las carátulas de las películas.

El proceso de **Web Scraping** se realiza con dos herramientas estas son BeautifulSoup y Selenium.

1. **BeautifulSoup**:
    - **BeautifulSoup** es una biblioteca de Python que facilita la extracción de datos de archivos HTML y XML. Una vez que se ha descargado una página web, **BeautifulSoup** permite analizar su contenido y extraer información de manera eficiente.
2. **Selenium**:
    - **Selenium** es una biblioteca de Python que permite simular la interacción del usuario con la página, como hacer clic en botones, desplazar la página, y esperar a que el contenido cargue completamente antes de extraer los datos. Esto es crucial cuando se desea acceder a contenido que no está disponible inicialmente en el HTML de la página.

Estado del arte

Ya se han creado múltiples recomendadores de películas que abordan la recomendación de películas de manera innovadora. Algunos ejemplos notables incluyen:

1. **Netflix**:
   - Netflix es uno de los pioneros en la recomendación personalizada de contenido. Utiliza un sistema que combina **filtrado colaborativo** (basado en el comportamiento de usuarios similares) y **modelos de contenido** (analizando características de las películas, como género, actores y sinopsis). Este enfoque permite a Netflix ofrecer recomendaciones altamente personalizadas y adaptadas a los gustos cambiantes de los usuarios.
2. **IMDb**:
   - IMDb utiliza un enfoque centrado en las **calificaciones** y **búsquedas de los usuarios**. Además del filtrado colaborativo, el sistema también tiene en cuenta la popularidad y las características de la película (género, sinopsis, etc.). Este sistema ofrece recomendaciones en función de las preferencias del usuario, pero también sugiere contenido basado en las tendencias y la popularidad general de las películas.
3. **Movielens**:
   - Movielens, se centra en la recomendación personalizada mediante **filtrado colaborativo**. Los usuarios reciben recomendaciones basadas en sus interacciones anteriores y en el comportamiento de otros usuarios con gustos similares. Movielens se ha convertido en una plataforma experimental importante para estudiar y evaluar algoritmos de recomendación.
4. **Letterboxd**:
   - Letterboxd es una red social para cinéfilos que permite a los usuarios registrar y puntuar películas. Genera recomendaciones basadas en las **afinidades de usuarios similares** y en los **patrones de puntuación**. A diferencia de otras plataformas, Letterboxd pone un fuerte énfasis en la comunidad, permitiendo a los usuarios interactuar y compartir opiniones sobre películas.

2.Descripción del Progreso

2.1 Descripción de los pasos a seguir:

El desarrollo del sistema de recomendación de películas sigue varios pasos clave, que incluyen la creación de la interfaz de usuario y la implementación de los cálculos para generar recomendaciones.

1. **Creación de la interfaz de usuario**
   Se diseña la estructura de la aplicación, creando las ventanas necesarias: una para el registro e inicio de sesión, otra para la búsqueda y visualización de la sinopsis de las películas, y una ventana para mostrar las recomendaciones. Cada acción del usuario requiere su propia ventana para interactuar de manera eficiente.
2. **Creación del sistema de registro e inicio de sesión**
   Implementación de un sistema de autenticación donde los usuarios pueden registrarse con su nombre de usuario y contraseña.
3. **Creación del sistema de recolección de votaciones**
   Los usuarios pueden calificar las películas con un rating entre 1 y 5 estrellas. Estas calificaciones se almacenan junto con el título de la película y el usuario correspondiente en el repositorio de usuarios, lo cual lo utilizaremos para generar las recomendaciones personalizadas
4. **Creación de la funcionalidad para calcular recomendaciones basadas en sinopsis**
   Se utiliza el modelo TF-IDF para analizar las sinopsis de las películas y calcular la similitud del coseno entre ellas. Este proceso permite recomendar películas con sinopsis similares, basándose en el contenido textual de las sinopsis.
   También generamos otra para las valoraciones personalizadas para que se calculen a través del Autor, género y sinopsis.
5. **Creación de la lógica para calcular recomendaciones basadas en puntuaciones**
   Se analizan las películas con mejores valoraciones del usuario (por ejemplo, 5, 4 y 3 estrellas) y se buscan otras películas similares en género, director y sinopsis. Este proceso ayuda a generar recomendaciones personalizadas, dando preferencia a las películas más valoradas.
6. **Creación de la ventana para mostrar las recomendaciones**
   Las recomendaciones generadas se muestran al usuario en una ventana de recomendaciones. Estas se actualizan cada vez que el usuario califica una nueva película.

7. **Creación de un sistema de interacción continua**
   Las recomendaciones se recalculan automáticamente cada vez que el usuario califica una nueva película. Además, se ofrece la posibilidad de explorar películas relacionadas basadas en la similitud de sinopsis, fomentando el descubrimiento de nuevas opciones fuera del historial de calificaciones.

2.2 Organización del código:

1. Organización General del Código:

Nuestro código se organiza principalmente en dos partes:

- Gestores: Lógica de la aplicación (GestorPeliculas, GestorUsuarios y GestorVentanas).
- Vista: Interfaz gráfica que maneja la interacción con el usuario (VistaRecomendaciones,VistaVotaciones,VistaLogin,VistaPrincipal).

Este enfoque separa la lógica de negocio (gestores) de la presentación (vistas), lo que es una buena práctica para mantener la escalabilidad, mantenimiento y claridad en el código.

2. Clases y Funciones Principales:

Clase GestorVentanas:

La clase GestorVentanas gestiona todas las vistas de la aplicación y la interacción entre ellas. Actúa como el controlador principal que coordina las acciones entre las distintas pantallas y las funcionalidades del programa.

- **Responsabilidad**: Gestiona la navegación entre las diferentes vistas (Login, Principal, Votaciones, Registro, etc.) y administra la creación de las vistas necesarias para mostrar la interfaz al usuario.
- **Composición**: Utiliza las vistas correspondientes para cada tipo de acción (login, registro, votaciones, etc.), gestionando la creación de instancias de esas vistas y su transición.

Métodos principales:

- __init__(self): Inicializa el gestor de ventanas y las vistas que serán utilizadas en la aplicación.
    - Crea instancias de las vistas necesarias: VistaLogin, VistaPrincipal, VistaVotaciones, etc.
    - Inicializa el GestorPeliculas, que se usa para gestionar las películas en la aplicación.
- set_user_info(self, user_id, username): Establece la información del usuario actual (ID y nombre de usuario) que se utilizará en varias vistas, como VistaVotaciones y VistaMisValoraciones.
- mostrar_login(self): Muestra la ventana de inicio de sesión (VistaLogin).
- mostrar_principal(self): Muestra la ventana principal (VistaPrincipal), que puede ser la pantalla principal una vez el usuario haya iniciado sesión.
- mostrar_votaciones(self): Muestra la ventana de votaciones (VistaVotaciones), donde los usuarios pueden votar por las películas.
- mostrar_mis_valoraciones(self, username): Muestra la ventana donde los usuarios pueden ver sus valoraciones previas de películas (VistaMisValoraciones).
- mostrar_registro(self): Muestra la ventana de registro de nuevos usuarios (VistaRegistro).
- mostrar_sinopsis(self, detalles_pelicula): Muestra la ventana de sinopsis de una película específica (VistaSinopsis), mostrando detalles como la descripción y más información sobre una película seleccionada.
- mostrar_recomendaciones(self, gestor_peliculas, username): Muestra la ventana de recomendaciones (VistaRecomendaciones), ofreciendo sugerencias de películas basadas en las valoraciones anteriores del usuario.
- _cambiar_ventana(self, nueva_ventana): Método privado que cierra todas las ventanas visibles y abre una nueva ventana proporcionada como parámetro. Esto permite la navegación fluida entre vistas.
- ejecutar(self): Inicia la aplicación mostrando inicialmente la ventana de inicio de sesión (VistaLogin), y luego ejecuta el ciclo de eventos de la aplicación con app.exec_().

Clases de las Vistas:

Las clases de vista son responsables de gestionar la interfaz gráfica de la aplicación. Cada vista tiene su propia clase y se encargan de mostrar una parte específica de la aplicacion.

Las vistas son :

- **VistaLogin**: Gestiona la pantalla de inicio de sesión. Permite a los usuarios autenticarse.
- **VistaPrincipal**: Pantalla principal, que se muestra después de que el usuario inicia sesión. Muestra un menú principal y un buscador de películas el cual te permite seleccionar una película para ver su sinopsis en VistaSinopsis.
- **VistaRecomendaciones**: Pantalla que ofrece recomendaciones personalizadas de películas basadas en las votaciones anteriores del usuario.
- **VistaRegistro**: Vista donde se pueden crear nuevos usuarios.
- **VistaSinopsis**: Muestra información detallada de una película específica, como su sinopsis y otros detalles, también ofrece unas recomendaciones basadas en la sinopsis.
- **VistaVotaciones**: Permite a los usuarios votar películas, también contiene un buscador por si se quiere votar una película en específico.
- **VistaMisValoraciones**: En esta vista se encuentra una tabla con las distintas puntuaciones que se le ha dado a las películas.

Clase GestorPeliculas :

La clase GestorPeliculas maneja la lógica de las recomendaciones y votaciones de las películas . Como se detalló anteriormente, esta clase es utilizada por el GestorVentanas para gestionar las operaciones como la búsqueda de películas, la obtención de recomendaciones y las valoraciones.

Métodos principales:

- **buscar_peliculas(self, nombre):** Realiza la búsqueda de películas en base al nombre proporcionado.
- **votar_pelicula(self, username, pelicula, valoracion):** Envía una valoración para una película.
- **peliculas_al_azar(self):** Devuelve una lista de películas seleccionadas al azar.
- **_calcular_similitudes(self):**Calcula la matriz de similitud de coseno utilizando las sinopsis de las películas.

- **_calcular_similitudes_recomendaciones(self):**Genera una matriz de similitud de coseno tomando los valores de sinopsis, director y género.
- **recomendar_peliculas(self, title):** Recomienda películas similares a la pelicula solicitada en función de la sinopsis.
- **recomendar_peliculas_por_usuario(self, username):** Genera recomendaciones personalizadas para cada usuario según sus valoraciones previas. Le da a las películas mejor votadas y devuelve una lista de recomendaciones ordenadas por la similitud, Para calcular esta similitud tiene en cuenta la sinopsis, el director y el género.

Clase GestorUsuarios :

La clase GestorUsuarios gestiona los datos relacionados con los usuarios, como el registro, almacenamiento de datos en el archivo usuarios.csv.

Métodos principales:

- **registrar_usuario(self, username, password)**:
  Registra un nuevo usuario en el sistema, validando que el nombre de usuario no esté vacío ni duplicado. Los datos se almacenan en el archivo CSV.
- **validar_usuario(self, username, password)**:
  Valida las credenciales de inicio de sesión de un usuario verificando su existencia y autenticidad en el archivo CSV.
- **guardar_datos(self)**:
  Almacena los datos de los usuarios en el archivo CSV para persistencia.
- **obtener_usuario_por_id(self, user_id)**:
  Recupera la información de un usuario específico utilizando su ID.
- **obtener_usuarios(self)**:
  Devuelve una lista con todos los usuarios registrados en el sistema.

3. Librerías utilizadas :

- **pandas:** Manejo y análisis de datos, nos permite manipular y trabajar con los csv
- **PyQt5:** Framework para crear interfaces gráficas de usuario basadas en Qt en Python.
- **scikit-learn:** Herramienta clave para aprendizaje automático,es la librería utilizada para el TF-IDF y la fórmula del coseno.

2.3 Proceso de web Scraping:

Fuentes de datos y categorías:

- **Fuente de datos:** El código utiliza enlaces específicos almacenados en un archivo CSV (peliculas.csv) para extraer datos de páginas web estos enlaces etsan ubicados dentro de la columna links. Estas páginas contienen información sobre películas, incluidas imágenes de póster.
- **Categorías:** Las principales categorías de datos manejadas son:
    - Enlaces (link) a las páginas web de cada película.
    - Títulos de las películas (title).
    - Imágenes de los pósters (poster_image), extraídas dinámicamente desde el contenido HTML de las páginas.

Análisis de las fuentes de datos:

- **Pasos para extraen los datos del HTML:**
  - Se utiliza Selenium.
  - El script abre cada enlace el cual esta en el archivo peliculas.csv utilizando un navegador controlado por Selenium.
  - Se incluye un time.sleep(3)(de tres segundos) para asegurar que la caratula de la imagen se carga completamente.
  - **Selección del elemento HTML:**
    - Se busca específicamente el elemento HTML <rt-img> con el atributo slot='posterImage' utilizando un selector CSS ("rt-img[slot='posterImage']").
    - Se extrae el atributo src, que contiene la URL de la imagen del póster.
  - **Almacenamiento de datos:**
    - Las URLs de las imágenes se almacenan en una nueva columna (poster_image) del DataFrame.
    - Los resultados se guardan en un archivo CSV actualizado (peliculas_final_imagenes.csv).

2.4 Descripción del dataset generado:

1. Dataset de Películas (peliculas_final_imagenes.csv):

- El dataset inicial proporcionado para realizar la tarea pero se le realizó una limpieza para eliminar películas duplicadas y se le añadió a través de Web Scraping la columna poster_image.
- Campos principales:
  - **title**: Título de la película. Utilizado para la busqueda de peliculas
  - **year**: Año de lanzamiento.
  - **synopsis:** Resumen de la trama de la película. Utilizada para calcular similitudes entre películas
  - **critic_score y people_score**: Puntajes otorgados por críticos y audiencia.
  - **genre**: Género de la película . Utilizado para cálculos de la similitud.
  - **director**: Nombre del director se utiliza para calcular similitudes.
  - **link:** Enlace a la página de la película. Utilizado para realizar web scraping de imágenes de los pósters.
  - **poster_image**: Enlace al póster de la película extraído mediante web scraping.
- Justificación de las modificaciones:
  - Columna poster_image: Se añade para poder mejorar la vista y visualizar el poster de la imagen..

2. Dataset de Usuarios (usuarios.csv):

- Creado para almacenar la información de los usuarios registrados en el sistema y sus votaciones por peliculas.
- Campos principales:
  - **Nombre de usuario**: Identificador de usuarios.
  - ID : id unico por usuario
  - **Contraseña:** Credenciales de acceso del usuario.
  - **votaciones:** Array que contiene objetos con dos atributos:
    - **title**: Título de la película que el usuario ha votado.
    - **rating**: Calificación que el usuario asigna a la película (por ejemplo, de 1 a 5)..
- Justificación del diseño:
  - Este csv nos permite almacenar los datos del login y del registro a la vez que las votaciones de los usuarios lo que nos permite tener mayor facilidad a la hora de recomendar las peliculas.

2.5 Descripción del proceso de procesamiento de lenguaje natural:

Stop word:

- Se eliminan palabras comunes en inglés, como "the", "and", "is", que no aportan significado relevante al contenido.
- Esto se controla mediante el argumento stop_words='english'.

Tokenización:

- El texto se divide en palabras o tokens individuales. Esto permite analizar cada palabra de forma independiente.
- Por ejemplo, el texto "A thrilling movie about space" se tokenizará en ["a", "thrilling", "movie", "about", "space"].

Filtrado basado en frecuencia:

- Se eliminan términos extremadamente comunes o raros . Esto se controla con:
  - **max_df=0.9:** Palabras que aparecen en más del 90% de los documentos se eliminan por ser demasiado frecuentes.
  - **min_df=0.01:** Palabras que aparecen en menos del 1% de los documentos se eliminan por ser demasiado raras.
- Este paso garantiza que solo se utilicen los términos más relevantes para identificar similitudes.

Representación mediante TF-IDF:

- Se convierte el texto en una matriz TF-IDF (Term Frequency-Inverse Document Frequency). Esta técnica:
  - Calcula la frecuencia de cada palabra en un documento (TF).
  - Ajusta esta frecuencia considerando cuántos documentos contienen esa palabra (IDF).
- Esto da más peso a palabras distintivas de cada documento y menos peso a palabras comunes.

Limitación de características:

- Se seleccionan las 1000 palabras más importantes del conjunto de datos para reducir la cantidad y mejorar . Esto se controla con max_features=1000.

2.6 El proceso de modelado:

Para realizar este apartado lo dividimos en dos funciones una centrada en hacer el TF-IDF para la recomendación de sinopsis que es la función calcular similitudes y la otra función _calcular_similitudes_recomendaciones , la cual se centra en hacer el TF-IDF para las recomendaciones personalizadas según las votaciones del usuario.

1. Creación de Vectores con TF-IDF:

En ambas funciones se utiliza el TF-IDF Vectorizer para convertir el texto en una matriz de características numéricas. Este proceso incluye:

1. **Selección del texto relevante:**
   - En _calcular_similitudes: Se utiliza únicamente el texto de las sinopsis.
   - En _calcular_similitudes_recomendaciones: Se crea una columna 'combined_features' que combina las sinopsis, los directores y los géneros ('synopsis' + 'director' + 'genre').
2. **Preprocesamiento del texto:**
   - Se eliminan valores nulos para evitar errores al procesar los datos.
   - Se eliminan palabras vacías y se ajusta la frecuencia mínima y máxima para los términos (mediante stop_words, min_df, y max_df).
3. **Vectorización:**
   - **TF-IDF Vectorizer** convierte cada documento en un vector numérico, donde:

        - **TF (Term Frequency):** Mide la frecuencia de cada palabra en el documento.
        - **IDF (Inverse Document Frequency):** Pondera la importancia de cada palabra en función de cuántos documentos la contienen, dando más peso a las palabras distintivas.
    - Se limita el número de características a las 1000 palabras más relevantes (max_features=1000), lo que mejora la eficiencia.
4. **Resultado:**
    - Se genera una matriz tfidf_matrix donde cada fila representa un documento (película) y cada columna representa una palabra clave.

2. Cálculo de Similitudes con la Matriz TF-IDF:

1. **Similitud de Coseno:**
    - Una vez obtenida la matriz tfidf_matrix, se utiliza la función cosine_similarity para calcular la similitud entre todas las combinaciones de documentos.
    - La similitud de coseno mide el grado de similitud entre dos vectores basándose en el ángulo entre ellos, sin importar su magnitud. Su valor está en el rango de [0, 1], donde:
        - 0 indica que los documentos no tienen similitud.
        - 1 indica que los documentos son idénticos.
2. **Resultados de Similitud:**
    - En _calcular_similitudes, el resultado se guarda en self.cosine_sim_synopsis, que contiene las similitudes basadas únicamente en las sinopsis.
    - En _calcular_similitudes_recomendaciones, el resultado se guarda en self.cosine_sim_recomendaciones, que considera sinopsis, directores y géneros.

2.7 Descripción y justificación del sistema ideado para el perfil del usuario:

El sistema de recomendación ha sido diseñado para adaptarse al perfil único de cada usuario, basándose en sus valoraciones anteriores de películas. Al registrar las valoraciones (de 1 a 5 estrellas), el sistema toma en cuenta las similitudes entre las películas valoradas positivamente y las características de las nuevas películas, como la sinopsis, el director y el género. Esto permite ofrecer recomendaciones personalizadas y relevantes, que se ajusten a las preferencias del usuario.

Además, el sistema ajusta las sugerencias a medida que el usuario interactúa con la plataforma, añadiendo nuevas valoraciones que modifican su perfil y mejoran las recomendaciones futuras. Se priorizan las películas que tienen similitudes con aquellas que el usuario ha calificado más alto, mientras que las valoraciones negativas o neutrales ayudan a filtrar aquellas películas menos atractivas.

El sistema también evita recomendar películas que ya han sido vistas o valoradas por el usuario, asegurando que las recomendaciones sean frescas y útiles. Este enfoque dinámico y personalizado crea una experiencia adaptativa que mejora a medida que el usuario proporciona más datos a través de sus valoraciones, haciendo que las sugerencias sean cada vez más precisas y ajustadas a sus gustos específicos.

Por último el sistema ofrece un apartado para ver las sinopsis y que el usuario pueda visualizar todo tipo de películas recomendadas a través de la sinopsis y que así se le recomienden otro tipo de películas aparte de las recomendadas en el apartado Recomendaciones.

2.8 El proceso de recomendación:

En este proceso, se utilizan las similitudes de coseno y los perfiles de usuario para recomendar películas relevantes. Existen dos enfoques principales en la recomendación: uno basado en la similitud de las sinopsis de las películas y otro basado en el perfil del usuario teniendo en cuenta el género y el director:

1. Recomendación Basada en la Similitud de Coseno para las Sinopsis:

En la función recomendar_peliculas, el proceso sigue estos pasos:

1. Entrada:
    - Se proporciona el título de una película como entrada para obtener recomendaciones.
2. Cálculo de la Similitud de Coseno:
    - Se obtiene el índice de la película dentro del csv peliculas_final_imagenes.
    - Luego, se calcula la similitud de coseno entre la película seleccionada y todas las demás películas utilizando la matriz de similitud de coseno self.cosine_sim_synopsis, que fue calculada previamente usando las sinopsis de las películas.
3. Selección de las Mejores Recomendaciones:
    - Las películas más similares se ordenan en función de su similitud, excluyendo la película seleccionada .
    - Las mejores 5 películas más similares se agregan a la lista de recomendaciones junto a su similitud.
4. Resultado:
    - Se devuelve una lista de las 5 películas más similares, ordenadas por similitud, basadas en las sinopsis de las películas.

2. Recomendación Basada en el Perfil del Usuario:

La función recomendar_peliculas_por_usuario realiza recomendaciones personalizadas teniendo en cuenta las votaciones previas del usuario. El proceso incluye:

1. Entrada:
    - Se proporciona el nombre de usuario para obtener recomendaciones personalizadas.
2. Recopilación de Votaciones del Usuario:
    - El sistema obtiene las votaciones realizadas por el usuario a través de la columna votaciones en el csv usuarios_df.
    - Las votaciones se almacenan en un formato arraylist que contiene tanto el título de la película como la puntuación que el usuario le ha asignado.
3. Clasificación de las Votaciones:
    - Se agrupan las películas votadas según la puntuación (de 5 a 1), lo que permite dar más peso a las películas con puntuaciones altas.
4. Cálculo de la Similitud de Coseno :
    - Para cada película votada, se calcula la similitud de coseno entre esta y todas las demás películas, utilizando la matriz de similitud combinada self.cosine_sim_recomendaciones, que incluye sinopsis, director y género.
5. Ajuste de la Similitud según la Puntuación del Usuario:
    - Para las películas votadas con una puntuación alta , se da un peso mayor en la similitud ajustada.
    - Los títulos recomendados se añaden a una lista, donde se ajusta la similitud real utilizando el peso correspondiente.
6. Eliminación de Recomendaciones Duplicadas:
    - Las recomendaciones duplicadas se eliminan para asegurar que cada película sea recomendada solo una vez.
    - Si una película ya ha sido votada por el usuario, no se recomienda nuevamente.
7. Resultado:
    - Se devuelve una lista de recomendaciones únicas ordenadas por la similitud ajustada, lo que mejora la personalización de las recomendaciones basadas en las preferencias pasadas del usuario.

3.Pruebas Realizadas

En este apartado de pruebas y validación se detalla como verificamos el correcto funcionamiento de la aplicación, para asegurarnos que cumple con los objetivos planteados. Realizamos pruebas funcionales y de validación en diferentes etapas del desarrollo. En esta tabla se describen las principales pruebas que realizamos y sus resultados:

Prueba	**Descripción**	**Entrada proporcionada**	**Salida esperada**	**Resultado obtenido**	**¿Prueba superada?**
Prueba de recomendación por sinopsis	Validar que el sistema recomienda películas similares basándose en la sinopsis.	Título de una película: "Inception".	Lista de películas con mayor similitud.	Lista con películas relacionadas correctamente generada.	Si
Prueba de recomendaciones para usuario	Verificar que las recomendaciones se ajustan al perfil del usuario según sus valoraciones previas.	Usuario con valoraciones altas en películas de acción y ciencia ficción.	Recomendaciones de películas del género acción/ciencia ficción con similitud ajustada.	Recomendaciones alineadas con las preferencias del usuario.	Si
Prueba de duplicados en recomendaciones	Comprobar que no se recomienden películas ya valoradas por el usuario.	Usuario con valoraciones en "Inception" y "The Matrix".	Lista de recomendaciones sin incluir "Inception" y "The Matrix".	Películas ya valoradas correctamente excluidas de la lista de recomendaciones	Si
Prueba de inicialización del sistema	Verificar que el sistema puede cargar los datasets correctamente	Archivos peliculas_final_imagenes.csv y usuarios.csv.	Sistema inicializado correctamente y sin errores al leer los	Datasets cargados correctamente y sin errores reportados.	Si

	al inicio.		datasets.		
Prueba de visualización de imágenes	Comprobar que los enlaces a los pósteres se incluyen correctamente	Recomendación basada en "Titanic".	Enlaces a los pósteres de las películas recomendadas correctamente adjuntados.	Los enlaces aparecen en la salida sin errores.	Si

4.Resultados Obtenidos

Se desarrolló un sistema de recomendación eficaz que permite a los usuarios descubrir nuevas películas basándose en sus intereses y valoraciones personales.

Se implementaron dos enfoques principales para generar recomendaciones:

1. **Recomendación por similitud de sinopsis**: Utilizando TF-IDF y similitud del coseno, el sistema identifica películas con sinopsis similares a las seleccionadas.
2. **Recomendación basada en el perfil del usuario**: A partir de las votaciones previas del usuario, se generan recomendaciones personalizadas.

El sistema combina métodos avanzados de procesamiento de lenguaje natural (PLN) y algoritmos de similitud, asegurando recomendaciones relevantes y precisas.

Además, se incorporó una interfaz fácil de usar, que permite a los usuarios interactuar de forma intuitiva y segura con la aplicación.

Finalmente, el sistema demostró ser robusto, manejando entradas inválidas y datos incompletos de manera eficiente, sin comprometer su rendimiento o funcionalidad.

5.Conclusiones

El sistema de recomendación de películas desarrollado cumple con los objetivos planteados, ofreciendo una herramienta efectiva para explorar contenido basado en intereses personales. Los algoritmos implementados combinan técnicas avanzadas de procesamiento de lenguaje natural.

La interfaz gráfica diseñada facilita la interacción del usuario, permitiendo una navegación intuitiva y segura a través de las funcionalidades principales del sistema. Este diseño centrado en el usuario incrementa la usabilidad y accesibilidad de la aplicación.

La escalabilidad y robustez del sistema se demostraron al manejar grandes datasets y gestionar entradas inválidas sin comprometer la funcionalidad. Esto garantiza que la solución sea adaptable a escenarios más amplios y complejos en el futuro.

Por último, el proyecto sienta las bases para posibles mejoras, como la integración de filtros avanzados, algoritmos híbridos de recomendación y una mayor personalización en la experiencia del usuario. En conjunto, esta solución representa un avance significativo en el campo de los sistemas de recomendación aplicados al entretenimiento.

6.Apéndices

Manual_Usuario_Sistema_recomendacion.pdf
Manual_Instalacion.pdf

7.Bibliografía

Python 3.10 (Lenguaje de programación)

- **Descripción: Versión utilizada para desarrollar el sistema de recomendación, compatible con todas las bibliotecas requeridas.**
- **Enlace: https://www.python.org/downloads/release/python-3100/**

Dataset utilizado: Kaggle

- **Descripción: Dataset principal de películas utilizado para entrenar y probar el sistema de recomendación. Contiene información como títulos, sinopsis, géneros, y equipo de producción.**
- **Enlace: https://www.kaggle.com/ (Especifica aquí el nombre del dataset si lo sabes)**

Documentación oficial de TF-IDF en scikit-learn

- **Descripción: Utilizado para vectorizar las sinopsis de las películas y calcular la similitud entre ellas.**
- **Enlace: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html**

BeautifulSoup para Web Scraping

- **Descripción: Herramienta utilizada para extraer imágenes de pósters desde los enlaces proporcionados en el dataset.**
- **Enlace: https://beautiful-soup-4.readthedocs.io/**

PyQt5 (Documentación oficial)

- **Descripción: Framework utilizado para diseñar la interfaz gráfica de usuario, incluyendo pantallas de búsqueda, votación y recomendaciones.**
- **Enlace: https://pypi.org/project/PyQt5/**

Ejemplos de Web Scraping con Python

- **Descripción: Guías prácticas utilizadas para implementar la extracción de datos adicionales desde páginas web.**
- **Enlace: https://realpython.com/tutorials/web-scraping/**

Documentación de Pandas: Manipulación de datos

- **Descripción: Guía utilizada para limpiar y procesar el dataset de películas, incluyendo la eliminación de columnas irrelevantes y el manejo de datos faltantes.**
- **Enlace: https://pandas.pydata.org/docs/**

NumPy: Operaciones matemáticas avanzadas

- **Descripción: Biblioteca utilizada para realizar cálculos vectoriales y manejar matrices de similitud generadas por TF-IDF.**
- **Enlace: https://numpy.org/doc/**

Manual de instalación de pip y gestión de dependencias

- **Descripción: Instrucciones utilizadas para instalar las bibliotecas necesarias desde el archivo requirements.txt.**
- **Enlace: https://pip.pypa.io/en/stable/installation/**

Diseño de sistemas de recomendación (Medium)

- **Descripción: Artículos consultados para comprender las bases de los sistemas de recomendación basados en contenido.**
- **Enlace: https://towardsdatascience.com/tagged/recommendation-system**

Kaggle Tutorial: Construyendo un sistema de recomendación

- **Descripción: Guía práctica consultada para implementar las recomendaciones basadas en similitud.**
- **Enlace: https://www.kaggle.com/general/20433**

TF-IDF explicado (Analytics Vidhya)

- **Descripción: Explicación teórica utilizada para implementar y optimizar la generación de vectores TF-IDF en el proyecto.**
- **Enlace: https://www.analyticsvidhya.com/blog/2021/06/tf-idf-a-quick-introduction/**

Stack Overflow

- **Descripción: Fuente de resolución de problemas durante la implementación del sistema, desde configuración hasta manejo de excepciones.**
- **Enlace: https://stackoverflow.com/**

[PÁGINA INTENCIONADAMENTE EN BLANCO]