Sistema de Recomendación

Prolog

Juan Pablo Casas Gómez

Alejandro Cañadas Parra

Juan Pablo Dávila Martínez

Eugenia Victoria Dayoub Barito

Pontificia Universidad Javeriana

Facultad de Ingeniería

Introducción a la Inteligencia Artificial

Bogotá D.C.

Marzo, 2025

Introducción:

El presente sistema de recomendación para una tienda de zapatos tiene como objetivo mejorar la experiencia de compra de los usuarios, facilitando la selección de productos a través de sugerencias basadas en compras y calificaciones previas. Para ello, el sistema modela la interacción entre los clientes y los productos disponibles, registrando tanto las compras como las ventas realizadas.

1. Diseño del problema

1.1 Variables:

El sistema está compuesto por las siguientes variables:

- Usuarios: representan a los clientes de la tienda y pueden interactuar con los productos. Cada usuario tiene los siguientes atributos:
    - Identificación (nombre)
    - Historial de compras
    - Calificaciones de los productos
- Productos (Zapatos): son los productos disponibles en la tienda y se clasifican en diferentes categorías. Cada producto tiene los siguientes atributos:
    - Identificación (nombre)
    - Categoría (deportivo, casual, formal, etc)
- Compras del usuario: los usuarios pueden comprar zapatos, lo que genera una relación entre un usuario y un producto. Cada compra del usuario tiene los siguientes atributos:
    - Usuario que realizó la compra (nombre)
    - Producto adquirido (nombre)
- Calificaciones de los productos: los usuarios pueden calificar los zapatos que han comprado, asignando un puntaje del 1 al 5. Cada calificación tiene los siguientes atributos:
    - Usuario que califica (nombre)

        - Producto calificado (nombre)
        - Valor de la calificación (1 al 5)
- Ventas: representan las transacciones completadas en el sistema. Cada venta tiene los siguientes atributos:
        - Usuario que realizó la compra (nombre)
        - Producto vendido (nombre)
        - Cantidad

1.2 Estados:

Se identifican los siguientes estados para el sistema:

- Estado inicial:
    - No hay usuarios ni productos registrados.
    - No hay compras ni calificaciones registradas.
    - No hay ventas registradas.
- Estado final:
    - Se han registrado tanto usuarios como productos.
    - Existen compras y calificaciones almacenadas.
    - Se han registrado ventas con la cantidad.
    - Se crean recomendaciones basadas en interacciones previas.

1.3 Acciones:

En el sistema se pueden realizar las siguientes acciones:

1. Registro del usuario: permite la creación de nuevos clientes en el sistema. Es necesario tener la identificación del usuario (nombre) para poder realizar el registro.
2. Registro de un producto: añade un nuevo producto (zapato) a la tienda. Es necesario tener los datos del zapato, la identificación (nombre) y la categoría a la que pertenece, para realizar el registro.

3. Registro de una compra: almacena la compra de un producto por parte del usuario. Es necesario tener la identificación (nombre) tanto del usuario como del producto, para poder relacionar el usuario con el producto.
4. Registro de una calificación: guarda la valoración que le da un usuario a un producto. Es necesario tener la identificación (nombre) del usuario, del producto y la calificación otorgada al producto, para registrar el valor en el sistema.
5. Generación de recomendaciones: sugiere productos a los usuarios, utilizando estrategias con datos adquiridos por compras previas y calificaciones.
    - Compras previas: se recomiendan productos similares a los adquiridos por el usuario.
    - Calificaciones: se recomiendan productos con altas calificaciones dadas por usuarios con intereses similares.
6. Registro de una venta: almacena la información de una transacción exitosa. Es necesario tener la identificación(nombre) del usuario, del producto y la cantidad adquirida.

El sistema se diseñó utilizando variables que representan las entidades clave del negocio, permitiendo modelar la interacción entre usuarios y productos. Se establecen dos estados principales (inicial y final) para reflejar la evolución del sistema desde una base de datos vacía hasta la generación de recomendaciones y el registro de ventas completadas. La metodología de recomendación emplea estrategias utilizadas en la industria, como el filtrado basado en compras previas y calificaciones, con el objetivo de ofrecer sugerencias personalizadas y relevantes para cada usuario, optimizando así la experiencia de compra y facilitando el seguimiento de las transacciones realizadas.

1.4 GRAFO:

A continuación, en la **Figura 1**, se presenta un grafo que representa el sistema con sus estados, variables y acciones

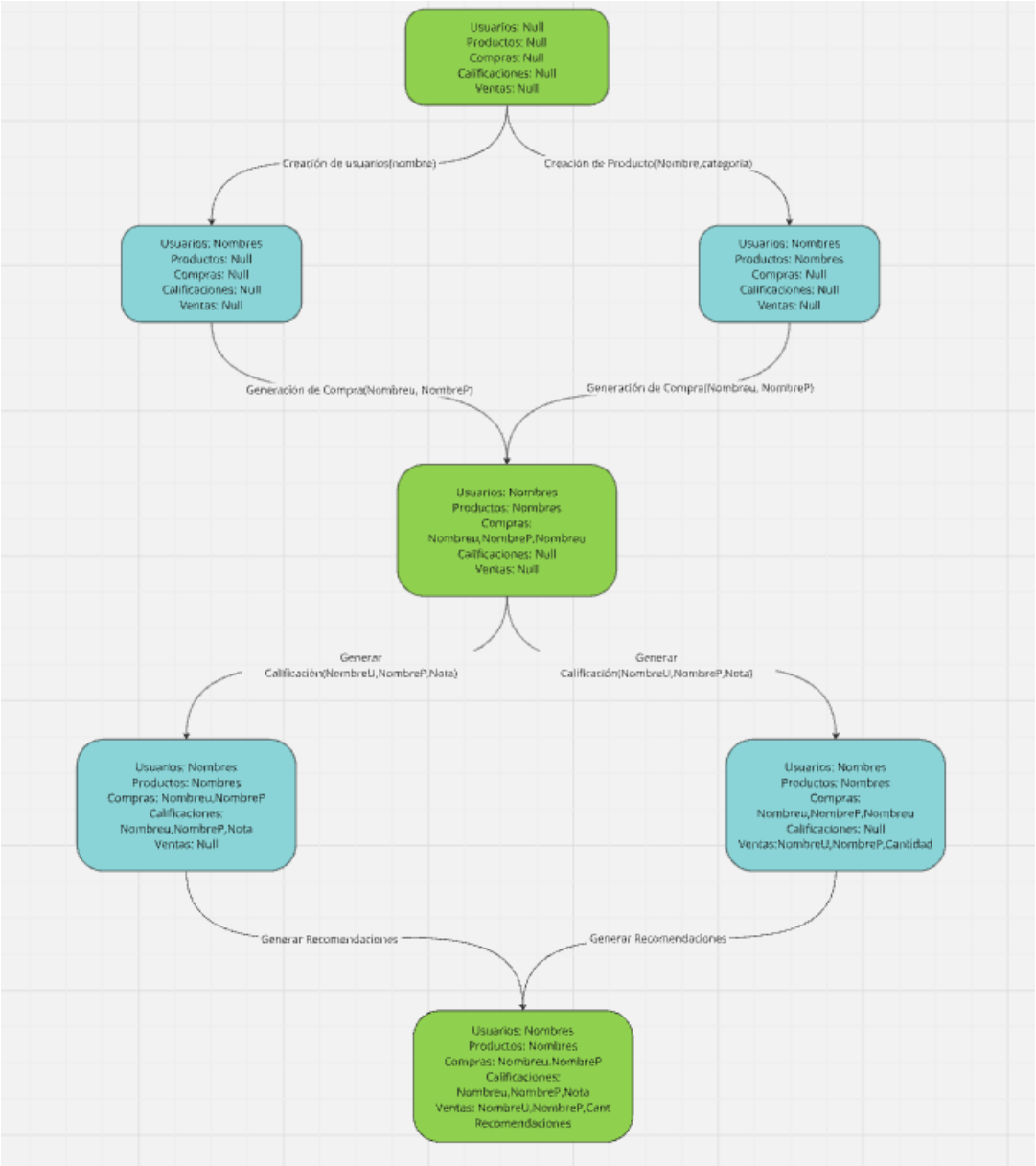


Figura 1

2. Solución del problema

IMPLEMENTACIÓN:

2.1 Realización de la base de hechos. Se debe tener varios casos de prueba para resolver las queries. (4%)

Ver adjunto base de hechos.pl

2.2 Realice una query que, dado un usuario, el sistema le recomiende un ítem, para ello investigue una forma de realizar la recomendación y descríbala (agregando la referencia). Diseñe las reglas que considere necesarias. (5%)

Para este punto investigamos en 3 diferentes fuentes, presentadas en la **Figura 2**

Fuente	**Concepto tomado**	**Aplicación en nuestra consulta**
Expert System Movies Suggestion	Uso de hechos y reglas, filtrado por categorías	Implementamos categoria/2 y producto/1
Recomendador Prolog	Filtrado por calificación, evitar sugerencias repetidas	Implementamos calificacion (_, Producto, Calificacion), Calificacion >= 4 y \+ compra (Usuario, Producto)
Travel Recommendation System	Uso de findall/3 para generar listas	Implementamos findall (Producto, recomendar_item(Usuario, Categoria, Producto), ListaProductos)

Figura 2

En base a esto se realizó la siguiente query (**Figura 3**):

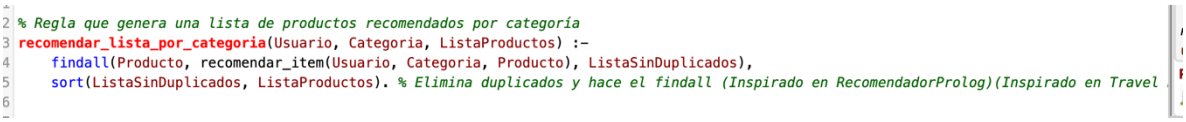
```
% Regla que recomienda un producto a un usuario basándose en categoría y calificación
recomendar_item(Usuario, Categoria, Producto) :-
    usuario(Usuario),                       % Asegura que el usuario existe (Inspirado en RecomendadorProlog)
    producto(Producto),                     % Asegura que el producto existe (Inspirado en Expert System Movies)
    categoria(Producto, Categoria),         % Filtra por categoría (Inspirado en Expert System Movies)
    calificacion(_, Producto, Calificacion), % Obtiene calificación del producto (Inspirado en RecomendadorProlog)
    Calificacion >= 4,                       % Solo recomienda productos con calificación >= 4 (Inspirado en RecomendadorProlog)
    \+ compra(Usuario, Producto).           % Evita recomendar productos comprados (Inspirado en RecomendadorProlog)
```

Figura 3

2.3 Realice una query que, dado un usuario, el sistema le recomiende una lista de ítems, para ello investigue una forma de realizar la recomendación y descríbala (agregando la referencia). Diseñe las reglas que considere necesarias. (5%)

Con las referencias anteriores utilizamos la de travel recomendation en donde se emplea el predicado findall/3 para recopilar elementos recomendados en una lista basándose en las preferencias del usuario. Este enfoque nos permite generar una lista para un usuario según su categoría de interés y asegurando que no haya sido comprado previamente.

A continuación, en la **Figura 4** se puede ver la query implementada:



```
% Regla que genera una lista de productos recomendados por categoría
recomendar_lista_por_categoria(Usuario, Categoria, ListaProductos) :-
    findall(Producto, recomendar_item(Usuario, Categoria, Producto), ListaSinDuplicados),
    sort(ListaSinDuplicados, ListaProductos). % Elimina duplicados y hace el findall (Inspirado en RecomendadorProlog)(Inspirado en Travel
```

Figura 4

2.4 Realice una query que genere una recomendación y que involucre el uso de la recurrencia (como restricción, no vale una regla que sea un recorrido en una lista). Diseñe las reglas que considere necesarias. (15%).

A continuación, en la **Figura 5** se puede ver la query implementada:

```
% Recomendación recursiva que usa el recomendar_aux donde se cumple la regla y la restricción dada
recomendar_recursivo(Usuario, Categoria, ListaRecomendaciones) :-
    recomendar_aux(Usuario, Categoria, [], ListaRecomendaciones).

% Caso base: si no hay más productos que añadir, se devuelve el acumulador.
recomendar_aux(_, _, Acumulador, Acumulador).

% Caso recursivo: buscar productos válidos y agregarlos a la lista de recomendaciones.
recomendar_aux(Usuario, Categoria, Acumulador, ListaRecomendaciones) :-
    producto(Producto), % Obtener un producto
    categoria(Producto, Categoria), % Verificar que pertenece a la categoría deseada
    calificacion(_, Producto, Calificacion), % Obtener su calificación.
    Calificacion >= 4, % Solo considerar productos con buena calificación.
    \+ compra(Usuario, Producto), % Evitar recomendar productos que ya compró.
    \+ member(Producto, Acumulador), % Evitar duplicados en la lista.
    !, % Evita que Prolog siga buscando mas productos en este nivel de nuestra recursion.
    recomendar_aux(Usuario, Categoria, [Producto | Acumulador], ListaRecomendaciones).


```

Figura 5

2.5 Realice una query que, dado una lista de usuarios, retorne un top 10 de la lista de ítems que le ha gustado a cada uno (calificaciones mayores a 3) (15%).

A continuación, en la **Figura 6** se puede ver la query implementada:

```
top_10_por_usuario(ListaUsuarios, Resultado) :-
    findall((Usuario, Top10),
        (member(Usuario, ListaUsuarios), % recorre sobre la lista de usuarios
         findall((Producto, Calificacion),
            (producto(Producto), % Verifica que sea un producto existente
             calificacion(Usuario, Producto, Calificacion),
             Calificacion > 3),
            ListaProductos), % Obtiene los productos con calificación mayor a 3
         length(ListaProductos, L),
         (L =< 10 -> Top10 = ListaProductos ; append(Top10, _, ListaProductos), length(Top10, 10))% Si hay más de 10, toma solo los primeros 10
        ),
    Resultado).

```

Figura 6

3. Realice conclusiones sobre el ejercicio (10%).

A modo de cierre, el ejercicio realizado evidencia cómo a partir del planteamiento inicial de un problema con sus variables, estados y acciones, y con la ayuda del entorno prolog, se puede lograr la construcción de un sistema de recomendación funcional que emula el funcionamiento de lo que es un algoritmo de recomendaciones en la vida real. Este sistema cumple con su propósito que recordemos es, utilizar información de los usuarios, productos

e interacciones pasadas para ofrecer recomendaciones personalizadas y relevantes para cada usuario registrado en el sistema. A partir de esto se pueden sacar las siguientes conclusiones del ejercicio:

3.1 El desarrollo del sistema de recomendación permitió explorar y aplicar los conceptos fundamentales de los sistemas expertos basados en reglas.

3.2 El sistema tiene capacidad de realizar recomendaciones de zapatos para cada usuario dependiendo de las compras previas y las calificaciones brindadas por usuarios con intereses similares.

3.3 La implementación de estrategias de filtrado, búsqueda y recursión, permitieron estructurar un mecanismo eficiente para la recomendación de productos, permitiendo así una experiencia personalizada para cada usuario.

3.3.1 Se utilizaron distintas estrategias para realizar recomendaciones, como el filtrado basado en compras previas y filtrado por calificaciones, lo que permitió ofrecer sugerencias más precisas a los usuarios.

3.3.2 Se exploró el uso del predicado findall/3 para generar listas de recomendaciones dinámicamente, lo que facilitó la estructuración de respuestas a consultas sobre preferencias de los usuarios.

3.3.3 Se hizo uso de recurrencia para optimizar y volver más eficiente la lógica de recomendación del sistema.

3.4 Aunque la implementación del sistema presentado es básica, su estructura está diseñada para recibir mejoras y lograr escalabilidad en caso de ser necesario.

3.5 La mezcla de diseño, implementación, graficación y análisis nos permite comprender de mejor manera la lógica detrás de este tipo de sistemas, siendo así

una base para la futura exploración de sistemas relacionados más complejos y temas asociados con la Inteligencia Artificial.

4. Referencias utilizadas

https://github.com/elaaatif/EXPERT-SYSTEM-MOVIES-SERIES-SUGGESTION

https://github.com/JYisus/RecomendadorProlog/blob/master/recomendador.pl

https://swish.swi-prolog.org/p/TravelRecommendation.swinb

https://www.swi-prolog.org/pldoc/man?predicate=%5C%2B/1