HENRY

Data Science

M4L8 | Aprendizaje no supervisado II

→ soyhenry.com

Objetivos

- Distinguir los tipos de sistemas de recomendación, analizando sus enfoques principales y aplicaciones en distintos contextos.
- Implementar métodos de filtrado colaborativo y basado en contenido, evaluando su eficacia en la personalización de recomendaciones.
- Examinar técnicas de evaluación de recomendadores, comparando métricas como precisión, cobertura y diversidad para optimizar resultados.

#TEMAS

Agenda

COMENCEMOS →

.01 Definición y clasificación

.02 Filtrado colaborativo

.03 Basado en contenidos

.04 Híbridos

.05 Avance de PI

<-->

¿Qué vimos en la lecture?

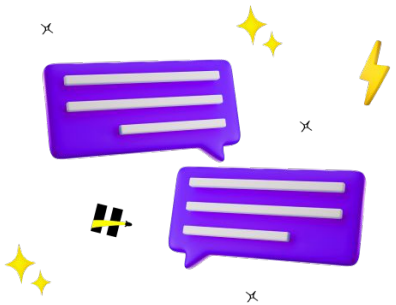
<01>

Definición y clasificación

Qué es un sistema de recomendación

- Modelos que **predicen intereses** y ofrecen **sugerencias personalizadas**
- Resuelven **sobrecarga de opciones**; aumentan **satisfacción** y **engagement**
- Funcionan como **interfaz inteligente** entre datos y experiencia humana
- **Predicción** (afinidad usuario–ítem) + **ranking** (orden por relevancia)
- Punto de partida: **similitudes**, **representaciones latentes** y **agrupamientos**

Recos en la vida cotidiana



- **Netflix**: sugiere por historial y **usuarios similares**
- **Spotify**: listas **Discover Weekly** por patrones de escucha
- **Amazon**: “Otros clientes también compraron...”
- **Coursera/Udemy**: cursos **complementarios** al finalizado
- Misma lógica: **comportamiento + atributos + similitudes**

Definición y clasificación: las 3 familias

- **Filtrado colaborativo**: aprende de **interacciones** (compras, ratings, clics)
- **Basado en contenidos**: usa **atributos** de ítems + **perfil** del usuario
- **Híbridos**: combinan ambos para **precisión, cobertura y diversidad**
- Objetivo común: **predecir preferencias individuales**

<02>

Filtrado colaborativo

Filtrado colaborativo: principio

- Premisa: **usuarios con comportamientos similares comparten gustos**
- No requiere atributos del ítem: aprende de la **matriz usuario–producto**
- Señales: **compras, calificaciones, clics, reproducciones**
- Predice **afinidades faltantes** en la matriz dispersa

Colaborativo user–user vs item–item

- **User–User**: encuentra **vecinos** con patrones parecidos; transfiere gustos
- **Item–Item**: ítems similares si **co-ocurren** con los mismos usuarios
- Ej.: RunMax Trail ⇒ **calcetines técnicos** / **mochilas deportivas**
- ShopSense: **item–item** útil por **catálogo estable** y atributos persistentes
- Selección depende de **densidad** y **escala** de interacciones

Métricas de similitud

- **Coseno**: ángulo entre vectores de interacción (escala-invariante)
- **Pearson**: corrige **media/varianza** (usuarios que puntúan distinto)
- ShopSense: coseno para **co-ocurrencia**; Pearson para **patrones de valoración**
- Salida: **matriz de similitud** (k ítems más cercanos por producto)
- Base del **ranking** personalizado

Factorización matricial (SVD)

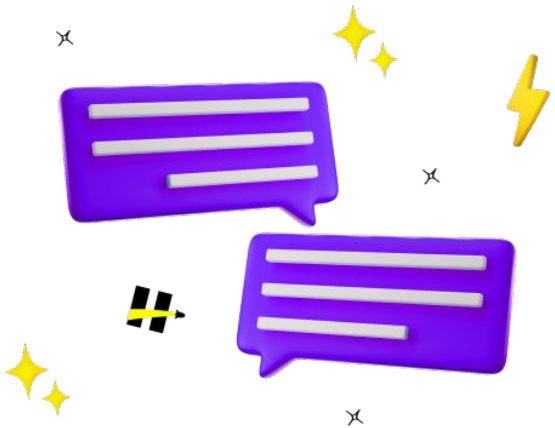
- Representa la matriz como **U × V^T** (factores latentes)
- Cada usuario/ítem → **vector** de baja dimensión (tecnología, deporte, precio...)
- Predicción = **producto punto** (rating esperado)
- Escala a **millones**; capta **patrones complejos** sin texto
- ShopSense: **Surprise + SVD** para ordenar por **puntuación esperada**

Sparsity y cold start: colaborativo

- **Dispersión**: pocos datos conocidos ⇒ similitudes **inestables**
- **Cold start**: usuarios/productos **nuevos** sin historial
- ShopSense: **combinar con contenido** para ítems nuevos (EcoFit)
- Onboarding: **encuesta inicial** de intereses para usuarios nuevos
- Mezcla resuelve **falta de datos** inicial

Evaluación del colaborativo (ranking)

- Importa el **top-K**: usuarios miran **primeras posiciones**
- **Precision@K**: proporción de recomendados **realmente relevantes**
- **Recall@K**: proporción de relevantes **capturados** en el top-K
- **NDCG@K**: **relevancia + posición** (premia aciertos arriba)
- ShopSense: optimiza **carruseles**, **portada** y **conversiones**



<03>

Basado en contenidos

Basado en contenidos: principio

- Si te gustó X por sus **características**, te gustarán **ítems similares**
- Usa **metadatos**: categoría, marca, precio, estilo, materiales, keywords
- Representa ítems con **TF-IDF**; similitud **coseno** entre vectores
- ShopSense: conecta **RunMax Trail** con **remera técnica** o **reloj deportivo**
- **No** depende de otros usuarios; arranca con **poco historial**

Ventajas y límites: contenido

- Pros: independencia de comunidad; cubre **ítems nuevos** (**cold start ítems**)
- Personaliza **nichos**; evita sesgo de **popularidad**
- Contras: **burbuja de similitud** (lista repetitiva)
- Mitigación: **diversidad mínima** (≥20% categorías complementarias)
- Requiere **metadatos de calidad** (normalización y etiquetado)

Métricas específicas: contenido

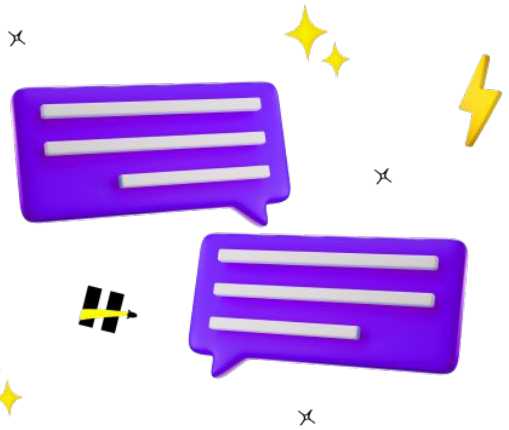
- **Precision@K / Recall@K**: relevancia y cobertura
- **Novedad**: ¿qué tan **nuevos** son los ítems para el usuario?
- **Diversidad**: **heterogeneidad** dentro de la lista (evitar clones)
- **Serendipity**: sorpresa **relevante** (baja similitud, alta utilidad)
- ShopSense: mezcla **familiaridad + descubrimiento** para retención

<04>

Híbridos

Híbridos: por qué y para qué

- Combinan **comportamiento colectivo** + **semántica de productos**
- Operan cuando un enfoque **falla** (datos escasos o metadatos pobres)
- Reducen **cold start**, **monotonía** y sesgos de popularidad
- ShopSense: **mejor rendimiento** con perfiles nuevos y catálogos nuevos
- Personalización **más rica, diversa y robusta**

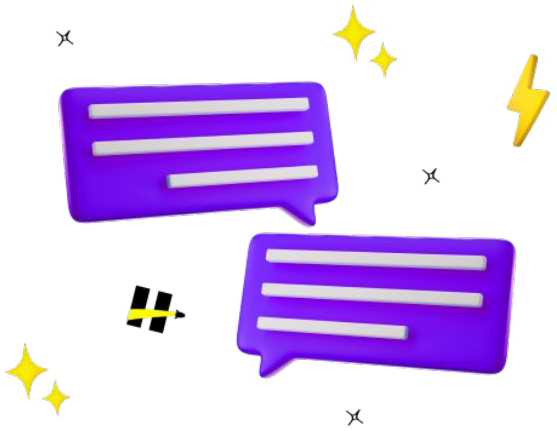


Estrategias de combinación

- **Ponderado**: score = **α·colab + (1−α)·contenido** (normalizado)
- **Two-Tower**: dos redes aprenden **embeddings** de usuarios e ítems; afinidad = **dot product**
- **Re-ranking**: contenido genera **candidatos**; colab **ordena** el top
- ShopSense: α **dinámico** (nuevos: 0.3 colab/0.7 contenido; activos: inverso)
- Equilibrio **precisión–diversidad–cobertura**

Impacto y métricas

- Pilotos ShopSense: **Precision@5 = 47%**, **Recall@10 = 65%**, **NDCG@5 = 0.79**
- **+18%** en **diversidad** de recomendados; menos repetición
- **+12%** en **frecuencia de compra** y mayor **engagement** app
- Mejora **conversión** y **ticket** vía cross-selling inteligente
- Valor: de “similares” a **experiencias complementarias**



Desafíos operativos

- **Sincronización**: colab semanal vs contenido por cambios de catálogo
- **Calibración** automática de **pesos** según densidad/recencia
- **Métricas unificadas** con negocio (retención, exposición estratégica)
- Monitoreo de **diversidad de marca** y **categoría** en listas
- Governanza del **pipeline** y escalado

<DATA SCIENCE/>

Vayamos a la práctica

Avance de PI

Consigna

FinanceGuard es un banco digital que enfrenta un desafío crítico: una tasa anual de abandono de clientes del 20%, lo que implica pérdidas significativas. Para revertir esta situación, la compañía busca aprovechar el potencial del análisis de datos y el aprendizaje automático. En este contexto, el estudiante, en su rol de Científico de Datos Junior, forma parte del equipo encargado de desarrollar un sistema predictivo que identifique a los clientes con mayor probabilidad de abandonar la entidad, facilitando estrategias de retención más efectivas.

En este **tercer avance**, asumes el rol de **analista de segmentación y exploración no supervisada.** Tu responsabilidad es aplicar técnicas de clustering y reducción de dimensionalidad para descubrir patrones ocultos en los datos de los clientes. A través de estas técnicas, deberás identificar segmentos con comportamientos diferenciados y analizar su relación con la tasa de churn, aportando una visión complementaria a los modelos supervisados desarrollados previamente.

Tareas a realizar

1. **Clustering básico para segmentación:**

- **K-Means clustering:**
  - Conceptos fundamentales: centroides, iteraciones
  - Selección del número de clusters K:
    - Método del codo (Elbow method)
    - Coeficiente de silueta (Silhouette score)
  - Implementación paso a paso
  - Interpretación de centroides
  - Visualización de clusters en 2D
- **DBSCAN (Density-Based clustering):**
  - Conceptos: core points, border points, noise
  - Parámetros básicos: eps (epsilon) y min_samples
  - Ventajas: detecta outliers, no requiere definir K
  - Comparación con K-means

2. **Reducción de dimensionalidad:**

- **PCA (Principal Component Analysis):**
  - Conceptos básicos: componentes principales
  - Varianza explicada por cada componente
  - Selección del número de componentes
  - Visualización de datos en 2D y 3D
  - Interpretación de los componentes principales
- **t-SNE básico:**
  - Visualización no lineal de datos
  - Parámetro perplexity (concepto básico)
  - Diferencias con PCA
  - Limitaciones y cuidados en interpretación

Tareas a realizar

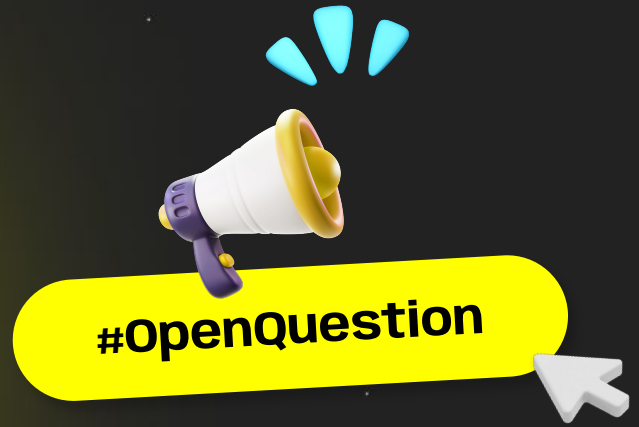
3. Aplicación al problema de churn:

- **Segmentación de clientes:**
  - Aplicar K-means al dataset de clientes
  - Identificar 3-5 segmentos principales
  - Analizar características de cada segmento
  - Tasa de churn por segmento identificado
- **Perfiles de clientes por cluster:**
  - Características demográficas y comportamiento por cluster
  - Crear features derivadas del clustering

Notas extra

La segmentación no supervisada puede revelar patrones “ocultos” en el comportamiento de los clientes. Usa estos insights para mejorar el feature engineering, y crear modelos más específicos.

HENRY



¿Preguntas?

HENRY

¡Muchas gracias!

→ soyhenry.com