Sistema de Recomendación de Optativas Universitarias

Proyecto para Modelos Matemáticos Aplicados

Junio 2025

Resumen

Este proyecto consiste en el desarrollo de un sistema inteligente para la recomendación personalizada de asignaturas optativas a estudiantes universitarios. Utiliza técnicas avanzadas de Procesamiento de Lenguaje Natural (NLP), extracción de etiquetas (tags), generación de embeddings semánticos y modelos de lenguaje (LLM) para analizar tanto los intereses de los estudiantes como las descripciones de los cursos, facilitando así la toma de decisiones académicas.

1. Objetivo

El objetivo principal de este proyecto es proporcionar una herramienta inteligente y automatizada que facilite la toma de decisiones académicas para estudiantes universitarios al momento de elegir asignaturas optativas. El sistema busca:

- **Personalización:** Ofrecer recomendaciones de cursos alineadas con los intereses, habilidades, trayectorias y expectativas individuales de cada estudiante, utilizando técnicas avanzadas de procesamiento de lenguaje natural y aprendizaje automático.
- **Optimización del proceso de orientación:** Reducir la incertidumbre y el tiempo que los estudiantes dedican a explorar la oferta de optativas, presentando sugerencias relevantes y justificadas.
- **Apoyo a la diversidad de perfiles:** Adaptarse a distintos perfiles estudiantiles, desde quienes tienen intereses muy definidos hasta quienes buscan explorar nuevas áreas, permitiendo una exploración guiada y flexible.
- **Facilitar la gestión docente:** Permitir a los docentes registrar y editar cursos de manera sencilla, asegurando que la información esté siempre actualizada y disponible para el sistema de recomendación.
- **Transparencia y explicabilidad:** Brindar a los usuarios información clara sobre por qué se recomienda cada curso, mostrando los intereses y similitudes detectadas.
- **Escalabilidad y replicabilidad:** Servir como base para sistemas similares en otras instituciones o contextos, gracias a su diseño modular y extensible.

En resumen, el sistema pretende ser un puente entre la oferta académica y las aspiraciones de los estudiantes, promoviendo una experiencia educativa más satisfactoria, informada y personalizada.

2. Funcionamiento General

El sistema sigue un flujo de trabajo estructurado y automatizado que integra diversas etapas de procesamiento de datos, análisis semántico y recomendación personalizada. A continuación se detalla cada fase del funcionamiento general:

1. **Preprocesamiento de datos:**
   - Limpieza y normalización de los textos de estudiantes y cursos, eliminando caracteres especiales, acentos y unificando el formato para facilitar el análisis posterior.
   - Conversión de los intereses y descripciones en formatos estructurados y homogéneos.
2. **Extracción de tags:**
   - Utilización de técnicas de NLP (spaCy) para extraer palabras clave relevantes de las descripciones de cursos y de los intereses de los estudiantes.
   - Sugerencia automática de etiquetas mediante modelos de lenguaje (LLM) como OpenRouter/mistral-7b-instruct, enriqueciendo el conjunto de tags con términos semánticamente relevantes.
   - Integración de una lista de etiquetas predefinidas para facilitar la selección manual por parte de los usuarios.
3. **Generación de embeddings:**
   - Conversión de los tags y descripciones en vectores semánticos (embeddings) utilizando modelos como distiluse-base-multilingual-cased-v1.
   - Almacenamiento de los embeddings generados para cursos y estudiantes, optimizando el cálculo de similitud y permitiendo actualizaciones eficientes.
4. **Cálculo de similitud:**
   - Cálculo de la similitud coseno entre los embeddings de cada estudiante y los de todos los cursos disponibles.
   - Generación de una matriz de afinidad que cuantifica el grado de correspondencia entre cada estudiante y cada curso.
5. **Recomendación:**
   - Para cada estudiante, el sistema rankea los cursos según la similitud calculada, presentando un listado personalizado de las optativas más afines a sus intereses y perfil.
   - Las recomendaciones se actualizan automáticamente ante cualquier cambio en los datos de estudiantes o cursos.
   - Se proporciona información explicativa sobre el motivo de cada recomendación, mostrando los intereses y similitudes detectadas.
6. **Gestión y actualización:**

- Los docentes pueden registrar y editar cursos, y los estudiantes pueden actualizar sus intereses y descripciones en cualquier momento.
- El sistema recalcula automáticamente los tags, embeddings y recomendaciones ante cualquier modificación, garantizando resultados actualizados y relevantes.

7. **Interfaz interactiva:**
   - Toda la interacción se realiza a través de una interfaz web amigable (Streamlit), que permite a los usuarios gestionar su información, consultar cursos y recibir recomendaciones de manera sencilla y visual.

Este flujo integral asegura que el sistema sea robusto, flexible y capaz de adaptarse dinámicamente a los cambios en la oferta académica y en los intereses de los estudiantes.

3. Componentes Principales

- **data/**: Contiene los datos de cursos y estudiantes, así como los embeddings y tags generados.
  - **courses.csv** y **students.csv**: Archivos principales con la información base de cursos y estudiantes.
  - `courses_with_tags.csv` y `students_with_tags.csv`: Versiones enriquecidas con etiquetas (tags) extraídas automáticamente.
  - `predefined_tags.py`: Lista de etiquetas predefinidas para la selección de intereses.
  - `courses_tags_embeddings.pkl` y `students_tags_embeddings.pkl`: Embeddings semánticos generados para cursos y estudiantes.
  - `models/`: Modelos de embeddings descargados (por ejemplo, distiluse-base-multilingual-cased-v1).
- **src/**: Código fuente principal para todo el procesamiento y lógica del sistema.
  - **data_preprocessing.py**: Limpieza y normalización de textos, carga de datos.
  - **tag_extraction.py**: Extracción automática de tags usando NLP (spaCy).
  - **tag_ia_suggestion.py**: Sugerencia de tags usando modelos de lenguaje (LLM) vía OpenRouter.
  - **embeddings.py**: Generación y gestión de embeddings semánticos para tags y descripciones.
  - **similarity.py**: Cálculo de similitud coseno entre embeddings de estudiantes y cursos.
  - **recommender.py**: Lógica principal de recomendación y ranking de cursos personalizados.
  - **run_workflow.py**: Script que ejecuta el flujo completo de procesamiento, extracción de tags, embeddings y recomendaciones.
  - **utils.py**: Funciones auxiliares para carga y manipulación de datos.

    - **api/elective_recommendation.py**: API de alto nivel para registrar, editar y consultar estudiantes/cursos, y recalcular recomendaciones.

- **app/**: Interfaz de usuario basada en Streamlit para interacción con estudiantes y docentes.
    - **webapp.py**: Aplicación web principal, permite registrar, editar, consultar y recomendar cursos y estudiantes desde una interfaz amigable.
- **README.md**: Documentación general y guía de uso del proyecto.
- **documentation/**: Documentación formal en formato LaTeX.

4. Tecnologías Utilizadas

El sistema integra diversas tecnologías modernas de ciencia de datos, procesamiento de lenguaje natural y desarrollo web para lograr recomendaciones personalizadas y eficientes. A continuación se describen en detalle:

- **Python**: Lenguaje principal de desarrollo, elegido por su ecosistema científico y de machine learning.
- **pandas**: Manipulación y análisis eficiente de datos tabulares (cursos, estudiantes, tags, embeddings).
- **sentence-transformers**: Biblioteca para generar embeddings semánticos de frases y palabras. Se emplea el modelo multilingüe 'distiluse-base-multilingual-cased-v1' para representar los intereses y descripciones en un espacio vectorial.
- **spaCy**: Procesamiento de lenguaje natural en español, extracción de palabras clave (tags) mediante lematización y análisis gramatical.
- **Streamlit**: Framework para construir la interfaz web interactiva, permitiendo a estudiantes y docentes interactuar con el sistema de manera sencilla y visual.
- **OpenRouter/mistral-7b-instruct**: Modelo de lenguaje grande (LLM) accesible vía API, utilizado para sugerir automáticamente etiquetas relevantes a partir de descripciones de cursos, enriqueciendo el sistema de tags.
- **dotenv**: Gestión de variables de entorno, como claves de API para servicios externos.
- **HuggingFace Hub**: Descarga y gestión de modelos preentrenados de embeddings.

Estas tecnologías permiten combinar procesamiento lingüístico avanzado, aprendizaje automático y una experiencia de usuario moderna, logrando un sistema robusto y flexible para la recomendación de optativas universitarias.

5. Interfaz de Usuario

La interfaz de usuario está implementada con Streamlit y permite la interacción tanto para estudiantes como para docentes de manera sencilla e intuitiva. A continuación se detallan sus funcionalidades principales:

- **Selección de rol**: Al iniciar la aplicación, el usuario puede elegir entre los roles de Estudiante o Docente desde la barra lateral.
- **Opciones para Estudiantes**:
  - **Registrar**: Permite a un nuevo estudiante registrarse, ingresando su nombre, seleccionando intereses desde una lista de etiquetas predefinidas y describiendo sus expectativas. Los datos se almacenan y procesan automáticamente.
  - **Editar información**: Un estudiante puede buscarse por su ID, cargar su información actual y modificar nombre, intereses o descripción. Los cambios se guardan y actualizan los datos procesados.
  - **Consultar información**: Permite consultar los datos de cualquier estudiante ingresando su ID.
  - **Ver cursos disponibles**: Muestra la lista de cursos registrados, con nombre, ID y descripción.
  - **Recomendar cursos**: El estudiante ingresa su ID y el sistema muestra un ranking personalizado de cursos recomendados, calculado en tiempo real según sus intereses y los embeddings generados.
- **Opciones para Docentes**:
  - **Registrar curso**: Permite registrar un nuevo curso, ingresando nombre y descripción.
  - **Editar curso**: Permite buscar un curso por ID, cargar su información y editar nombre o descripción.
  - **Ver cursos disponibles**: Visualiza todos los cursos registrados, con detalles ampliados para docentes.
- **Interacción y validación**: La interfaz valida los campos obligatorios (por ejemplo, nombre no vacío) y muestra mensajes de éxito o error según la acción realizada.
- **Actualización dinámica**: Los formularios y listados se actualizan dinámicamente según las acciones del usuario, garantizando una experiencia fluida.

Toda la lógica de la interfaz se encuentra en el archivo `app/webapp.py`, que conecta la capa visual con la API de recomendación y los módulos de procesamiento de datos. Esto permite que tanto estudiantes como docentes gestionen y consulten información, y accedan a recomendaciones personalizadas de manera centralizada y amigable.

6. Casos de Uso

A continuación se describen los principales casos de uso del sistema, detallando el flujo y las acciones involucradas en cada uno:

- **Registro de un nuevo estudiante:**
  - El usuario selecciona el rol de Estudiante y elige la opción Registrar".
  - Completa un formulario con su nombre, selecciona intereses desde una lista de etiquetas predefinidas y puede agregar una descripción personalizada de sus expectativas.
  - Al enviar el formulario, el sistema valida los datos y almacena la información en la base de datos (archivo CSV), generando automáticamente los tags y embeddings asociados.
  - El estudiante queda registrado y listo para recibir recomendaciones personalizadas.
- **Edición de información de estudiante:**
  - El estudiante ingresa su ID y carga su información actual.
  - Puede modificar su nombre, intereses o descripción.
  - Al guardar los cambios, el sistema actualiza los datos y recalcula los embeddings y recomendaciones asociadas.
- **Consulta de información de estudiante:**
  - Permite a cualquier usuario consultar los datos de un estudiante ingresando su ID.
  - Se muestra la información registrada, incluyendo intereses y descripción.
- **Visualización de cursos disponibles:**
  - Tanto estudiantes como docentes pueden ver la lista completa de cursos registrados.
  - Se muestran detalles como nombre, ID y descripción de cada curso.
- **Recomendación personalizada de cursos:**
  - El estudiante ingresa su ID y solicita recomendaciones.
  - El sistema calcula la similitud entre los intereses del estudiante y los cursos disponibles usando embeddings y similitud coseno.
  - Se muestra un ranking de los cursos más recomendados, junto con el puntaje de afinidad y detalles de cada curso.
- **Registro y edición de cursos (Docente):**
  - El docente puede registrar un nuevo curso ingresando nombre y descripción, o editar un curso existente buscando por ID.

- Los cambios se reflejan en la base de datos y se actualizan los tags y embeddings del curso automáticamente.
- Los cursos editados o nuevos quedan disponibles para ser recomendados a los estudiantes.

Estos casos de uso cubren los principales flujos de interacción previstos en la plataforma, asegurando una experiencia completa tanto para estudiantes como para docentes.

7. Requisitos

- Python 3.8 o superior
- Dependencias listadas en requirements.txt:
  - pandas
  - sentence-transformers
  - spacy
  - flask
  - streamlit
  - dotenv
  - requests
  - huggingface-hub
- Acceso a internet para usar la API de OpenRouter (opcional, solo para sugerencia automática de tags)

8. Validación Experimental

La validación experimental del sistema se estructura en torno a los siguientes aspectos clave: extracción de etiquetas (tags) y recomendaciones personalizadas.

8.1. Extracción de Tags

Ejemplo de Cursos:

- **Curso:** Inteligencia Artificial
  Descripción: Introducción a los conceptos y técnicas de IA, aprendizaje automático, redes neuronales y aplicaciones.
  Tags extraídos (NLP): aprendizaje, red, inteligencia, aplicación, concepto
  Tags IA (LLM): inteligencia artificial, machine learning, redes neuronales, automatización, aprendizaje automático
- **Curso:** Criptografía
  Descripción: Este curso explora los principios matemáticos detrás de la criptografía moderna, incluyendo algoritmos de cifrado, funciones hash y protocolos de seguridad.
  Tags extraídos (NLP): criptografia, algoritmo cifrado, algoritmo, cifrado, función, hash, protocolo, seguridad
  Tags IA (LLM): criptografía, seguridad, cifrado, funciones hash, protocolos

8.2. Ejemplo de Estudiantes

- **Estudiante:** Valentina
  Descripción: Me apasiona el big data y la ciencia de datos. Quiero mejorar mis habilidades en estadística y visualización.
  Intereses: big data, analisis de datos, estadistica, visualizacion, ciencia de datos
  Tags extraídos: analisis de dato, big, big datar, ciencia, ciencia de dato, data, dato, estadistica, estadisticar, visualizacion

- **Estudiante:** Martí
  Descripción: Me interesaría saber cómo funciona el internet y las redes de computadoras.
  Intereses: redes
  Tags extraídos: computadora, internet, red

8.3. Recomendación

```
Matriz de afinidad (similitud coseno):

Top 3 cursos recomendados para el estudiante 1:
CursoID: 1 | Score: 0.828
CursoID: 8 | Score: 0.776
CursoID: 3 | Score: 0.758

Top 3 cursos recomendados para todos los estudiantes:
Estudiante 1: ['CursoID 1 (score 0.828)', 'CursoID 8 (score 0.776)', 'CursoID 3 (score 0.758)']
Estudiante 2: ['CursoID 2 (score 0.842)', 'CursoID 4 (score 0.824)', 'CursoID 5 (score 0.708)']
Estudiante 3: ['CursoID 8 (score 0.809)', 'CursoID 3 (score 0.768)', 'CursoID 1 (score 0.725)']
Estudiante 4: ['CursoID 5 (score 0.839)', 'CursoID 8 (score 0.725)', 'CursoID 1 (score 0.722)']
Estudiante 5: ['CursoID 3 (score 0.768)', 'CursoID 1 (score 0.743)', 'CursoID 8 (score 0.736)']
Estudiante 6: ['CursoID 1 (score 0.799)', 'CursoID 3 (score 0.749)', 'CursoID 8 (score 0.749)']
Estudiante 7: ['CursoID 3 (score 0.682)', 'CursoID 5 (score 0.609)', 'CursoID 1 (score 0.607)']
Estudiante 8: ['CursoID 1 (score 0.735)', 'CursoID 8 (score 0.701)', 'CursoID 6 (score 0.670)']
Estudiante 9: ['CursoID 5 (score 0.776)', 'CursoID 3 (score 0.757)', 'CursoID 8 (score 0.721)']
Estudiante 10: ['CursoID 4 (score 0.781)', 'CursoID 8 (score 0.772)', 'CursoID 6 (score 0.767)']
Estudiante 11: ['CursoID 1 (score 0.778)', 'CursoID 4 (score 0.771)', 'CursoID 5 (score 0.696)']
Estudiante 12: ['CursoID 8 (score 0.717)', 'CursoID 1 (score 0.686)', 'CursoID 5 (score 0.657)']
Estudiante 13: ['CursoID 1 (score 0.731)', 'CursoID 3 (score 0.657)', 'CursoID 8 (score 0.634)']
Estudiante 14: ['CursoID 6 (score 0.815)', 'CursoID 8 (score 0.790)', 'CursoID 2 (score 0.764)']
Estudiante 15: ['CursoID 3 (score 0.775)', 'CursoID 1 (score 0.753)', 'CursoID 2 (score 0.737)']
Estudiante 16: ['CursoID 2 (score 0.859)', 'CursoID 1 (score 0.712)', 'CursoID 3 (score 0.679)']
Estudiante 17: ['CursoID 8 (score 0.901)', 'CursoID 3 (score 0.706)', 'CursoID 1 (score 0.675)']
```

Top Recomendaciones para Estudiantes:

- **Estudiante:** Valentina
  Cursos recomendados:

  1. Matemática Discreta (ID: 3) – Score: 0.809
     Lógica, conjuntos, grafos y combinatoria para ciencias de la computación.
  2. Base de Datos (ID: 5) – Score: 0.782
     Modelado, diseño y administración de bases de datos relacionales y no relacionales.
  3. Optimización (ID: 8) – Score: 0.763
     Este curso se enfoca en técnicas para encontrar el mejor resultado en problemas matemáticos, como programación lineal y no lineal, algoritmos genéticos y optimización convexa.

- **Estudiante:** Martí
  Cursos recomendados:

  1. Redes de Computadoras (ID: 2) – Score: 0.838
     Estudio de protocolos, arquitecturas y seguridad en redes de computadoras.
  2. Inteligencia Artificial (ID: 1) – Score: 0.712
     Introducción a los conceptos y técnicas de IA, aprendizaje automático, redes neuronales y aplicaciones.
  3. Matemática Discreta (ID: 3) – Score: 0.698
     Lógica, conjuntos, grafos y combinatoria para ciencias de la computación.

9. Créditos

Desarrollado para la asignatura de Modelos Matemáticos Aplicados.