INSTITUTO TECNOLÓGICO DE CULIACÁN

INGENIERÍA EN SISTEMA COMPUTACIONALES

TAREA

Resumen Sistema de Recomendación

NOMBRE DE LA MATERIA:

Inteligencia Artificial

ALUMNOS:

Cruz Méndez Eymardh Sahid

Cardenas Quiñonez Angel

NOMBRE DEL DOCENTE:

Zuriel Dathan Mora Felix

Culiacán, Sin., 16 de Febrero de 2025

Cómo Construir un Sistema de Recomendación

Un sistema de recomendación suele seguir estas etapas:

- **Recolección de datos**: Fuentes como historial de usuarios, interacciones (clicks, compras), metadatos (género, categorías) o contexto (ubicación, dispositivo).
- **Preprocesamiento**: Limpieza (manejo de valores nulos), normalización y creación de *features* (ej: embeddings de texto).
- **Selección del modelo**:
    - **Filtrado colaborativo**: Basado en similitudes entre usuarios/ítems (ej: SVD, KNN).
    - **Basado en contenido**: Recomienda ítems similares a los que el usuario ya interactuó (ej: TF-IDF, NLP).
    - **Híbridos**: Combina ambos enfoques (ej: modelos de deep learning con embeddings).
- **Implementación**: Entrenamiento con librerías como TensorFlow o Scikit-learn.
- **Evaluación**: Métricas como RMSE, precisión@k, AUC-ROC o pruebas A/B.
- **Despliegue**: APIs en tiempo real (Flask, FastAPI) o procesamiento por lotes (Airflow).

Tecnologías y Frameworks

- **Lenguajes**: Python (dominante), Java/Scala (para escalabilidad).
- **Frameworks de ML**:
    - **Scikit-learn**: Para modelos clásicos (KNN, regresión).
    - **TensorFlow/PyTorch**: Redes neuronales (Autoencoders, Transformers).

    - **Apache Spark MLlib**: Procesamiento distribuido (ALS para filtrado colaborativo).
    - **Surprise/LightFM**: Especializados en recomendaciones.
- **Bases de datos**: PostgreSQL (extensiones como pg_vector), Redis (caché), Cassandra (escalabilidad).
- **Streaming**: Apache Kafka o AWS Kinesis para datos en tiempo real.

Herramientas en AWS y Google Cloud Platform

- **AWS**:
    - **SageMaker**: Entrenamiento y despliegue de modelos (incluye algoritmos como Factorization Machines).
    - **Personalize**: Servicio gestionado para crear recomendaciones sin código.
    - **EMR/Glue**: Procesamiento de big data (Spark, Hadoop).
    - **Lambda/S3**: Para arquitecturas serverless y almacenamiento.
- **GCP**:
    - **Vertex AI**: Plataforma unificada para entrenar modelos (TensorFlow, XGBoost).
    - **Recommendations AI**: Servicio especializado con integración de BigQuery.
    - **Dataflow**: Procesamiento de flujos (Apache Beam).
    - **Bigtable**: Base de datos NoSQL para baja latencia.

Algoritmos y Frameworks para Optimización de Recursos

- **Optimización de cómputo**:
    - **Hyperopt/Optuna**: Ajuste automático de hiperparámetros.
    - **TensorFlow Lite/ONNX**: Modelos ligeros para edge computing.
    - **Pruning/Cuantización**: Reducir tamaño de modelos (ej: TensorFlow Model Optimization).
- **Arquitecturas eficientes**:
    - **Wide & Deep (Google)**: Combina regresión lineal y redes neuronales.
    - **Two-Tower (Facebook)**: Embeddings separados para usuarios e ítems.
- **Gestión de recursos**:
    - **Kubeflow**: Orchestación de pipelines en Kubernetes.
    - **Horovod**: Entrenamiento distribuido.
    - **AutoML (GCP/AWS)**: Automatiza diseño de modelos para reducir costos.

Fuentes de Consulta

1. **AWS Documentation**: Amazon SageMaker, AWS Personalize.
2. **GCP Documentation**: Vertex AI, Recommendations AI.
3. **Frameworks**: TensorFlow, Apache Spark MLlib.