Para iniciar el resumen, primero tenemos que saber que es un Agente inteligente, la explicación del compañero en la clase lo dejo muy claro, pero no está demás recodarlo

Los agentes inteligentes son sistemas que perciben su entorno, procesan la información y actúan de manera autónoma para lograr un objetivo. Se pueden basar en reglas predefinidas o utilizar inteligencia artificial para aprender y mejorar su desempeño con el tiempo, algunas de sus características son:

Autonomía – Pueden operar sin intervención humana.
Percepción – Captan información del entorno mediante sensores o datos.
Razonamiento – Procesan la información para tomar decisiones.
Acción – Ejecutan tareas para alcanzar sus objetivos.
Adaptabilidad – Pueden aprender y mejorar con la experiencia.

Y algunos tipos de estos también son:

Reactivos: Responden a estímulos sin planificar (ej. un termostato).
Basados en objetivos: Toman decisiones para alcanzar metas específicas.
Basados en utilidad: Optimizan acciones para maximizar beneficios.
Aprendices: Usan técnicas como Machine Learning para mejorar su desempeño.

Un ejemplo común es **Siri** o **Google Assistant**, que reciben comandos de voz, procesan la información y responden de manera autónoma.

Estos están relacionados con varias tecnologías claves que son

1. **Inteligencia Artificial (IA) y Machine Learning**: Para tomar decisiones y aprender de los datos.
2. **Procesamiento del Lenguaje Natural (NLP)**: Para interactuar con humanos mediante lenguaje.
3. **Sistemas de Recomendación**: Para ofrecer sugerencias personalizadas.
4. **Visión por Computadora**: Para analizar imágenes o videos.
5. **Robótica**: En agentes físicos que interactúan con el entorno.
6. **Computación en la Nube y Edge Computing**: Para escalabilidad y procesamiento rápido.
7. **Bases de Datos y Big Data**: Para manejar y analizar grandes volúmenes de datos.
8. **APIs y Microservicios**: Para integración con otros sistemas.

9. **Blockchain**: Para seguridad y transparencia en transacciones.
10. **Frameworks y Herramientas**: Como TensorFlow, PyTorch, o Dialogflow para desarrollo.

Estas sirven para crear agentes inteligentes capaces de realizar tareas complejas, aprender y mejorar continuamente.

Para saber cómo construir un sistema de recomendación primero debemos saber cuáles existen, aquí unos ejemplos

Filtrado Colaborativo: Basado en el comportamiento de usuarios similares.

Basado en usuarios: Recomienda ítems que usuarios similares han preferido.

Basado en ítems: Recomienda ítems similares a los que el usuario ya ha interactuado.

Filtrado Basado en Contenido: Recomienda ítems similares en características (por ejemplo, género de películas o palabras clave).

Sistemas Híbridos: Combina filtrado colaborativo y basado en contenido para mejorar la precisión.

Basado en Popularidad: Recomienda ítems populares en general (por ejemplo, tendencias).

Basado en Conocimiento: Usa reglas o conocimiento específico del dominio (por ejemplo, recomendaciones de viajes basadas en preferencias explícitas).

Estos son los algoritmos comunes

- **K-Nearest Neighbors (KNN)**: Para similitud.
- **Factorización de Matrices (SVD, ALS)**: Para datos dispersos.
- **Redes Neuronales**: Autoencoders, Wide & Deep, Transformers.
- **Clustering (K-Means)**: Para agrupar usuarios o ítems.

Las tecnologías

- **Lenguajes**: Python (principalmente), R, Java.
- **Bases de Datos**: MySQL, MongoDB, Neo4j.
- **Big Data**: Apache Spark, Hadoop.
- **Cloud**: AWS, Google Cloud, Azure.

Los frameworks y librerias

- **Machine Learning**: Scikit-learn, TensorFlow, PyTorch.
- **Recomendación**: Surprise, Implicit, Cornac.
- **Procesamiento de Datos**: Pandas, NumPy, Spark MLlib.

Y estos son los pasos para construirlo

1. **Recopilar Datos**: Interacciones, contenido y perfiles de usuarios.
2. **Preprocesar**: Limpieza, normalización y manejo de datos faltantes.
3. **Elegir Modelo**: Filtrado colaborativo, basado en contenido o híbrido.
4. **Entrenar y Evaluar**: Usar métricas como precisión, recall o RMSE.
5. **Desplegar**: Integrar con APIs (Flask, FastAPI) y usar Docker.
6. **Monitorear y Mejorar**: Ajustar hiperparámetros y actualizar con nuevos datos.

Herramientras de Google Cloud Platform (GCP) y Amazon Web Services (AWS)

Google Cloud Platform (GCP)

1. **Recommendations AI**: Servicio especializado para recomendaciones personalizadas.
2. **Vertex AI**: Plataforma unificada para entrenar y desplegar modelos de machine learning.
3. **BigQuery**: Base de datos analítica para grandes volúmenes de datos.
4. **TensorFlow Extended (TFX)**: Para pipelines de machine learning en producción.
5. **Cloud Functions**: Ejecución de código sin servidor (serverless).

Amazon Web Services (AWS)

1. **Amazon Personalize**: Servicio especializado para recomendaciones personalizadas.
2. **Amazon SageMaker**: Plataforma para construir y desplegar modelos de machine learning.
3. **S3**: Almacenamiento escalable para datos.
4. **DynamoDB**: Base de datos NoSQL para interacciones de usuarios.

5. **AWS Lambda**: Ejecución de código sin servidor (serverless).

Las diferencias de estas 2 es que GCP es ideal para la integración con herramientas de IA y machine learning y AWS es mas como big data y machine learning también

Algortimos y framewords para la optimización de recursos

Algoritmos Clave

1. **Optimización Clásica**:
    - Programación Lineal (LP), Entera (IP) y No Lineal (NLP).
2. **Heurísticas y Metaheurísticas**:
    - Algoritmos Genéticos, Recocido Simulado, Optimización por Enjambre de Partículas (PSO).
3. **Machine Learning**:
    - Refuerzo (Reinforcement Learning), Redes Neuronales, Gradient Boosting (XGBoost, LightGBM).
4. **Grafos**:
    - Dijkstra, Floyd-Warshall, Algoritmo Húngaro.

Frameworks y Herramientas

1. **Librerías**:
    - SciPy, PuLP, CVXPY, Google OR-Tools.
2. **Machine Learning**:
    - TensorFlow, PyTorch, Scikit-learn.
3. **Simulación**:
    - SimPy, AnyLogic.
4. **Solvers**:
    - Gurobi, CPLEX, AMPL.
5. **Plataformas Cloud**:
    - Google Cloud (Vertex AI), AWS (SageMaker), Azure (Machine Learning).

Con este conjunto de frameworks, herramientas y algoritmos permite resolver problemas complejos de optimización de recursos de una manera muy eficiente.