Manual de Estudio: AWS Certified AI Practitioner

Dominio 3.1: Consideraciones de Diseño de Aplicaciones con Modelos Fundacionales

Material de Preparación Detallado

1. Introducción a los Modelos Fundacionales (FM)

Los Modelos Fundacionales se diferencian del Machine Learning tradicional por su **adaptabilidad** y naturaleza de **uso general**. Mientras que el ML tradicional se enfoca en tareas específicas (ej. análisis de sentimiento), los FM pueden realizar múltiples tareas (NLP, visión artificial, generación de código) basándose únicamente en la petición de entrada.

2. Criterios de Selección de Modelos

Al diseñar una aplicación, el arquitecto de IA debe equilibrar varios factores críticos:

2.1. El Triángulo de Compromiso: Coste, Latencia y Rendimiento

- **Coste:** Incluye el entrenamiento previo (muy caro), el refinamiento y el coste de inferencia (hardware/tokens). Debes decidir si un 1 % extra de exactitud justifica miles de dólares adicionales en infraestructura.
- **Latencia:** Crucial para aplicaciones en tiempo real. Un modelo muy complejo puede ser lento. Por ejemplo, en vehículos autónomos, un modelo con alta latencia es inaceptable, descartando algoritmos lentos como KNN en grandes dimensiones.
- **Modalidad:** Determinar si el modelo debe ser unimodal (solo texto) o multimodal (texto e imágenes) y si requiere soporte multilingüe.

2.2. Complejidad, Interpretabilidad y Explicabilidad

- **Cajas Negras:** Los FM son extremadamente complejos, lo que impide la **interpretabilidad** (entender matemáticamente cada coeficiente, como en una regresión lineal).
- **Explicabilidad:** Técnica que intenta explicar una çaja negra.ªproximándola localmente con un modelo más simple. Si la transparencia total es un requisito legal, los FM pueden no ser la mejor opción.

3. Parámetros de Inferencia

Son valores que ajustamos para influir en la respuesta del modelo sin cambiar sus pesos internos. En Amazon Bedrock, los más importantes son:

Parámetro	Efecto en la Respuesta
Temperatura	Controla la aleatoriedad. Valores bajos (0.1) dan respuestas deterministas; valores altos (1.0) dan respuestas creativas/diversas.
Top K	Limita el modelo a elegir entre los K tokens más probables. Reduce la posibilidad de respuestas incoherentes.
Top P (Nucleus)	El modelo elige del subconjunto de tokens cuya probabilidad acumulada suma P. Ayuda a mantener la diversidad dinámica.
Longitud	Limita el número de tokens en la salida (*max tokens*).
Secuencias de parada	Define tokens específicos que, al aparecer, hacen que el modelo deje de escribir.

4. Personalización: RAG y Bases de Datos Vectoriales

4.1. RAG (Generación Aumentada por Recuperación)

Es una técnica para mitigar **alucinaciones** y proporcionar datos actualizados. En lugar de confiar solo en el entrenamiento previo, el modelo consulta una base de conocimientos externa.

4.2. Bases de Datos Vectoriales en AWS

Almacenan **incrustaciones (embeddings)**, que son representaciones matemáticas del significado de los datos. AWS ofrece varios servicios para esto:

- **Amazon OpenSearch Serverless:** Motor vectorial dedicado para búsqueda semántica.
- **Amazon RDS para PostgreSQL:** Mediante la extensión `pgvector`.
- **Amazon Aurora:** Soporte para almacenamiento vectorial.
- **Amazon Bedrock Knowledge Bases:** Solución completamente administrada que implementa RAG conectando modelos a datos privados de S3.

5. Agentes en Amazon Bedrock

Los modelos por sí solos no pueden ejecutar acciones (ej. comprar un billete). Los **Agentes** actúan como orquestadores que:

1. Desglosan tareas complejas en varios pasos.

2. Llaman automáticamente a **APIs** externas para realizar acciones.
3. Consultan bases de conocimiento para obtener contexto adicional.
4. Generan la lógica de orquestación necesaria sin intervención manual del desarrollador.