EXTRACCIÓN DE CONOCIMIENTOS EN BASES DE DATOS

ING. LUIS ENRIQUE MASCOTE CANO

Universidad Tecnológica de Chihuahua

INVESTIGACIÓN DE LOS TIPOS DE APLICACIONES, PROCESAMIENTO Y HERRAMIENTAS

Lic. Ricardo Hernández Martínez

Fecha de Entrega: 25/SEPTIEMBRE/2025

Índice

Introducción

Este informe compara y documenta los cuatro dominios estrechamente relacionados: Inteligencia Artificial (IA), Machine Learning (ML), Data Mining (DM) y Big Data. Para cada dominio se presentan: una definición breve, al menos tres tipos de aplicaciones con ejemplos concretos, modalidades de procesamiento (batch vs. streaming) y una selección de 4–5 herramientas/tecnologías clave con su funcionalidad y casos de uso. Al final se incluye una conclusión comparativa y referencias bibliográficas en formato APA.

Inteligencia Artificial (IA)

IA es el campo que busca crear sistemas capaces de realizar tareas que, si las realizara un humano, requerirían inteligencia. Incluye desde sistemas simbólicos y reglas hasta modelos de aprendizaje profundo (deep learning) y agentes inteligentes

Tipos de aplicaciones

1. **Visión por computador**
    - **Descripción:** Procesamiento de imágenes y vídeo para extraer información (detección, reconocimiento, segmentación, etc.).
    - **Ejemplo concreto:** Sistemas de inspección visual en fábricas (p. ej., uso de modelos de detección de defectos en líneas de producción) o la tecnología de visión usada por empresas como **Siemens** o startups como **Gleematic/Roboflow** en inspección industrial.
2. **Procesamiento de Lenguaje Natural (NLP)**
    - **Descripción:** Comprensión y generación de texto y lenguaje hablado (chatbots, resumen automático, búsqueda semántica, traducción automática, etc.).
    - **Ejemplo concreto:** Aplicaciones de atención al cliente basadas en LLMs (por ejemplo, chatbots empresariales que usan modelos como GPT o soluciones de **Hugging Face**).
3. **Agentes y robótica autónoma**
    - **Descripción:** Agentes que perciben, planean y actúan en entornos físicos o virtuales, integrando percepción, toma de decisiones y control.
    - **Ejemplo concreto:** Robots autónomos en almacenes (Amazon Robotics), vehículos autónomos (Waymo/Tesla) o drones con navegación autónoma.

4. **Generación de contenido (Generative AI)**
    - **Descripción:** Creación automática de texto, imágenes, audio o video a partir de instrucciones (prompt-to-content).
    - **Ejemplo concreto:** Plataformas que usan modelos generativos para marketing y diseño (p. ej., campañas que emplean imágenes generadas por IA; soluciones de Google Cloud y OpenAI usadas por compañías creativas).

Procesamiento: batch vs. streaming en IA

- **Batch (por lotes):** Se emplea típicamente en fases de entrenamiento de modelos grandes (entrenamiento offline con datasets completos, re-entrenamientos periódicos, validación, experimentación). Batch permite procesar grandes volúmenes de datos con optimización de throughput.
    - **Cuándo usar:** Entrenamiento de modelos de deep learning, reentrenamientos programados, pipelines ETL previos al modelado.
- **Streaming (tiempo real / near-real-time):** Se emplea en inferencia en línea, sistemas de recomendación en tiempo real, detección de fraude en transacciones, sistemas de control en robótica donde la latencia es crítica.
    - **Cuándo usar:** Inferencia en producción con baja latencia, monitorización continua de modelos (drift detection), sistemas de alerta inmediata.

Ejemplos prácticos: entrenar una red de visión por computador en batch usando GPUs durante horas/días; desplegar el modelo para inferencia en streaming en una línea de producción con latencia <100 ms.

Herramientas y tecnologías

1. **TensorFlow**

- **Funcionalidad clave:** Framework de código abierto para construir, entrenar y desplegar modelos de ML/IA (incluye Keras y TFX para producción).
- **Caso de uso:** Modelado deep learning (visión, NLP), despliegue en edge y servidores, soporte para MLOps. (Fuente: TensorFlow docs)

2. **PyTorch**
    - **Funcionalidad clave:** Biblioteca para deep learning con enfoque en dinamismo y facilidad de investigación; soporte fuerte para entrenamiento distribuido y producción (TorchServe).
    - **Caso de uso:** Investigación y prototipado rápido en NLP y visión, escalado a entrenamiento distribuido. (Fuente: PyTorch docs)
3. **Hugging Face (Transformers)**
    - **Funcionalidad clave:** Biblioteca y ecosistema de modelos preentrenados (transformers) para NLP, con repositorio de modelos y utilidades de despliegue.
    - **Caso de uso:** Chatbots, clasificación de texto, RAG (recall-augmented generation). (Fuente: Hugging Face)
4. **OpenAI / APIs LLM**
    - **Funcionalidad clave:** Acceso a grandes modelos de lenguaje como servicio (completions, embeddings, etc.).
    - **Caso de uso:** Generación de texto, resumen automático, asistentes conversacionales.
5. **ROS / Frameworks de robótica**
    - **Funcionalidad clave:** Middleware y herramientas para construir sistemas robóticos integrados (percepción, control, simulación).
    - **Caso de uso:** Robots autónomos en investigación y producción.

Machine Learning (ML)

ML es una subdisciplina de la IA que estudia algoritmos y modelos que permiten a las máquinas aprender patrones a partir de datos y hacer predicciones o tomar decisiones sin programación explícita para cada tarea.

Tipos de aplicaciones

1. **Clasificación y detección de fraude**
    - **Descripción:** Modelos supervisados que detectan eventos anómalos o transacciones fraudulentas.
    - **Ejemplo concreto:** Sistemas de detección de fraude en tarjetas de crédito (bancos y procesadores de pago usan modelos de gradient boosting o redes neuronales).
2. **Sistemas de recomendación**
    - **Descripción:** Recomendaciones personalizadas (productos, contenido) utilizando filtrado colaborativo, modelos híbridos y embeddings.
    - **Ejemplo concreto:** Recomendaciones en e-commerce (Amazon, Netflix) que mezclan modelos offline y sistemas de ranking en tiempo real.
3. **Regresión y forecasting**
    - **Descripción:** Predicción de valores continuos como demanda, precios o métricas operativas.
    - **Ejemplo concreto:** Forecast de demanda para inventario (retail) o predicción de consumo energético.
4. **Segmentación y clustering**
    - **Descripción:** Agrupar clientes o eventos según similitud para marketing y análisis.

- **Ejemplo concreto:** Segmentación de clientes para campañas personalizadas.

Procesamiento: batch vs. streaming en ML

- **Batch:** Entrenamiento de modelos (supervisado/ensamble), evaluación y backtesting se realizan usualmente en batch. Las tareas de feature engineering sobre grandes históricos suelen ser por lotes.
  - **Cuándo usar:** Reentrenamiento, feature store recomputations, experiments.
- **Streaming:** Se usa para inferencia en tiempo real, aprendizaje en línea (online learning) y actualizaciones incrementales (p. ej., modelos que adaptan pesos con nuevos datos). También para pipelines de features en near-real-time.
  - **Cuándo usar:** Scoring en tiempo real (detección de fraude), sistemas de recomendaciones en sesiones de usuario, actualizaciones de modelos ligeros estilo online learning.

Herramientas y tecnologías

1. **scikit-learn**
   - **Funcionalidad clave:** Biblioteca Python para algoritmos clásicos de ML (regresión, clasificación, clustering, selección de modelos).
   - **Caso de uso:** Prototipado rápido de modelos tradicionales, pipelines de ML para datos tabulares. (Fuente: scikit-learn docs)
2. **XGBoost / LightGBM / CatBoost**
   - **Funcionalidad clave:** Implementaciones eficientes de boosting de árboles (alto rendimiento para tabular data).
   - **Caso de uso:** Competencias ML, producción para predicción tabular (fraude, churn). (Fuentes: documentación de cada proyecto)

3. **MLflow / TFX / Kubeflow**
    - **Funcionalidad clave:** Plataformas de MLOps para tracking de experimentos, packaging, despliegue y gestión de modelos.
    - **Caso de uso:** Reproducibilidad y despliegue en pipeline de producción.
4. **TensorFlow / PyTorch (uso en ML moderno)**
    - **Funcionalidad clave:** Además de deep learning, se usan para modelos complejos y redes neuronales aplicadas a problemas típicos de ML.
    - **Caso de uso:** Modelos de series temporales, embeddings para tabular+texto, modelos híbridos.
5. **Feature stores (Feast, Tecton)**
    - **Funcionalidad clave:** Sistemas para almacenar, servir y garantizar consistencia de features entre training e inferencia.
    - **Caso de uso:** Evitar discrepancias entre features offline y online y facilitar pipelines de producción.

Data Mining (DM)

Data Mining es el proceso de descubrir patrones, relaciones y conocimiento útil a partir de datos. Suele enfocarse en técnicas exploratorias, extracción de reglas, clustering y modelos predictivos — con un énfasis en el descubrimiento más que en el despliegue masivo.

Tipos de aplicaciones

1. **Análisis de asociación (market basket analysis)**
    - **Descripción:** Descubrir reglas de co-ocurrencia (p. ej., A → B) para estrategias de cross-sell y layout de tiendas.
    - **Ejemplo concreto:** Reglas de cesta de la compra en retail para promociones.
2. **Segmentación de clientes (clustering)**
    - **Descripción:** Agrupar clientes en segmentos con comportamientos similares para campañas de marketing.
    - **Ejemplo concreto:** Agrupación de clientes por valor de vida (CLV) y comportamiento de compra.
3. **Detección de patrones y outliers**
    - **Descripción:** Identificación de anomalías y patrones inusuales en logs, sensores o transacciones.
    - **Ejemplo concreto:** Detección de fraudes o anomalías en sensores industriales.
4. **Minería de secuencias y análisis temporal**
    - **Descripción:** Analizar secuencias de eventos para entender comportamientos y procesos.
    - **Ejemplo concreto:** Análisis de clics en navegadores o secuencia de pasos en procesos de negocio.

Procesamiento: batch vs. streaming en DM

- **Batch:** Tradicionalmente, las tareas de data mining (minería de reglas, clustering a gran escala) se realizan en batch con muestras históricas.
    - **Cuándo usar:** Exploración profunda, generación de reglas y modelos que no requieren respuesta inmediata.
- **Streaming:** Cada vez más útil para detección de anomalías en tiempo real y minería de patrones en flujos (p. ej., logs de eventos), donde se necesita respuesta inmediata.
    - **Cuándo usar:** Monitorización de seguridad, detección de fraude en tiempo real, análisis de clickstream continuo.

Herramientas y tecnologías

1. **Weka**
    - **Funcionalidad clave:** Conjunto de herramientas para data mining y aprendizaje automático clásico con GUI para experimentación.
    - **Caso de uso:** Enseñanza, prototipado rápido y exploración de algoritmos clásicos. (Fuente: Weka docs)
2. **RapidMiner**
    - **Funcionalidad clave:** Plataforma visual para diseñar workflows de minería de datos, incluye conectores empresariales.
    - **Caso de uso:** Usuarios empresariales que necesitan pipelines sin programar (ETL + modelado + scoring). (Fuente: RapidMiner)
3. **KNIME / Orange**
    - **Funcionalidad clave:** Plataformas visuales para análisis y minería de datos con muchos nodos para transformaciones y ML.
    - **Caso de uso:** Análisis exploratorio, prototipado y automatización de workflows analíticos. (Fuente: KNIME)

4. **R (paquetes: arules, caret, tidyverse)**
    - **Funcionalidad clave:** Lenguaje y ecosistema para análisis estadístico y minería de datos, con paquetes especializados.
    - **Caso de uso:** Análisis exploratorio avanzado, minería de reglas, visualización estadística.
5. **Python (pandas, mlxtend, scikit-learn)**
    - **Funcionalidad clave:** Bibliotecas para manipulación de datos, extracción de reglas, clustering y modelado.
    - **Caso de uso:** Pipelines reproducibles y automatización de tareas de minería y análisis.

Big Data

Big Data se refiere a la gestión, almacenamiento y procesamiento de volúmenes de datos que superan la capacidad de las soluciones tradicionales, tanto por tamaño como por velocidad y variedad. Incluye arquitecturas distribuidas y tecnologías para escalabilidad y tolerancia a fallos.

Tipos de aplicaciones

1. **Almacén de datos y analytics a gran escala**
    - **Descripción:** Agregación y análisis de datos empresariales para reporting, BI y análisis histórico.
    - **Ejemplo concreto:** Data lakes y pipelines en empresas que analizan terabytes/petabytes (p. ej., telcos y plataformas digitales).
2. **Procesamiento de logs y telemetría (observability)**
    - **Descripción:** Ingesta y análisis de grandes volúmenes de logs y métricas para monitorización y diagnóstico.
    - **Ejemplo concreto:** Plataformas de observabilidad que almacenan y procesan grandes tasas de eventos (p. ej., ELK stack + soluciones cloud).
3. **Stream processing a escala (event-driven architectures)**
    - **Descripción:** Procesamiento continuo de flujos para alertas, enriquecimiento y acciones en tiempo real.
    - **Ejemplo concreto:** Sistemas de procesamiento de eventos en banca o telecom para detección de fraude y enrutamiento en tiempo real.
4. **Machine learning a escala (feature engineering distribuido)**
    - **Descripción:** Preparación de features y entrenamiento de modelos con datasets muy grandes.

- **Ejemplo concreto:** Entrenamiento distribuido de modelos con Spark/Databricks sobre petabytes de datos.

Procesamiento: batch vs. streaming en Big Data

- **Batch:** Ideal para trabajos que requieren procesar grandes volúmenes en ventanas temporales (ETL nocturno, agregaciones históricas, entrenamientos masivos).
    - **Cuándo usar:** Reporting, ETL/ELT, backfills, cargas masivas a data warehouses.
- **Streaming:** Ideal cuando se requiere baja latencia y procesamiento continuo (event-processing, detección de anomalías, enriquecimiento en tiempo real).
    - **Cuándo usar:** Pipelines de eventos, sistemas de personalización en tiempo real, monitorización continua.

Nota: En arquitecturas modernas se combinan ambos (lambda o kappa architectures) para aprovechar fortalezas de cada enfoque.

Herramientas y tecnologías

1. **Apache Hadoop (HDFS + MapReduce + YARN)**
    - **Funcionalidad clave:** Almacenamiento distribuido y procesamiento por lotes para grandes datasets.
    - **Caso de uso:** Data lakes basados en HDFS y procesamiento por MapReduce / ecosistema Hadoop. (Fuente: Apache Hadoop)
2. **Apache Spark**
    - **Funcionalidad clave:** Motor de procesamiento en memoria para batch y streaming (Spark Structured Streaming), MLlib para ML distribuido.
    - **Caso de uso:** ETL masivo, procesamiento interactivo y entrenamiento distribuido. (Fuente: Spark docs)

3. **Apache Kafka**
    - **Funcionalidad clave:** Plataforma distribuida de mensajería y streaming para ingestión y transporte de eventos.
    - **Caso de uso:** Backbone de ingesta en arquitecturas event-driven, buffering y pub/sub a escala. (Fuente: Kafka docs)
4. **Flink / Samza / Beam**
    - **Funcionalidad clave:** Frameworks para stream processing con garantías de estado y latencia baja (Flink es popular por exactly-once semánticas en estado manejado).
    - **Caso de uso:** Procesamiento de eventos en tiempo real con requisitos estrictos de estado y latencia. (Fuente: Flink docs)
5. **Bases de datos NoSQL orientadas a Big Data (Cassandra, HBase, ClickHouse)**
    - **Funcionalidad clave:** Almacenamiento horizontalmente escalable para lecturas/ escrituras a gran escala (baja latencia, alta disponibilidad).
    - **Caso de uso:** Time-series a escala, OLAP rápido (ClickHouse), almacenamiento de eventos en alta tasa (Cassandra, HBase).

Conclusión

Los cuatro dominios —IA, ML, DM y Big Data— se solapan considerablemente, pero cada uno tiene un foco y preocupaciones distintas. **IA** es el objetivo más amplio (crear sistemas inteligentes) y suele apoyarse en técnicas de ML y, cuando hay volúmenes grandes de datos, en infraestructuras de Big Data. **ML** se centra en métodos para aprender de datos (modelado, evaluación y despliegue). **DM** pone énfasis en el descubrimiento de patrones y conocimiento útil, siendo frecuentemente más exploratorio. **Big Data** es la capa de infraestructura y arquitectura que permite almacenar y procesar datos a escalas que las tecnologías tradicionales no soportan.

Solapamientos: - ML e IA comparten algoritmos y herramientas (p. ej., TensorFlow/PyTorch); la diferencia principal es el alcance: ML es una técnica, IA es el objetivo. - DM y ML usan muchas de las mismas técnicas analíticas; DM se enfoca más en descubrimiento y ML en predicción y automatización. - Big Data proporciona el sustrato sobre el que ML/IA/DM pueden escalar.

Diferencias importantes: - **Objetivo:** DM → conocimiento; ML → predicción/automación; IA → comportamiento inteligente; Big Data → infraestructura. - **Procesamiento:** Big Data y ML requieren planes claros para batch vs streaming; IA tiende a entrenamientos batch y inferencias streaming.

Importancia de elegir herramientas y modo de procesamiento: - Seleccionar entre batch y streaming depende de requisitos de latencia, consistencia y coste. Por ejemplo, la detección de fraude exige streaming (latencia baja) mientras que un reentrenamiento de modelos se realizará en batch. - La elección de herramientas depende del tipo de problema, volumen de datos y equipo: Spark/Kafka para data-driven architectures; TensorFlow/PyTorch para modelos avanzados; scikit-learn/XGBoost para problemas tabulares; Weka/KNIME/RapidMiner para prototipado y DM.

En resumen, diseñar soluciones efectivas requiere mapear requisitos del negocio a las elecciones correctas de procesamiento y tecnología, combinando técnicas cuando sea necesario (p. ej., training en batch + serving en streaming + feature store para consistencia).

Referencias

TensorFlow. (2023). *TensorFlow Guide*. https://www.tensorflow.org/guide

PyTorch. (s. f.). *PyTorch Documentation*. https://pytorch.org/docs/

scikit-learn developers. (s. f.). *scikit-learn: User Guide*. https://scikit-learn.org/stable/user_guide.html

Apache Hadoop. (s. f.). *Apache Hadoop Project*. https://hadoop.apache.org/

Apache Spark. (s. f.). *Spark Documentation*. https://spark.apache.org/docs/latest/

Datacamp. (2024). *Batch vs. Stream Processing*. https://www.datacamp.com/blog/batch-vs-stream-processing

Databricks. (2025). *Batch vs. streaming data processing in Databricks*. https://docs.databricks.com/