Ciencia de Datos y de BigData.

Nombre: Roberto Bustamante

Fecha: 12/04/2025

Resumen de Ciencia de Datos y Big Data

---

1. Tipos de Análisis para Generación de Conocimiento

Tipo de Análisis	Descripción
Descriptivo	Resume y describe datos para identificar patrones y tendencias.
Exploratorio	Descubre relaciones ocultas y genera hipótesis.
Inferencial	Generaliza conclusiones de una muestra a una población.
Predictivo	Usa datos históricos para predecir eventos futuros.
Prescriptivo	Recomienda acciones óptimas basadas en datos.
Series Temporales	Analiza datos ordenados en el tiempo para predecir tendencias.
Datos No Estructurados	Extrae información de textos, imágenes o videos con NLP y ML.

---

2. Técnicas de Recolección de Datos

Enfoques:

- **Cuantitativo: Datos numéricos (ej: encuestas).**
- **Cualitativo: Datos descriptivos (ej: entrevistas).**

Métodos Tradicionales:

Método	Ejemplo de Uso
Encuestas	Preferencias de consumidores sobre un producto.
Entrevistas	Experiencias de pacientes con enfermedades crónicas.
Observación	Comportamiento de niños en el aula.
Grupos focales	Percepción de una campaña publicitaria.
Revisión de registros	Incidencia de enfermedades respiratorias en hospitales.
Experimentos	Nivel de estrés en trabajadores bajo diferentes condiciones.

Big Data:

Técnica	Herramientas	Ejemplo
Extracción de logs	Apache Flume, Logstash	Monitoreo de tráfico en e-commerce.
Web Scraping	BeautifulSoup, Scrapy	Comparación de precios en tiendas en línea.
Sensores IoT	Apache Kafka, AWS IoT Core	Mantenimiento predictivo en fábricas.
Redes sociales	Twitter API, Hootsuite	Análisis de sentimiento sobre un producto.
Bases de datos (SQL/NoSQL)	Sqoop, MongoDB	Detección de fraudes en transacciones.
APIs externas	REST APIs, Postman	Datos meteorológicos para agricultura.

3. Lenguajes y Frameworks para Ciencia de Datos

Lenguaje	Ventajas	Desventajas
R	Paquetes estadísticos avanzados, visualización.	Lento, sintaxis poco intuitiva.
Python	Versátil, fácil aprendizaje, librerías (pandas).	Tipado dinámico (errores potenciales).
SQL	Eficiente para consultas en bases de datos.	Capacidades analíticas limitadas.
Java	Alto rendimiento, seguridad de tipos.	Verbosidad, menos librerías estadísticas.
Scala	Ideal para Big Data (Apache Spark).	Curva de aprendizaje pronunciada.
Julia	Rendimiento alto, legibilidad.	Comunidad pequeña, pocos paquetes.
MATLAB	Herramientas matemáticas avanzadas.	Costoso, no es de propósito general.

1.Tipos de análisis que se pueden realizar para la generación de conocimiento

1. Análisis descriptivo: Este tipo de análisis se enfoca en describir y resumir los datos de forma clara y concisa. Permite identificar patrones, tendencias y características principales de los datos.

2. Análisis exploratorio: El análisis exploratorio se utiliza para descubrir relaciones y patrones ocultos en los datos. Ayuda a identificar variables relevantes y a generar hipótesis para investigaciones más profundas.

3. Análisis inferencial: Este tipo de análisis se utiliza para hacer inferencias y sacar conclusiones sobre una población a partir de una muestra de datos. Permite generalizar los resultados obtenidos y hacer predicciones.

4. Análisis predictivo: El análisis predictivo se utiliza para predecir eventos o resultados futuros. Se basa en modelos y algoritmos que utilizan datos históricos para hacer proyecciones y estimaciones.

5. Análisis prescriptivo: El análisis prescriptivo se enfoca en encontrar la mejor solución o curso de acción a seguir. Utiliza técnicas de optimización y simulación para tomar decisiones basadas en datos.

6. Análisis de series temporales: Este tipo de análisis se utiliza para analizar datos que están organizados en función del tiempo. Permite identificar patrones y tendencias a lo largo del tiempo y hacer predicciones futuras.

7. Análisis de datos no estructurados: El análisis de datos no estructurados se enfoca en analizar datos que no se encuentran organizados en una estructura específica, como textos, imágenes o videos. Se utilizan técnicas de procesamiento de lenguaje natural y aprendizaje automático para extraer información relevante.

2. Técnicas que se pueden aplicar para la extracción de información y conocimiento.

Qué son las técnicas de recolección de datos

Son herramientas que te permiten obtener la información necesaria para analizar una situación o resolver una pregunta de investigación.

Estas técnicas son esenciales para garantizar que los datos que recolectas sean útiles, confiables y relevantes. Si no eliges el método adecuado, podrías terminar con información incompleta o inexacta, lo que afectaría el análisis final.

Tipos de recolección de datos

¿Cuáles son las técnicas de recolección de datos? Existen dos tipos de recolección de datos, y cada uno tiene su propio enfoque para obtener la información necesaria.

— Recolección de datos cuantitativos: se enfoca en obtener información numérica que puede medirse y analizarse de manera estadística.

— Recolección de datos cualitativos: busca captar experiencias, opiniones o comportamientos. Estos datos no se traducen en números, sino en descripciones detalladas que ayudan a entender mejor un fenómeno.

Instrumentos y métodos de recolección de datos

Existen diversos **instrumentos y métodos de recolección de datos** que permiten obtener información precisa. Entre ellos:

- **Encuestas:** son cuestionarios estructurados que se aplican a un grupo amplio de personas. Se pueden realizar de manera presencial, por teléfono o en línea. Permiten recolectar datos de forma rápida y eficiente.

Ejemplo:

Estudio de mercado para conocer las preferencias de los consumidores sobre un nuevo producto lanzado al mercado.

- **Entrevistas:** consisten en hacer preguntas directas a una persona o grupo, ya sea de manera estructurada (con preguntas fijas) o semiestructurada (más flexible). Es ideal para obtener información detallada.

Ejemplo:

Investigación sobre las experiencias de pacientes con enfermedades crónicas en el sistema de salud.

- **Observación:** este método implica observar el comportamiento de los sujetos en su entorno natural sin intervenir. Es útil para estudios de comportamiento o para entender cómo actúan las personas en situaciones reales.

Ejemplo:

Análisis del comportamiento de niños en el aula durante actividades grupales.

- **Grupos focales** (focus groups): se reúnen entre 6 y 12 personas para discutir un tema específico. El moderador guía la conversación para obtener percepciones, opiniones o ideas sobre un tema en particular.

Ejemplo:

Discusión sobre la percepción de una campaña publicitaria antes de su lanzamiento.

- **Revisión de registros:** consiste en analizar información existente, como registros médicos, informes o bases de datos, para extraer datos relevantes sin tener que realizar nuevas encuestas o entrevistas.

Ejemplo:

Estudio sobre la incidencia de enfermedades respiratorias utilizando registros médicos hospitalarios.

- **Pruebas o experimentos:** se trata de aplicar pruebas físicas, mentales o técnicas específicas para medir habilidades, conocimientos o respuestas ante ciertos estímulos. Son muy útiles para estudios en áreas científicas o académicas.

Ejemplo:

Evaluación del nivel de estrés en trabajadores mediante pruebas cognitivas bajo diferentes condiciones laborales.

Cada uno de estos **instrumentos de recolección de datos** tiene sus ventajas y desafíos. La elección dependerá de los objetivos de la investigación y del tipo de información que se busque recolectar.

Técnicas de recolección de datos en Big Data

Estos son algunos **ejemplos de técnicas de recolección de datos** que se utilizan en Big Data:

1. Extracción de datos de registros (Logs)

Una de las técnicas más comunes para recolectar datos es la captura de los registros de eventos o logs. Estos registros son generados automáticamente por sistemas, aplicaciones y servidores cada vez que ocurre un evento significativo, como una transacción, un inicio de sesión o un error. En el contexto de big data, los logs de aplicaciones web, servidores y dispositivos IoT proporcionan una rica fuente de datos que se puede analizar para monitorear rendimiento, detectar anomalías o predecir comportamientos.

Herramientas que se utilizan:
Apache Flume: Usado para recolectar, agregar y mover grandes cantidades de datos de logs a un sistema de almacenamiento centralizado.
Logstash: Una herramienta para recopilar, analizar y transformar datos de logs de múltiples fuentes.

Ejemplo:
Una empresa de e-commerce analiza los logs de su servidor web para detectar picos de tráfico, identificar intentos de acceso no autorizados y medir la velocidad de carga de las páginas.

2. Rastreo Web (Web Scraping)

El web scraping es otra técnica fundamental que se utiliza para extraer información de sitios web. Implica el uso de bots que acceden a páginas web, recuperan su contenido (HTML, XML, JSON) y lo procesan para extraer datos específicos. Esta técnica es particularmente útil para capturar datos no estructurados o semiestructurados de sitios web públicos, como datos de redes sociales, reseñas de productos o noticias.

Herramientas utilizadas:
BeautifulSoup y Scrapy: Librerías en Python especializadas en la extracción y estructuración de datos desde el contenido web.
Apache Nutch: Un framework extensible de rastreo web que puede usarse junto a Apache Hadoop para realizar el procesamiento de grandes volúmenes de datos de sitios web.

Ejemplo:
Un sitio comparador de precios utiliza web scraping para recolectar información diaria de productos y precios de varias tiendas en línea y mostrar comparativas actualizadas.

3. Sensores y dispositivos IoT

Con la expansión del Internet de las Cosas (IoT), la recolección de datos de dispositivos inteligentes y sensores ha tomado relevancia. Los sensores IoT generan datos en tiempo real que provienen de dispositivos conectados, como cámaras,

termostatos, wearables, entre otros. Esta información, que puede ser en formato de secuencias de datos o flujos de tiempo real, es clave para aplicaciones como el monitoreo de entornos, la automatización industrial y el mantenimiento predictivo.

Herramientas utilizadas:
Apache Kafka: Se usa para gestionar y procesar grandes flujos de datos en tiempo real provenientes de dispositivos IoT.
AWS IoT Core: Plataforma en la nube que facilita la conexión y gestión de millones de dispositivos IoT para recolectar datos.

Ejemplo:
Una planta industrial recoge datos de sensores de temperatura y vibración en sus máquinas para detectar señales de fallo inminente y aplicar mantenimiento predictivo.

4. Recolección de datos de redes sociales

Las plataformas de redes sociales son una fuente rica en datos no estructurados, como publicaciones, comentarios, likes, shares, imágenes y videos. Las empresas utilizan APIs ofrecidas por las redes sociales para recolectar datos a gran escala con el objetivo de analizar tendencias, medir la satisfacción del cliente o realizar estudios de mercado. El análisis de sentimiento y la minería de opiniones son casos comunes de uso en este contexto.

Herramientas utilizadas:
Twitter API, Facebook Graph API: Son interfaces que permiten a los desarrolladores acceder y recolectar datos desde las respectivas plataformas.
Hootsuite o Buffer: Herramientas que ayudan a la gestión y análisis de datos provenientes de múltiples redes sociales.

Ejemplo:
Una agencia de marketing digital utiliza la API de Twitter para analizar miles de tuits sobre un producto recién lanzado y realizar un análisis de sentimiento para evaluar la percepción del público.

5. Extracción de Datos de Bases de Datos Relacionales y NoSQL

En el mundo del big data, es crucial la capacidad de recolectar datos tanto de bases de datos tradicionales (relacionales) como de bases de datos NoSQL que están optimizadas para grandes volúmenes de datos no estructurados o semiestructurados.

La extracción de datos desde estas fuentes implica la ejecución de consultas eficientes para obtener grandes cantidades de información y luego transformarla o integrarla en sistemas de almacenamiento y análisis de big data.

Herramientas utilizadas:
Sqoop: Herramienta para la transferencia de grandes volúmenes de datos entre bases de datos relacionales y sistemas Hadoop.
Cassandra, MongoDB: Son bases de datos NoSQL que almacenan grandes cantidades de datos distribuidos, perfectas para los entornos de big data.

Ejemplo:
Una institución financiera extrae datos desde su base de datos relacional (MySQL) hacia un clúster Hadoop usando Sqoop, para hacer análisis históricos de transacciones y detectar fraudes.

6. APIs y Fuentes Externas

El acceso a datos a través de APIs es otra técnica extendida en big data, sobre todo cuando se requiere recolectar datos de servicios externos, como datos meteorológicos, económicos o demográficos. Estas interfaces proporcionan un acceso estructurado a datos en tiempo real o históricos que pueden complementar fuentes internas.

Herramientas utilizadas:
REST APIs y SOAP APIs: Son protocolos que permiten la interacción entre sistemas y la transferencia de datos en formatos como JSON o XML.
Postman: Una herramienta muy usada para interactuar y probar APIs antes de integrarlas a sistemas más grandes de recolección de datos.

Ejemplo:
Una app agrícola recopila datos meteorológicos en tiempo real desde una API pública para ofrecer recomendaciones personalizadas a los agricultores sobre riego y siembra.

Lenguajes y frameworks que se usan actualmente para proyectos de ciencia de datos.

R.

Lanzado en 1995 como descendiente directo del antiguo lenguaje de programación S, R se ha ido fortaleciendo. Escrito en C, Fortran y en sí mismo, el proyecto cuenta actualmente con el apoyo de la *R Foundation for Statistical Computing*.

Licencia: ¡Gratis!

Ventajas

- Excelente gama de paquetes de código abierto y de alta calidad. R tiene un paquete para casi todas las aplicaciones cuantitativas y estadísticas imaginables. Esto incluye redes neuronales, regresión no lineal, filogenia, cartografía, mapas y muchos, muchos otros.
- La instalación básica viene con funciones y métodos estadísticos integrales muy completos. R también maneja el álgebra de matriz particularmente bien.
- La visualización de datos es una fortaleza clave con el uso de bibliotecas como ggplot2.

Contras

- Rendimiento. R no es un lenguaje rápido. Esto no es un accidente. R fue diseñado a propósito para facilitarle el análisis de datos y las estadísticas. No fue diseñado para hacer la vida más fácil para tu computadora. Mientras que R es lento en comparación con otros lenguajes de programación, para la mayoría de los propósitos, es lo suficientemente rápido.
- Especificidad de dominio. R es fantástico para fines estadísticos y científicos de datos. Pero no es tan fantástico para programaciones de propósito general.
- Raro. R tiene algunas características poco frecuentes que pueden atrapar a los programadores con experiencia en otros idiomas. Por ejemplo: indexación desde 1, utilizando operadores de asignación múltiple, estructuras de datos no convencionales.

Python.

Guido van Rossum presentó Python en 1991. Desde entonces, se ha convertido en un lenguaje de uso general extremadamente popular, y se utiliza ampliamente en la comunidad de *data science*.

Licencia: ¡Gratis!

Ventajas

- Python es un lenguaje de programación de uso general muy popular y general. Cuenta con una amplia gama de módulos específicos y soporte comunitario. Los principales GIS de escritorio como ArcGIS (con la ArpPy), QGIS (con PyQGIS) o gvSIG la introducción de Python.
- Python es un lenguaje fácil de aprender. La baja barrera de entrada lo convierte en un primer idioma, lo que es ideal para aquellos que son nuevos en programación.
- Paquetes como pandas, scikit-learn y Tensorflow hacen de Python una opción sólida para aplicaciones avanzadas de aprendizaje automático.

Contras

- Seguridad de tipos: Python es un lenguaje de tipo dinámico, lo que significa que debemos ser muy cuidadosos. Los errores de tipo (como pasar una *string* como un argumento a un método que espera un número entero) deben esperarse de vez en cuando.
- Para los fines específicos de análisis estadístico y de datos, **la amplia gama de paquetes de R** le da una ligera ventaja sobre Python. Para los lenguajes de propósito general, hay alternativas más rápidas y seguras que Python.

SQL.

SQL («lenguaje de consulta estructurado») define, administra y consulta bases de datos relacionales. El lenguaje apareció en 1974 y desde entonces ha sufrido muchas implementaciones, pero los principios básicos siguen siendo los mismos.

Licencia: Varía, ya que algunas implementaciones son gratuitas y otras son propietarias.

Ventajas

- Muy eficiente en consultas, actualización y manipulación de bases de datos relacionales.
- La sintaxis declarativa hace de SQL un lenguaje muy legible. ¡No hay ambigüedad sobre lo que de debe hacer

SELECT name FROM users WHERE age > 18

- SQL utilizado en una amplia gama de aplicaciones, por lo que es un lenguaje muy útil para estar familiarizado. Los módulos como SQLAlchemy hacen que la integración de SQL con otros lenguajes sea sencillo.

Contras

- Las capacidades analíticas de SQL son bastante limitadas: más allá de agregar y sumar, contar y promediar datos, sus opciones son limitadas.
- Para los programadores que vienen de un contexto imperativo, la sintaxis declarativa de SQL puede presentar una curva de aprendizaje.

- Hay muchas implementaciones de SQL como PostgreSQL, SQLite, MariaDB. Todas son lo suficientemente diferentes como para hacer que la interoperabilidad sea un dolor de cabeza.

Java.

Java es un lenguaje extremadamente popular que se ejecuta en la Máquina Virtual Java (JVM). Es un sistema informático abstracto que permite una portabilidad perfecta entre plataformas. Actualmente respaldado por Oracle Corporation.

Licencia: ¡Gratis! Versiones heredadas, propietarias.

Ventajas

- Ubicuidad. Muchos sistemas y aplicaciones modernas se basan en un back-end de Java. La capacidad de integrar métodos de ciencia de datos directamente en la base de código existente es poderosa.
- Fuertemente tipado. Java es un buen lenguaje cuando se trata de garantizar la seguridad de tipos. Para aplicaciones de big data de misión crítica, esto es muy importante.
- Java es un lenguaje compilado de propósito general y **alto rendimiento**. Lo que lo hace adecuado para escribir eficientes códigos de producción **ETL** y algoritmos de *machine learning* muy intensivos computacionalmente.

Contras

- Para análisis ad-hoc y aplicaciones estadísticas más dedicadas, la verbosidad de Java hace que sea una primera opción poco probable. Los lenguajes de script de tipado dinámico como R y Python se prestan a una productividad mucho mayor.
- En comparación con los lenguajes específicos de dominio como R, no dispone de muchas librerías disponibles para métodos estadísticos avanzados.

Scala.

Desarrollado por **Martin Odersky** y lanzado en 2004, Scala es un lenguaje que se ejecuta en la Máquina Virtual Java (JVM). Es un lenguaje de múlti paradigmático, que permite tanto enfoques orientados a objetos como funcionales. El framework de computación de cluster Apache Spark está escrito en Scala.

Licencia: ¡Gratis!

Ventajas

- Scala + Spark = Computación en clúster de alto rendimiento. Scala es un lenguaje ideal para quienes trabajan con conjuntos de datos de gran volumen.
- Multi-paradigmático: los programadores de Scala pueden tener lo mejor de ambos mundos. Tanto la programación orientada a objetos como funcional.
- Scala se compila en el bytecode de Java y se ejecuta en una JVM. Esto permite la interoperabilidad con el lenguaje Java en sí, haciendo de Scala un lenguaje de propósito general muy poderoso, además de ser adecuado para la ciencia de datos.

Contras

- Scala no es un lenguaje sencillo para comenzar a utilizar si está empezando. Lo mejor es descargar sbt y configurar un IDE como Eclipse o IntelliJ con un complemento específico de Scala.
- La sintaxis y el sistema de tipos se describen con frecuencia como complejos. Esto hace que la curva de aprendizaje sea pronunciada para aquellos que vienen de lenguajes dinámicos como Python.

Julia.

Lanzada en 2011, Julia impresionó al mundo de la computación numérica. Su perfil se elevó gracias a la adopción temprana por parte de varias organizaciones importantes, incluidas muchas de la industria financiera.

Licencia: ¡Gratis!

Ventajas

- Julia es un lenguaje compilado JIT (‘*just-in-time*‘), que le permite ofrecer un buen rendimiento. También ofrece las capacidades de simplicidad, tipado dinámico y *scripting* de un lenguaje interpretado como Python.
- Julia fue diseñada específicamente para el análisis numérico. Pero también ofrece programación de propósitos generales.
- Legibilidad. Muchos usuarios del lenguaje mencionan ésto como una ventaja clave.

Contras

- Madurez. Como nuevo idioma, algunos usuarios de Julia han experimentado inestabilidad al usar paquetes complementarios. Pero el núcleo del lenguaje es, al parecer, lo suficientemente estable para usar en producción.
- Los paquetes limitados son otra consecuencia de la juventud del lenguaje y de la pequeña comunidad de desarrollo. A diferencia de R y Python, Julia no tiene la posibilidad de disponer de paquetes (todavía).

MATLAB.

MATLAB es un lenguaje de **computación numérica** que se utiliza en el mundo académico y en la industria. Desarrollado y licenciado por MathWorks, una compañía establecida en 1984 para comercializar el software.

Licencia: Propietario – los precios varían dependiendo del caso.

Ventajas

- Diseñado para la computación numérica. MATLAB es adecuado para aplicaciones cuantitativas con requisitos matemáticos sofisticados, como procesamiento de señales, transformaciones Fourier, álgebra matricial y procesamiento de imágenes.
- Visualización de datos. MATLAB tiene incorporadas grandes capacidades de ploteado.
- MATLAB se enseña con frecuencia como parte de cursos de pregrado en asignaturas cuantitativas como Física, Ingeniería y Matemáticas Aplicadas. Como consecuencia, es ámpliamente utilizado en estos campos.

Contras

- Licencia propietaria. Dependiendo del caso (uso académico, personal o empresarial) es posible que tengamos que desembolsar una gran cantidad de dinero. Existen alternativas gratuitas disponibles como Octave.
- MATLAB no es una opción obvia para programación de propósito general.

Referencias

De Redacción de la Universidad Internacional de la Rioja, E. (2024, 29 noviembre). Técnicas de recolección de datos en Big Data. *UNIR México*. https://acortar.link/TpbISc

MisApuntes. (2023, 6 octubre). Los 7 tipos de análisis de datos que debes conocer - MisApuntes. *MisApuntes*. https://acortar.link/sHb2TN

Morales, A. (2023, 22 mayo). *Lenguajes de programación para realizar ciencia de datos*. MappingGIS. https://acortar.link/Z12DCD