**Ciencia de datos PILARES**

**Conceptos básicos**

o ¿Qué es la ciencia de datos?

La ciencia de datos es un campo interdisciplinario que combina métodos, procesos, algoritmos y sistemas para extraer conocimiento y obtener información valiosa a partir de datos estructurados y no estructurados. Se apoya en diversas disciplinas como las matemáticas, la estadística, la informática y el aprendizaje automático, y tiene como objetivo encontrar patrones, hacer predicciones y tomar decisiones basadas en datos.

En términos generales, la ciencia de datos sigue un ciclo de vida que incluye la recolección de datos, su limpieza, análisis y visualización, así como la creación de modelos predictivos que ayuden a resolver problemas específicos. Esto la convierte en una herramienta clave en una variedad de industrias como la salud, finanzas, marketing y tecnología, entre muchas otras.

Se pueden cubrir las siguientes bases:

1. **Recolección de datos**: Métodos para obtener datos de diferentes fuentes (bases de datos, APIs, web scraping, etc.).
2. **Limpieza de datos**: Procesos para corregir, eliminar inconsistencias, manejar valores faltantes y preparar los datos para su análisis.
3. **Análisis exploratorio de datos (EDA)**: Herramientas para explorar los datos, identificar patrones, tendencias y relaciones entre variables.
4. **Modelado**: Técnicas estadísticas y algoritmos de machine learning para construir modelos predictivos o descriptivos.
5. **Visualización**: Uso de herramientas como Matplotlib, Seaborn o Tableau para comunicar los hallazgos a través de gráficos e informes.
6. **Toma de decisiones basada en datos**: Cómo usar los resultados obtenidos para generar insights y apoyar la toma de decisiones estratégicas.

o Aplicaciones de la ciencia de datos

La ciencia de datos tiene una amplia gama de aplicaciones en múltiples sectores, ayudando a mejorar procesos, tomar decisiones estratégicas y crear nuevos productos y servicios. A continuación, te presento algunas de las áreas más relevantes con ejemplos prácticos:

### **1. Salud**

* **Aplicación**: Predicción de enfermedades.
* **Ejemplo práctico**: Utilizar modelos predictivos basados en grandes cantidades de datos médicos (como historias clínicas electrónicas, análisis genéticos y estilos de vida) para predecir la aparición de enfermedades como el cáncer o la diabetes. Esto permite una intervención temprana y personalizada.
* **Herramientas**: Modelos de machine learning, algoritmos de clasificación (como regresión logística o árboles de decisión).

### **2. Finanzas**

* **Aplicación**: Detección de fraudes.
* **Ejemplo práctico**: Los bancos y las instituciones financieras utilizan la ciencia de datos para analizar transacciones en tiempo real, identificar patrones sospechosos y detectar actividades fraudulentas. Un modelo entrenado puede alertar sobre comportamientos anómalos, como múltiples transacciones en un corto periodo desde diferentes ubicaciones.
* **Herramientas**: Algoritmos de detección de anomalías, redes neuronales, clustering.

### **3. Marketing y Ventas**

* **Aplicación**: Personalización de campañas publicitarias.
* **Ejemplo práctico**: Las empresas pueden analizar datos de clientes (historial de compras, navegación en sitios web, interacción en redes sociales) para crear campañas de marketing dirigidas, personalizando ofertas según los intereses y comportamientos del usuario. Plataformas como Netflix o Amazon utilizan este tipo de análisis para recomendar productos o contenido.
* **Herramientas**: Sistemas de recomendación, análisis de sentimiento, algoritmos de clustering.

### **4. Logística y Transporte**

* **Aplicación**: Optimización de rutas y tiempos de entrega.
* **Ejemplo práctico**: Empresas de transporte y logística como UPS o DHL utilizan modelos de ciencia de datos para optimizar rutas de entrega, minimizando los costos y tiempos. Estos modelos tienen en cuenta factores como el tráfico, el clima, y las distancias entre puntos de entrega.
* **Herramientas**: Algoritmos de optimización, análisis geoespacial, modelos predictivos.

### **5. Manufactura**

* **Aplicación**: Mantenimiento predictivo.
* **Ejemplo práctico**: Las fábricas y empresas manufactureras recopilan datos de sensores en sus máquinas para predecir cuándo fallará una pieza o equipo. Esto permite hacer el mantenimiento antes de que ocurran fallos, evitando costosos tiempos de inactividad.
* **Herramientas**: Análisis de series temporales, IoT (Internet de las Cosas), machine learning.

### **6. Deportes**

* **Aplicación**: Análisis de rendimiento y estrategia.
* **Ejemplo práctico**: Equipos deportivos como los de la NBA o la Premier League utilizan análisis de datos para evaluar el rendimiento de jugadores, optimizar estrategias de juego y decidir alineaciones. Los datos se recopilan de sensores en los jugadores, análisis de video y estadísticas de juego.
* **Herramientas**: Machine learning, análisis de datos en tiempo real, visualización de datos.

### **7. Ciencia y Medio Ambiente**

* **Aplicación**: Predicción de cambios climáticos.
* **Ejemplo práctico**: Los científicos utilizan grandes volúmenes de datos climáticos históricos y modelos predictivos para prever fenómenos naturales como huracanes, olas de calor o el deshielo polar. Esto ayuda a mitigar riesgos y prepararse para desastres naturales.
* **Herramientas**: Modelos predictivos, simulaciones, análisis de series temporales.

### **8. E-commerce**

* **Aplicación**: Análisis del comportamiento del cliente.
* **Ejemplo práctico**: Empresas de comercio electrónico utilizan los datos de las interacciones de los usuarios en sus plataformas (clics, tiempo de permanencia, compras, productos visualizados) para identificar patrones de comportamiento y mejorar la experiencia del usuario. Esto puede llevar a la optimización del diseño del sitio y mejores conversiones.
* **Herramientas**: Análisis de cohortes, algoritmos de clustering, minería de datos.

### **9. Educación**

* **Aplicación**: Mejora del rendimiento estudiantil.
* **Ejemplo práctico**: Las instituciones educativas pueden analizar el rendimiento de los estudiantes, su asistencia, participación en actividades extracurriculares y otros factores para identificar los que tienen riesgo de abandono o que necesitan apoyo adicional. Esto les permite intervenir de manera temprana y personalizada.
* **Herramientas**: Regresión logística, análisis de cohortes, clustering.

### **10. Seguridad Informática**

* **Aplicación**: Respuesta a amenazas cibernéticas.
* **Ejemplo práctico**: En la ciberseguridad, se utilizan técnicas de ciencia de datos para monitorear redes y sistemas en busca de actividades inusuales que puedan indicar ataques cibernéticos, como infecciones por malware o intentos de acceso no autorizados.
* **Herramientas**: Detección de anomalías, machine learning supervisado y no supervisado, redes neuronales.

o Introducción al flujo de trabajo de un proyecto de ciencia de datos

El flujo de trabajo de un proyecto de ciencia de datos es un proceso estructurado que permite abordar problemas complejos de manera sistemática, desde la recolección de datos hasta la entrega de resultados y la toma de decisiones basadas en ellos. Este flujo de trabajo garantiza que se sigan pasos lógicos, mejorando la eficiencia y la calidad de los análisis. A continuación, se describen las etapas más comunes en el ciclo de vida de un proyecto de ciencia de datos:

### **Definición del problema**

Antes de cualquier análisis, es crucial comprender claramente el problema que se desea resolver. Esto incluye:

* **Identificación del objetivo**: ¿Qué se espera lograr? Por ejemplo, predecir ventas, mejorar un proceso, o detectar fraudes.
* **Entender el contexto**: Conocer el negocio o el entorno en el que se va a aplicar el análisis es esencial para tomar decisiones acertadas.
* **Establecer métricas de éxito**: Definir cómo se medirá el éxito del proyecto, ya sea a través de precisión, ahorro de costos, o mejoras en la eficiencia.

Este flujo de trabajo es flexible y puede adaptarse a diferentes tipos de proyectos de ciencia de datos. En un curso introductorio, cada etapa puede explicarse y explorarse con ejemplos prácticos para que los estudiantes entiendan cómo aplicar la metodología en casos del mundo real.

o Otras herramientas de trabajo en la ciencia de datos.

### **1. Jupyter Notebook / JupyterLab**

* **Descripción**: Una de las herramientas más usadas por científicos de datos. Es un entorno interactivo basado en web que permite escribir y ejecutar código en diferentes lenguajes de programación, como Python, Julia y R.
* **Ventajas**:
  + Ideal para análisis exploratorio de datos, visualización y desarrollo de modelos.
  + Integración sencilla con bibliotecas populares como NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow.
  + Permite combinar código, texto (con Markdown), gráficos y comentarios en un solo documento.
* **Uso común**: Proyectos locales en los que se necesita un entorno de pruebas y desarrollo.

### **2. Anaconda**

* **Descripción**: Una distribución de Python y R para ciencia de datos, que incluye más de 1,500 paquetes y herramientas de administración de entornos.
* **Ventajas**:
  + Ofrece una instalación sencilla de múltiples bibliotecas y entornos virtuales.
  + Viene con Jupyter Notebook, JupyterLab y otras herramientas preinstaladas.
  + Incluye **Conda**, una herramienta para la gestión de entornos y dependencias.
* **Uso común**: Desarrollar y gestionar proyectos de ciencia de datos en el entorno local con una gran cantidad de bibliotecas listas para usar.

### **3. Kaggle**

* **Descripción**: Kaggle no solo es una plataforma de competiciones de ciencia de datos, sino también un entorno en la nube que permite crear notebooks con Python y R, similar a Google Colab.
* **Ventajas**:
  + Acceso gratuito a GPUs para proyectos de machine learning.
  + Comunidad activa y datasets abiertos para experimentar.
  + Permite compartir y colaborar en notebooks con otros científicos de datos.
* **Uso común**: Competir en desafíos de ciencia de datos, compartir proyectos y trabajar con datasets disponibles públicamente.

### **4. Apache Spark**

* **Descripción**: Una herramienta para el procesamiento de grandes volúmenes de datos de manera distribuida. Ideal para análisis a gran escala.
* **Ventajas**:
  + Procesa grandes conjuntos de datos distribuidos en clústeres.
  + Compatible con múltiples lenguajes como Python (con PySpark), Scala, Java, y R.
  + Alta velocidad en el procesamiento de datos en memoria y capacidad de manejar tareas complejas como machine learning y consultas de SQL.
* **Uso común**: Procesamiento de big data en entornos empresariales o con grandes volúmenes de datos.

### **5. RStudio**

* **Descripción**: Un entorno de desarrollo integrado (IDE) para el lenguaje R, pero también compatible con Python.
* **Ventajas**:
  + Muy popular en la comunidad de estadística y ciencia de datos, especialmente para análisis y visualización de datos.
  + Gran cantidad de bibliotecas para análisis estadístico y modelado.
  + Herramientas avanzadas para gráficos interactivos y reporting.
* **Uso común**: Proyectos enfocados en estadísticas, análisis de datos y visualización con el lenguaje R.

### **6. VS Code (Visual Studio Code)**

* **Descripción**: Un editor de código ligero pero muy potente, ampliamente usado por desarrolladores y científicos de datos.
* **Ventajas**:
  + Soporta múltiples lenguajes de programación con extensiones, incluyendo Python, R, Julia y más.
  + Integración con notebooks Jupyter.
  + Control de versiones (Git) incorporado.
  + Extensiones para análisis de datos y machine learning.
* **Uso común**: Desarrollo de ciencia de datos en un entorno robusto y personalizable con características de depuración y control de versiones.

### **7. Apache Hadoop**

* **Descripción**: Un framework de código abierto para el almacenamiento y procesamiento distribuido de grandes volúmenes de datos usando clústeres de ordenadores.
* **Ventajas**:
  + Ideal para manejar big data, especialmente en empresas que requieren almacenamiento y análisis de petabytes de datos.
  + Compatible con Apache Spark para procesamiento rápido en memoria.
* **Uso común**: Proyectos empresariales a gran escala donde se maneja big data.

### **8. Power BI / Tableau**

* **Descripción**: Herramientas de visualización de datos que permiten crear dashboards interactivos y gráficos a partir de diferentes fuentes de datos.
* **Ventajas**:
  + Facilidad de uso y gran cantidad de plantillas gráficas para representar datos complejos.
  + Integración con diversas fuentes de datos, tanto locales como en la nube.
  + Posibilidad de compartir visualizaciones y dashboards interactivos con equipos y clientes.
* **Uso común**: Presentación y visualización de datos en informes ejecutivos o análisis de negocios.

### **9. DataRobot**

* **Descripción**: Una plataforma automatizada de machine learning (AutoML) que permite construir y desplegar modelos predictivos con muy poca intervención manual.
* **Ventajas**:
  + Permite automatizar el proceso de creación de modelos, desde la limpieza de datos hasta el ajuste de hiperparámetros.
  + Fácil de usar incluso para personas con menos experiencia en machine learning.
  + Buena para la implementación rápida de modelos en producción.
* **Uso común**: Proyectos que requieren modelos rápidos y efectivos sin mucho esfuerzo manual.

### **10. Dask**

* **Descripción**: Un framework de computación paralela en Python, diseñado para escalar el procesamiento de datos de forma local o distribuida.
* **Ventajas**:
  + Permite trabajar con datasets que no caben en la memoria usando APIs similares a Pandas.
  + Fácil de integrar con otras herramientas populares de Python como NumPy, Pandas y Scikit-learn.
* **Uso común**: Procesamiento de grandes volúmenes de datos cuando Pandas u otras herramientas locales no son suficientes.

### **11. Azure Machine Learning Studio / AWS SageMaker**

* **Descripción**: Plataformas en la nube de Microsoft Azure y Amazon Web Services (AWS) para el desarrollo de modelos de machine learning.
* **Ventajas**:
  + Integración con la infraestructura en la nube, permitiendo escalar modelos de manera eficiente.
  + Acceso a herramientas avanzadas de machine learning, incluyendo AutoML y despliegue sencillo de modelos.
  + Integración con otros servicios en la nube para análisis de datos, almacenamiento y visualización.
* **Uso común**: Desarrollar y desplegar modelos a gran escala en entornos empresariales en la nube.

Estas herramientas cubren diferentes etapas del ciclo de vida de un proyecto de ciencia de datos, desde el análisis exploratorio hasta la implementación de modelos. Según el tipo de proyecto (pequeña escala, big data, visualización), puedes elegir la que mejor se adapte a tus necesidades.

Introducción a Google Colab.

**Google Colab** (abreviatura de "Colaboratory") es una plataforma de computación en la nube que permite escribir y ejecutar código Python directamente desde el navegador. Es especialmente popular en la comunidad de ciencia de datos y aprendizaje automático por su facilidad de uso y acceso gratuito a recursos computacionales potentes como GPUs y TPUs. Colab es una herramienta que facilita el desarrollo de proyectos, colaboraciones, experimentos y la enseñanza de ciencia de datos y machine learning.

### **Características principales de Google Colab**

1. **Entorno basado en Jupyter Notebooks**:
   * Google Colab se basa en **Jupyter Notebooks**, lo que permite combinar código Python con celdas de texto enriquecido en un mismo documento. Esto facilita la documentación y el análisis exploratorio.
   * Las celdas de código pueden contener scripts Python y generar visualizaciones directamente, lo que es útil para mostrar resultados inmediatos.
2. **Ejecutar código en la nube**:
   * A diferencia de Jupyter Notebook, que generalmente se ejecuta en tu máquina local, Colab utiliza la infraestructura de Google para ejecutar el código en la nube. No necesitas configurar entornos locales o preocuparte por la instalación de bibliotecas.
   * Ofrece **acceso gratuito a GPUs y TPUs** (unidades de procesamiento gráfico y tensorial), lo que es ideal para proyectos de machine learning y deep learning que requieren gran capacidad de cómputo.
3. **Colaboración en tiempo real**:
   * Como otros productos de Google (como Google Docs), Colab permite la **colaboración en tiempo real**. Varias personas pueden editar y ejecutar el mismo notebook al mismo tiempo, lo que facilita el trabajo en equipo.
   * Puedes compartir tu notebook con colegas o compañeros de estudio con permisos para solo visualizar o editar.
4. **Integración con Google Drive**:
   * Los notebooks se almacenan directamente en Google Drive, lo que facilita el acceso desde cualquier dispositivo y la sincronización automática.
   * Además, puedes montar tu Google Drive dentro de Colab para acceder a archivos y datasets directamente desde el notebook.
5. **Instalación de bibliotecas**:
   * Colab viene con muchas bibliotecas preinstaladas (como NumPy, Pandas, Matplotlib, TensorFlow, Scikit-learn), pero también puedes instalar otras bibliotecas con comandos como !pip install.
   * Es flexible y permite trabajar con una amplia gama de herramientas de Python para ciencia de datos y machine learning.
6. **Facilidad para importar datasets**:
   * Colab facilita la carga de datos desde diversas fuentes. Puedes cargar datasets directamente desde tu computadora, Google Drive, APIs o incluso desde bases de datos en la nube.
   * También puedes acceder a datasets públicos de Kaggle o Google Cloud con facilidad.

4. Entorno de trabajo en Google Colab.

El entorno de trabajo en **Google Colab** es intuitivo y está diseñado para facilitar el flujo de trabajo de científicos de datos y desarrolladores. Es similar a Jupyter Notebooks, pero con la ventaja de estar basado en la nube, lo que significa que no necesitas configuraciones locales y puedes acceder a recursos avanzados como GPUs y TPUs. A continuación, se describen los componentes clave del entorno de trabajo en Google Colab:

### **1. Interfaz del Notebook**

El entorno de trabajo en Google Colab se organiza principalmente en forma de notebooks, que son documentos interactivos donde puedes escribir código, incluir texto, imágenes, gráficos, y más.

* **Celdas de código**: Aquí es donde escribes tu código Python (u otros lenguajes soportados). Cada celda puede ejecutarse de manera independiente, permitiendo realizar pruebas rápidas y ver los resultados de manera inmediata.
  + Para ejecutar una celda de código, puedes hacer clic en el ícono de "play" al lado de la celda o presionar Shift + Enter.
  + Puedes combinar comandos en Python, instalar paquetes con !pip install, acceder a archivos con comandos !ls y más, todo en la misma celda.
* **Celdas de texto**: Estas celdas usan el lenguaje de marcado **Markdown** y permiten agregar comentarios, explicaciones, títulos, enlaces y fórmulas matemáticas con sintaxis LaTeX.
  + Son útiles para documentar el proceso, explicar el código o agregar información adicional al notebook.
  + Puedes formatear el texto fácilmente usando Markdown para crear listas, encabezados, tablas, etc.