**Proyecto de Datos-Enfoque en CRISP-DM General (Basico 3 Sprints)**

Tiempo: 3 sprints de una semana cada uno.

Fases de CRISP-DM:

* Comprensión del Negocio
* Comprensión de los Datos
* Preparación de los Datos
* Modelado
* Evaluación
* Despliegue

**Semana 0**

**Fase 1: Comprensión del Negocio**

**Definir el problema:**

* Identificar la pregunta de negocio a responder.
* Establecer los objetivos del proyecto.
* Definir los criterios de éxito.
* Crear un documento de requisitos detallado.
* Definir las métricas clave de éxito (KPIs).
* Elaborar un diagrama de flujo del proceso actual

**Semana 1 (Sprint 1)**

**Fase 2: Comprensión de los Datos**

* Identificar las fuentes de datos disponibles y su formato. (bases de datos, archivos, APIs).
* Evaluar la calidad y relevancia de cada fuente.
* Evaluar la calidad de los datos (completitud, consistencia, relevancia).
* Crear un inventario de datos.
* Realizar un análisis exploratorio de los datos (EDA) utilizando Python (Pandas, NumPy), librerías automáticas, otros…
* Visualizar los datos para identificar patrones y relaciones (Matplotlib, Seaborn).

**Semana 2 (Sprint 2)**

**Fase 3: Preparación y Modelado Básico**

**Limpiar los datos:**

* manejar valores faltantes, outliers, imputación, eliminación,inconsistencias.
* Identificar y corregir outliers.
* Transformar los datos: normalización, estandarización, codificación de variables categóricas.
* Crear nuevas características si es necesario (feature engineering).
* Dividir los datos en conjuntos de entrenamiento y prueba.

**Fase 4: Modelado**

**Selección de modelos:**

* Elegir modelos iniciales basados en el tipo de problema (clasificación, regresión, etc.).
* Entrenar modelos iniciales con hiperparámetros por defecto.
* Evaluar el rendimiento de los modelos en el conjunto de prueba.
* Considerar la complejidad y el tiempo de entrenamiento.

**Semana 3 (Sprint 3)**

**Fase 5: Evaluación**

* Calcular métricas de evaluación (accuracy, precision, recall, F1-score).
* Afinar los hiperparámetros de los modelos.
* Realizar validación cruzada para obtener una evaluación más robusta.
* Comparar el rendimiento de diferentes modelos.
* Seleccionar el mejor modelo basado en las métricas y los objetivos del negocio.
* Interpretar los resultados del modelo.

**Fase 6: Despliegue**

* **Planificación del despliegue:**
* Definir la infraestructura necesaria (local, nube).
* Crear un prototipo de la aplicación.
* Implementar una interfaz de usuario sencilla.
* Planificar el monitoreo del modelo en producción.
* Publicacion en web de Tableros interactivos

**Consideraciones:**

* **Iteración:** Las fases no son lineales. Es posible volver a fases anteriores para realizar ajustes.
* **Colaboración:** Utilizar herramientas de colaboración para facilitar el trabajo en equipo.
* **Versionamiento:** Utilizar Git para gestionar el código.
* **Documentación:** Documentar cada etapa para facilitar la comprensión y el mantenimiento.
* **MVP:** Centrarse en las funcionalidades más importantes para el MVP.

**Herramientas y Tecnologías Sugeridas:**

* **Lenguajes de programación:** Python
* **Librerías:** Pandas, NumPy, Scikit-learn, TensorFlow, PyTorch
* **Herramientas de visualización:** Matplotlib, Seaborn, Tableau, Power BI
* **Herramientas de ETL:** Webscraping, sql
* **Herramientas de colaboración:** GitHub, Slack, Trello, Jira
* **Nube:** AWS, GCP, Azure
* **Librerias/Entornos de Despliegue:** streamlite
* **Frameworks de desarrollo web:** Flask, Django

!pip install opencv-python

import cv2

import os

def Write\_Image\_Sizes(filenames, storage\_file):

    """

    Takes the File names, writes the width and height of images in csv along with file names

    """

    store\_file = open(storage\_file, 'w+')

    store\_file.write("ImageName,Height,Width")

    store\_file.write("\n")

    cnt = 0

    for file in filenames:

        # Check if the file exists before attempting to read it

        if not os.path.exists(file):

            print(f"Warning: File not found: {file}")

            continue  # Skip to the next file

        cv\_img = cv2.imread(file)

        # Check if the image was loaded successfully

        if cv\_img is None:

            print(f"Warning: Could not read image: {file}")

            continue  # Skip to the next file

        # img.shape gives (img\_height, img\_width, img\_channel)

        store\_file.write(str(file) + "," + str(cv\_img.shape[0]) + "," + str(cv\_img.shape[1]))

        store\_file.write("\n")

        cnt += 1

        if cnt % 10000 == 0:

            print("Processed Images: ", cnt)

    store\_file.close()

train\_data=pd.read\_csv('/content/drive/MyDrive/OCR/SynthImageDataset/Train\_data1.csv')

train\_image\_names=list(train\_data['ImageName'].values)

train\_data['ImageName'].values

Write\_Image\_Sizes(train\_image\_names,'/content/drive/MyDrive/OCR/Train\_image\_sizes1.csv')