# 1. Objetivo General

En este segundo proyecto, deberás:

1. **Realizar Feature Engineering**:
    - Crear, transformar y seleccionar variables (features) relevantes que puedan mejorar el desempeño del modelo.
    - Justificar la estrategia utilizada para crear nuevas variables y/o transformar las existentes.
2. **Investigar y Probar Múltiples Modelos (Modeling)**:
    - Elegir un conjunto de modelos **adecuados** para tu tipo de problema (regresión o clasificación).
    - **Entrenar y comparar** estos modelos a través de una evaluación consistente (validación cruzada o train/test splits bien configurados).
    - **Seleccionar** el modelo o conjunto de modelos más prometedores, explicando por qué.

Este entregable se enfocará, dentro del ciclo CRISP-DM, en la fase de **Modeling** y en la continuación de la **Data Preparation** (feature engineering), previo a la evaluación y despliegue en siguientes etapas.

---

# 2. Fases CRISP-DM relacionadas

1. **Data Preparation (Feature Engineering)**
    - Extensión de la fase previa de Data Wrangling.
    - Creación y/o selección de variables que añadan valor predictivo.
2. **Modeling**
    - Selección y configuración de modelos (hiperparámetros).

    - Entrenamiento y comparación inicial de los modelos.

3. **Evaluation (inicial)**

    - Métricas preliminares para comparar modelos y justificar la elección de uno u otro.

    - (La evaluación final y el refinamiento continuo se realizarán en fases posteriores, pero aquí debes mostrar tu **estrategia inicial de evaluación**).

---

# 3. Feature Engineering

## 3.1 Creación de Nuevas Variables

- **Variables derivadas**: Genera nuevas características a partir de las existentes (por ejemplo, variables agregadas, ratios, transformaciones logarítmicas, etc.).

- **Codificación** de variables categóricas: Label encoding, one-hot encoding u otras técnicas (según convenga).

- **Variables temporales** (si aplica): Extraer día, mes, estación, festivos, etc., cuando se trabaje con fechas.

## 3.2 Selección de Variables

- Realiza un **análisis** para descartar variables que no aportan valor (colinealidad extrema, poca variabilidad, etc.).

- Considera métodos de **reducción de dimensionalidad** (PCA, por ejemplo) si tu dataset es muy grande, y justifica tu elección.

## 3.3 Justificación

- Documenta por qué creaste ciertas variables y por qué descartaste otras.

- Explica **cómo** cada nueva variable podría impactar el rendimiento del modelo.

---

# 4. Modelado (Modeling)

Dependiendo de si tu proyecto es de **regresión** o de **clasificación**, deberás **probar varios modelos**. Abajo se listan algunos modelos sugeridos:

## 4.1 Regresión (La lista siguiente no es exhaustiva)

- **Regresión Lineal** (ej., `LinearRegression,` de scikit-learn).
- **Árboles de Decisión** (ej., `DecisionTreeRegressor`).
- **Random Forest** (ej., `RandomForestRegressor`).
- **Gradient Boosting** o **XGBoost** (ej., `GradientBoostingRegressor`, `XGBRegressor`).
- **Todos los algoritmos que vamos a ver en clase**

## 4.2 Clasificación (La lista siguiente no es exhaustiva)

- **Regresión Logística** (ej., `LogisticRegression`).
- **Árboles de Decisión** (ej., `DecisionTreeClassifier`).
- **Random Forest** (ej., `RandomForestClassifier`).
- **Ensambladores** de Boosting (ej., `GradientBoostingClassifier`, `XGBClassifier`).
- **Todos los algoritmos que vamos a ver en clase**

**Nota:**

Si utilizas otros modelos (p.ej., SVM, redes neuronales, etc.), documenta adecuadamente el **por qué** y **cómo**.

---

# 5. Estrategia de Evaluación

## 5.1 Particionamiento de Datos

- Utiliza un **train/test split** o, preferiblemente, **validación cruzada** para obtener una estimación más robusta.

- Explica cuántos “folds” utilizas y por qué.

### 5.2 Métricas de Desempeño

- **Regresión**: RMSE, MAE, R2 u otras métricas que consideres relevantes.

- **Clasificación**: Accuracy, Precisión, Recall, F1-score, AUC-ROC, etc.

- Elige métricas adecuadas a tu problema y justifica tu elección.

### 5.3 Comparación de Resultados

- Para cada modelo probado, reporta las métricas obtenidas.

- Incluye tablas y/o gráficas que faciliten la comparación (por ejemplo, un dataframe con los resultados promedio de cross-validation y su desviación estándar).

### 5.4 Ajuste de Hiperparámetros (Opcional en esta fase)

- Si cuentas con tiempo, puedes realizar una **búsqueda de hiperparámetros** (Grid Search o Random Search) para los modelos más prometedores.

- Justifica los hiperparámetros elegidos.

---

## 6. Organización del Trabajo (Repositorio de GitHub)

Conservando la estructura propuesta en la **Parte 1**, puedes agregar o modificar notebooks y documentación como sigue:

```
├── data/
│   ├── raw/
│   ├── interim/
│   └── processed/
│
```

```
├── notebooks/
│   ├── 01_EDA.ipynb                <- (Proyecto 1)
│   ├── 02_DataWrangling.ipynb       <- (Proyecto 1)
│   ├── 03_FeatureEngineering.ipynb  <- Nuevo notebook para crear y
describir tus nuevas features.
│   └── 04_Modeling.ipynb            <- Entrenamiento y comparación de
modelos (regresión / clasificación).
│
├── src/
│   ├── data_prep.py
│   ├── feature_engineering.py       <- Nuevo script para tus
funciones de feature engineering.
│   └── modeling.py                  <- Funciones de entrenamiento y
evaluación de los modelos.
│
├── reports/
│   ├── figures/
│   ├── primer_proyecto.md
│   └── segundo_proyecto.md          <- Documento principal del
segundo proyecto.
│
├── .gitignore
├── README.md
└── requirements.txt
```

## Notebooks Recomendados

1. **03_FeatureEngineering.ipynb**:
    - Explica y muestra el proceso de creación y selección de variables.
    - Incluye la justificación de cada transformación.
2. **04_Modeling.ipynb**:
    - Configura la experimentación con varios modelos.
    - Muestra tablas, gráficas y métricas que comparen el desempeño de cada modelo.

- Finaliza con un análisis de resultados y la selección **tentativa** del mejor modelo.

### Documento del Segundo Proyecto: `segundo_proyecto.md`

En la carpeta `reports/`, agrega un informe (o README específico) que incluya:

1. **Resumen de Feature Engineering**:
   - Principales transformaciones y por qué se implementaron.
2. **Resumen de Modelos Evaluados**:
   - Cuáles se probaron y por qué.
3. **Métricas Obtenidas**:
   - Comparaciones y conclusiones sobre qué modelos funcionan mejor.
4. **Lecciones Aprendidas**:
   - Dificultades y hallazgos clave.

---

## 7. Formato y Buenas Prácticas

1. **Documenta** tu código en los notebooks y en los scripts (`.py` en la carpeta `src/`).
2. **Usa comentarios** para aclarar la lógica y los pasos de procesamiento.
3. **Versiona** con frecuencia en GitHub: sube cambios parciales de manera ordenada.
4. **Incluye conclusiones** y recomendaciones en cada notebook. No dejes el análisis "oculto" sin explicar.

---

## 8. Entregables y Evaluación

### Entregable: Repositorio Actualizado

- El repositorio debe contener los nuevos notebooks, scripts y el documento `segundo_proyecto.pdf` en la carpeta `reports/`.
- Los **notebooks** (03 y 04) deben tener:
    - Código, visualizaciones y justificaciones detalladas.
    - Sección de conclusiones finales.

## Criterios de Evaluación

1. **Calidad y profundidad** en el Feature Engineering (creatividad y justificación).
2. **Rigor en la comparación de modelos**:
    - Evidencia de experimentación con múltiples enfoques.
    - Métricas y metodología de evaluación adecuadas.
3. **Claridad en la presentación** de resultados y conclusiones.
4. **Organización y limpieza** del repositorio (uso correcto de notebooks, scripts, `reports/`, etc.).
5. **Documentación** apropiada (comentarios, README, notas en los notebooks).