Taller 3a – Preparación de Datos y Modelos Predictivos

Estudiante: Juan Felipe Gómez Carmona
Maestría en Analítica Aplicada
Universidad de La Sabana
Profesor: Hugo Franco, Ph.D.

Octubre 2025

Índice

1. Introducción

El presente taller tiene como objetivo aplicar los principios del flujo de trabajo de Ciencia de Datos en la preparación, imputación y modelado de conjuntos de datos con valores faltantes. A partir del *Cleveland Heart Disease Dataset*, se busca evaluar el impacto de distintas estrategias de imputación en el rendimiento de modelos supervisados, así como comparar dos algoritmos ampliamente usados en clasificación: **Random Forest** y **XGBoost**.

2. Metodología

2.1. Dataset y Preprocesamiento

Se emplearon dos datasets:

- **Penguins Dataset**: utilizado para practicar la creación de *pipelines* con imputación, escalamiento y detección de valores atípicos.
- **Cleveland Heart Disease Dataset**: empleado para evaluar el impacto de diferentes estrategias de imputación y comparar modelos de clasificación.

En el conjunto de Cleveland, compuesto por 303 registros y 14 atributos clínicos, las columnas `ca` y `thal` presentaban valores faltantes, los cuales fueron convertidos a `NaN` para su imputación posterior.

2.2. Estrategias de Imputación

Se evaluaron distintas estrategias mediante `SimpleImputer` y `KNNImputer`, integradas en un pipeline junto con estandarización (`StandardScaler`) y el clasificador correspondiente. Las estrategias incluyeron:

- **mean**: reemplazo por la media de la columna.
- **median**: reemplazo por la mediana.
- **most_frequent**: reemplazo por el valor más frecuente.
- **constant**: reemplazo por un valor fijo (0).
- **KNN**: imputación basada en vecinos más cercanos ($k = 5$).

2.3. Comparación de Modelos

Luego de determinar la mejor estrategia de imputación, se compararon los modelos **Random Forest** y **XGBoost**. El flujo de procesamiento incluyó:

1. Imputación con el método *constant* (reemplazo por 0).
2. Estandarización con `StandardScaler`.
3. Entrenamiento de los clasificadores sobre el conjunto de entrenamiento.
4. Evaluación mediante métricas de *accuracy*, *recall* y *precision*.

Cómo ejecutar y archivos entregados

Los scripts entregados están disponibles en GitHub (`taller3-ml-analisis`). Para replicar los resultados:

1. Crear un entorno virtual e instalar dependencias:

```
python -m venv .venv
source .venv/bin/activate   # (Windows: .venv\Scripts\activate)
pip install -r requirements.txt
```

2. Ejecutar:

```
python data_prep.py
```

3. Las figuras generadas se almacenan en la carpeta `figures/`.

Algoritmo del pipeline propuesto

Algorithm 1: Pipeline de preparación e imputación

Input: Datos clínicos X, etiquetas y
Output: Modelo entrenado $\mathcal{M}$ y métricas
Dividir X, y en entrenamiento y prueba (stratificado);
Imputar valores faltantes (`constant`/`median`/`KNN`);
Estandarizar con `StandardScaler`;
Entrenar clasificador (`RandomForest` o `XGBoost`);
Evaluar con `accuracy`, `recall`, `precision`;
return $\mathcal{M}$ y métricas;

3. Resultados

3.1. Validación con el dataset Penguins

Como se observa en la Figura 1, el modelo logra clasificar correctamente las especies del dataset original. Tras aplicar la técnica de *capping* de valores atípicos (Figura 2), el desempeño se mantiene estable con una exactitud superior al 95 %.

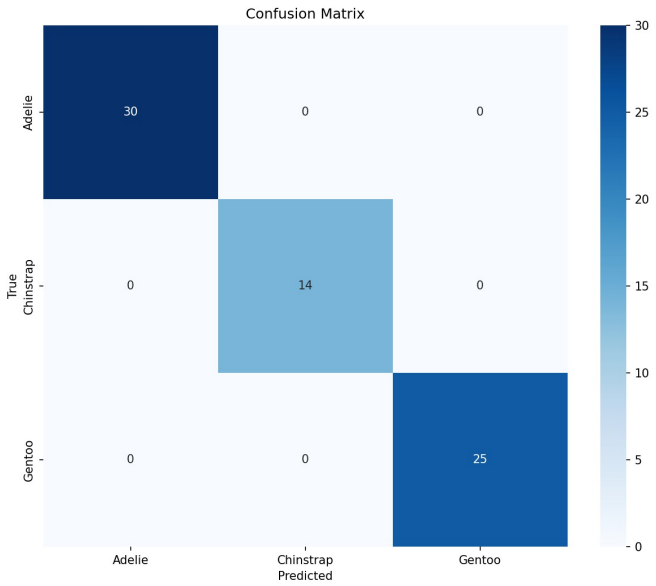

Figura 1: Matriz de confusión inicial del modelo en el dataset Penguins.

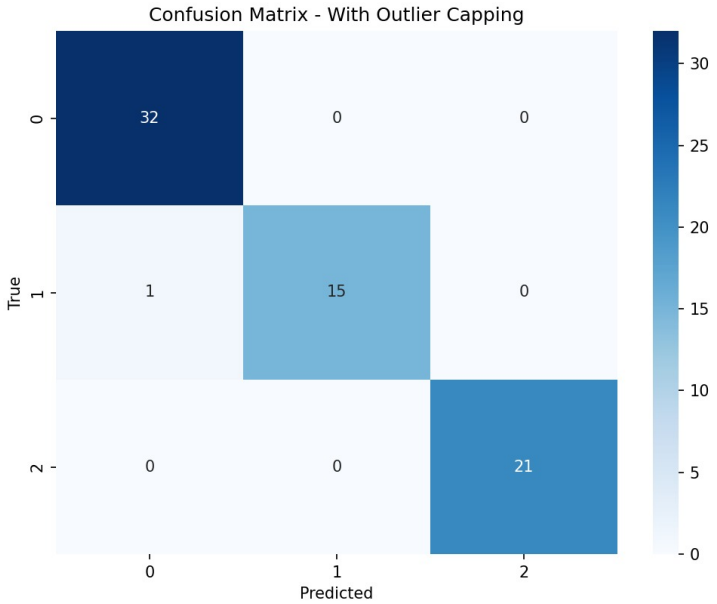

Figura 2: Matriz de confusión posterior al capping de valores atípicos.

3.2. Impacto de la Imputación de Datos (Cleveland)

Los resultados de las estrategias de imputación se resumen en el Cuadro 1. La imputación **constant** alcanzó la mayor precisión (90.16 %), mientras que *median* y *most frequent* también ofrecieron resultados competitivos.

Estrategia	**Accuracy**
Mean	0.8688
Median	0.8852
Most Frequent	0.8852
Constant	**0.9016**
KNN	0.8688

Cuadro 1: Comparación del rendimiento según la estrategia de imputación.

3.3. Comparación de Modelos

En la Tabla 2 se comparan los resultados de ambos clasificadores. **Random Forest** superó a **XGBoost** en todas las métricas evaluadas.

Modelo	Accuracy	Recall (Clase 1)	Precision (Clase 1)
Random Forest	**0.90**	0.96	0.84
XGBoost	0.85	0.93	0.79

Cuadro 2: Comparación de desempeño entre Random Forest y XGBoost.

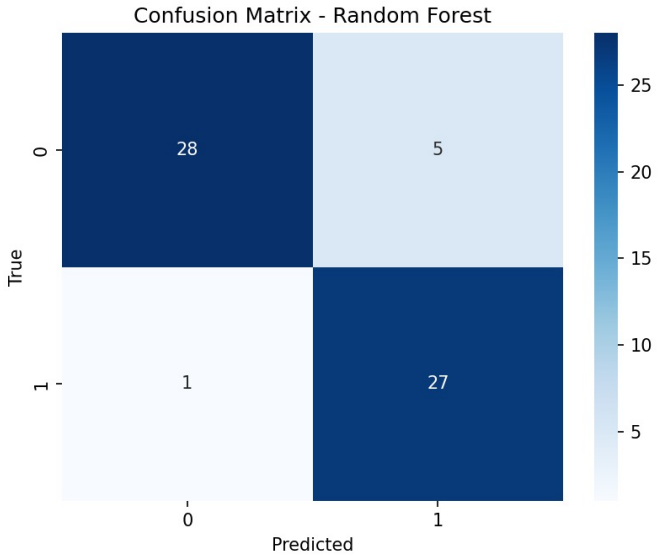

Figura 3: Matriz de confusión del modelo Random Forest.

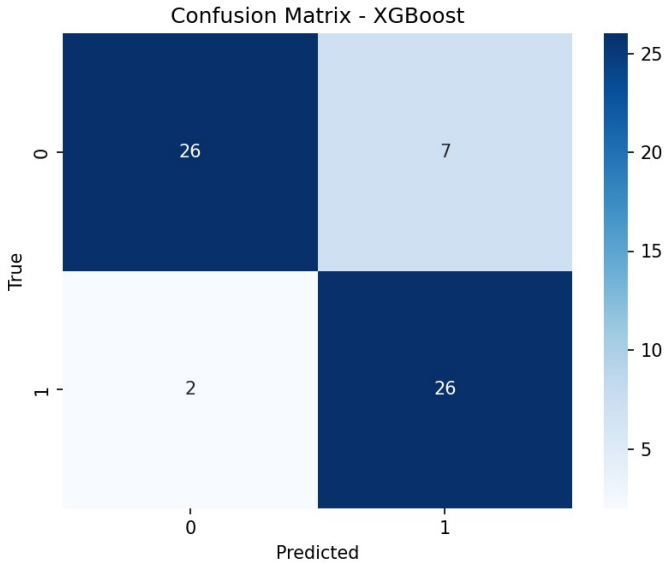

Figura 4: Matriz de confusión del modelo XGBoost.

4. Discusión

Los resultados demuestran que el proceso de preprocesamiento influye significativamente en el rendimiento final del modelo. El método de imputación **constant** ofreció un equilibrio adecuado entre simplicidad y desempeño. En cuanto a los algoritmos, **Random Forest** superó a **XGBoost** en todas las métricas, mostrando mayor estabilidad y precisión en la detección de pacientes con enfermedad. Este hallazgo es relevante en contextos clínicos donde el *recall* es crítico para reducir falsos negativos.

5. Conclusiones

- La imputación de valores faltantes mediante la estrategia **constant** logró la mejor precisión global (90 %).
- **Random Forest** se destacó por su mayor capacidad de generalización y balance entre *recall* y *precision*.
- Las diferencias en desempeño entre modelos refuerzan la importancia

del preprocesamiento de datos dentro del flujo de trabajo de Machine Learning.

- Un pipeline bien diseñado, con imputación adecuada y estandarización, permite maximizar la efectividad de los modelos predictivos.

6. Referencias

- UCI Machine Learning Repository. (2025). *Cleveland Heart Disease Dataset*. Disponible en: `https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/`
- Scikit-learn Documentation. (2025). *Imputation and Pipelines*. Disponible en: `https://scikit-learn.org/stable/modules/impute.html`
- Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. KDD.
- Waskom, M. (2021). *Seaborn: statistical data visualization*. Journal of Open Source Software, 6(60), 3021.