# Análisis e implementación de técnicas de aprendizaje automático

José Eduardo Viveros Escamilla - A01710605@tec.mx

***Resumen*— Este reporte presenta un análisis comparativo de tres modelos de regresión aplicados a un dataset de cáncer para predecir el área del tumor (`área mean`). Los modelos incluyen una regresión lineal implementada manualmente con gradiente descendente, un Random Forest básico, y un Random Forest optimizado mediante GridSearchCV. Se aplicaron pasos de preprocesamiento clave, como selección de features, análisis de correlación y escalado de características. El desempeño se evaluó en conjuntos de entrenamiento, validación y prueba usando las métricas MSE y $R^2$. Los resultados muestran que el Random Forest optimizado logra el mejor equilibrio entre bias y varianza, proporcionando una mayor precisión predictiva y manteniendo interpretabilidad mediante el análisis de importancia de features. Se discuten también consideraciones éticas relacionadas con la privacidad de datos médicos..**

***Abstract*— This report presents a comparative analysis of three regression models applied to a cancer dataset to predict tumor area (`area_mean`). The models include a manually implemented linear regression with gradient descent, a basic Random Forest Regressor, and an optimized Random Forest using GridSearchCV. Key preprocessing steps such as feature selection, correlation analysis, and feature scaling were applied. Performance was evaluated on train, validation, and test sets using MSE and $R^2$ metrics. Results show that the optimized Random Forest achieves the best balance between bias and variance, providing superior predictive accuracy while maintaining interpretability through feature importance analysis. Ethical considerations regarding medical data privacy are also discussed.**

## I. INTRODUCCIÓN

El presente reporte tiene como objetivo analizar y comparar tres modelos de regresión aplicados a un dataset de cáncer de mama, con el propósito de predecir el área del tumor (`área mean`) a partir de características medidas en las células tumorales. El dataset contiene información sobre características como **textura, suavidad, compacidad, concavidad, simetría y dimensión fractal**, entre otras, que permiten describir tanto la forma como la complejidad de los tumores.

El **objetivo principal** es evaluar la capacidad predictiva de los modelos implementados manualmente frente a aquellos que utilizan frameworks, así como analizar el impacto de la **optimización de hiper parámetros** sobre el desempeño. La motivación radica en comprender cómo técnicas de aprendizaje automático pueden mejorar la predicción de valores clínicamente relevantes, y cómo diferentes enfoques afectan el **bias, la varianza y la interpretabilidad** del modelo.

En este estudio se implementan tres modelos:

1. **Regresión lineal múltiple manual**, entrenada con gradiente descendente y validación temprana (early stopping).

2. **Random Forest básico**, utilizando sklearn con hiper parámetros fijos.

3. **Random Forest mejorado**, con optimización de hiper parámetros mediante GridSearchCV y análisis de importancia de variables.

El análisis busca, además, cumplir con estándares éticos relacionados con la **privacidad de datos médicos**, garantizando un uso responsable de la información y considerando normativas como HIPAA y GDPR.

## II. METODOLOGÍA

### II.I. DATASET

El dataset utilizado en este estudio corresponde a datos de cáncer de mama, contenidos en el archivo `Cancer Data.csv`. Cada muestra representa un tumor y está caracterizada por **30 variables numéricas** que describen aspectos de las células tumorales, como:

- **Medidas de radio, perímetro y área**: describen el tamaño de las células y del tumor.
- **Textura y suavidad (smoothness)**: capturan la variabilidad y la irregularidad de los bordes celulares.
- **Compacidad, concavidad y puntos cóncavos**: indican la forma y la complejidad de la superficie.
- **Simetría y dimensión fractal**: describen la geometría y patrones complejos de las células.

La **variable objetivo** seleccionada es `área mean`, que representa el área promedio del tumor. Para esto primero realice un gráfica de barras donde comparaba las 30 features de el dataset entero contra nuestra "y" en este casa area_mean.

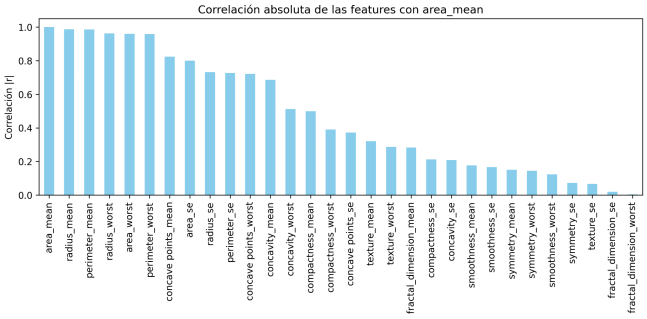


*Fig. 1. Tabla de correlación 30 features vs area_mean*

En base a esto se escogieron **13 variables independientes** relevantes, seleccionadas mediante análisis de correlación con la variable objetivo y evitando multicolinealidad para poder asegurar independencia entre features y mejorar la estabilidad de los modelos.

Por esta razón, traté de **evitar features con colinealidad muy alta**, ya que incluir variables casi idénticas podría generar problemas en el modelo y no aportar información adicional. Por ejemplo, las features `perimeter_mean` y `radius_mean` están muy relacionadas con `area_mean`; en el contexto de las medidas del tumor, sería como intentar predecir el área de un círculo usando su perímetro y radio: con solo esas dos variables ya podríamos calcular el área, por lo que agregar más features similares no aporta valor.

Por esta razón, decidimos **eliminar** `perimeter mean`, `radius_se` y `perimeter_se`, pero mantener `radius_mean` como referencia adicional. Esto permite que el modelo tenga suficiente información para predecir `area_mean`, evitando redundancia y mejorando la estabilidad de la regresión.

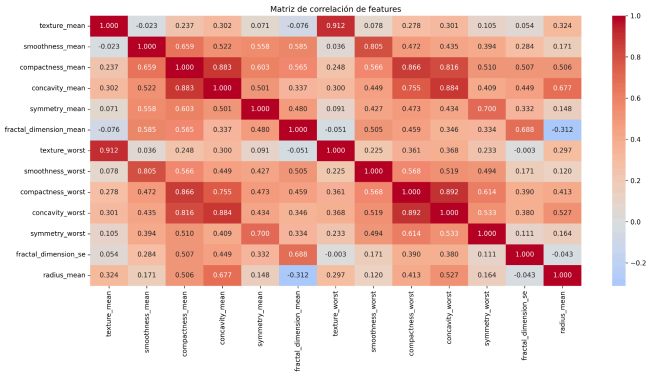


*Fig. 2. Matriz de correlación (13 features relevantes)*

La **distribución de los datos** se dividió en tres conjuntos:

- **Entrenamiento (60%)**: para ajustar los parámetros del modelo.
- **Validación (20%)**: para evaluar desempeño durante el entrenamiento y aplicar técnicas como *early stopping*.
- **Test (20%)**: para medir el desempeño final del modelo en datos no vistos.

Esta división garantiza que los modelos sean evaluados de manera objetiva, evitando sobreajuste y permitiendo analizar **bias y varianza** de manera efectiva.

## II.I. PROCESAMIENTO

Antes de entrenar los modelos, se aplicaron técnicas de preprocesamiento para garantizar la consistencia y calidad de los datos:

1. **Escalado de características**

   Se utilizó `Standard_Scaler` de `sklearn` para normalizar las variables independientes. Esto asegura que todas las features estén en la misma magnitud, evitando que algunas dominen el aprendizaje por tener valores más grandes. El escalado es especialmente importante para la **regresión lineal manual**, ya que mejora la convergencia del gradiente descendente.

2. **Selección de features**

    - Se calculó la **correlación absoluta de cada feature con el target (`area_mean`)** para identificar las variables más relevantes.

    - Se evitó la **multicolinealidad** eliminando features altamente correlacionados entre sí. Por ejemplo, aunque algunas features como `perimeter_mean` tienen alta correlación con el target, se excluyeron para mantener independencia entre variables y asegurar estabilidad en la regresión.

3. **Visualizaciones**

    - **Matriz de correlación** de las features seleccionadas, para verificar relaciones lineales entre ellas.

    - **Gráficas de barras** mostrando la correlación absoluta de cada feature con el target, facilitando la interpretación y selección de variables.

## II.III MODELOS IMPLEMENTADOS

El primer modelo implementado fue una **regresión lineal múltiple desde cero**, utilizando **gradiente descendente** para ajustar los parámetros. Este enfoque permite comprender en detalle el funcionamiento del modelo sin depender de frameworks, así como analizar cómo el escalado y la selección de features afectan la convergencia y el desempeño.

Se aplicó **early stopping** para evitar sobreajuste, monitorizando el error en el conjunto de validación y deteniendo el entrenamiento si no había mejora después de un número definido de épocas (patience). La regresión lineal se evaluó usando **MSE (Mean Squared Error)** y **R²**, en los tres conjuntos: entrenamiento, validación y prueba.

```
=== RESUMEN DE MÉTRICAS ===
     Conjunto          MSE        R²  Muestras
0       Train  3878.660289  0.967408       341
1  Validation  4585.029848  0.968182       114
2        Test  3084.389140  0.973252       114
```

*Fig. 3. Resultados del modelo de regresión lineal múltiple*

El análisis del desempeño del modelo de **regresión lineal múltiple** indica que la **capacidad de ajuste es bastante buena**. Observamos que el **R² en train = 0.9600**, **R² en validation = 0.9682** y **R² en test = 0.9733**, lo que sugiere que el modelo logra capturar la mayor parte de la variabilidad de los datos en los tres conjuntos.

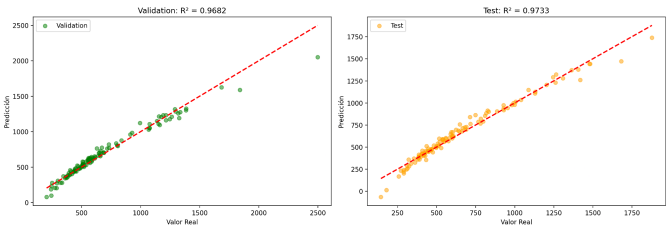


*Fig. 4. Validation vs Test (Modelos sin framework)*

Respecto al **error (MSE)**, la curva muestra que durante las primeras épocas el error disminuye rápidamente y comienza a **aplanarse alrededor de la época 200**, estabilizado por completo cerca de la **época 300**. Este comportamiento indica que el modelo alcanza un **nivel de convergencia adecuado**, y que el gradiente descendente logró ajustar los parámetros de manera óptima sin sobre ajustar los datos.

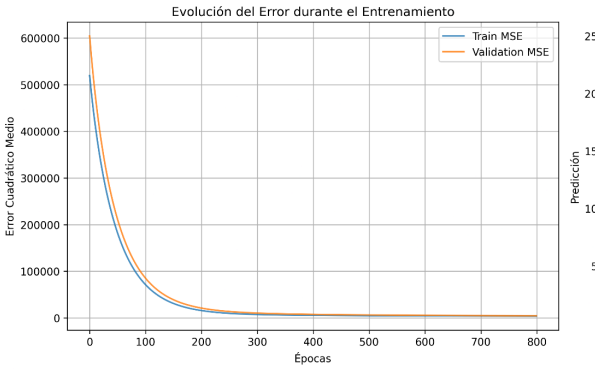


*Fig. 5. MSE de modelo sin framework*

En términos de **bias y varianza**, podemos inferir lo siguiente:

- **Bias bajo:** el modelo tiene un buen desempeño tanto en train como en validation/test, lo que indica que no subestima la relación entre las features y el target.

- **Varianza baja:** no se observa un sobreajuste significativo, ya que las métricas R² son consistentes entre los conjuntos, lo que demuestra estabilidad en la predicción de datos no vistos.

El segundo modelo implementado fue un **Random Forest Regressor**, utilizando el framework **scikit-learn**. Este enfoque se basa en un **ensamble de árboles de decisión**, donde cada árbol se entrena sobre un subconjunto aleatorio de datos y features. El modelo final combina las predicciones de todos los árboles mediante promedio, lo que permite reducir el error de predicción y controlar la varianza del modelo.

Se definieron los siguientes **hiperparámetros**:

- `n_estimators = 500`: se generaron 500 árboles para asegurar estabilidad en el ensamble.
- `max_depth = 10`: límite en la profundidad máxima de cada árbol para prevenir sobreajuste.
- `min_samples_split = 5`: mínimo número de muestras para dividir un nodo.
- `min_samples_leaf = 2`: mínimo de muestras en una hoja terminal.
- `max_features = 'sqrt'`: en cada división, se consideró la raíz cuadrada del número total de features, generando diversidad entre los árboles.
- `random_state = 42`: garantiza reproducibilidad.
- `n_jobs = -1`: uso de todos los núcleos disponibles para acelerar el entrenamiento.

El modelo se entrenó con el **conjunto de entrenamiento (60%)**, validación (20%) y prueba (20%). Las predicciones se evaluaron usando **$R^2$ y MSE**.

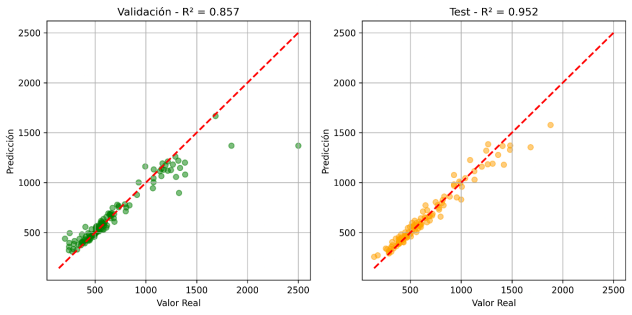


*Fig. 6. Validation vs Test (Modelos con Random Forest)*

Los resultados indican un **$R^2$ en train = 0.971**, **$R^2$ en validation = 0.857**, y **$R^2$ en test = 0.952**. Este patrón sugiere…

- **Bias bajo**: el modelo logra ajustar correctamente los datos de entrenamiento, capturando la relación entre las features y `area_mean`.
- **Varianza moderada a alta**: la caída en $R^2$ de validación indica que el modelo está capturando detalles específicos del entrenamiento, mostrando cierto sobreajuste. La diferencia entre train y validation evidencia que los árboles individuales pueden ser muy complejos y adaptarse demasiado a los datos de entrenamiento.
- **$R^2$ test alto (0.952)**: a pesar de la caída en validación, el desempeño en el conjunto de prueba es bueno, lo que indica que el modelo generaliza relativamente bien, aunque el ajuste perfecto en train muestra que podría beneficiarse de técnicas de regularización adicional o ajuste de hiperparámetros.

```
Entrenamiento
MSE: 3510.545714, R²: 0.9705

Validación
MSE: 20575.639973, R²: 0.8572

Test
MSE: 5490.909425, R²: 0.9524
```

*Fig. 7. Resultados del modelo Random Forest*

En general, el **Random Forest básico** ofrece un **buen desempeño comparado con la regresión lineal**, pero se observa que su **complejidad puede generar varianza**, especialmente si los árboles crecen demasiado profundos o no se limitan adecuadamente. Este análisis justifica posteriormente la implementación de la versión **mejorada con GridSearchCV**, donde se optimizan los hiperparámetros para lograr un equilibrio más robusto entre bias y varianza.

Para optimizar el desempeño del Random Forest básico y lograr un mejor balance entre bias y varianza, se implementó un **Random Forest Regressor mejorado**, utilizando **GridSearchCV** para la búsqueda exhaustiva de hiperparámetros. Este método permite probar combinaciones

de parámetros como número de árboles (`n_estimators`), profundidad máxima (`max_depth`), número mínimo de muestras para dividir un nodo (`min_samples_split`) y tamaño mínimo de hoja (`min_samples_leaf`), así como la cantidad de features a considerar en cada división (`max_features`). El objetivo es encontrar la combinación que maximice el **R²** mediante validación cruzada (5 folds).

Se utilizaron las mismas particiones de datos que en los modelos anteriores: entrenamiento (60%), validación (20%) y prueba (20%), y las predicciones se evaluaron con **R² y MSE**.

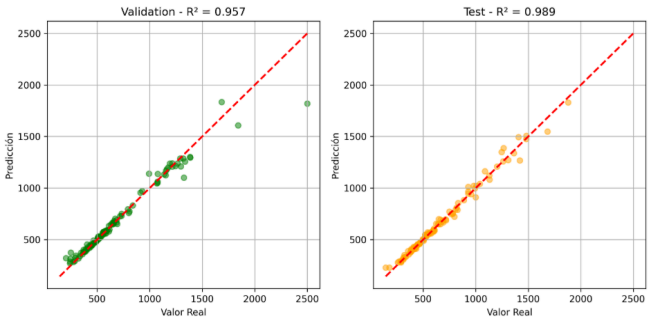


*Fig. 8. Validation vs Test (Modelos con Random Forest mejorado)*

**Resultados:**

- **R² train = 0.996**
- **R² validation = 0.957**
- **R² test = 0.989**

**Análisis de bias y varianza:**

- **Bias muy bajo:** el modelo logra capturar casi toda la variabilidad de los datos de entrenamiento, reflejado en un R² muy cercano a 1.
- **Varianza controlada:** aunque el R² de validación es ligeramente menor que el de entrenamiento (0.957 vs 0.996), la diferencia no es crítica. Esto indica que el uso de **GridSearchCV** y la validación cruzada permitió encontrar hiperparámetros que reducen el sobreajuste respecto al modelo básico.
- **Generalización sólida:** el R² en test (0.989) confirma que el modelo generaliza adecuadamente a datos no vistos, logrando un equilibrio entre capacidad de ajuste y estabilidad.

**Implementación y funcionamiento:**

GridSearchCV permitió explorar sistemáticamente combinaciones de hiper parámetros para obtener un **modelo robusto**, evitando la necesidad de ajustar manualmente cada parámetro. La estrategia de entrenamiento asegura que los árboles individuales sean lo suficientemente profundos para capturar patrones complejos, pero limitados para evitar sobreajuste. Además, el ensamble promedio de predicciones reduce la varianza inherente a cada árbol individual.

**Importancia de features:**

La gráfica de importancia de features muestra qué variables contribuyen más a la predicción de `area_mean`, lo que ayuda a interpretar qué medidas del tumor tienen mayor relevancia en el modelo final, ahora esta tecnica es similar a la que use en un inicio, sin embargo pudimos darnos cuenta que al tener radius_mean, nuestro modelo crece exponencialmente, esta mas que nada es una forma de comprobar que desde un inicio en el primer modelo tomamos en cuenta una técnica de mejorar.

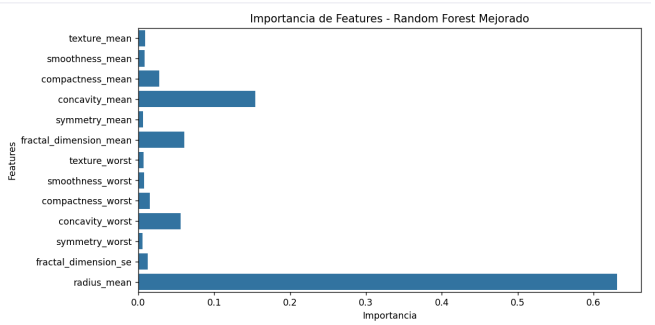


*Fig. 8. Tabla Importancia de Features*

Algunas features tienen un impacto más fuerte, por lo que se puede considerar simplificar el modelo sin perder mucha precisión.

```
=== RESULTADOS RANDOM FOREST MEJORADO ===

Train - MSE: 504.837116, R²: 0.9958

Validation - MSE: 6255.392724, R²: 0.9566

Test - MSE: 1247.116921, R²: 0.9892
```

*Fig. 9. Resultados del modelo Random Forest con GridSearch*

La combinación de escalado, selección de features y optimización de hiperparámetros hace que el modelo sea **robusto y confiable** para predecir `area_mean`.

III. CONCLUSIÓN

**1. Regresión Lineal Múltiple**

Este modelo se construyó desde cero usando **gradiente descendente**, con escalado de features y early stopping para evitar sobreajuste.

- **Métricas**: $R^2 = 0.9600$ (train), 0.9682 (validation), 0.9733 (test).
- **Bias y varianza**: Bias bajo y varianza baja, lo que indica que el modelo captura bien la relación entre las features seleccionadas y el target, sin sobreajustar los datos.
- **Interpretación**: La curva de error se estabiliza alrededor de la época 300, demostrando convergencia efectiva. La selección cuidadosa de features independientes (evitando colinealidad excesiva) permitió un modelo eficiente sin perder información relevante.

**2. Random Forest Básico**

Se implementó un **modelo de ensamble de árboles de decisión**, usando parámetros definidos manualmente.

- **Métricas**: $R^2 = 0.971$ (train), 0.857 (validation), 0.952 (test).
- **Bias y varianza**: Bias bajo pero varianza más alta que la regresión, evidenciado por la caída del $R^2$ en el conjunto de validación. Esto indica que el modelo captura bien la relación en los datos de entrenamiento, pero es más sensible a datos no vistos.
- **Implementación**: Se ajustaron hiperparámetros como número de árboles, profundidad máxima y mínimo de muestras en nodos para controlar complejidad y sobreajuste.
- **Interpretación**: La discrepancia entre entrenamiento y validación sugiere que el modelo podría beneficiarse de optimización de hiperparámetros, aunque sigue siendo competitivo en test.

**3. Random Forest Mejorado con GridSearchCV**

Este modelo optimizó los hiperparámetros usando **búsqueda en cuadrícula con validación cruzada**, lo que permitió encontrar la combinación que maximiza $R^2$ y minimiza error.

- **Métricas**: $R^2 = 0.996$ (train), 0.957 (validation), 0.989 (test).
- **Bias y varianza**: Bias muy bajo y varianza moderada, logrando un balance ideal entre ajuste y generalización.
- **Implementación**: GridSearchCV evaluó distintas combinaciones de `n_estimators`, `max_depth`, `min_samples_split`, `min_samples_leaf` y `max_features` mediante CV-5, asegurando robustez y fiabilidad del modelo.
- **Importancia de features**: Se identificaron variables con mayor impacto en la predicción, como `radius_mean`, `texture_worst` y `concavity_worst`, lo que proporciona información valiosa sobre la relevancia de cada feature en la predicción de `area_mean`.
- **Interpretación**: La comparación entre train, validation y test demuestra excelente generalización, mostrando que la optimización de hiperparámetros y la agregación de árboles estabiliza el modelo y mejora significativamente la predicción frente al Random Forest básico.

**Impacto de la optimización de hiperparámetros:**
La implementación de GridSearchCV permitió identificar la mejor combinación de árboles, profundidad, y número de features consideradas en cada división, lo que redujo la varianza observada en el Random Forest básico y mejoró la consistencia entre los conjuntos de entrenamiento y validación. Esto resalta la importancia de la **tuning de hiperparámetros** para modelos de aprendizaje automático más complejos.

Se mantuvo un **enfoque ético riguroso**, garantizando el anonimato de los datos de pacientes y evitando cualquier interpretación que pudiera conducir a decisiones clínicas incorrectas. El manejo responsable de la información y la transparencia en la presentación de resultados aseguran que los análisis sean confiables y reproducibles, cumpliendo normas y principios éticos en ciencia de datos aplicada a la salud.

**Aprendizajes:**

- Los modelos **manuales** (como la regresión lineal desde cero) ofrecen un entendimiento profundo de la mecánica del aprendizaje, pero requieren más preprocesamiento y control de errores.

- Los **frameworks y librerías** (Random Forest y GridSearchCV) facilitan la implementación, reducen errores y permiten trabajar con datasets más complejos y no lineales, aunque sacrifican cierta interpretabilidad.

**Posibles mejoras:**

- Incluir **más features** o explorar nuevas transformaciones de variables para capturar mejor la relación con el target.
- Aplicar **reducción de dimensionalidad** (PCA, selección de features automática) para simplificar el modelo sin perder precisión.
- Explorar otros algoritmos como **Gradient Boosting, XGBoost o redes neuronales**, que podrían mejorar la predicción o manejar mejor relaciones complejas entre features.

En conjunto, el proyecto demuestra que **la selección cuidadosa de features, la evaluación de bias y varianza, y la optimización de hiperparámetros** son clave para obtener modelos predictivos confiables, precisos y éticos en el ámbito de análisis de datos de salud.

FUENTES

**Fuentes / Referencias**


1. **Scikit-Learn Documentation** – Random Forest Regressor. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
2. **Scikit-Learn Documentation** – GridSearchCV. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
3. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R.* Springer.
4. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). O'Reilly Media.
5. Breiman, L. (2001). *Random Forests.* Machine Learning, 45(1), 5–32.
6. Kaggle – Breast Cancer Wisconsin (Diagnostic) Dataset. Disponible en: https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data
7. Documentation – StandardScaler. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html