Caso III

Anyelin Arias Camacho, Guissell Margarita Betancur Oviedo,
Pablo Garro Telles, Ericka Carolina Salas Moreno,
Kevin Fernando Sibaja Alvarado, Rodrigo Arturo Vásquez Chavarría

Lead University, Costa Rica

{anyelin.arias, guissell.betancur, pablo.garro,
ericka.salas, kevin.sibaja, arturo.vasquez}@ulead.ac.cr

Julio 2025

1. Definición de Modelos Implementados

1.1. Modelo de Preprocesamiento

El modelo de preprocesamiento se define como un *pipeline* que transforma los datos brutos en un conjunto de características listo para el entrenamiento. Consta de las siguientes etapas:

1. **Imputación de valores faltantes**
   Estrategia de imputación (`media`, `mediana`, `moda`) como hiperparámetro:

$$\hat{x}_j^{(i)} = \begin{cases} \dfrac{1}{n_j} \displaystyle\sum_{k:\, x_j^{(k)} \text{ no faltante}} x_j^{(k)}, & \text{(media)} \\ \text{mediana}\{x_j^{(k)}\}, & \text{(mediana or moda)} \end{cases}$$

2. **Escalado de características**
   Selección de escalador: `StandardScaler` o `MinMaxScaler` como hiperparámetro:

$$z_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{\sigma_j}, \quad \sigma_j = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(x_j^{(k)} - \mu_j\right)^2}$$

   o bien

$$z_j^{(i)} = \frac{x_j^{(i)} - \min_k x_j^{(k)}}{\max_k x_j^{(k)} - \min_k x_j^{(k)}}.$$

3. **Codificación de variables categóricas**
   Codificador: `OneHotEncoder` u `OrdinalEncoder` como hiperparámetro:

$$e_{j,c}^{(i)} = \begin{cases} 1 & \text{si } x_j^{(i)} = c, \\ 0 & \text{en otro caso.} \end{cases}$$

4. **Generación de características polinomiales (opcional)**
   Grado d como hiperparámetro. Se agregan términos $\{x_1^a\, x_2^b \cdots x_p^c\}$ con $a+b+\cdots+c \le d$.

Hiperparámetros a optimizar Mediante búsqueda exhaustiva o aproximación genética:

- `imputer__strategy` $\in$ {`media`, `mediana`, `moda`}
- `scaler` $\in$ {`StandardScaler`, `MinMaxScaler`}
- `poly__degree` $\in$ {1,2,3}

1.2. Modelo de Algoritmo de Aprendizaje

El algoritmo de aprendizaje (LA) es el componente encargado de ajustar un modelo predictivo a los datos procesados. Formalmente, dados un conjunto de entrenamiento $(\mathbf{X}, \mathbf{y})$, el objetivo del algoritmo es encontrar una función $f(\mathbf{x}; \theta)$ parametrizada por θ, que minimice una función de pérdida $\mathcal{L}$ sobre los datos:

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\left(f(\mathbf{x}^{(i)}; \theta), y^{(i)}\right)$$

Componentes principales:

1. **Función modelo** $f(\mathbf{x}; \theta)$: puede ser una regresión logística, árbol de decisión, red neuronal, SVM, etc.
2. **Función de pérdida** $\mathcal{L}$: depende de la tarea (clasificación, regresión). Ejemplos:

$$\mathcal{L}_{\text{clasif.}} = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}), \quad \mathcal{L}_{\text{reg.}} = (y - \hat{y})^2$$

3. **Método de optimización**: algoritmo que ajusta los parámetros del modelo. Ejemplos: Gradiente Descendente, Algoritmos Genéticos, Búsqueda Aleatoria.

Algoritmos de búsqueda utilizados:

- **Búsqueda Aleatoria** (`RandomSearch`): evalúa combinaciones aleatorias de hiperparámetros.
- **Búsqueda Genética** (`GeneticSearch`): aplica evolución de poblaciones con operadores de cruce, mutación y selección:

$$\text{población}_{t+1} \leftarrow \text{selección}\left(\text{mutación}\left(\text{cruce}\left(\text{población}_t\right)\right)\right)$$

Hiperparámetros del modelo a optimizar Dependiendo del tipo de modelo:

- `Regresión logística:` `C` (regularización), `penalty` (`l1`, `l2`)
- `Árbol de decisión:` `max_depth`, `min_samples_split`
- `Red neuronal:` número de capas, neuronas por capa, función de activación, tasa de aprendizaje

Objetivo general de optimización Maximizar la métrica de validación cruzada sobre el conjunto de validación, por ejemplo:

$$\text{Accuracy} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\left[\hat{y}^{(i)} = y^{(i)}\right] \quad \text{o} \quad \text{F1-score, AUC, RMSE, } \mathrm{R}^2, \text{etc.}$$

Salida del algoritmo de aprendizaje

- $\hat{\theta}$: parámetros entrenados del modelo
- $f(\mathbf{x}; \hat{\theta})$: modelo final que puede ser usado para predicción

1.3 Modelo de Selección de Algoritmos

El objetivo del modelo de selección de algoritmos (AS) es identificar el modelo de aprendizaje más adecuado para un problema dado, considerando el rendimiento de múltiples algoritmos sobre un conjunto de datos preprocesado.

1.1. Componentes del modelo AS

- **Entrada:** conjunto de datos preprocesado (X, y), donde $X \in R^{n \times p}$ y $y \in R^n$.
- **Salida:** algoritmo de aprendizaje óptimo A^* con sus hiperparámetros óptimos θ^*.

1.2. Algoritmos evaluados

- **Clasificación:** `LogisticRegression`, `RandomForestClassifier`, `SVC`, `KNeighborsClassifier`, `XGBoostClassifier`.
- **Regresión:** `LinearRegression`, `Ridge`, `Lasso`, `RandomForestRegressor`, `XGBoostRegressor`.

Cada algoritmo tiene sus propios hiperparámetros. La tarea consiste en buscar tanto el mejor algoritmo como la mejor combinación de hiperparámetros, bajo un criterio de evaluación definido (por ejemplo, `accuracy`, `RMSE`, etc.).

2. Especificación matemática

Dado un conjunto de algoritmos candidatos $\mathcal{A} = \{A_1, A_2, \ldots, A_m\}$ y un conjunto de hiperparámetros posibles Θ_i para cada A_i, buscamos:

$$(A^*, \theta^*) = \arg \max_{A_i \in \mathcal{A},\ \theta \in \Theta_i} \text{Score}_{cv}(A_i(\theta), X, y)$$

Donde:

- Score_{cv} es la métrica de validación cruzada (por ejemplo, `accuracy` para clasificación o R^2 para regresión).
- $A_i(\theta)$ es el modelo A_i configurado con los hiperparámetros θ.

3. Métodos de búsqueda utilizados

3.1. Fuerza bruta (Grid Search)

Se realiza una búsqueda exhaustiva sobre un espacio discreto de hiperparámetros:

$$\forall A_i \in \mathcal{A},\ \forall \theta \in \Theta_i,\ \text{evaluar Score}_{cv}(A_i(\theta), X, y)$$

Ventaja: garantiza encontrar la mejor combinación dentro del espacio evaluado.
Desventaja: alto costo computacional.

3.2. Algoritmos Genéticos (Genetic Algorithm, GA)

Los algoritmos genéticos modelan el problema como una población de soluciones (individuos), que evolucionan mediante operadores genéticos (selección, cruce, mutación).

- Cada individuo representa un algoritmo A_i y una combinación de hiperparámetros θ.
- La función de aptitud es Score_{cv}.

La evolución sigue el siguiente proceso:

1. Inicialización de población aleatoria.
2. Evaluación de *fitness*.
3. Selección de los mejores individuos.
4. Cruce y mutación.
5. Repetición por generaciones.

Formulación matemática:

$$\text{Fitness}(x) = \text{Score}_{cv}(A_i(\theta), X, y), \quad x = \text{codificación}(A_i, \theta)$$