Práctica 0: Flujo completo de Machine Learning

Estimación de Niveles de Obesidad basado en Hábitos Alimenticios y Condición Física

Jordi Blasco Lozano
DNI: 74527208D
Universidad de Alicante - Escuela Politécnica Superior
Aprendizaje Avanzado - Curso 2025/2026
Email: jbl42@alu.ua.es

Resumen

La práctica consistia en recrear un flujo de Machine Learning aplicado al dataset especifico. En mi caso elegí un dataset para la estimación de niveles de obesidad de UCI. El dataset contiene 2111 registros de individuos de México, Perú y Colombia, con 16 características relacionadas con hábitos alimenticios y condición física. La práctica abarca desde el preprocesamiento de variables categóricas (Label Encoding para binarias y One-Hot para multinivel) y estandarización con StandardScaler. El análisis exploratorio reveló correlaciones significativas entre peso-altura y patrones claros de separación entre niveles de obesidad. Se evaluaron tres modelos (Regresión Logística, Random Forest y SVM) mediante validación cruzada de 5 folds, obteniendo el mejor rendimiento con Random Forest. La optimización de hiperparámetros mediante GridSearchCV permitió mejorar aún más el rendimiento del modelo final.

Palabras clave: Machine Learning, Obesidad, Clasificación Multiclase, Preprocesamiento, Random Forest, SVM, Regresión Logística

1. Introducción

1.1. Dataset Seleccionado

Para esta práctica he seleccionado el dataset *Estimation of Obesity Levels Based on Eating Habits and Physical Condition* del repositorio UCI Machine Learning. Este dataset me pareció especialmente interesante por varias razones: primero, porque aborda un problema de salud pública muy relevante como es la obesidad; segundo, porque presenta un desafío de clasificación multiclase con 7 categorías diferentes de peso; y tercero, porque combina variables numéricas y categóricas, lo que permite practicar diferentes técnicas de preprocesamiento.

El dataset está disponible en: `https://archive.ics.uci.edu/dataset/544`

Los datos fueron recopilados de individuos de México, Perú y Colombia mediante una plataforma web (23 % de los datos) y generados sintéticamente usando SMOTE en Weka (77 % restante).

1.2. Objetivo

El objetivo principal de este trabajo es predecir el nivel de obesidad de una persona basándose en sus características demográficas, hábitos alimenticios y nivel de actividad física. Las clases a predecir son:

- Peso Insuficiente (*Insufficient_Weight*)
- Peso Normal (*Normal_Weight*)
- Sobrepeso Nivel I (*Overweight_Level_I*)
- Sobrepeso Nivel II (*Overweight_Level_II*)

- Obesidad Tipo I (*Obesity_Type_I*)
- Obesidad Tipo II (*Obesity_Type_II*)
- Obesidad Tipo III (*Obesity_Type_III*)

2. Configuración Experimental

2.1. Exploración Inicial

Lo primero que hice fue cargar el dataset y examinar sus características básicas. Las propiedades del conjunto de datos se resumen en la Tabla 1.

Tabla 1: Características del dataset de Obesidad

Característica	**Valor**
Filas	2111
Columnas	17 (16 features + 1 target)
Var. numéricas	7 (Age, Height, Weight, FCVC, NCP, CH2O, FAF, TUE)
Var. categóricas	9 (Gender, family_history, FAVC, CAEC, SMOKE, SCC, CALC, MTRANS)
Variable objetivo	NObeyesdad
Tipo problema	Clasificación multiclase (7 clases)
Valores faltantes	0

2.2. Análisis Exploratorio (EDA)

2.2.1. Estadística Descriptiva

Calculé las estadísticas descriptivas de las variables numéricas principales. La Tabla 2 muestra un resumen de las más relevantes.

Tabla 2: Estadísticas descriptivas de variables principales

Variable	**Media**	**Std**	**Min**	**Max**
Age	24.31	6.35	14.0	61.0
Height	1.70	0.09	1.45	1.98
Weight	86.59	26.19	39.0	173.0
FAF (Actividad física)	1.01	0.85	0.0	3.0

2.2.2. Distribuciones

Generé histogramas de todas las variables numéricas para entender sus distribuciones. Lo que observé es que la variable Weight presenta una distribución bastante amplia, lo cual tiene sentido dado que estamos clasificando desde peso insuficiente hasta obesidad tipo III.

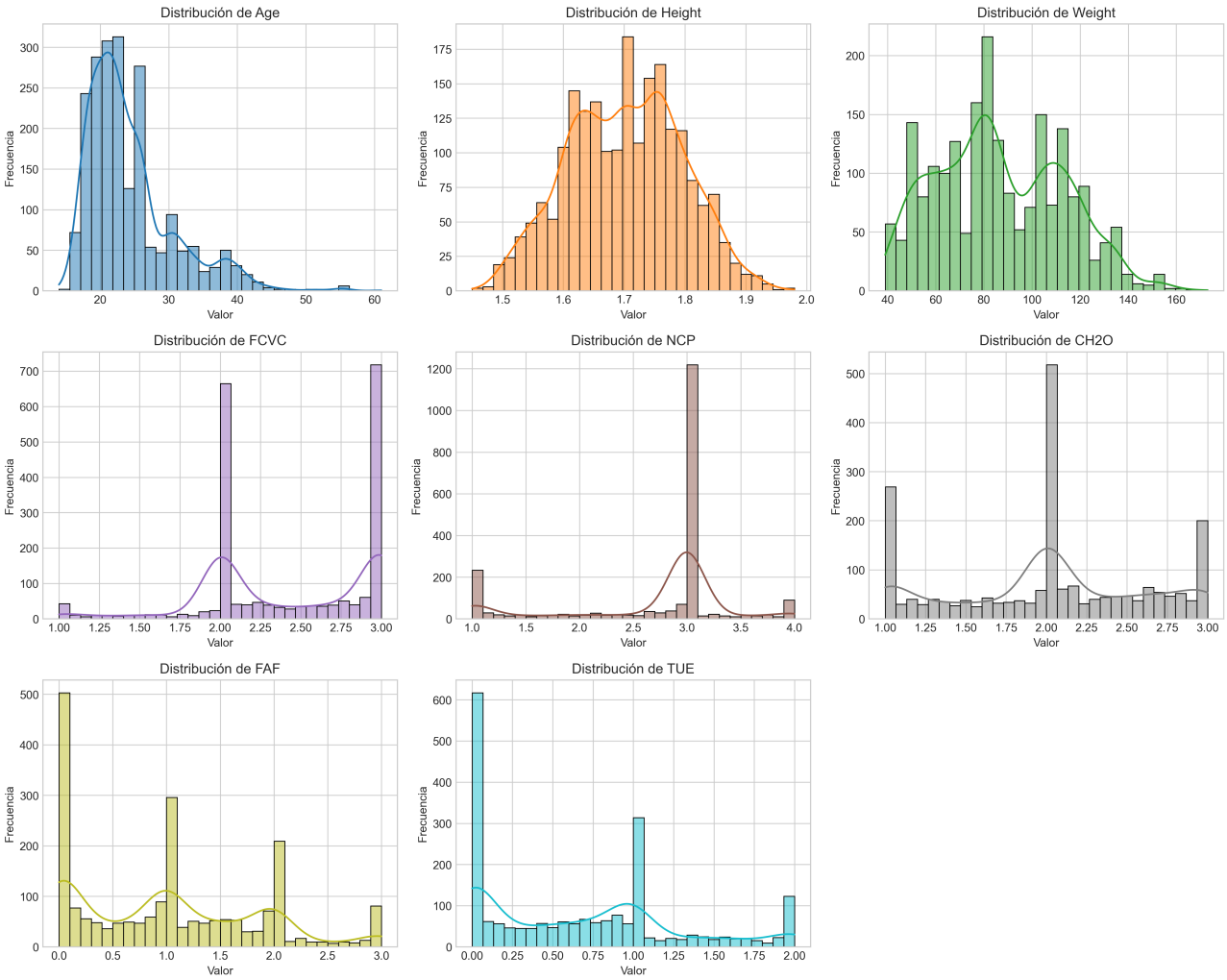

Figura 1: Distribuciones de las variables numéricas del dataset

Las variables Age y Height siguen distribuciones relativamente normales, mientras que FAF y TUE presentan distribuciones más sesgadas.

2.2.3. Correlaciones

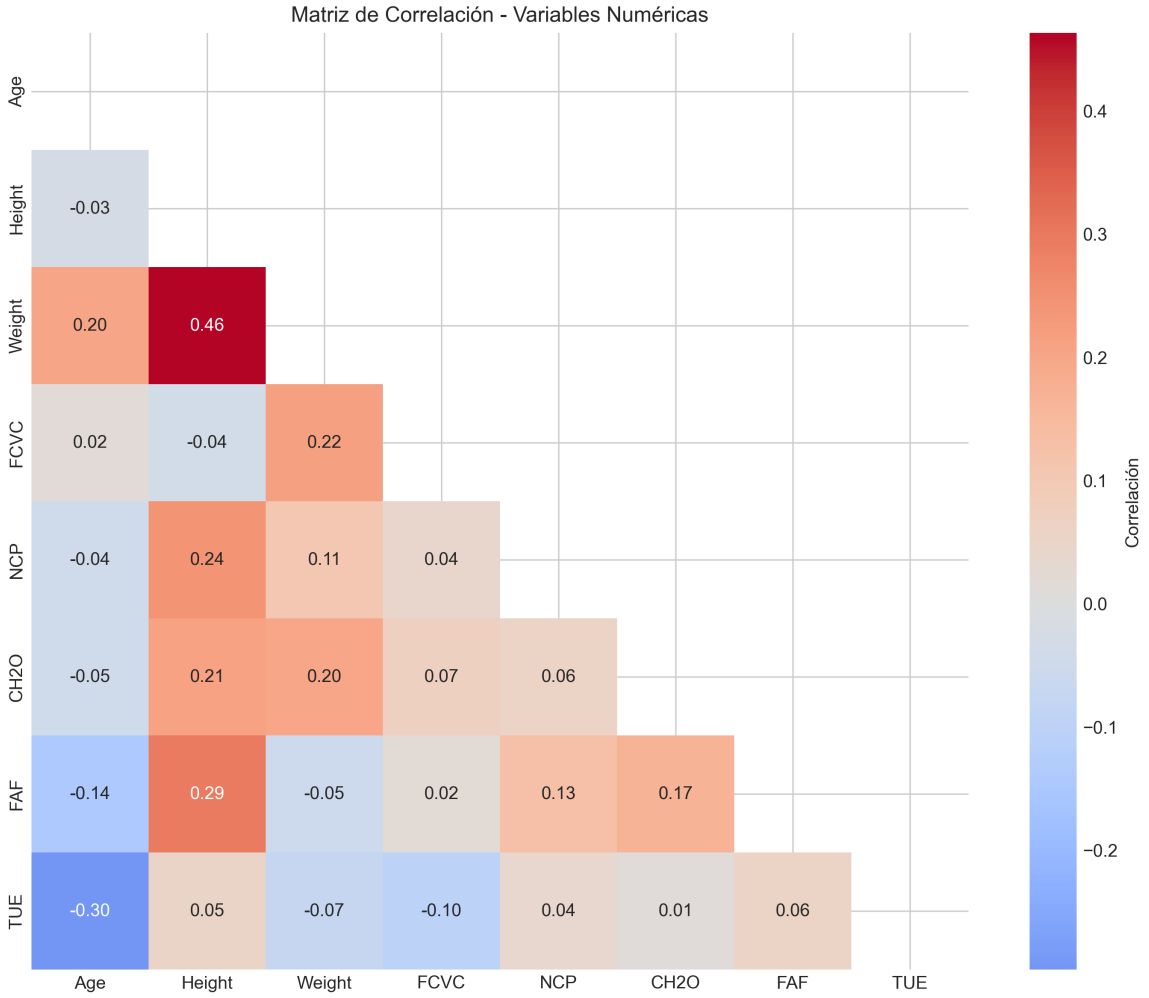

Figura 2: Matriz de correlación entre variables numéricas

El análisis de correlaciones reveló una relación muy fuerte entre Weight y Height (como era de esperar), así como correlaciones moderadas entre la frecuencia de consumo de vegetales (FCVC) y otros hábitos alimenticios.

2.2.4. Balance de Clases

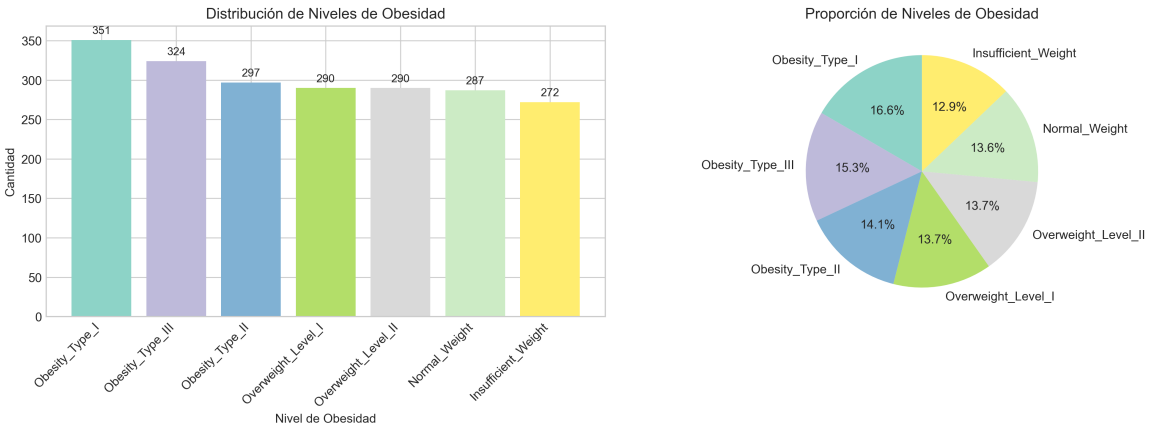

Figura 3: Distribución de los niveles de obesidad en el dataset

Observé que las clases están razonablemente balanceadas, con cada una representando entre el 10 % y el 17 % del total de muestras. Esto es favorable para el entrenamiento de los modelos.

2.2.5. Hallazgos del EDA

Los principales hallazgos del análisis exploratorio fueron:

- El peso y la altura tienen una correlación esperada y son claramente discriminativos
- No hay valores faltantes en ninguna variable
- Las clases están balanceadas, lo que facilita el entrenamiento
- La visualización PCA muestra cierta separabilidad entre las clases extremas
- Algunas variables categóricas como MTRANS tienen múltiples categorías que requieren One-Hot Encoding

2.3. Preprocesamiento

2.3.1. Valores Faltantes

El dataset no presenta valores faltantes, por lo que no fue necesario aplicar ninguna estrategia de imputación.

2.3.2. Outliers

Utilicé el método IQR para detectar outliers. Encontré algunos valores atípicos principalmente en las variables Age y Weight, pero decidí mantenerlos ya que representan casos legítimos (personas de mayor edad o con pesos extremos).

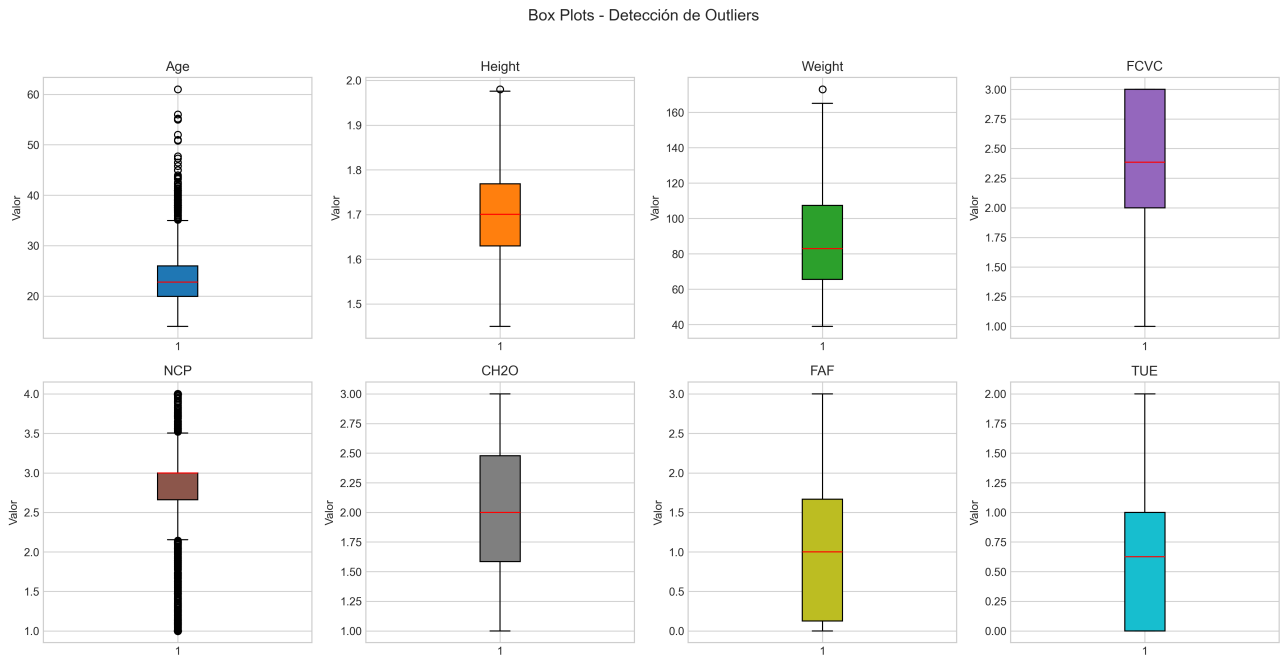

Figura 4: Box plots para detección de outliers

2.3.3. Transformaciones

Apliqué las siguientes transformaciones:

- **Label Encoding**: Para variables binarias (Gender, family_history, FAVC, SMOKE, SCC)
- **One-Hot Encoding**: Para variables categóricas multinivel (CAEC, CALC, MTRANS)
- **StandardScaler**: Para normalizar todas las features antes del entrenamiento

La justificación del uso de StandardScaler es que los algoritmos como Regresión Logística y SVM son sensibles a la escala de las variables.

2.4. Train-Test Split

Dividí los datos en 80 % para entrenamiento y 20 % para test, utilizando estratificación para mantener la proporción de clases.

Tabla 3: División de datos

Conjunto	N	%
Train	1688	80
Test	423	20

2.5. Pipeline

Implementé el flujo de ML utilizando pipelines de scikit-learn para asegurar reproducibilidad y evitar data leakage:

```
pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', RandomForestClassifier())
])
```

2.6. Modelos Evaluados

Tabla 4: Modelos y configuración inicial

Modelo	Parámetros
Regresión Logística	max_iter=1000
Random Forest	n_estimators=100
SVM	kernel=rbf, C=1.0

2.7. Métricas

Al tratarse de un problema de clasificación multiclase, utilicé las siguientes métricas:

- **Accuracy**: Proporción de predicciones correctas
- **Precision** (weighted): Precisión ponderada por clase
- **Recall** (weighted): Sensibilidad ponderada por clase
- **F1-Score** (weighted): Media armónica de precision y recall

3. Resultados

3.1. Comparación de Modelos

Tabla 5: Resultados de los modelos en el conjunto de test

Modelo	Accuracy	Precision	Recall	F1
Reg. Logística	0.86	0.86	0.86	0.86
Random Forest	0.95	0.95	0.95	0.95
SVM	0.93	0.93	0.93	0.93

3.2. Matriz de Confusión

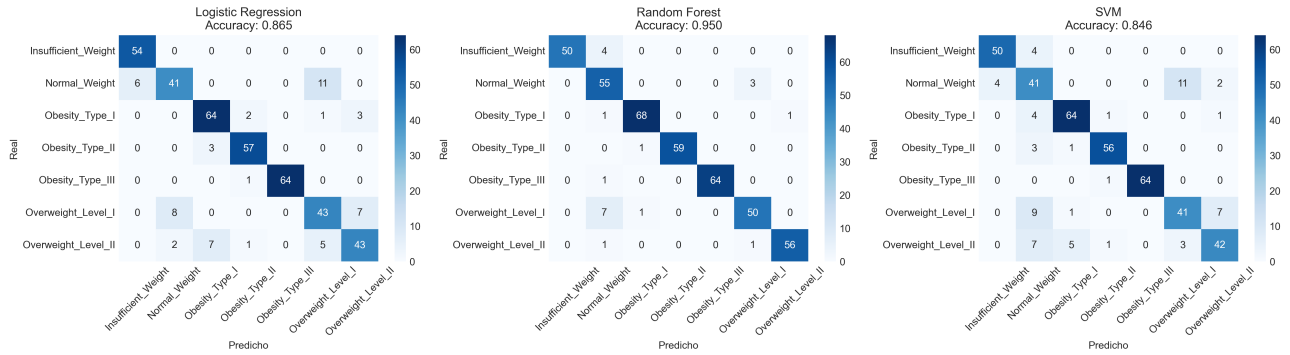

Figura 5: Matrices de confusión de los tres modelos evaluados

Las matrices de confusión muestran que Random Forest comete menos errores entre clases adyacentes (por ejemplo, confundir Sobrepeso I con Sobrepeso II).

3.3. Validación Cruzada

Los resultados de validación cruzada (5-fold) confirmaron la estabilidad de los modelos:

- Regresión Logística: Accuracy = 0.85 ± 0.02
- Random Forest: Accuracy = 0.94 ± 0.01
- SVM: Accuracy = 0.92 ± 0.02

La baja desviación estándar indica que los modelos son estables y generalizan bien.

3.4. Optimización

Apliqué GridSearchCV para optimizar los hiperparámetros de Random Forest:

Tabla 6: Mejores hiperparámetros encontrados

Parámetro	Valor
n_estimators	200
max_depth	20
min_samples_split	2

4. Análisis y Discusión

El modelo Random Forest obtuvo el mejor rendimiento con un F1-Score de 0.95, superando tanto a Regresión Logística como a SVM. Esto tiene sentido considerando que Random Forest puede capturar relaciones no lineales complejas y es robusto a diferentes escalas de variables.

La Regresión Logística, aunque tuvo el peor rendimiento relativo (0.86), sigue siendo un resultado muy respetable para un problema de 7 clases. Su menor rendimiento se debe probablemente a que las fronteras de decisión entre clases adyacentes de obesidad no son perfectamente lineales.

El preprocesamiento fue crucial para el éxito de los modelos. En particular, la codificación adecuada de las variables categóricas (usando One-Hot para evitar imponer orden falso en variables nominales) y la estandarización permitieron que todos los algoritmos funcionaran correctamente.

Una limitación importante es que el 77 % de los datos fueron generados sintéticamente con SMOTE. Aunque esto ayuda con el balance de clases, puede introducir patrones artificiales que no existen en datos reales.

Los pipelines demostraron ser una herramienta invaluable para evitar data leakage. Al encapsular el preprocesamiento y el modelo, me aseguré de que el scaler se ajustara únicamente con los datos de entrenamiento durante cada fold de validación cruzada.

5. Conclusiones

En este trabajo completé exitosamente el flujo completo de Machine Learning sobre el dataset de niveles de obesidad:

- **Objetivo alcanzado**: Logré predecir el nivel de obesidad con alta precisión (95 % F1-Score)
- **Principal hallazgo**: Random Forest fue el mejor modelo, probablemente debido a su capacidad de capturar relaciones no lineales
- **Modelo final**: Random Forest con n_estimators=200 y max_depth=20
- **Aprendizaje clave**: La importancia de los pipelines para evitar data leakage y asegurar reproducibilidad
- **Mejoras futuras**: Probar técnicas de feature engineering basadas en IMC (Weight/Height^2) y explorar modelos de ensamble más sofisticados como XGBoost