PREDICCIÓN DEL ABANDONO Y ÉXITO ACADÉMICO DE ESTUDIANTES

Análisis de datos con WEKA

Manuel Pérez Vélez, Julia Sánchez Márquez

COMPLEMENTO DE BASES DE DATOS

02/04/2025

Contenido

1. Introducción

La educación es un derecho humano fundamental que desempeña un papel clave en el progreso de la humanidad como sociedad. Es la herramienta más importante para combatir la pobreza, mejorar la salud y promover la paz y la estabilidad social. Sin embargo, en el mundo actual, el abandono escolar es uno de los mayores riesgos para el futuro de la sociedad y el desarrollo personal de miles de personas. Esta problemática, cuanto menos compleja y que atraviesa variedad de posibles factores, no solo afecta a la persona que deja sus estudios, sino que también tiene implicaciones económicas y sociales a nivel global. Es una realidad que existe una crisis de aprendizaje, por ejemplo, durante la pandemia de COVID-19 más de 70 millones de personas cayeron en la pobreza, millones de niños perdieron un año de escolaridad y, tres años después, las pérdidas de aprendizaje sufridas no se han recuperado. Si los niños no pueden comprender un texto a los 10 años, es poco probable que lleguen a leer con fluidez, no prosperarán más adelante en la escuela y no podrán impulsar sus carreras profesionales y las economías de sus países en un futuro.

Uno de los desafíos que enfrentan las instituciones a nivel mundial es cómo abordar eficazmente los distintos estilos de aprendizaje de cada alumno y el rendimiento académico de los estudiantes, y, con esta información mejorar la calidad del proceso educativo. En este contexto, la capacidad de predecir y anticipar posibles dificultades cobra una gran importancia, especialmente para aquellas instituciones que buscan implementar estrategias de apoyo y orientación a estudiantes en riesgo de fracaso académico o abandono.

Gracias al constante progreso tecnológico, y al desarrollo de la ciencia de datos, se han podido desarrollar herramientas como Weka, que nos permiten aplicar algoritmos de minería de datos para analizar grandes volúmenes de información, en este caso, información educativa, que nos permitirá descubrir recurrencia de patrones que podrán predecir con precisión el riesgo de deserción.

En este trabajo de investigación se tiene como objetivo examinar la herramienta Weka y todas las funcionalidades que puede ofrecer para llegar a unos resultados coherentes y fiables. Para ello, este trabajo busca contribuir a la reducción del fracaso académico en la educación superior mediante técnicas de aprendizaje automático para identificar a estudiantes en riesgo de fracaso en una etapa temprana de su trayectoria académica, así, se podrían implementar estrategias de apoyo en los centros. Para ello se utilizan datos del Instituto Politécnico de Portalegre (IPP), Portugal, con el fin de construir modelos de clasificación que permitan predecir el riesgo de que los estudiantes no terminen sus estudios a tiempo. El objetivo principal es proporcionar un sistema que permita identificar, desde el momento en que el estudiante se matricula en la institución, a los estudiantes con posibles dificultades en su trayectoria académica, de modo que se puedan implementar estrategias de apoyo preventivas.

Este conjunto de datos, extraído del *UCI Machine Learning Repository*, incluye información conocida al momento de la matrícula del estudiante, como su trayectoria académica previa, datos demográficos y factores socioeconómicos. Además, también incorpora el rendimiento académico durante los dos cuatrimestres del curso del que los estudiantes están matriculados, lo que permite un análisis más completo del proceso formativo. Esta información es clave para desarrollar sistemas que ayuden a identificar y segmentar a los estudiantes en riesgo desde las primeras etapas de su trayectoria universitaria. A diferencia de otros estudios que solo consideran dos clases (éxito o fracaso), este dataset utiliza una tercera categoría: "inscrito", que representa a aquellos estudiantes que aún permanecen en el sistema. Esta clase intermedia permite diseñar estrategias de apoyo más específicas, diferenciando entre estudiantes con riesgo moderado y aquellos con alto riesgo de abandono.

Los resultados obtenidos mediante los modelos desarrollados permiten una predicción temprana del riesgo de abandono con un buen nivel de precisión. Esto demuestra la utilidad del aprendizaje automático en contextos educativos y refuerza la importancia de implementar soluciones tecnológicas en el ámbito académico. Las conclusiones extraídas apuntan a la viabilidad de intervenir de manera oportuna para mejorar la permanencia estudiantil.

El resto de este documento se estructura de la siguiente manera: la sección 2 describe la metodología, incluyendo una descripción de los datos, los métodos utilizados para abordar el conjunto de datos desequilibrado y los procedimientos de entrenamiento y evaluación de los modelos de clasificación; la sección 3 presenta y analiza los resultados e indica algunas líneas de trabajo futuro; la sección 4 contienes las conclusiones y la sección 5 contiene la bibliografía.

2. Contenido

Conjunto de datos utilizado

Los datos contienen variables relacionadas con factores demográficos (edad al momento de la matriculación, sexo, estado civil, nacionalidad, código postal, necesidades especiales), factores socioeconómicos (estudiante-trabajador, habilidades de los padres, profesión de los padres, situación laboral de los padres, beca estudiantil, deuda estudiantil) y la trayectoria académica del estudiante (calificación de admisión, años de repetición en la escuela secundaria, orden de elección del curso matriculado, tipo de curso en la escuela secundaria).

Atributos del dataset que usaremos para la predicción del abandono o éxito académico de estudiantes universitarios:

Datos Demográficos, Académicos y Socioeconómicos al momento de la matrícula:

1. **Estado civil**: Estado civil del estudiante (soltero, casado, viudo, etc.).
2. **Modo de aplicación**: Vía de ingreso a la universidad (contingente general, internacional, transferencia, etc.).
3. **Orden de postulación**: Prioridad del curso en la solicitud (0 = primera opción, hasta 9 = última opción).
4. **Carrera**: Programa académico al que se postuló (Agronomía, Diseño, Enfermería, etc.).
5. **Régimen horario**: Si el estudiante asiste en el turno diurno (1) o nocturno (0).
6. **Estudios previos**: Nivel educativo antes de ingresar (educación básica, secundaria, superior, etc.).
7. **Nota de estudios previos**: Calificación obtenida en la formación previa (0–200).
8. **Nacionalidad**: País de origen del estudiante.
9. **Formación de la madre**: Nivel educativo de la madre.
10. **Formación del padre**: Nivel educativo del padre.
11. **Ocupación de la madre**: Ocupación profesional de la madre.
12. **Ocupación del padre**: Ocupación profesional del padre.
13. **Nota de admisión**: Calificación obtenida para el ingreso a la universidad (0–200).
14. **Desplazado**: Si el estudiante reside fuera de su zona de origen (1 = sí, 0 = no).

15. **Necesidades educativas especiales**: Si el estudiante presenta alguna necesidad especial (1 = sí, 0 = no).
16. **Deudor**: Si tiene deudas pendientes con la universidad (1 = sí, 0 = no).
17. **Género**: Sexo del estudiante (1 = masculino, 0 = femenino).
18. **Becado**: Si es beneficiario de una beca (1 = sí, 0 = no).
19. **Edad al matricularse**: Edad del estudiante al momento de la matrícula.
20. **Estudiante internacional**: Si se trata de un estudiante extranjero (1 = sí, 0 = no).

Indicadores económicos nacionales:

21. **Tasa de desempleo**: Porcentaje de desempleo nacional en ese período.
22. **Tasa de inflación**: Porcentaje de inflación.
23. **PIB**: Producto Interno Bruto del país.

Variable objetivo (Target):

24. **Resultado académico**: Estado final del estudiante al terminar el curso (clasificación en tres categorías: **abandono**, **inscrito**, **graduado**).

Los atributos del 1 al 23 son de tipo Integer. Sin embargo, muchos de ellos representan variables categóricas, ya sean nominales u ordinales, codificadas numéricamente en el dataset. Por ejemplo, el atributo estado civil se codifica con valores como: 1 = soltero, 2 = casado, 3 = viudo, 4 = divorciado, 5 = unión de hecho y 6 = separados legalmente. Otro ejemplo es el atributo género, donde 0 representa femenino y 1 masculino.

La única variable nominal no codificada numéricamente es el atributo 24, que corresponde a la **variable objetivo**. Esta variable presenta tres categorías: *abandono, inscrito y graduado,* que serán utilizadas en este estudio para evaluar la precisión de los diferentes algoritmos de clasificación ofrecidos por la herramienta Weka. El enfoque que hemos tomado según estas tres variables es el siguiente: se excluirán los registros con target “enrolled” del entrenamiento para construir un modelo binario: dropout vs graduated. Luego aplicaremos dicho modelo sobre los datos “enrolled” para obtener la predicción: ¿será más probable que el estudiante se gradúe o abandone? E identificaremos patrones como los perfiles de dichos estudiantes, qué características tienen similares, etc. Se aplicará el modelo entrenado tanto sin atributos del desarrollo académico durante el curso (enfoque de predicción anticipada) como con dichos

atributos incluidos (enfoque de predicción basada en el rendimiento parcial). Esto nos permitirá comparar los resultados y entender mejor los perfiles de los estudiantes que se encuentran actualmente en estado "enrolled".

En resumen, el dataset proporciona un perfil integral del estudiante universitario, facilitando una predicción temprana del éxito o abandono académico.

3. Preprocesamiento de los datos y análisis

3.1. Procesamiento de los datos

Realizamos un riguroso preprocesamiento de datos para gestionar anomalías, valores atípicos inexplicables, valores faltantes y descartaremos los registros que no tienen concordancia con el propósito del estudio. Detectar antes de comenzar el curso asignaturas matriculadas en el primer o segundo cuatrimestre, asignaturas convalidadas, evaluadas o sin evaluar, promedio de notas del primer o segundo cuatrimestre, evaluaciones realizadas, notas de cursos específicos y pago de matrícula al día, generaría un sesgo ya que estaríamos usando información del "futuro" (después del inicio del curso) para predecir una situación previa. Es decir, el modelo tendría una ventaja que no tendría en una situación real de uso.

No obstante, como se detallará más adelante en los resultados, también se realizó un análisis exploratorio para evaluar cómo variaba la precisión del modelo al incluir o excluir dichos atributos.

El archivo original proporcionado por el repositorio UCI Machine Learning Repository estaba en formato .csv. Si bien Weka afirma soportar este formato, nos encontramos con dificultades al configurar correctamente los parámetros de importación (caracteres delimitadores de cadenas, separador de campos, representación de valores faltantes, definición de atributos nominales y numéricos, presencia de cabecera, etc.). (Figura 1)

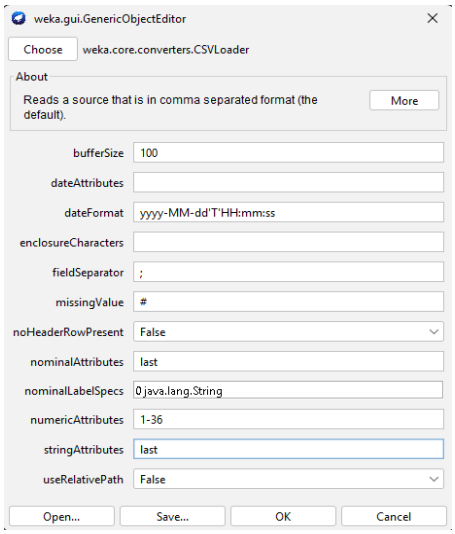

Ilustración 1. Filtro Lectura CSV.

Uno de los errores más relevantes fue el de *"Unique Columns"*, provocado por la mala interpretación de nombres de columnas con apóstrofes, como *'Mother's qualification'* y *'Mother's occupation'*. Para solucionarlo, desarrollamos un pequeño script en Python con las siguientes funcionalidades:

1. Carga del archivo .csv original, utilizando el separador correcto (;), dado que el archivo del repositorio UCI no sigue el formato típico de WEKA (.arff) separado por comas.
2. Eliminación de columnas que contienen información del futuro académico del estudiante, como asignaturas matriculadas, notas y evaluaciones, o el estado del pago de matrícula, para evitar sesgos en el modelo.

```
# Eliminación de columnas que contienen información del futuro del estudiante
columns_to_drop = [
      "Tuition fees up to date",
      "Curricular units 1st sem (enrolled)",
      "Curricular units 1st sem (grade)",
      ...
]
df_filtered = df_complete.drop(columns=columns_to_drop, errors='ignore')
```

3. Adaptar los nombres de atributos, reemplazando espacios y caracteres problemáticos (como comillas simples ') por guiones bajos _, para asegurar una correcta interpretación en Weka.
4. Inferencia automática del tipo de atributo (NUMERIC o STRING), según los datos reales de cada columna.

```
def infer_arff_type(series):
    if pd.api.types.is_numeric_dtype(series):
        return 'NUMERIC'
    else:
        return 'STRING'
```

5. Conversión de los datos al formato. arff, respetando la sintaxis de Weka:
    a. Definición de la relación (@RELATION).
    b. Declaración de cada atributo (@ATTRIBUTE).
    c. Inclusión de los datos (@DATA), con tratamiento adecuado de valores faltantes (?) y texto entre comillas simple.
6. Cambio de la variable dependiente de tipo String a tipo nominal para poder realizar el entrenamiento y posterior análisis.

```
@attribute TARGET {Dropout,Graduate}
```

Se generaron dos archivos. arff: uno con todos los atributos originales (archivo_completo.arff) y otro con los atributos filtrados para entrenar el modelo de predicción sin sesgo (archivo_filtrado.arff).

Gracias a este script fue posible importar correctamente el conjunto de datos en Weka y comenzar a trabajar con él. No obstante, aún quedaba por separar aquellos registros con Target = 'Enrolled' para utilizarlos posteriormente como conjunto de predicción,

dado que nuestro objetivo era estudiar si un estudiante se graduaría (Target = 'Graduate') o abandonaría sus estudios (Target = 'Dropout').

Para ello, Weka ofrecía una opción muy cómoda que permite generar nuevos conjuntos de datos en formato. arff, excluyendo o seleccionando únicamente los registros con una determinada clase en el atributo objetivo. Así, pudimos separar adecuadamente ambos subconjuntos (entrenamiento y predicción) de manera rápida y sencilla. (Figura 2)

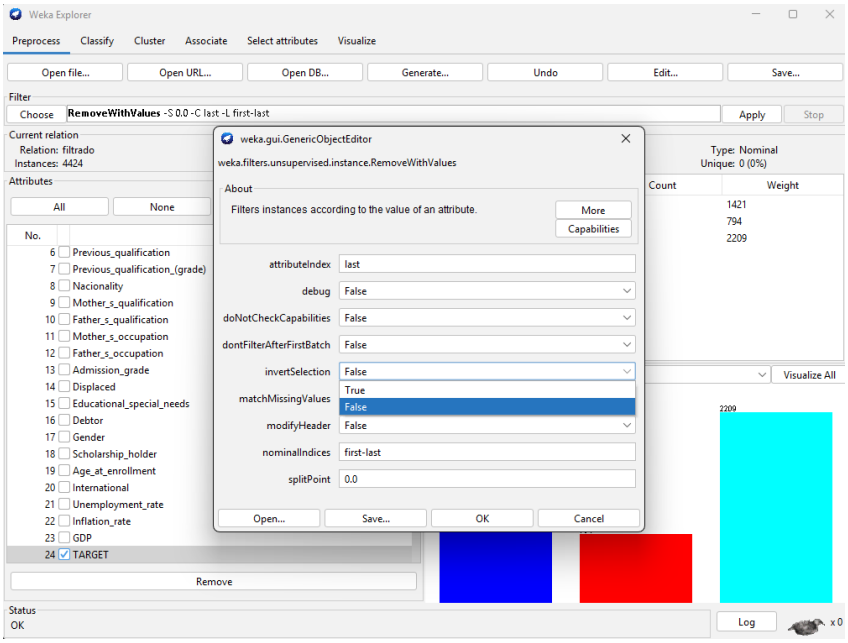

Ilustración 2. Filtro RemoveWithValues.

Específicamente, empleamos el filtro RemoveWithValues, disponible en la pestaña de Preprocess. En este filtro, seleccionamos el atributo Target y configuramos la condición en invertSelection para seleccionar (o eliminar) los registros con valor "Enrolled".

Cabe destacar que Weka ofrece una amplia variedad de filtros para modificar tanto atributos como clases, incluyendo funcionalidades como: convertir tipos de atributos (por ejemplo, de numérico a nominal), duplicar registros, aplicar transformaciones matemáticas a atributos numéricos, eliminar atributos que contengan determinadas palabras clave, estandarizar valores, entre otros. Sin embargo, en nuestro caso, al haber desarrollado previamente el script en Python para el preprocesamiento, decidimos implementar directamente algunas de estas operaciones desde allí.

El conjunto de datos final consta de 3630 registros, 23 variables independientes y una dependiente (target).

3.2. Métricas usadas

Durante el análisis de los algoritmos hemos usado dos métricas:

- **Cross-Validation**:

La validación cruzada es una técnica de evaluación de modelos en aprendizaje automático que se utiliza para estimar el rendimiento de un modelo de forma más robusta. En lugar de entrenar y probar el modelo en un único conjunto de datos, la validación cruzada divide el conjunto de datos en varias partes. En cada iteración, se usa una de esas particiones como conjunto de prueba y el resto como conjunto de entrenamiento. Este proceso se repite para cada parte, y finalmente se calcula el promedio de las métricas obtenidas en cada iteración. El resultado es una estimación más confiable del rendimiento del modelo, ya que tiene en cuenta diferentes subconjuntos de los datos.

- **Percentage Split**:

La métrica de percentage split o división porcentual es una técnica sencilla para evaluar un modelo de aprendizaje automático. Consiste en dividir el conjunto de datos en dos subconjuntos: uno para entrenamiento y otro para prueba. Por lo general, el conjunto de entrenamiento se utiliza entre el 70% y el 80% de los datos, mientras que el conjunto de prueba ocupa el resto, es decir, entre el 20% y el 30%. En nuestro caso hemos usado el 65% de entrenamiento y el 35% de prueba.

Primero se hace la división, es decir, se separan aleatoriamente los datos en dos partes, una para entrenar el modelo y otra para probarlo. Después, el modelo se entrena con el conjunto de entrenamiento. Por último, se procede a evaluar el modelo utilizando el conjunto de prueba para medir su rendimiento (precisión, error, etc.).

Aunque es rápida y fácil de implementar, la principal limitación del percentage split es que depende de cómo se dividen los datos. Si se elige una división no representativa, el modelo puede dar una evaluación sesgada. En nuestro caso, hemos visto como varían los resultados aplicando el mismo algoritmo, principalmente por que el conjunto que se queda de entrenamiento es distinto y esto da lugar a variaciones significativas con respecto a los aciertos.

En resumen, la validación cruzada es más precisa y confiable, pero más costosa en términos de tiempo y recursos, mientras que el percentage split es rápido y sencillo, pero puede ser menos confiable debido a la aleatoriedad en la división de los datos

3.3. Clasificadores usados

Comenzamos usando un clasificador de árbol, concretamente J48 para ver cómo se comportaba el dataset. Para ello WEKA contiene múltiples clasificadores tanto de árboles, regresión, bayes, etc. Además, puedes ajustar los parámetros de cada clasificador cómo desees. De la misma forma ofrece distintas formas de probar el clasificador como por ejemplo mediante validación cruzada especificando el número de conjuntos o por división porcentual entre conjunto de entrenamiento y prueba.

Al ejecutar este clasificador J48 pudimos observar una anomalía, el árbol era demasiado grande para un conjunto de datos mediano como el nuestro. Mostraba un total de 603 nodos, esto nos dio una pista sobre qué podía estar pasando. La primera opción fue que los atributos no tuviesen relación directa entre sí, es decir, fueran demasiados independientes por lo que no era fácil separar las clases por un conjunto claro de reglas. La segunda opción fue que hubiera inconsistencia en los datos o variables que realmente no estuvieran aportando información a nuestro caso, información que confundían más que ayudaban, es decir, ruido en el dataset. O también podía ser por un problema de sobreajuste (complejidad más ruido).

Ante esto decidimos usar la herramienta "*Select Attributes*" que proporciona Weka con la cual puedes evaluar la importancia de cada atributo respecto a la variable objetivo. Además, también permite seleccionar un subconjunto óptimo de atributos o ver qué atributos tienen bajo aporte o nulo. También consta de distintos métodos de búsqueda como *Ranker* el cual muestra todos los atributos ordenados por importancia, *BestFirst* o *GreedyStepwise* los cuales proporcionan una combinación óptima de atributos. Una vez usamos la herramienta pudimos observar que había 5 atributos que no estaban aportando valor a nuestra tarea de clasificación. Esos atributos eran "*Unemployement_rate*", "*Nacionality*", "*Inflation_rate*", "*Educational_special_needs*" e "International". ¿Por qué estos atributos no aportaban valor? Después de un estudio sacamos las siguientes conclusiones:

Los atributos "*Nacionality*", "*International*", "*Educational_special_needs*" no aportan valor pues sus valores varían demasiado poco para la cantidad de registros que hay. Es decir, si "Nacionality" puede tomar un valor entre 0 y 109 cada uno representando a un país, hay un 97.5% de casos con valor a 1 por lo que el 2.5% restante no son registros suficientes para determinar algún tipo de regla para clasificar la clase. Lo mismo ocurre con "International" y "Educational_special_needs". Esto pudo observarse gracias a los histogramas que aporta Weka (Figura 3).

Ilustración 3. Gráficas de distribución de Weka.

Sin embargo "*Unemployment_rate*" y "*Inflation_rate*" no se podía ver de forma tan clara en el histograma por qué no estaba aportando valor. Esto nos llevó a optar por la segunda opción, analizar la relación de estos atributos con los demás atributos. Para ello Weka proporciona un apartado llamado "Visualize" donde aparece una matriz (Figura 4) con todos los atributos y permite acceder a la relación entre cada uno de ellos.

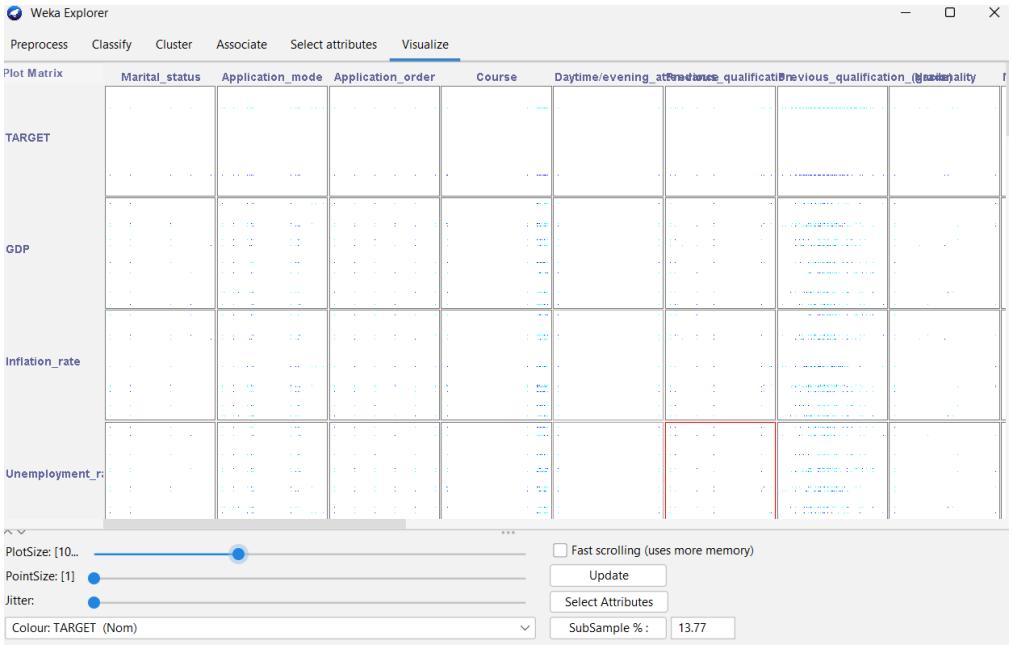

Ilustración 4. Plot Matrix Weka.

De esta forma podemos ir uno a uno viendo una gráfica de dispersión donde cada cruz representa una combinación de dos atributos de una instancia. Gracias a esto hemos podido concluir que entre los atributos "*Unemployment_rate*" y "*Admission_grade*" no tienen una fuerte relación entre sí. Mientras que "*Admission_grade*" parece ser muy útil

para predecir el abandono, "*Unemployment_rate*" parece no variar según la nota por lo que se puede concluir que no es clave para entender la deserción en el dataset (Figura 5). Además, se puede ver como el desempleo no cambia mucho según la nota de admisión, parece independiente, pues en caso contrario aparecerían líneas curvas o inclinadas en lugar de horizontales.

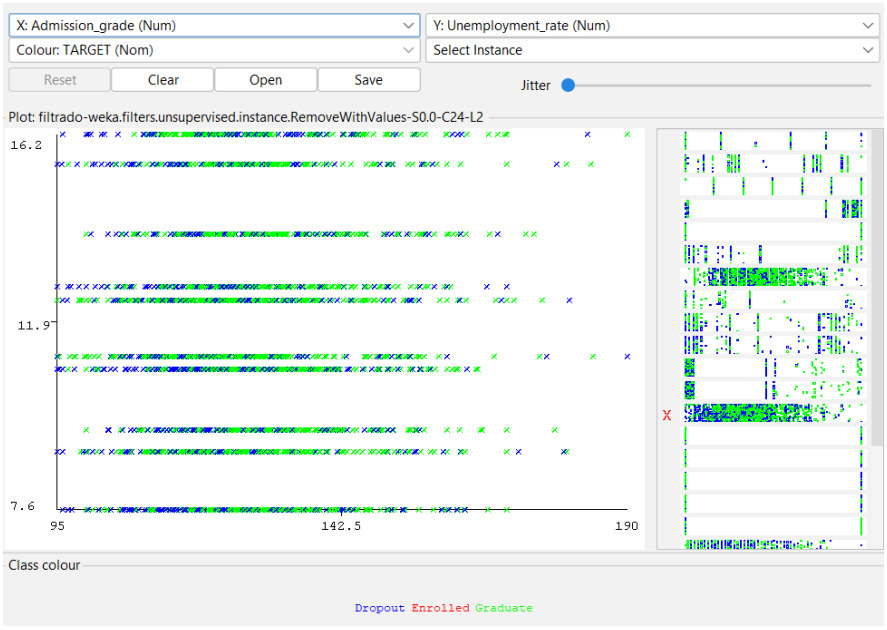

Ilustración 5. Gráfica de dispersión "Admission_grade" y "Unemployment_rate"

De la misma forma ocurría entre los atributos "*Admission_grade*" y "*Inflation_rate*" (Figura 6).

Ilustración 6. Gráfica de dispersión "Admission_grade" y "Inflation_rate"

3.4. Mejor clasificador encontrado (contando solo matrícula). Archivo filtrado

Hemos descartado todos los atributos con valores posteriores a la matriculación, asegurándonos de predecir solo con información disponible desde el inicio. Lo hicimos mediante una herramienta muy parecida a la mencionada anteriormente para descartar atributos en Weka. En este caso hemos usado el filtro "*Remove*" que permite seleccionar un rango de atributos y eliminarlos directamente del conjunto del conjunto de datos.

Tras aplicar estas modificaciones a los datos, procedimos a evaluar los diferentes algoritmos que nos ofrecía weka. Después de probarlos exhaustivamente, el algoritmo más estable y que sistemáticamente ofrecía mejores resultados fue LogitBoost con REPTree. Antes de detallar por qué, conviene analizar mejor nuestro conjunto de datos para entender por qué esta combinación fue tan adecuada.

El dataset preprocesado presenta relaciones complejas entre variables, había muchas variables categóricas y numéricas, lo que genera interacciones no lineales difíciles de modelar con algoritmos simples. Además, su tamaño medio (~3600 instancias) implica un riesgo moderado de sobreajuste, especialmente con modelos muy complejos. Por último, aunque la clase "Enrolled" fue descartada, las clases "Dropout" y "Graduate" tienen una proporción desigual (aproximadamente 1:1.7), añadiendo otro nivel de dificultad al proceso de clasificación.

Entonces, **¿por qué LogitBoost con REPTree funciona bien aquí?**

LogitBoost (6) es muy efectivo para corregir errores incrementales, lo cual es útil si las clases tienen cierta complejidad, algo que pasaba en nuestro conjunto (patrones de deserción que dependen de combinaciones de variables). Además, este enfoque de corrección incremental de errores permite ajustar finamente las predicciones, algo esencial cuando hay desbalance de clases y los errores (como confundir "Dropout" con "Graduate") necesitan ser tratados con precisión.

Por otro lado, los árboles de decisión pueden volverse excesivamente específicos si no se controlan adecuadamente, lo que lleva al sobreajuste. Para evitar esto se "poda" dicho árbol que lo obliga a eliminar ramas con escasa contribución, conservando solo las reglas más generales y relevantes. Esto reduce el riesgo de sobreajuste y mejora la capacidad de generalización del modelo. Por esta razón se usó una poda agresiva (-N 15) que obliga al árbol a eliminar ramas poco significativas, dejando solo las reglas más relevantes.

Durante las pruebas, se identificó que 8 iteraciones de boosting ofrecían el mejor rendimiento, maximizando la precisión sin comprometer la generalización. A partir de esa cantidad, más iteraciones (10 o 50) no solo no mejoraban el resultado, sino que lo deterioraban levemente, indicando un inicio de sobreajuste. Pero, ¿por qué?

Esto se explica porque en boosting, cada nueva iteración entrena un nuevo árbol sobre los errores del anterior. Si se usan demasiadas iteraciones, el modelo puede sobreajustarse a errores muy específicos provocando ruido. Con 8 iteraciones se alcanzó un punto donde el modelo ha aprendido los patrones clave, pero sin capturar ruido. A partir de ahí, más iteraciones no mejoran, empeoran el rendimiento.

En cuanto a la configuración del árbol:

- Hojas mínimas de tamaño 2, de esta forma permitía árboles más detallados que capturan sutilezas del dataset. Pero lo suficientemente pequeño para evitar que los árboles sean triviales.
- Poda agresiva (parámetro -N 15) evitaba que los árboles se sobreajustaran a los datos de entrenamiento. La poda agresiva hace que cada árbol se enfoque solo en patrones claros y relevantes, algo crucial cuando hay muchas variables, como en nuestro caso, pero no todas son igual de útiles.
- No restringimos el límite de profundidad del árbol. El árbol podía crecer según las necesidades de los datos, pero la poda controlaba su complejidad, permitiendo flexibilidad sin rigidez.
- Error mínimo como criterio de poda: se aseguraba que solo se mantuvieran ramas que aportaran a la precisión.

El buen desempeño del modelo se explica por la naturaleza de nuestro conjunto de datos: con múltiples variables de diversa relevancia y una distribución desbalanceada de clases. Para este tipo de conjuntos de datos es crucial contar con un modelo que sea capaz de capturar interacciones complejas sin sobreajustarse. La poda agresiva de los árboles REPTree evitó que los árboles memoricen los datos, obligándolos a retener solo las reglas generales. Esto, combinado con un número óptimo de iteraciones (8), permite que el modelo aprenda de los errores de forma incremental, refinando la predicción sin caer en el sobreajuste. Así, se logra un balance ideal entre precisión y capacidad de generalización, reflejado en una precisión del 87.1% y un AUC de 0.92.

Así, se construyó un modelo eficiente y eficaz para predecir la deserción estudiantil desde la matrícula.

Este resultado (Figura 7) nos da tras aplicar la **validación cruzada** con 10 6 particiones

```
=== Summary ===

Correctly Classified Instances        3162               87.1074 %
Incorrectly Classified Instances       468               12.8926 %
Kappa statistic                          0.7323
Mean absolute error                      0.0972
Root mean squared error                  0.2674
Relative absolute error                 30.3351 %
Root relative squared error             66.8165 %
Total Number of Instances             3630
```

Ilustración 7. Resultados Validación Cruzada.

```
=== Summary ===

Correctly Classified Instances        1063               83.7008 %
Incorrectly Classified Instances       207               16.2992 %
Kappa statistic                          0.6643
Mean absolute error                      0.1279
Root mean squared error                  0.2994
Relative absolute error                 39.736  %
Root relative squared error             74.1409 %
Total Number of Instances             1270
```

Y este (Figura 8) tras dividir el fichero entre un **65%** de datos que usaremos de **entrenamiento** y un **35%** para **probarlo**

Ilustración 8. Resultados Split Percentage.

Tras estos resultados procedemos a un análisis más detallado:

- **Porcentaje de Clasificación Correcta**:
  El modelo alcanzó una precisión del 87.1%, lo que representa un incremento notable respecto a configuraciones anteriores que rondaban el 85%. Esto implica que el modelo clasifica correctamente a casi 9 de cada 10 estudiantes en términos de su riesgo de deserción desde el momento de la matrícula.

  La estabilidad del modelo fue consistente tanto en validación cruzada como en pruebas por porcentaje (percentage split), con fluctuaciones mínimas, lo que demuestra una buena capacidad de generalización.

- **Kappa Statistic**:
  - ✓ El valor de Kappa fue 0.732, lo que indica un alto grado de acuerdo entre las predicciones del modelo y las clases reales, más allá de lo esperado por azar. Ya que este modelo varía entre 0 a 1, representando el 1 una predicción perfecta y 0 como si eligiera al azar.
- **Errores Promedio**:
  - ✓ El Mean Absolute Error (MAE) y el Root Mean Squared Error (RMSE) son muy similares en ambos casos (MAE alrededor de 0.1 y RMSE alrededor de 0.27), indicando que no hay predicciones con errores extremos.
- **Errores Relativos**:
  - ✓ El Relative Absolute Error varía entre 30% y 39%, mientras que el Root Relative Squared Error está entre 66% y 74%.
  - ✓ Estos valores confirman que el modelo se ajusta bien a la variabilidad interna del conjunto de datos, siendo más eficiente comparado, por ejemplo, un clasificador simple basado en la media.
- **Número Total de Instancias**:

- ✓ En cross-validation, se utilizaron 3630 instancias, evaluadas en 10 particiones estratificadas, asegurando que todas las clases estuvieran representadas proporcionalmente en cada fold.
- ✓ En percentage split, se evaluaron 1270 instancias (aproximadamente el 30% del conjunto total), separadas previamente para simular un escenario de datos nuevos.

Ambos métodos (validación cruzada y percentage split) mostraron resultados consistentes, reforzando la confianza en la capacidad del modelo para generalizar. La alta precisión y bajo error reflejan que la combinación de boosting y poda agresiva permitió capturar las relaciones importantes del dataset sin caer en sobreajuste.

AUC: 0.92, otro indicador clave, muestra una excelente capacidad de discriminación del modelo entre las clases.

Con respecto a las **matrices de confusión** tenemos:

1. Para **Cross-Validation** (Figura 9)

```
=== Confusion Matrix ===

    a    b    c   <-- classified as
 1232    0  224 |    a = Dropout
    0    0    0 |    b = Enrolled
  244    0 1930 |    c = Graduate
```

Ilustración 9. Matriz de confusión para Cross-Validation.

Es decir, los valores representan lo siguiente:

Dropout:

- Correctamente clasificados: 1232
- Incorrectamente clasificados como Graduate: 224
- Precisión para Dropout: 83.5%
- Recall para Dropout: 84.6%

Graduate:

- Correctamente clasificados: 1930
- Incorrectamente clasificados como Dropout: 244
- Precisión para Graduate: 89.6%
- Recall para Graduate: 88.8%

Enrolled: Esta clase no se utiliza (no relevante en el análisis final), por lo tanto, se ignora.

```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,846    0,112    0,835      0,846    0,840      0,732    0,920     0,877     Dropout
                 ?        0,000    ?          ?        ?          ?        ?         ?         Enrolled
                 0,888    0,154    0,896      0,888    0,892      0,732    0,920     0,933     Graduate
Weighted Avg.    0,871    0,137    0,871      0,871    0,871      0,732    0,920     0,911
```

Por lo que podemos decir que el modelo tiende a confundir una parte de los Dropout con Graduate y viceversa, pero mantiene un buen equilibrio. La simetría en los errores (224 vs. 244) muestra que el modelo no está sesgado excesivamente hacia una clase. No hay errores entre Dropout y Enrolled ni entre Graduate y Enrolled, reforzando que el modelo no comete errores absurdos.

Para **Percentage Split** los valores representan lo siguiente (Figura 10):

```
=== Confusion Matrix ===

   a   b   c   <-- classified as
 423   0 111 |   a = Dropout
   0   0   0 |   b = Enrolled
  96   0 640 |   c = Graduate
```

Ilustración 10. Matriz de confusión para Percentage Split.

Dropout:

- Correctamente clasificados: 423
- Incorrectamente clasificados como Graduate: 96
- Precisión para Dropout: 81.5%
- Recall para Dropout: 79.2%

Graduate:

- Correctamente clasificados: 640
- Incorrectamente clasificados como Dropout: 111
- Precisión para Graduate: 85.2%
- Recall para Graduate: 87.0%

Enrolled: Esta clase no se utiliza (no relevante en el análisis final), por lo tanto, se ignora.

La proporción de errores se mantiene muy similar. El modelo sigue reconociendo bien los Dropouts. La confusión entre Dropout y Graduate es la principal fuente de error, pero se mantiene controlada. El comportamiento es muy consistente: el patrón de errores y aciertos es casi idéntico entre métodos. La simetría en los errores refuerza la idea de que el modelo generaliza bien fuera de la muestra de entrenamiento.

En general, los resultados de ambas matrices de confusión muestran que el modelo es altamente eficaz para identificar correctamente a los estudiantes que se gradúan,

logrando una clasificación precisa. Además, el modelo ha demostrado una gran capacidad para predecir la deserción estudiantil, acertando en la mayoría de los casos, aunque aún hay cierto margen de mejora en esta área.

Es importante destacar que, aunque se producen algunas confusiones entre estudiantes que abandonan y los que se gradúan, esta situación es comprensible pues la complejidad de predecir el abandono basándose únicamente en información disponible al momento de la matrícula es mayor. A pesar de esto, el modelo mantiene un excelente equilibrio, logrando un rendimiento general muy elevado y una gran capacidad de generalización. El hecho de que el modelo haya alcanzado una precisión superior al 87% y un AUC de 0.92, indica que es una herramienta fiable y efectiva para apoyar la toma de decisiones tempranas en el ámbito educativo. Aunque si es cierto que seguir mejorando la sensibilidad hacia los casos de abandono sería beneficioso, el desempeño actual ya ofrece una base muy sólida sobre la cual se pueden construir intervenciones preventivas.

En resumen, el modelo no solo es el mejor evaluado hasta ahora, sino que también ofrece un excelente punto de partida para detectar patrones clave en la trayectoria estudiantil desde el inicio, con un rendimiento destacable en ambos métodos de validación.

3.5. Mejor clasificador encontrado (contando el curso completo). Archivo completo.

Con todos los atributos que nos aportan el fichero, que representan datos desde la matriculación hasta finalizar el curso. Hemos procedido a hacer el análisis con el mismo algoritmo y mismas métricas.

El algoritmo usado para este caso es LogitBoost + REPTree, igual que el anterior con el fin de poder comparar como varía la precisión de uno con respecto a otro:

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        3242               89.3113 %
Incorrectly Classified Instances       388               10.6887 %
Kappa statistic                          0.7731
Mean absolute error                      0.0826
Root mean squared error                  0.2447
Relative absolute error                 25.9828 %
Root relative squared error             61.4063 %
Total Number of Instances             3630

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,832    0,067    0,888      0,832    0,859      0,774    0,937     0,932     Dropout
                 ?        0,000    ?          ?        ?          ?        ?         ?         Enrolled
                 0,933    0,168    0,896      0,933    0,914      0,774    0,937     0,941     Graduate
Weighted Avg.    0,893    0,129    0,893      0,893    0,892      0,774    0,937     0,937

=== Confusion Matrix ===

    a    b    c   <-- classified as
 1182    0  239 |    a = Dropout
    0    0    0 |    b = Enrolled
  149    0 2060 |    c = Graduate
```

Ilustración 11. Métricas de evaluación con Cross Validation para el conjunto completo con REPTree.

Este resultado nos da tras aplicar la validación cruzada con 8 particiones

Y este tras usar LogitBoost + DecisionStump con 100 iteraciones con validación cruzada:

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        3288               90.5785 %
Incorrectly Classified Instances       342                9.4215 %
Kappa statistic                          0.7994
Mean absolute error                      0.0888
Root mean squared error                  0.2181
Relative absolute error                 27.9341 %
Root relative squared error             54.7256 %
Total Number of Instances             3630

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,840    0,052    0,913      0,840    0,875      0,801    0,952     0,949     Dropout
                 ?        0,000    ?          ?        ?          ?        ?         ?         Enrolled
                 0,948    0,160    0,902      0,948    0,925      0,801    0,952     0,957     Graduate
Weighted Avg.    0,906    0,118    0,906      0,906    0,905      0,801    0,952     0,954

=== Confusion Matrix ===

    a    b    c   <-- classified as
 1193    0  228 |    a = Dropout
    0    0    0 |    b = Enrolled
  114    0 2095 |    c = Graduate
```

Ilustración 12. Métricas de evaluación con Cross Validation para el conjunto completo con DecisionStump.

Hagamos la tabla comparativa:

Métrica	LogitBoost + REPTree (8 iter.)	LogitBoost + DecisionStump (100 iter.)
Precisión Global	89.31%	90.58%
Kappa Statistic	0.7731	0.7994
MAE (Error Absoluto Medio)	0.0826	0.0888
RMSE (Raíz Error Cuadrático)	0.2447	0.2181
AUC (Área bajo la curva ROC)	0.937	0.952
TP Rate Dropout	0.832	0.840
TP Rate Graduate	0.933	0.948
PRC Area (Precisión-Recall)	0.937	0.954
Tiempo de entrenamiento	1.15s	1.95s

Tabla 1. Comparativa REPTree con DecisionStump

¿Por qué ocurre esto?

Hay que saber que DecisionStump es un modelo extremadamente simple, es un árbol con una sola decisión (un solo nodo). Por sí solo, es casi inútil para clasificaciones complejas, pero tiene una ventaja clave, es muy difícil que sobreajuste, mientras que LogitBoost funciona sumando muchos modelos débiles para construir un modelo fuerte. Entonces, al añadir notas semestrales, el dataset ahora contiene mucha más información relevante, lo que facilita que incluso modelos simples encuentren patrones útiles. DecisionStump puede, en cada iteración, centrarse en una sola nota o atributo y mejorar el modelo de forma incremental, sin complicarse. Esto hace que el conjunto de 100 stumps trabaje muy eficientemente: cada uno aporta un poco, y juntos logran una gran precisión. Con más atributos útiles, se necesita menos "complejidad interna" en cada árbol y más "cooperación" entre muchos modelos simples.

Para entenderlo mejor, es como imaginar que tienes que resolver un problema complicado. Puedes tener:

1. 8 expertos (REPTree), cada uno muy inteligente, pero si se equivocan un poco, pueden insistir demasiado en su opinión.

2. 100 personas con una idea muy simple (DecisionStump), que poco a poco, corrigen los errores y se van acercando juntas a la mejor solución.

En este caso, con más información en el dataset, la estrategia de los 100 pasos pequeños (2) funcionó mejor que la de los 8 pasos grandes (1).

3.6. Comparación entre el archivo filtrado y completo

A lo largo del proyecto, se han construido dos modelos con enfoques claramente diferenciados:

1. **Modelo Preventivo** (solo datos hasta la matrícula): diseñado para anticipar el riesgo de deserción desde el primer momento, sin necesidad de esperar a resultados académicos posteriores.
2. **Modelo Predictivo Completo** (con notas semestrales): enfocado en afinar las predicciones una vez se dispone de información más detallada del desempeño académico.

Dataset	Precisión (%)	Kappa	AUC
Sin notas semestrales	87.1%	0.7323	0.920
Con notas semestrales	90.6%	0.7994	0.952

Tabla 2. Comparativa dataset completo y filtrado

Ambos modelos han demostrado un alto rendimiento, especialmente si consideramos que el modelo basado solo en datos hasta la matrícula, sin información académica posterior, alcanza un 87.1% de precisión. Este nivel de exactitud es muy alto para un sistema predictivo temprano, lo que lo convierte en una herramienta valiosa para la intervención temprana.

El modelo basado únicamente en datos previos a la matrícula tiene una importancia estratégica clave. Permite a las instituciones anticiparse a situaciones de riesgo incluso antes de que los estudiantes comiencen sus estudios, posibilitando la implementación de políticas de prevención proactivas. Si bien es cierto que el modelo completo ofrece una precisión superior, esto es esperable dado que cuenta con información más rica. Sin embargo, el valor del modelo preventivo radica en su capacidad de actuar con antelación, lo cual es fundamental en entornos educativos donde la prevención es esencial.

Lejos de ser modelos en competencia, ambos se complementan. El modelo sin notas es ideal para una primera evaluación del riesgo, mientras que el modelo con notas puede servir como seguimiento continuo, ajustando las predicciones conforme se obtiene más información. Esta aproximación escalonada es, de hecho, la más eficiente desde el punto de vista de la gestión educativa: primero prevenir, luego refinar.

En definitiva, aunque la adición de las notas semestrales mejora el rendimiento predictivo (de 87.1% a 90.6%), el modelo basado solo en datos previos sigue siendo altamente eficaz para un uso temprano. La capacidad de identificar patrones de riesgo desde la matrícula con alta precisión confirma que incluso sin datos posteriores, se pueden detectar indicadores clave de deserción. Esto respalda el uso de este modelo en contextos donde la intervención temprana es prioritaria y muestra que un análisis inteligente de la información disponible desde el inicio es suficiente para tomar decisiones informadas.

4. Conclusiones

El objetivo de este proyecto ha sido predecir la deserción estudiantil únicamente con información disponible en el momento de la matrícula, sin depender de las notas obtenidas en los semestres. Esta restricción es clave, ya que permite intervenir tempranamente y tomar decisiones estratégicas antes del inicio del curso académico.

Los resultados obtenidos con el modelo LogitBoost + REPTree han sido muy satisfactorios, es una gran herramienta con una excelente capacidad de discriminación entre estudiantes que abandonan y los que se gradúan. A pesar de no contar con notas, el modelo ha podido capturar patrones significativos basados en variables socioeconómicas, académicas previas y características personales.

4.1. Perfil de los Estudiantes:

Estudiantes que tienden a **graduarse** suelen presentar:

- Buenas notas de acceso (Admission_grade).
- No tener deudas (Debtor = No).
- Padres con mayores niveles de cualificación.
- Edad al ingreso en rangos moderados.
- Becarios (Scholarship_holder = Sí).
- No ser desplazados ni presentar necesidades educativas especiales.

Estudiantes con riesgo de **abandono** suelen:

- Tener deudas activas.
- Provenir de entornos socioeconómicos más desfavorecidos (ocupaciones y cualificaciones de los padres bajas).
- Tener un historial académico previo más débil.
- Ser mayores al ingresar (edad más alta al momento de la matrícula).
- No recibir becas.

El modelo ha identificado que variables como *Admission_grade, Debtor, Previous_qualification_(grade), Course, Scholarship_holder* y *Age_at_enrollment* son particularmente influyentes apareciendo primeras en el árbol de decisión. Estas

variables, accesibles antes de comenzar el curso, permiten construir perfiles de riesgo desde el inicio.

Gracias a estos resultados, se podrían diseñar estrategias de apoyo personalizadas para estudiantes en riesgo de abandono como programas de tutoría enfocados en estudiantes con baja nota de ingreso, ayudas financieras o asesoramiento a estudiantes con deudas, orientación vocacional y académica para mayores al ingresar o apoyo psicosocial a estudiantes desplazados o con necesidades educativas.

4.2. Efectividad del modelo

Si bien el modelo que incluyen notas académicas de los estudiantes proporciona una mayor precisión, el modelo sin estas notas ofrece una ventaja práctica. Esto se debe a que, al basarse solo en los datos de matrícula, permite intervenir mucho antes de que los estudiantes enfrenten problemas académicos. En situaciones reales, esto es crucial, ya que las intervenciones tempranas pueden reducir significativamente las tasas de abandono escolar y brindar apoyo antes de que los estudiantes alcancen un nivel académico bajo. Por lo tanto, aunque la precisión es algo menor, la capacidad de prever y actuar de forma anticipada compensa esta diferencia.

Para ver la efectividad del modelo vamos a probarlo con el conjunto de datos con variable objetivo “Enrolled” es decir, que aún están cursando. Para ello hemos exportado nuestro modelo ya entrenado e importado el conjunto de prueba cambiando los valores de Target que contenían “Enrolled” por “?”. (4) Posteriormente en la pestaña classify hemos importado nuestro modelo, hemos seleccionado la opción “Supplied test set” para importar el conjunto de prueba y seleccinado la variable “Target” como clase. Ajustamos que la salida nos la proporcione en texto plano y le hemos dado a Start y estos han sido los resultados:

```
=== Predictions on test set ===

    inst#     actual  predicted error prediction
        1        1:? 3:Graduate       0.983
        2        1:? 3:Graduate       0.73
        3        1:?  1:Dropout       0.511
        4        1:? 3:Graduate       1
        5        1:? 3:Graduate       0.99
        6        1:?  1:Dropout       0.694
        7        1:? 3:Graduate       0.929
        8        1:?  1:Dropout       0.691
        9        1:? 3:Graduate       0.629
       10        1:? 3:Graduate       0.938
       11        1:? 3:Graduate       0.812
       12        1:?  1:Dropout       0.625
       13        1:? 3:Graduate       0.913
       14        1:? 3:Graduate       0.708
       15        1:?  1:Dropout       0.985
       16        1:?  1:Dropout       0.998
       17        1:?  1:Dropout       0.989
       18        1:?  1:Dropout       0.994
       19        1:? 3:Graduate       0.648
       20        1:?  1:Dropout       1
       21        1:?  1:Dropout       0.62
       22        1:? 3:Graduate       0.93
       23        1:?  1:Dropout       0.919
       24        1:? 3:Graduate       0.998
       25        1:? 3:Graduate       0.993
       26        1:? 3:Graduate       0.996
       27        1:? 3:Graduate       0.986
       28        1:? 3:Graduate       0.938
       29        1:? 3:Graduate       0.994
       30        1:?  1:Dropout       0.769
       31        1:?  1:Dropout       0.977
       32        1:? 3:Graduate       0.808
       33        1:?  1:Dropout       0.972
       34        1:?  1:Dropout       0.857
       35        1:? 3:Graduate       0.867
       36        1:? 3:Graduate       0.941
       37        1:? 3:Graduate       0.638
       38        1:? 3:Graduate       0.599
       39        1:? 3:Graduate       0.991
       40        1:? 3:Graduate       0.999

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Total Number of Instances               0
Ignored Class Unknown Instances                40

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 ?        ?        ?          ?        ?          ?        ?         ?         Dropout
                 ?        ?        ?          ?        ?          ?        ?         ?         Enrolled
                 ?        ?        ?          ?        ?          ?        ?         ?         Graduate
Weighted Avg.    ?        ?        ?          ?        ?          ?        ?         ?

=== Confusion Matrix ===

 a b c   <-- classified as
 0 0 0 | a = Dropout
 0 0 0 | b = Enrolled
 0 0 0 | c = Graduate
```

Ilustración 13. Resultados predicción conjunto de prueba.

Se puede ver como no proporciona métricas de evaluación pues no está entrenando el modelo realmente, pero, en las predicciones si vemos que nos da valores para los registros que tiene el archivo test.

Las predicciones proporcionadas por el modelo indican una alta confianza para algunos de los registros, especialmente aquellos con una puntuación de predicción cercana a 1, lo que sugiere que el modelo está bastante seguro de la clasificación. Esto refleja que el modelo está identificando patrones confiables para las clases de estudiantes que se graduarán y aquellos que abandonarán. Así es, pues si observamos las predicciones que hemos obtenido coinciden con los perfiles previamente estudiados para los estudiantes en riesgo de abandono (Dropout) o graduación (Graduate).

Si nos fijamos, el modelo predice correctamente a los estudiantes como “Graduados” con una alta probabilidad de certeza (por encima de 0.9 en muchos casos). Esto es consistente con el análisis previo, donde los estudiantes con buen rendimiento académico, buenos antecedentes familiares y apoyo financiero tienen más probabilidades de completar sus estudios. Por otro lado, los estudiantes clasificados como “Abandonados” (Dropout) también tienen predicciones con una probabilidad significativa (mayores a 0.5 en muchos registros). Este comportamiento se alinea con los perfiles de estudiantes que muestran un rendimiento académico bajo, dificultades familiares o carecen de apoyo económico, características que previamente habíamos identificado como factores de riesgo para el abandono escolar. Las características más relevantes que contribuyen a la predicción de Dropout y Graduate en el modelo son consistentes con lo que se analizó anteriormente. Los estudiantes con buenas “*Admission_grades*” y antecedentes académicos sólidos tienen una mayor probabilidad de ser clasificados como Graduados, mientras que aquellos con menor rendimiento académico o sin apoyo económico, familiar o social, tienden a ser clasificados como Abandonados (Dropout).

Las predicciones realizadas por el modelo son consistentes con los perfiles previamente identificados. Los Graduados tienen perfiles académicos sólidos, un buen apoyo familiar y acceso a recursos como becas. Los estudiantes clasificados como Dropout presentan perfiles con menos apoyo, bajo rendimiento académico y posibles obstáculos personales o económicos. El modelo, por tanto, confirma que los estudiantes que abandonan o se gradúan pueden ser diferenciados con precisión utilizando los atributos identificados, lo que resalta la efectividad del modelo para predecir estos dos resultados clave.

En resumen, el modelo desarrollado demuestra que, aun sin datos posteriores, es posible predecir la deserción con un alto grado de exactitud, permitiendo a las instituciones educativas implementar políticas preventivas que mejoren la retención y éxito estudiantil desde el primer momento. El hecho de que el modelo pueda diferenciar de manera efectiva entre estudiantes con diferentes perfiles, basándose solo en los datos de matrícula, demuestra que ha logrado generalizar bien y puede ser útil en escenarios educativos reales donde la toma de decisiones debe realizarse a tiempo.

5. Bibliografía

1. **UNESCO. (2022).** *Reimaginar juntos nuestros futuros: Un nuevo contrato social para la educación*. Reimaginar juntos nuestros futuros: un nuevo contrato social para la educación - UNESCO Biblioteca Digital
2. **Predict Students' Dropout and Academic Success.** Predict Students' Dropout and Academic Success - UCI Machine Learning Repository

3. **Vídeo YouTube de introducción a la herramienta de Weka.** Introducción a WEKA
4. **Vídeo YouTube de evaluación del modelo en Weka.** Clasificación o Predicción usando archivos Test en WEKA
5. **Documentación de Weka.** Documentación Weka
6. **¿Qué es boosting?** ¿Qué es el boosting?