Programación Lógica

Guía CRISP-DM

1. Fase de comprensión del problema

1.1. Evaluación de la situación:

El objetivo general del proyecto es desarrollar un modelo de minería de datos capaz de predecir si un estudiante aprobará o no, a partir de variables relacionadas con sus hábitos académicos y personales.
El proyecto se realizó en la herramienta RapidMiner Studio, que permite implementar procesos de aprendizaje supervisado de forma visual y modular.
Recursos disponibles:

- Datos: dataset en formato CSV con información de estudiantes obtenida de kaggle.
- Software: RapidMiner Studio (versión gratuita).
- Recursos computacionales: PC de los integrantes.
- Personal: equipo de trabajo compuesto por alumnos:
    - Amarilla, Aldo
    - Eitner, Bianca
    - Sosa, Nicolas

1.2. Determinación de los objetivos de DM.

- Construir un modelo predictivo que estime la variable `condition`.
- Identificar los factores más influyentes sobre el rendimiento académico.
- Generar reglas interpretables que permitan extraer conocimiento útil sobre el resultado de los estudiantes.

El resultado exitoso se define como obtener un modelo con una precisión superior a 80%(ochenta por ciento).

2. Fase de comprensión de los datos

2.1. Recolección inicial de datos:

A continuación, se detalla el conjunto de datos adquirido para el proyecto, el cual se denomina "Student Exam Score Dataset", el cual fue adquirido de la plataforma de repositorios de datos Kaggle URL:
https://www.kaggle.com/datasets/grandmaster07/student-exam-score-dataset-analysis.

Se trata de un conjunto de datos tabulares (en formato CSV) proveniente de una base de datos educativa. Cada registro representa a un estudiante y cuantifica sus hábitos de estudio y su desempeño académico previo.
En primer lugar, se realizó una búsqueda exploratoria en la plataforma Kaggle utilizando términos clave como "rendimiento académico" y "student performance". Durante esta búsqueda, se identificaron dos conjuntos de datos potenciales que se ajustaban a los objetivos del proyecto. El descarte de uno de los datasets candidatos fue debido a que una revisión preliminar reveló una cantidad excesiva de campos con datos faltantes (ETL *missing values*). Dicha inconsistencia comprometía la calidad del análisis y requería un esfuerzo de limpieza desproporcionado. Lo cual nos llevó a la selección del dataset "Student Exam Score Dataset".

Descripción de los atributos:

El conjunto de datos original en formato .csv contiene los siguientes atributos.

Atributo	Descripción
id_student	Identificador único de los estudiantes.
hours_studied	Número de horas dedicadas al estudio.
previus_scores	Promedio de notas obtenidas anteriormente.
sleep_hours	Horas promedio de sueño diario.
attendance_percent	Porcentaje de asistencia a clases.
exam_score	Nota del examen final.

Durante la fase de adquisición y la exploración inicial de los datos (previa a la preparación), se identificaron las siguientes limitaciones y problemas inherentes al dataset:

- El dataset cuenta con un número reducido de registros. Esta escasez puede limitar la capacidad de generalización de los modelos a entrenar y aumenta el riesgo de sobreajuste (overfitting).
- Se detectó una inconsistencia crítica en la variable objetivo exam_score. Aunque la descripción del dataset indica una escala de 1 a 100, la distribución real de los datos mostró que la gran mayoría de las puntuaciones eran anormalmente bajas (la mayoria por debajo del 50).
- Como consecuencia de la anomalía anterior, se constató un fuerte desbalance. Existe una baja representación de estudiantes con notas altas (aprobados) en comparación con aquellos con notas bajas (desaprobados), independientemente del umbral de aprobación que se defina.

3. Fase de preparación de los datos

Esta fase fue crucial para adecuar los datos brutos adquiridos de Kaggle, transformándolos en un conjunto de datos listo y optimizado para el proceso de minería de datos y la construcción de modelos. Este proceso se dividió en las siguientes tres tareas:

3.1. Selección de datos:

El objetivo de esta tarea fue definir qué atributos (columnas) y tuplas (filas) del conjunto de datos original se utilizarían en el modelado.
Se decidió conservar el 100% de los registros (filas). Esta decisión se tomó tras una inspección que confirmó que el dataset no presentaba valores faltantes (missing values).
Se realizó una selección de atributos basada en su rol y relevancia predictiva:

- Atributos Predictivos Incluidos: Se seleccionaron todos los atributos que aportan información relevante sobre los hábitos y el contexto del estudiante. Estos fueron:
    - hours_studied
    - previus_scores
    - sleep_hours
    - attendance_percent
- Atributos Excluidos (No Predictivos):
    - id_student: Se excluyó de los predictores, ya que es un metadato (identificador único) que no posee ningún valor predictivo sobre el rendimiento del alumno.
    - exam_score: se quitó porque lo reemplace con una nueva columna generada con valores binarios(aprobó, no aprobó)

3.2. Construcción de datos:

Esta tarea implicó la generación de nuevos atributos derivados para cumplir con los requisitos del modelo a implementar.
En primer lugar, la tarea principal fue la ingeniería de una nueva característica para posibilitar el análisis de clasificación (específicamente, el Árbol de Decisión).
Se creó una nueva columna categórica llamada condition, ya que el modelo de clasificación requerirá esta etiqueta de clase categórica y no un valor continuo como lo es exam_score. Se estableció una regla de negocio para "discretizar" la variable exam_score. Basado en el problema de distribución anómala detectado en la fase de adquisición (donde la mayoría de las notas eran inferiores a 50), se definió un umbral de aprobación funcional en > 4.
La implementación fue:

- condition = 'Y' (Aprueba) si exam_score > 4
- condition = 'N' (Desaprueba) si exam_score <= 4

3.3. Formateo de los datos:

El formateo implicó las transformaciones sintácticas y estructurales necesarias para que los datos fueran compatibles con las herramientas de software utilizadas.
Para facilitar la adición manual del atributo condition, el archivo .csv original descargado de Kaggle fue convertido e importado a formato Excel (.xlsx). En esta hoja de cálculo se aplicó la fórmula condicional SI(...) para generar la nueva columna.
Una vez cargado el archivo Excel en RapidMiner, se utilizó el operador Set Role para definir la variable objetivo, se establece condition con el rol `label`.
El resto de las variables (hours_studied, previus_scores, etc.) se mantuvieron con su rol por defecto de attribute (predictor).

Finalmente, se utilizó el operador Split Data para particionar el conjunto de datos en dos subconjuntos:

- 70% para Entrenamiento (para construir el modelo).
- 30% para Prueba (para evaluar el rendimiento del modelo en datos nuevos).

Se aplicó la opción de muestreo estratificado (stratified sampling) sobre la variable objetivo (condition). Esto garantiza que la distribución de los datos (especialmente la proporción de "Y" y "N") se mantenga de forma equitativa en ambos conjuntos, lo cual es vital dado el desbalance detectado.

4. Fase de modelado

4.1. Selección de la técnica de modelado:

Se seleccionó la técnica de Árbol de Decisión (Clasificación). Esta técnica se implementó en RapidMiner utilizando el operador Decision Tree.
Para la construcción del árbol, se configuró el parámetro criterion (criterio de división) con el valor gain_ratio (ratio de ganancia). Se optó por gain_ratio en lugar de otros criterios (como information gain) porque penaliza los atributos que tienen una gran cantidad de valores únicos, evitando así el sobreajuste (overfitting) y seleccionando divisiones más generalizables.

4.2. Criterio de división:

El modelo emplea el criterio de mínimos cuadrados (least squares), que minimiza la suma de los errores al cuadrado entre los valores reales y los predichos en cada hoja.
La predicción en cada hoja corresponde al promedio de los valores reales del conjunto de entrenamiento que cumplieron esas condiciones.

4.3. Validación del modelo:

Para encontrar la configuración óptima, se realizaron varias pruebas experimentales variando la partición de los datos (operador Split Data):

- Prueba 1 (60% Entrenamiento / 40% Prueba): Resultó en una precisión global (accuracy) de aproximadamente 50%.
- Prueba 2 (80% Entrenamiento / 20% Prueba): Resultó en una precisión global de aproximadamente 55%.
- Partición Final (70% Entrenamiento / 30% Prueba): Esta partición fue la que arrojó el mejor rendimiento, como se detalla en la siguiente fase.

Estos resultados sugieren que la partición 70/30 ofrece el mejor equilibrio entre proporcionar suficientes datos para que el modelo aprenda (entrenamiento) y mantener un conjunto de datos lo suficientemente grande y representativo para una evaluación fiable (prueba).
El modelo final se generó utilizando esta partición 70/30.

5. Fase de Evaluación

El primer paso de la evaluación consiste en verificar si el modelo cumple con los criterios de éxito definidos en la Fase 1. El objetivo de negocio establecido era alcanzar una precisión global (accuracy) en el modelo predictivo de al menos el 80%.
Tras aplicar el modelo entrenado (con la partición 70/30) sobre el conjunto de datos de prueba, se obtuvo una precisión global del 88.33%.
Con este resultado, se concluye que el modelo de Árbol de Decisión cumple y supera el criterio de éxito principal definido para el proyecto.

Para una interpretación detallada del rendimiento, se empleó la herramienta de evaluación principal: la **Matriz de Confusión**. Esta matriz compara las clases reales del conjunto de prueba con las clases que el modelo predijo para esos mismos datos.

accuracy: 88.33%

	true N	true Y	class precision
pred. N	47	5	90.38%
pred. Y	2	6	75.00%
class recall	95.92%	54.55%	

El análisis detallado de la matriz arroja las siguientes conclusiones clave:

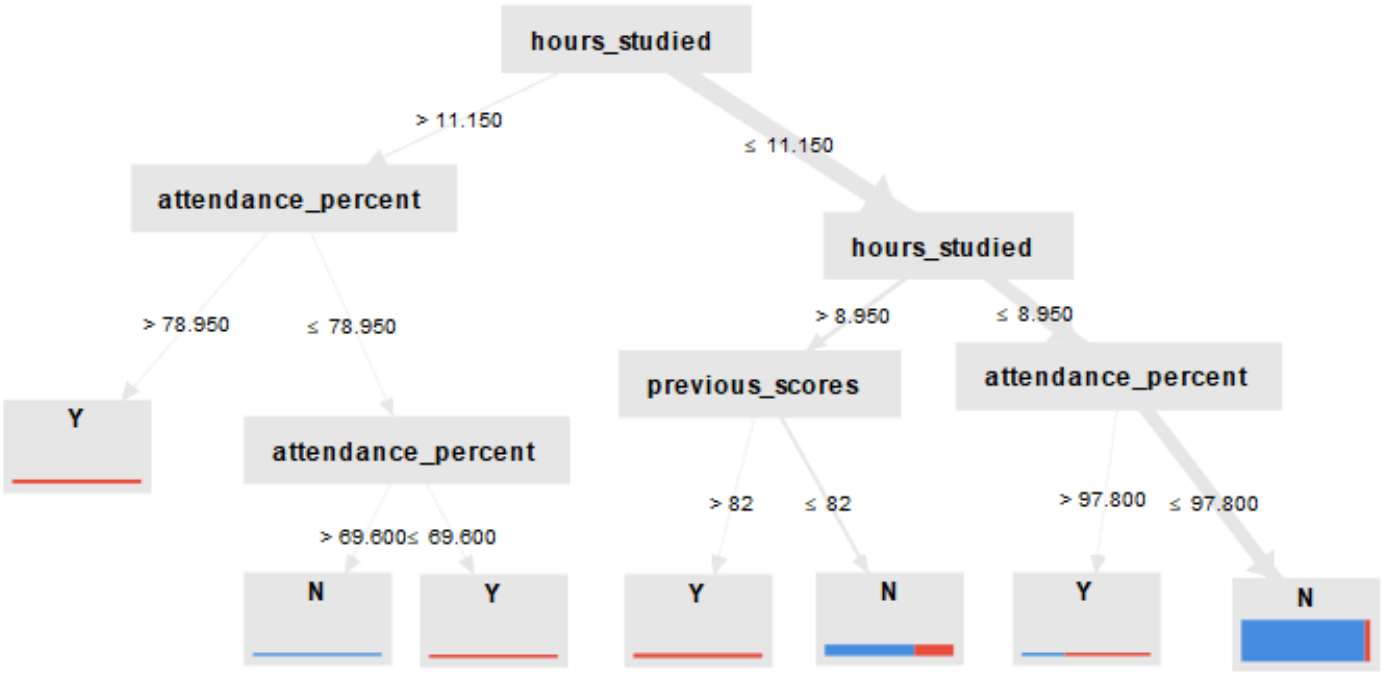
- **Precisión Global:** 88.33%. El modelo clasificó correctamente a 53 de los 60 estudiantes del conjunto de prueba (47 Verdaderos Negativos + 6 Verdaderos Positivos).
- **Errores:** El modelo cometió un total de 7 errores (5 Falsos Negativos + 2 Falsos Positivos).
- **Fortaleza (Clase "N" - Desaprueba):** El modelo es extremadamente fiable para identificar a los estudiantes que van a **desaprobar**.
    - `class recall (` **True N): 95.92%**. El modelo detectó correctamente a 47 de los 49 estudiantes que realmente desaprobaron.
- **Debilidad (Clase "Y" - Aprueba):** La fiabilidad del modelo disminuye drásticamente al predecir a los estudiantes que **aprueban**.
    - `class recall` **(True Y): 54.55%**. Esta es la métrica más débil. El modelo solo pudo identificar correctamente a 6 de los 11 estudiantes que realmente aprobaron. Los otros 5 fueron clasificados erróneamente como "N" (Falsos Negativos).

Teniendo en cuenta los resultados obtenidos, es preciso revisar el proceso global:

- Los resultados justifican la selección de la partición 70/30. Como se documentó en la fase de modelado, las pruebas con otras particiones (60/40 o 80/20) arrojaron precisiones significativamente menores (aprox. 50-55%). Esto indica que la configuración 70/30 fue la óptima para este conjunto de datos.

- La debilidad en predecir la clase "Y" (aprueba) no es un fallo del algoritmo, sino una consecuencia directa de los problemas identificados en la Fase de Adquisición de Datos: el desbalance de clases (muy pocos "Y") y el tamaño de muestra limitado. El modelo tuvo muy pocos ejemplos de "aprobados" para aprender a reconocerlos eficazmente.
- Como establece la metodología, debe considerarse que la fiabilidad calculada del 88.33% se aplica solamente para los datos sobre los que se realizó el análisis. Dada la limitación del tamaño del dataset, no se puede garantizar que el modelo mantenga este mismo rendimiento en un conjunto de datos de producción mucho más grande, diferente o más balanceado.

5.1. Interpretación del árbol



El árbol de decisión generado proporciona un mapa visual de la lógica que el modelo utiliza para clasificar a los estudiantes. La interpretación del árbol revela los siguientes puntos clave:

- **El atributo principal:** El nodo raíz (la primera división) es hours_studied (horas estudiadas). Esto significa que el criterio gain_ratio determinó que esta es la variable más importante para predecir si un alumno aprueba o desaprueba.
- **Atributos secundarios:** Las siguientes divisiones más importantes son attendance_percent (porcentaje de asistencia) y previous_scores (notas previas).
- **Atributo excluido:** Es notable que el atributo sleep_hours (horas de sueño) no fue seleccionado por el algoritmo. Esto indica que, según los datos de entrenamiento, esta variable no aportaba información relevante para la predicción en comparación con las otras.
- Lógica del modelo: a nivel general el arbol muestra que:

- Estudiar mucho (`> 11.150` horas) casi siempre lleva a un resultado "Y" (aprueba), aunque es moderado por la asistencia.
- Estudiar poco (≤ 8.950 horas) está fuertemente correlacionado con un resultado "N" (desaprueba). La gran mayoría de los casos "N" del dataset caen en esta rama (como se ve en la hoja N {N=94, Y=4}).

5.2. Ejemplo de regla:

Del árbol completo se extraen múltiples reglas. A modo de ejemplo, se analiza una de las reglas que predice la clase "Y" (Aprueba):

SI `hours_studied > 11.150` **Y** `attendance_percent > 78.950` **ENTONCES** `condition = Y` (Aprueba)

Interpretación de la regla: "Si un estudiante dedicó más de 11.15 horas de estudio Y su asistencia a clases fue superior al 78.95%, el modelo predice que el estudiante aprueba".

Fiabilidad de la regla: Esta regla específica se generó a partir de 5 casos en el conjunto de entrenamiento. La "pureza" de la hoja es del 100%, ya que los 5 casos que cumplieron esta condición resultaron en "Y" (`{N=0, Y=5}`).

5.3. Conclusión de la evaluación:

Tras analizar tanto las métricas de rendimiento (Matriz de Confusión) como la lógica interna del modelo (Árbol de Decisión), la evaluación concluye lo siguiente:

1. **Objetivo Cumplido:** El modelo logra el criterio de éxito establecido en la Fase 1, alcanzando una precisión global del 88.33%, superando el mínimo requerido del 80%.
2. **Rendimiento Desbalanceado:** El modelo es altamente eficaz para identificar a los estudiantes que desaprueban (Clase "N"), con un *recall* del 95.92%. Sin embargo, es deficiente para identificar a los estudiantes que aprueban (Clase "Y"), con un *recall* de solo 54.55%.
3. **Causa Raíz:** La interpretación del árbol y la matriz confirma que este rendimiento desbalanceado es un reflejo directo de los datos de origen. Los problemas de tamaño de muestra limitado y fuerte desbalance de clases (identificados en la Fase de Adquisición) han provocado que el modelo esté "sesgado" hacia la clase "N", ya que tenía muchos más ejemplos de esta.
4. **Recomendación Futura:** Si bien el modelo es funcional, para mejorar su fiabilidad (especialmente para predecir el éxito de un estudiante), es imperativo volver a entrenarlo con un conjunto de datos más grande y, sobre todo, más balanceado.

6. Fase de Implementación (Prolog)

```
hours_studied > 11.150
|   attendance_percent > 78.950: Y {N=0, Y=5}
|   attendance_percent ≤ 78.950
|   |   attendance_percent > 69.600: N {N=2, Y=0}
|   |   attendance_percent ≤ 69.600: Y {N=0, Y=3}
hours_studied ≤ 11.150
```

| hours_studied > 8.950
| | previous_scores > 82: Y {N=0, Y=6}
| | previous_scores ≤ 82: N {N=16, Y=7}
| hours_studied ≤ 8.950
| | attendance_percent > 97.800: Y {N=1, Y=2}
| | attendance_percent ≤ 97.800: N {N=94, Y=4}