Predecir el abandono y éxito académico de los estudiantes

1. Introducción

Este informe presenta un análisis detallado del proceso que se usó por medio de Machine Learning (ML) para predecir el abandono y éxito académico de los estudiantes con un dataset disponible en Kaggle. El conjunto de datos proporciona una visión integral de los estudiantes matriculados en varios programas de licenciatura ofrecidos por una institución de educación superior. Incluye datos demográficos, factores socioeconómicos e información sobre el rendimiento académico que pueden utilizarse para analizar los posibles factores predictivos del abandono y éxito académico de los estudiantes [1].

El objetivo de este conjunto de datos es contribuir a la reducción del abandono y el fracaso académico en la educación superior mediante el uso de técnicas de inteligencia artificial para identificar tempranamente a los estudiantes en riesgo en las primeras etapas de su trayectoria académica. De esta manera, se podrán implementar estrategias de apoyo específicas para cada caso.

Vamos a usar el **dataset** de Kaggle Predict students' dropout and academic success | Kaggle, el conjunto de datos incluye una variedad de factores demográficos, socioeconómicos y de rendimiento académico relacionados con los estudiantes matriculados en instituciones de educación superior. A continuación se pueden observar sus variables, un total de 35 variables (columnas) :

- Marital status: The marital status of the student.
- Application mode: The method of application used by the student.
- Application order: The order in which the student applied.
- Course: The course taken by the student.
- Daytime/evening attendance: Whether the student attends classes during the day or in the evening.

- Previous qualification: The qualification obtained by the student before enrolling in higher education.
- Nacionality: The nationality of the student.
- Mother's qualification: The qualification of the student's mother.
- Father's qualification: The qualification of the student's father.
- Mother's occupation: The occupation of the student's mother.
- Father's occupation: The occupation of the student's father.
- Displaced: Whether the student is a displaced person.
- Educational special needs: Whether the student has any special educational needs.
- Debtor: Whether the student is a debtor.
- Tuition fees up to date: Whether the student's tuition fees are up to date.
- Gender: The gender of the student.
- Scholarship holder: Whether the student is a scholarship holder.
- Age at enrollment: The age of the student at the time of enrollment. (Numerical)
- International: Whether the student is an international student. (Categorical)
- Curricular units 1st sem (credited): The number of curricular units credited by the student in the first semester.
- Curricular units 1st sem (enrolled): The number of curricular units enrolled by the student in the first semester.
- Curricular units 1st sem (evaluations): The number of curricular units evaluated by the student in the first semester.
- Curricular units 1st sem (approved): The number of curricular units approved by the student in the first semester
- **Target**: Variable to predict, says if the student is Dropout, Graduate or Enrolled.

Nuestra **variable a predecir** es Target, una variable categórica, que es la que nos dice si el estudiante se graduó, si desertó de la carrera y si aun la esta usando.

Cómo **métrica de desempeño** vamos a usar el **Log-loss** es una métrica que mide el desempeño de un clasificador con respecto a cuánto divergen las probabilidades predichas de la etiqueta de clase verdadera. Un log-loss más bajo indica un mejor modelo. Un modelo perfecto, que predice una probabilidad de 1 para la clase verdadera, tendrá un log-loss de 0.

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}),$$

Imagen 1. Formula del logloss

La función Log Loss proporciona una medida de cuán seguras son las predicciones de un clasificador, en lugar de simplemente medir cuán correctas son. Por ejemplo, una

probabilidad predicha de 0.80 para una etiqueta verdadera de 1 se penaliza más que una probabilidad predicha de 0.99.

Además de la métrica de desempeño logloss, se vio necesario utilizar la métrica de precisión (accuracy) en el proyecto . Esto debido a que a diferencia del logloss, que se enfoca en la estimación de probabilidades y la confianza de las predicciones, la precisión proporciona una medida directa de la proporción de predicciones correctas en relación con el total de muestras. Esto es especialmente útil cuando el objetivo principal es evaluar la exactitud general del modelo en términos de clasificación correcta de las instancias. La métrica de precisión es especialmente útil en casos donde todas las clases tienen una importancia similar y no hay una necesidad específica de evaluar la confianza de las predicciones. Al utilizar tanto logloss como precisión en el análisis de un modelo de Machine Learning, se obtiene una evaluación más completa que considera tanto la calidad de las probabilidades predichas como la tasa general de clasificación correcta.

$$Accuracy = \frac{Correct\ prediction}{Total\ cases} * 100\%$$

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} * 100\%$$

Imagen 2. Fórmula del Accuracy

Teniendo en cuenta estas métricas, esperamos tener un accuracy mayor a XXX y un Logloss de XXX hacia abajo. Esto con el fin de decir si el modelo está clasificando correctamente o no.

2. Exploración descriptiva del dataset

Para explorar el dataset se usó primero un histograma.

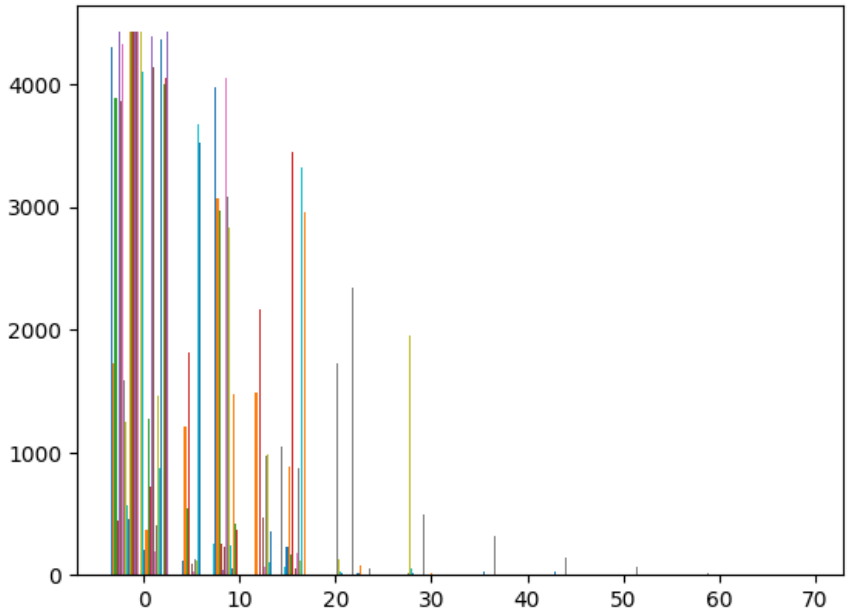


Imagen 3. Histograma

Se observa del histograma que los valores están agrupados en su mayoría alrededor del valor de 4000 y hay unos datos atípicos que hay que analizar.

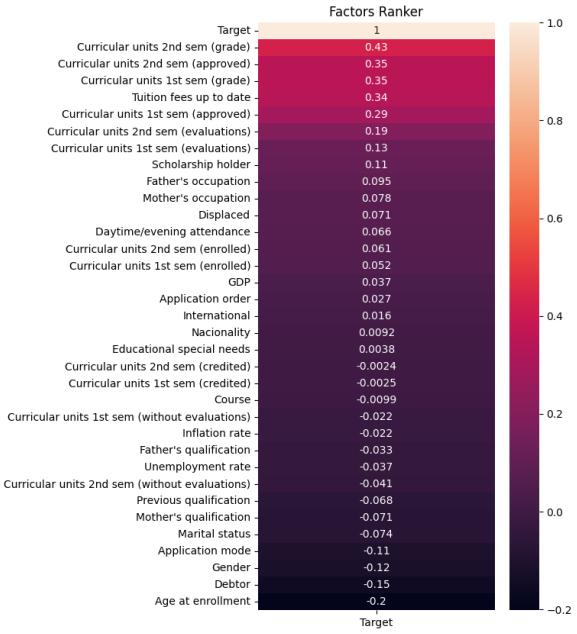


Imagen 4. Mapa de calor

Se hizo un mapa de calor (Imagen 4) para ver la relación entre cada variable con nuestra columna objetivo Target y así poder notar cuales son mas relacionadas y cuáles no.

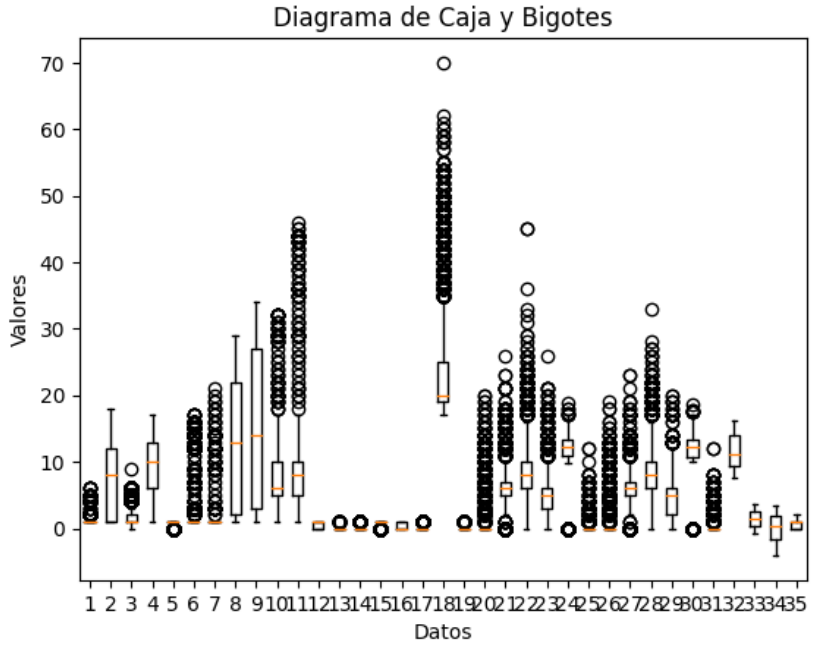


Imagen 5. Diagrama de Caja y Bigotes

De este diagrama se puedo observar varias variables con valores atípicos, se noto una distribución razonable y logramos observar la mediana de cada variable que es la línea anaranjada.

3. Iteraciones de desarrollo

Antes de comenzar con cualquier modelo se optó por hacer un **preprocesado**, que podemos encontrar en el archivo "*02 - preprocesado.ipynb*" del dataset en el siguiente orden:

1. Se realizó una transformación de variables categóricas en numericas.
2. Se eliminó los datos duplicados, usando el método "*df.**drop_duplicates**(inplace=True)*"
3. Para los datos faltantes en el dataset se optó por eliminar las filas con esos valores. Haciendo uso de *df.**dropna**(inplace=True)*.

4. Se hizo una selección de características basada en la correlación frente a la variable objetivo (Target) pero después de pruebas se llega a la conclusión que no es relevante este paso.
5. Se eliminaron manualmente Variables que no aportan al modelo, variables que casi todos sus datos eran iguales. Un ejemplo era la Variable *Nacionality*, la mayoría de sus estudiantes son del mismo país, por lo que no es algo relevante.

3.1 Modelos usados:

Los modelos usados fueron modelo de regresión logística, árbol de decisión, Random Forest y Gradiente Boosting Tree. Se usó estos 4 modelos, todos pertenecen a modelos de aprendizaje supervisado.

1. **Modelo de árboles de Decisión.**

En este modelo, se construye un árbol en el que cada nodo interno representa una característica o atributo, cada rama representa una posible salida para esa característica y cada hoja representa una clase o valor de salida. El objetivo es dividir los datos en subconjuntos más pequeños y homogéneos en función de las características, para poder tomar decisiones precisas en la clasificación o la predicción de valores.

El código completo se puede encontrar en "*03 - modelo de árbol de decisión.ipynb*". Para hacer uso de este se importaron las librerías, se dividió los datos en conjuntos de entrenamiento y prueba, crea el modelo, se entrena por medio de *fit* y obtenemos el Accuracy y el Logloss.

De estron los siguientes:

- `LogLoss: 12.14`
- `Acurracy: 66.33%`

Con base en estos resultados podemos decir que no es eficiente este modelo, pues se ve muy alejado de los parámetros definidos. El Logloss es muy lejano a cero y la precisión no es tan alta. Es necesario analizar otros modelos para comparar.

2. **Modelo de regresión logística.**

Este modelo busca estimar la probabilidad de que una instancia pertenezca a una clase específica. Utiliza una función logística o sigmoide para mapear las características de entrada a un valor entre 0 y 1, que representa la probabilidad de pertenencia a la clase positiva.

El código completo se puede encontrar en "*04 - modelo de regresión logística.ipynb*". Para hacer uso de este se importaron las librerías, se dividió los datos en conjuntos de entrenamiento y prueba, crea el modelo, se entrena por medio de *fit* y obtenemos el Accuracy y el Logloss.

De este modelo los resultados fueron los siguientes:

- `LogLoss: 0.62`
- `Accuracy: 75.48%`

Este modelo tuvo un rendimiento significativamente mejor que el de árboles de decisión. El Logloss se acerca mucho más a uno y el accuracy subió un 10%.

3. **Modelo de Random Forest.**

Este modelo se basa en la creación de múltiples árboles de decisión, donde cada árbol se entrena con una muestra aleatoria de los datos y un subconjunto aleatorio de las características. Luego, se combina la predicción de cada árbol para obtener una predicción final. El objetivo es reducir el sobreajuste y mejorar la precisión del modelo.

El código completo se puede encontrar en "*05 - modelo de Random Forest.ipynb*". Para hacer uso de este se importaron las librerías, se dividió los datos en conjuntos de

entrenamiento y prueba, crea el modelo, se entrena por medio de *fit* y obtenemos el Accuracy y el Logloss.

De este modelo los resultados fueron los siguientes:

- `LogLoss: 5.94`
- `Accuracy: 66.78%`

Se puede deducir que este modelo tampoco es eficiente para nuestro problema, haciendo una comparación exhaustiva nos damos cuenta que otros modelos llegan a dar un predicción más correcta.

4. **Modelo de Gradiente Boosting Tree.**

El modelo utiliza un enfoque de "impulso" para mejorar la precisión de las predicciones, lo que significa que cada árbol se enfoca en los errores cometidos por el árbol anterior para mejorar la precisión general del modelo.

El código completo se puede encontrar en "*06 - modelo Gradiente Boosting Tree.ipynb*". Para hacer uso de este se importaron las librerías, se dividió los datos en conjuntos de entrenamiento y prueba, crea el modelo, se entrena por medio de *fit* y obtenemos el Accuracy y el Logloss.

- `LogLoss: 0.59`
- `Accuracy: 76.38%`

Este modelo en comparación con los anteriores se torna más eficiente, tiene buenos números. Queda hacer una comparación más exhaustiva frente al otro modelo de regresión logística.

3.2 Resultados generales:

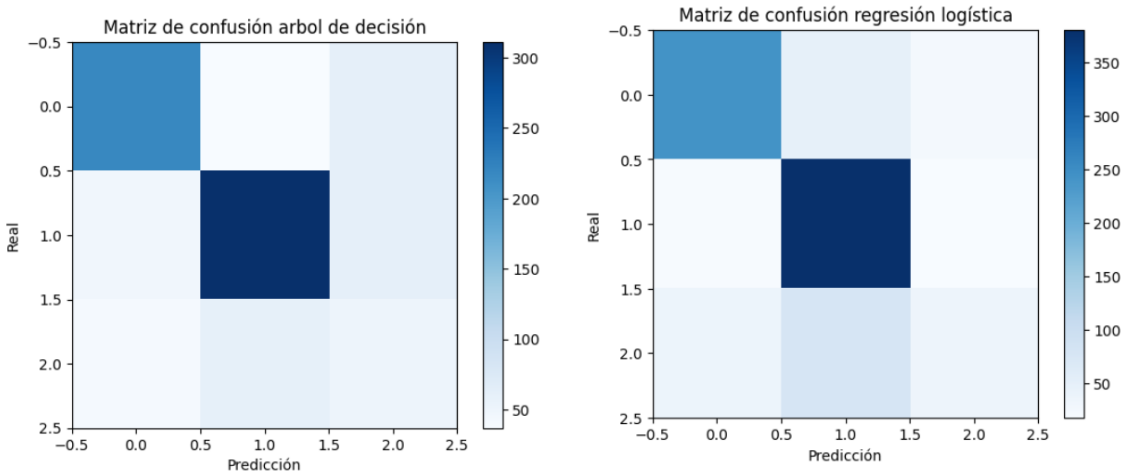
	Arboles de Decisión	Regresión logística	Random Forest	Gradiente Boosting Tree
LogLoss	12.14	0.62	5.94	0.59
Accuracy	66.33%	75.48%	66.78%	76.38%

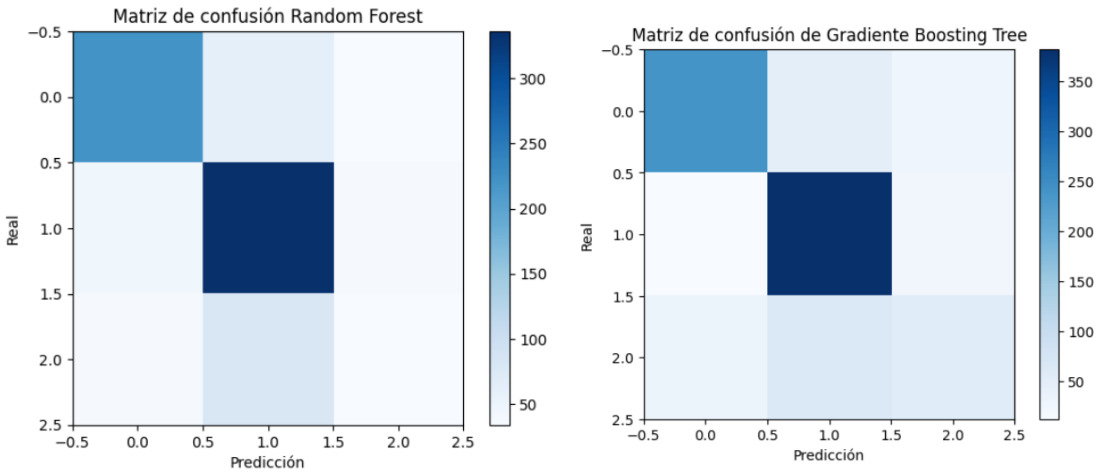
Tabla 1. Resultados de los modelos (Métricas)

Teniendo en cuenta los resultados, los modelos con la mayor precisión son el modelo de Regresión Logística y el modelo Gradiente Boosting Tree, ambos con una precisión alrededor del 75%. Sin embargo, el modelo Gradiente Boosting Tree tiene un log loss significativamente menor en comparación con los demás modelos, lo que indica una mejor capacidad para hacer predicciones con una mayor confianza.

En conclusión, el modelo Gradiente Boosting Tree parece ser el mejor modelo en función de la precisión y el log loss proporcionados.

También podemos ver los resultados visualmente con la matriz de confusión:





De las matrices de confusión de cada modelo, podemos deducir que casi todas tienen la misma tendencia. Pero si verificamos la diagonal principal, vemos que la Gradiente Boosting Tree tiene su diagonal más colorida, la clase 0, la clase 1 es persistente en todos los modelos pero la clase 2 tiene más color en este modelo, lo que indica que hizo una mejor predicción.

Recordemos cual es cada clase: Dropout: 0, Graduate: 1, Enrolled': 2

4. Retos y consideraciones de despliegue

El primer reto que se tuvo en proyecto, fue encontrar los modelos correctos. En diferentes búsquedas se encontraban muchos modelos, pero era difícil saber cuál iba a ser el correcto, la forma de solucionarlo fue probando. Pues, es importante elegir el modelo adecuado para el problema en cuestión, teniendo en cuenta las características de los datos, la cantidad de datos disponibles, la complejidad del modelo y la precisión requerida.

El segundo reto fue con la normalización de los datos, al principio al hacer esto, se sobreentrenaba el modelo, con cualquier métrica daba un precisión del 100%, aun si se tomaba tan solo un 50% de datos para entrenamiento. Después se logró hacer un estandarizado y esa fue la solución al problema.

5. Conclusiones

Basado en los resultados obtenidos y en el análisis realizado, se evaluaron varios modelos de clasificación utilizando diferentes métricas como precisión y log loss. Se observó que el modelo de Gradiente Boosting Tree demostró la mayor precisión, con un valor de alrededor del 75%, superando ligeramente al modelo de Regresión Logística. Además, el modelo de Gradiente Boosting Tree también presentó el log loss más bajo, lo que indica una mayor capacidad para hacer predicciones confiables. Estos resultados sugieren que el modelo de Gradiente Boosting Tree puede ser la mejor opción para el conjunto de datos en cuestión, ya que ofrece un equilibrio entre precisión y capacidad predictiva.

El estandarizado de los datos y el uso de gráficas son elementos esenciales en el proceso de modelado y evaluación de modelos de clasificación. Al estandarizar los datos, se garantiza un tratamiento equitativo de las variables, mientras que las gráficas proporcionan una representación visual que facilita la interpretación de los resultados y la toma de decisiones informadas. Estos enfoques combinados contribuyen a mejorar la precisión y la comprensión de los modelos de clasificación.

En resumen, el modelo de Gradiente Boosting Tree se destacó como el mejor modelo en términos de precisión y log loss, lo que sugiere que tiene la capacidad de realizar predicciones más precisas y confiables en el conjunto de datos dado. Sin embargo, se recomienda realizar más evaluaciones y pruebas con diferentes métricas y técnicas de validación para obtener una visión más completa y precisa de la capacidad de los modelos. Además, es importante considerar el contexto específico del problema y las características del conjunto de datos al seleccionar el modelo más adecuado.

[1] Predict students' dropout and academic success | Kaggle