Éxito estudiantil: Análisis Comparativo de Métodos de Selección de Características y Modelos de Aprendizaje Automático

Facundo Mazzola
Estudiante de Lic. Ciencias de datos
Universidad Católica Argentina (UCA)
Buenos Aires, Argentina
facundomazzola@uca.edu.ar

***Abstract*—This paper aims to be able to determine which are the features that are most relevant for their use in predicting wether a student drops out of their superior studies, they graduate at the time their course is set to finish or they are still enrolled at that time. For this, 12 features were extracted as most important using the *Random Forest* and *Boruta* algorythm sequentially.**

These features were used to train and test three machine learning models. The same machine learning models were also trained on other sets of features provided by various feature selection models done in a previous paper. When comparing all machine learning and feature selection models, it was observed that the *Random Forest* and *Boruta* combination for feature selection had an overall better performace with all three models and the *Random Forest* machine learning model had a better performance across all feature selection models.

***Index Terms*—Ensemble, dropout, feature selection, algorythm comparison**

ESTADO DEL ARTE

Anterior a este trabajo, durante la creación del dataset a analizar, se produjo en la *Escola Superior de Tecnologia e Gestão* -en Portalegre, Portugal- un paper (referencia 2), donde se explicó la creación del mismo. En el mismo se realizaron distintas selecciones de características utilizando diferentes métodos y/o algoritmos, los cuales arrojaron distintos resultados, es decir, un diferente set de características *rankeadas* como las más importantes -considerando las diez mejores por método-. En el presente trabajo, se busca evaluar los resultados de los diferentes métodos de selección provistos por los autores del susodicho paper y sus resultados, así como la selección provista por el autor del presente trabajo, a partir de su utilización en modelos de *machine learning*.

I. INTRODUCCIÓN Y OBJETIVOS

El objetivo principal de este trabajo es poder determinar cuáles son las variables cuyo valor determina si un alumno abandonará sus estudios superiores o si, en caso de terminarlos, lo hará en tiempo y forma o no.

Para esto, primero se procesarán los datos, y luego se realizará una selección de características, la cual será evaluada y comparada con las obtenidas de el paper antes mencionado, a través de su utilización en modelos de aprendizaje automático tanto simples como de ensamble y evaluados a través de distintas métricas, que determinarán la mejor selección y modelo.

Objetivos secundarios

1) Evaluar el rendimiento de modelos de ensamble respecto de modelos más simples.
2) Utilizar los datos de las variables que más determinan si un alumno abandona los estudios o no los puede completar dentro del plazo esperado, para predecir el comportamiento o resultado del estudiante.
3) Evaluar a distintos modelos en la predicción de la variable objetivo.
4) Evaluar distintos métodos de selección de características.

Hipotesis

"Los algoritmos de ensamble de *machine learning* tendrán un mejor rendimiento que los algoritmos más simples a la hora de clasificar".

II. METODOLOGÍA

A. Los datos

El conjunto de datos que se empleará para este trabajo es un dataset extraído de UCI Machine Learning Repository que consta de 36 variables que se utilizaran como predictores y una variable objetivo, del tipo categórico, que nos indica si cada una de las 4424 observaciones -es decir, cada alumno- abandonó los estudios, no los completó en el plazo previsto, o se graduó a tiempo. El dataset ha sido previamente preprocesado. Se han eliminado tanto datos faltantes como outliers.

B. Procesamiento

1) Codificación de la variable $Target$: La variable $Target$ es la variable objetivo en este trabajo. Es decir, aquella que se busca predecir. Esta es una variable categórica con tres valores: $Dropout$, $Enrolled$ y $Graduate$. Para poder introducirla a modelos de aprendizaje automático, a esta se la codificó a una forma numérica de la siguiente

manera: a las observaciones cuyo valor de la variable target era $Dropout$ se lo reemplazó por el valor 0, a aquellas con el valor $Enrolled$ se lo reemplazo con el valor 1 y a aquellas con el valor $Graduate$ con el valor 2.

2) Datos Faltantes: Para el procesamiento de los datos, primero se trataron los datos faltantes. En la descripción del dataset (referencia 1) se menciona que no se encuentran datos faltantes ya que no hay datos del tipo NA. Aun así, dentro de la descripción de las variables (referencia 3), algunas de las variables categóricas tienen un código de categoría asignado a los datos cuyo valor es desconocido.

Para los datos faltantes de tipo categórico, lo que se realizó fue una imputación de la moda de su respectiva variable a los elementos que aparecen como desconocidos.

3) Balance de datos: Lo siguiente que se observó de la variable $Target$ es que la cantidad de observaciones para cada clase es muy distinta, es decir, es un dataset desbalanceado, lo que puede generar sesgo en los modelos y dar una peor capacidad para predecir instancias de las clases minoritarias.

TABLE I
COMPOSICIÓN DE LA VARIABLE $Target$

Clase	N° Observaciónes	%
Dropout (0)	1421	32%
Enrolled(1)	794	18%
Graduate(2)	2209	50%

Para este problema, la solución que se aplicó fue la utilización del *oversampling*. Esta técnica consta de crear n nuevas observaciones de las clases minoritarias que son copias de las observaciones de cada clase minoritaria, tomadas aleatoriamente con reposición del conjunto de observaciones de cada clase minoritaria. La cantidad de observaciones nuevas para cada clase se calcula como:

$$n_i = M - m_i$$

Donde n_i es la cantidad de observaciones nuevas, M es la cantidad de observaciones de la clase mayoritaria y m_i es la cantidad de observaciones de la clase minoritaria i.

4) One-hot Encoding: Por ultimo, todas las variables categóricas se separaron en columnas binarias con el nombre de cada categoría donde la presencia de un 1 en el elemento j de la columna binaria indica la presencia de la respectiva categoría en el elemento j de la columna original y el 0 su ausencia. Con este procedimiento las 36 variables se convirtieron en 226 variables.

C. Selección de características

Para la selección de características, se utilizó el modelo de *Random Forest*, con el fin de obtener las variables con mayor ganancia de información. Aquellas con una ganancia mayor al percentil 95 del conjunto de importancias obtenidas, se usaron para confirmar que estas resulten importantes utilizando el algoritmo *Boruta*.

1) Random Forest: El algoritmo *Random Forest* que se utilizó es un algoritmo de *bagging* que genera una cantidad n de arboles de decisión que utilizan utilizan un criterio *Gini* para decidir cuál es la variable más importante de un subset. Siendo p_i la probabilidad de que una observación pertenezca a la clase i y j el numero de clases:

$$Gini = 1 - \sum_{i=1}^{j} p_i^2 \tag{1}$$

Cada árbol calcula la impureza de *Gini* para un subset aleatorio de m cantidad de variables, que es la raíz de la cantidad de variables que se introducen, en cada nodo. Luego, calcula el promedio de las diferencias en impuridad antes y después de separar por la variable i, las suma, y las divide por la cantidad de árboles.

2) Boruta: Este algoritmo utiliza en parte al anterior. Lo que hace es crear varias copias de cada una de las variables, pero con sus valores permutados a lo largo de la columna *shadow variables*. Luego el dataset con todas las variables (*originales* + *shadow*) se introduce a un *Random Forest* para ver si la modificación en los valores cambia significativamente el resultado del árbol. Si cambia significativamente, se designa a la variable como *confirmada importante*.
Esto lo hace computando el valor Z para cada característica:

$$Z_j = \frac{imp(j) - \bar{imp}(j_s)}{ds(imp(j_s))} \tag{2}$$

Donde $imp(j)$ es la importancia $Gini$ de la característica original j, $\bar{imp}(j_s)$ es la media de las importancias de sus copias y $ds(imp(j_s))$ el desvío estándar de las mismas. Luego de esto realiza un test con un nivel de significancia $\alpha = 0.05$ donde si el valor Z de la variable es mayor a su valor crítico (1.64), la variable se confirma como importante.

D. Modelización

Seguido de esto, se implementaron los siguientes modelos para poder predecir a la variable $Target$ con las variables salientes de $Boruta$: *árbol de decisión, Random Forest* y *AdaBoost*.

1) Árbol de decisión: El *árbol de decisión* utiliza el criterio de impureza *Gini* (ecuación 1) para realizar separaciones lógicas entre los datos. El algoritmo calcula la impureza potencial que puede tener los nodos subsiguientes a hacer una partición o separación de los datos, teniendo en cuenta cada característica por la cual se puede separar, y dentro de estas cada valor que aparece en el conjunto de valores de las respectivas características -es decir, cada valor que aparece en la columna de una característica j- y decide un valor y característica para separar los datos, donde la impureza *Gini* potencial sea la mínima.

Luego clasifica los nuevos datos según de que lado de las divisiones hechas por el árbol se encuentren estos.

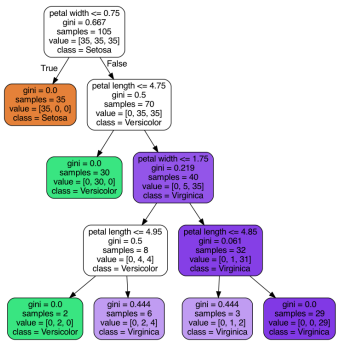

Fig. 1. Ejemplo gráfico de un árbol de decisión

2) Random Forest: Para hacer predicciones, este algoritmo crea n árboles de decisión y separa los datos como se mencionó en el apartado II-C1. Luego, se le introduce al algoritmo los datos a predecir y dentro de este, para cada dato a predecir, cada árbol realiza una predicción y su clasificación final es dada por una votación de las predicciones de cada uno de ellos.

3) AdaBoost: Es un algoritmo de *boosting*, donde crea N árboles de decisión iterativamente, corrigiendo los errores del árbol anterior en la creación de cada uno nuevo. Esto lo hace modificando los pesos de las observaciones. Inicialmente cada observación tiene un peso de $\frac{1}{n}$ donde n es la cantidad de observaciones. Luego, se calcula el error del árbol de la siguiente manera:

$$\epsilon = \frac{\sum_{i=1}^{n} w_i * incorrecto_i}{\sum_{i=1}^{n} w_i} \tag{3}$$

Donde w_i es el peso de la observación i e $incorrecto_i$ es una variable binaria que nos informa si la predicción para la observación i fue correcta.

Seguido, se calcula la importancia α del modelo.

$$\alpha = \frac{1}{2} * \log(\frac{1-\epsilon}{\epsilon}) \tag{4}$$

Y finalmente se actualizan los pesos y se normalizan para que su suma sea igual a 1.

$$w_i = w_i * \exp(-\alpha * y_i * \hat{y}_i) \tag{5}$$

$$w_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \tag{6}$$

Aquí, y_i es el valor real de la observación i, mientras que $\hat{y}_i$ es el valor predicho para la observación i.

De esta manera se crean de forma iterativa nuevos árboles, modificando los pesos de las observaciones en cada iteración. La predicción final para una observación i se da por votación ponderada de todos los árboles, donde cada árbol tiene su peso α.

4) Entrenamiento y validación: Para el entrenamiento de los modelos y validación de los datos, se seleccionó una muestra aleatoria del conjunto de características obtenidas, correspondiente al 70% de las observaciones para entrenar los modelos, y el 30% restante se utilizó para evaluarlos con las siguientes métricas, que resultan útiles para problemas con datos desbalanceados:

$$F1Score = 2 \times \frac{\frac{TP}{TP+FP} \times \frac{TP}{TP+FN}}{\frac{TP}{TP+FP} + \frac{TP}{TP+FN}} \tag{7}$$

De manera simplificada:

$$F1Score_j = \frac{TP}{2TP + FP + FN} \tag{8}$$

Donde TP es la cantidad de predicciones correctas para la clase j, FP es la cantidad de predicciones a las que se les asignó la clase j pero eran de otra clase y FN es la cantidad de predicciones a las que se les asignó otra clase pero realmente eran de la clase j. Esta medida se toma para cada clase y se promedia para obtener una medida global.

El área bajo la curva roc:

$$AUC \simeq \frac{1}{2} \sum_{i=1}^{n-1} (x_{i+1} - x_i) \cdot (y_i + y_{i+1}) \tag{9}$$

Donde n es la cantidad de puntos (x_i, y_i) de una función dada por la relación entre la sensibilidad y especificidad de un modelo de clasificación ante cambios en el umbral de decisión.

Y el *accuracy* por clase:

$$accuracy_j = \frac{TP}{n} \tag{10}$$

En este caso n es la cantidad de predicciones realizadas. Al igual que con el *F1Score*, se promedió el *accuracy* de cada clase para obtener una métrica global de cada modelo.

Este proceso de dividir los datos en conjuntos de entrenamiento y prueba, ajustar los modelos al conjunto de entrenamiento, realizar predicciones y evaluarlas se realizó 100 veces donde cada vez se seleccionaron nuevos conjuntos aleatorios de entrenamiento y prueba, para promediar los resultados de las métricas y así obtener las más representativas de cada modelo.

$$F1Score = \frac{1}{100} \sum_{k=1}^{100} F1Score_k$$

$$AUC = \frac{1}{100} \sum_{k=1}^{100} AUC_k$$

$$accuracy = \frac{1}{100} \sum_{k=1}^{100} accuracy_k$$

III. Resultados

A. Selección de características

Para el caso del *Random Forest*, se seleccionaron las características con una importancia mayor a su percentil 95 del conjunto de importancias. Esta selección dejó 12 características que a continuación se presentan en orden de importancia:

- La cantidad de unidades curriculares aprobadas en el segundo semestre de cursada
- El promedio de las notas finales para las unidades curriculares cursadas en el segundo semestre
- La cantidad de unidades curriculares aprobadas en el primer semestre de cursada
- El promedio de las notas finales para las unidades curriculares cursadas en el primer semestre
- La cantidad de evaluaciones tomadas en el segundo semestre
- La cantidad de evaluaciones tomadas en el primer semestre
- La nota que se obtuvo en el examen de admisión
- La edad a la que se anotó el alumno en la carrera que se esta teniendo en cuenta
- La nota o promedio que obtuvo en su etapa educativa anterior
- Si tiene los pagos de su educación al día
- El producto bruto interno(PBI)
- La tasa de desempleo

Luego,al introducir estas variables al algoritmo *Boruta*, este las confirmó a todas como importantes.

B. Comparación

Luego de realizar el entrenamiento y las predicciones, se evaluaron a todos los modelos de *machine learning*. Se realizó este mismo proceso pero introduciendo a los modelos cada uno de los resultados de las selecciones de características mencionadas en el paper (referencia 2) - aunque sin las transformaciones realizadas a los datos que se realizaron en el apartado II-B ya que estas no fueron realizadas en el mencionado paper y son un aporte de este trabajo- y se los evaluó a cada uno de ellos utilizando las métricas mencionadas anteriormente. A cada uno de estos resultados, incluidos los del presente trabajo, se los trasladó a la siguiente tabla:

TABLE II
RESULTADOS DE LA EVALUACIÓN DE DISTINTAS MÉTRICAS POR MODELO Y MÉTODO DE SELECCIÓN DE CARACTERÍSTICAS

Feature Selection	Modelo	F1 Score	AUC	Balanced Accuracy
RF+Boruta	Random forest	0.87	0.90	0.90
	ADABoost	0.70	0.81	0.77
	Árbol de decisión	0.68	0.80	0.76
XGBoost	Random forest	0.69	0.81	0.78
	ADABoost	0.65	0.79	0.75
	Árbol de decisión	0.66	0.78	0.75
Random Forest	Random forest	0.68	0.80	0.77
	ADABoost	0.64	0.78	0.74
	Árbol de decisión	0.66	0.78	0.74
CatBoost	Random forest	0.68	0.80	0.77
	ADABoost	0.64	0.78	0.75
	Árbol de decisión	0.64	0.78	0.74
LightGBM	Random forest	0.69	0.81	0.77
	ADABoost	0.64	0.78	0.75
	Árbol de decisión	0.65	0.77	0.73

Como se puede observar, para todos los métodos de selección de características, el modelo de *machine learning* que mejor rendimiento mostró es el *Random Forest*, cuestión que se ve mayormente reflejada en las métricas de *accuracy* y *AUC* que presenta. Seguido de este, encontramos el algoritmo *AdaBoost* como el segundo mejor en términos de rendimiento.

Por otro lado, el método de selección de características que mejores resultados exhibió fue el provisto en el presente trabajo: La combinación de *Random Forest* y *Boruta*.

IV. CONCLUSIONES

Luego de realizar la selección de características y compararla con las demás selecciones de características, de este trabajo se pudo aprender que las variables que mayor determinan si un alumno abandonará sus estudios superiores, o si los terminará en tiempo y forma o no, son las provistas por el método de selección de características *Random Forest* combinado con *Boruta*. Además de esto se pudo comprobar, para este caso, que los modelos de ensamble de *bagging* como el *Random Forest* y los de *boosting* como *AdaBoost* tienen mejor rendimiento que otros algoritmos más simples.

No queda descartado sin embargo, que pueda haber otros métodos de selección de características o modelos de *machine learning* que puedan dar resultados distintos, mejores o peores al provisto por el presente trabajo. A futuro, se podrían realizar comparaciones que consideren otras posibilidades del vasto universo de modelos de *machine learning* y métodos de selección de características para reunir más evidencia que soporte o refute la hipótesis aquí planteada.

REFERENCIAS

[1] Valentim Realinho, Jorge Machado, Luís Baptistaand Mónica V. Martins, "Predict students' dropout and academic success"(dataset). Zenodo, Dec. 13, 2021. doi: 10.5281/zenodo.5777340.

[2] M.V.Martins, D. Tolledo, J. Machado, L. M.T. Baptista, V.Realinho. (2021) "Early prediction of student's performance in higher education: a case study" Trends and Applications in Information Systems and Technologies, vol.1, in Advances in Intelligent Systems and Computing series. Springer. DOI: 10.1007/978-3-030-72657-7_16

[3] Información sobre las variables https://storage.googleapis.com/kaggle-forum-message-attachments/1832313/17922/Features%20information.pdf

[4] Peter Flach.(2012). Captitulo 11 "Model ensembles" en "MACHINE LEARNING, The Art and Science of Algorithms that Make Sense of Data"(pp. 330- 342)

[5] Jim. (2022). How to handle Imbalanced Data? R - Bloggers. https://www.r-bloggers.com/2022/08/how-to-handle-imbalanced-data/

[6] Selva Prabhakaran. (2018). Feature Selection – Ten Effective Techniques with Examples. machinelearning+. https://www.machinelearningplus.com/machine-learning/feature-selection/