

PREDICCIÓN DEL ÉXITO ACADÉMICO EN EDUCACIÓN SUPERIOR UTILIZANDO ÁRBOLES DE DECISIÓN

Miguel Ángel Sarmiento Aguiar
Universidad Eafit
Colombia
msarmie4@eafit.edu.co

Marlon Pérez Ríos
Universidad Eafit
Colombia
mperezr@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

RESUMEN

El objetivo de este informe es mostrar una predicción al puntaje de los estudiantes de pregrado en las universidades colombianas en las pruebas saber pro y mejorar lo que se podría sacar en ellas, ya que esto servirá para aconsejar a los jóvenes antes de tomar la decisión de escoger una carrera, y así salgan más preparados para presentarla y ayuden al desarrollo del país aumentando el nivel académico de este a nivel mundial.

Se busca poder predecir el posible éxito de la población estudiantil en algún programa de educación superior, y con esta predicción, aconsejar respecto a una decisión posiblemente más acertada.

Palabras clave

- Estructuras de datos.
- Árboles de decisión.
- Predicción éxito académico.

Palabras clave de la clasificación de la ACM

- Modelos de computación.
- Diseño y análisis de algoritmos.
- Sistemas de gestión de datos.
- Aprendizaje automático.
- Gráficos de computadora.

[1]

1. INTRODUCCIÓN

En la actualidad el nivel de las pruebas saber pro en la población de jóvenes estudiantes de pregrado es baja, con respecto a otras universidades del mundo, debido a que la mayoría al finalizar la educación secundaria no sabe en qué área poseen mejores competencias, por esto toman una decisión con mayor probabilidad de ser errónea respecto a la probabilidad de éxito. Casi en la totalidad de los casos dicha decisión se toma basándose en el gusto por alguna profesión o por el deseo de una gran remuneración monetaria.

Por esto se pretende mostrar la metodología por la cual se va a desarrollar un algoritmo basado en árboles de decisión, con el fin de predecir el posible éxito de los jóvenes en las diferentes carreras universitarias, reflejándose en las pruebas saber pro, todo esto en base a la información de variables sociodemográficas y de las pruebas saber 11

suministrada por el gobierno colombiano en los últimos años.

2. PROBLEMA

El problema a resolver es la creación de un algoritmo basado en árboles de decisión y en los datos suministrados por el ICFES, mencionados anteriormente. La solución a este problema podrá predecir si el puntaje en las pruebas Saber Pro estará o no por encima del promedio, teniendo en cuenta que los datos del ICFES no solo incluyen variables académicas como el puntaje sino también variables sociodemográficas como género, edad, información de los padres, estrato, entre muchas otras.

Así, se pretende disminuir el nivel de deserción de los estudiantes de pregrado y aumentar los puntajes en la prueba saber pro, guiándolos por la opción en la que tienen mayor probabilidad de éxito, así se puede ahorrar una gran cantidad de recursos tanto para ellos como para el gobierno, añadiendo que con dicha predicción se puede aumentar el interés y las ganas por el estudio, incrementando el número de posibles profesionales en el país.

3. TRABAJOS RELACIONADOS

3.1 Algoritmo ID3

Se basa en la búsqueda de hipótesis dado un conjunto de ejemplos de datos, estos deben estar conformados por una serie de listas ordenadas de valores, cada valor es denominado atributo, uno de estos es el atributo a clasificar (objetivo), el cual es de tipo binario.

La elección del mejor atributo se establece mediante la entropía, eligiendo aquel que proporcione una mejor ganancia de información. Para buscar la hipótesis se utiliza un árbol de decisión que mediante nuevas instancias dirá si el ejemplo dado va a ser positivo o negativo.

Sus elementos son:

- Nodos: contienen los atributos.
- Arcos: contienen valores posibles del nodo padre.
- Hojas: nodos que clasifican el ejemplo como positivo o negativo.

Como ejemplo se propone predecir si es buena idea jugar o no tenis dependiendo de atributos como temperatura, humedad, entre otros:

| Estado | Humedad | Viento | Juego tenis |
|---------|---------|--------|-------------|
| ? | Alta | Leve | No |
| Soleado | Alta | Fuerte | No |
| Nublado | Alta | Leve | Si |
| Lluvia | Alta | Leve | Si |
| Lluvia | Normal | Leve | Si |
| Lluvia | Normal | Fuerte | No |
| Nublado | Normal | Fuerte | Si |
| Soleado | Alta | Leve | No |
| Soleado | Normal | Leve | Si |
| Lluvia | Normal | Leve | Si |
| Soleado | Normal | Fuerte | Si |
| Nublado | Alta | Fuerte | Si |
| Nublado | Normal | Leve | Si |
| Lluvia | Alta | Fuerte | Si |

Figura 1 - Ejemplo ID3

El árbol de decisión sería:

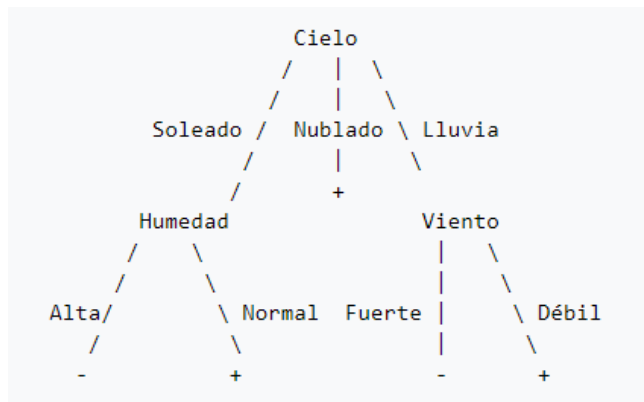


Figura 2 - Árbol de decisión ID3 [2]

3.2 Algoritmo C4.5

Este algoritmo construye árboles de decisión desde un grupo de datos de entrenamiento usando el concepto de entropía de información, donde dichos datos son ejemplos ya clasificados. Cada ejemplo es un vector con los atributos o características de este. Los datos de entrenamiento son aumentados con otro vector cuyos componentes representan la clase a la que pertenece cada muestra.

En cada nodo del árbol elige un atributo de los datos que más eficazmente dividen el conjunto de muestras en subconjuntos enriquecidos en una clase, su criterio es el normalizado para ganancia de información que resulta en la elección de un atributo para dividir los datos. El atributo con la mayor ganancia de información normalizada se elige como parámetro de decisión. [3]

Como ejemplo se toma el mismo caso del algoritmo ID3, donde la distribución de datos para el atributo Estado es:

| | Desconocido | Soleado | Nublado | Lluvia |
|---------|-------------|---------|---------|--------|
| No | 1 | 2 | 0 | 1 |
| Si | 0 | 2 | 4 | 4 |
| Totales | 1 | 4 | 4 | 5 |

Figura 3 - Ejemplo C4.5

Tomando la división de los datos para el valor Nublado, Lluvia y Soleado del atributo Estado:

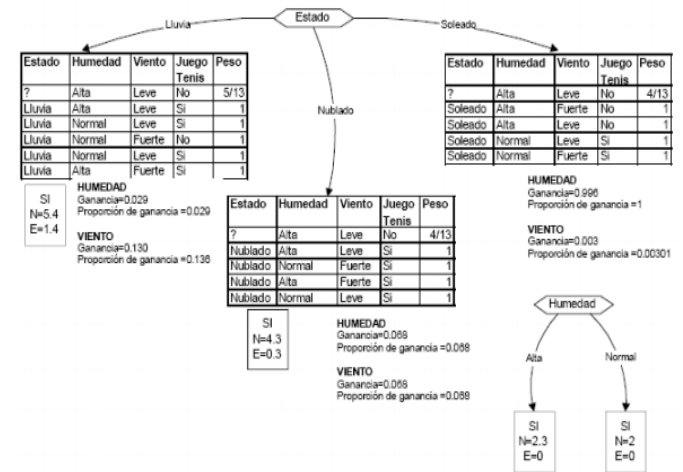


Figura 4 - Árbol de decisión C4.5 [4]

3.3 Algoritmo Random Forest

Este algoritmo consiste en un gran número de árboles de decisión individuales que operan como un conjunto, para así mejorar el rendimiento de la predicción. Cada árbol individual genera un tipo de predicción y el que tenga mejor desempeño es el que va a ser el modelo de predicción de todo el Random Forest.

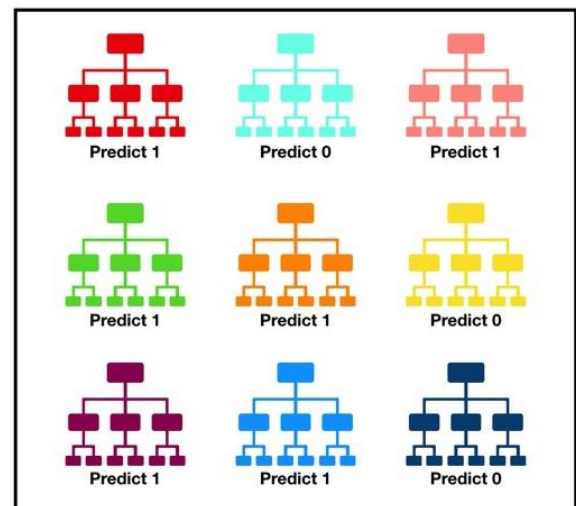


Figura 5 - Algoritmo Random Forest

El aspecto principal del Random Forest es el conocimiento de cada parte individual. Cada árbol individual no es tan potente como cuando se juntan todos los árboles, aún sin estar relacionados o con poca relación, todos se complementan, y si hay algún error en alguno de los individuales, los otros lo pueden corregir, y este es una de las principales partes claves de este algoritmo. Esto es como una empresa que está dividida por diferentes áreas, que muchas no están muy relacionadas entre sí, pero cuando todas se juntan crean una empresa que da grandes resultados.

Algunos prerrequisitos para que funcione bien son:

- 1- Que haya una señal real para que las predicciones no sean aleatorias sino acertadas.
- 2- Las predicciones hechas por los árboles individuales deben tener bajas correlaciones entre sí. [5]

3.4 Algoritmo CART

Es una técnica de aprendizaje de árbol de decisión no paramétrica que produce árboles de clasificación o regresión, dependiendo de si la variable dependiente es categórica o numérica, respectivamente. La palabra binario implica que un nodo en un árbol de decisión solo puede dividirse en dos grupos. CART utiliza el índice de Gini como medida de impureza para seleccionar el atributo. El atributo con la mayor reducción de impurezas se utiliza para dividir los registros del nodo. CART acepta datos con valores numéricos o categóricos y también maneja valores de atributos faltantes. Utiliza la poda de complejidad de costos y también genera árboles de regresión.

El análisis de árboles de clasificación y regresión (CART) generalmente consiste en tres pasos:

1. Construcción del árbol máximo.
2. Poda del árbol.
3. Selección del árbol óptimo mediante un procedimiento de validación cruzada.

Como ejemplo se supone el árbol y los datos en la Figura 6, donde se quiere determinar un conjunto de reglas que indiquen si un conductor vive o no en los suburbios.

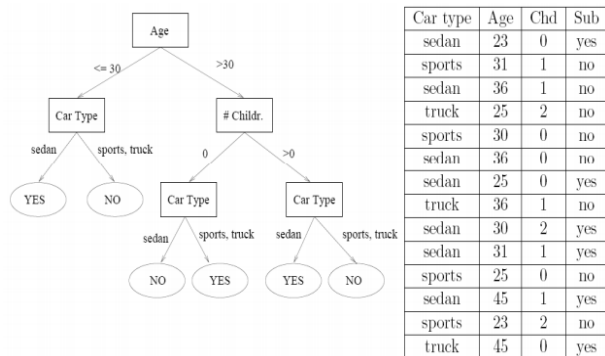


Figura 6 - Ejemplo CART

Se concluye:

- Si $Age \leq 30$ y CarType = Sedan entonces Si
- Si $Age \leq 30$ y CarType = truck/Sports entonces No
- Si $Age > 30$, Children = 0 y CarType = Sedan entonces No
- Si $Age > 30$, Children = 0 y CarType = truck/Sports entonces Si
- Si $Age > 30$, Children > 0 y CarType = Sedan entonces Si
- Si $Age > 30$, Children > 0 y CarType = truck/Sports entonces No

[6]

4. Estructura de datos arreglo de arreglos

Se decide crear una estructura de datos que sea un arreglo de arreglos, donde cada estudiante es un arreglo que contiene las variables de estudio como trabajo de la madre, tipo de piso, colegio, estrato, entre otros.

| | estu_conse cutivo.1 | estu_exter ior | periodo | ... | exito |
|--------------|------------------------|-------------------|---------|-----|-------|
| Estudiante 1 | 1-1 | 1-2 | 1-3 | ... | 1-m |
| Estudiante 2 | 2-1 | 2-2 | 2-3 | ... | 2-m |
| Estudiante 3 | 3-1 | 3-2 | 3-3 | ... | 3-m |
| ... | ... | ... | ... | ... | ... |
| Estudiante n | n-1 | n-2 | n-3 | ... | n-m |

Figura 7 - Estructura de datos

4.1 Operaciones de la estructura de datos

Para solucionar el problema eficientemente, primero se decide crear una estructura de datos que contenga toda la información de la base de datos como texto, después se divide dicha estructura en arreglos, cada uno de los cuales es una fila de la base, es decir, cada fila es un estudiante. Luego se procede a dividir cada una de estas filas en arreglos, cada uno de los cuales es uno de los atributos de cada estudiante.

La operación de llenado de la estructura de datos se puede resumir en la siguiente figura:

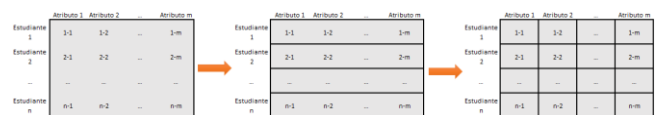
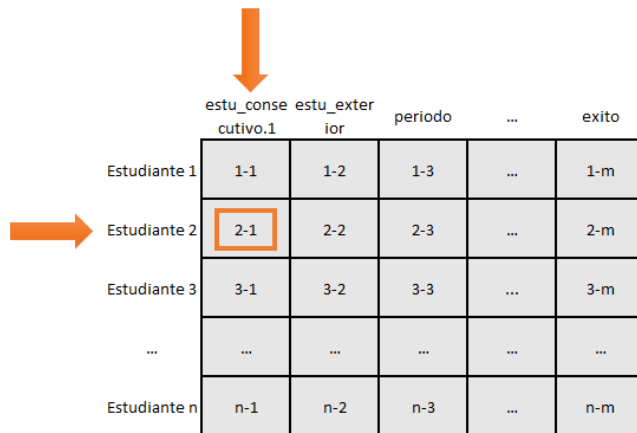


Figura 8 - Llenar estructura de datos

La otra operación de la estructura de datos es acceder a un elemento de ella, lo cual, por ser un arreglo de arreglos, no implica recorrer el número de filas o el número de columnas completamente, como se muestra en la siguiente figura:



| | estu_consecutivo.1 | estu_exterior | periodo | ... | exito |
|--------------|--------------------|---------------|---------|-----|-------|
| Estudiante 1 | 1-1 | 1-2 | 1-3 | ... | 1-m |
| Estudiante 2 | 2-1 | 2-2 | 2-3 | ... | 2-m |
| Estudiante 3 | 3-1 | 3-2 | 3-3 | ... | 3-m |
| ... | ... | ... | ... | ... | ... |
| Estudiante n | n-1 | n-2 | n-3 | ... | n-m |

Figura 9 - Acceder a estructura de datos

4.2 Criterios de diseño de la estructura de datos

La estructura de datos se diseñó para reducir el tiempo de ejecución y la memoria utilizada al ejecutar el código en Python, ya que se utilizaron operaciones evitando el aumento de la complejidad de la variable n, que corresponde al número de filas (estudiantes). Esto es importante porque dicha variable tiene un valor considerable entre 15000 (archivo más pequeño) y 135000 (archivo más grande), por lo cual puede aumentar significativamente la memoria y el tiempo de ejecución si se multiplica por ella misma.

De esta forma, en el peor de los casos, el algoritmo tiene una complejidad $O(n*m)$, donde el valor de m (78) no es tan significativo, comparándolo con n, que como ya se dijo, representa el número de filas del arreglo.

4.3 Análisis de Complejidad

Según el algoritmo obtenido, el cálculo de la complejidad para el llenado de la estructura de datos en el peor de los casos queda de la siguiente manera, donde n es el número de filas y m el número de columnas:

$$T(n, m) = C1 + C2 + C3 + C4*n + C5*n + C6*n*m + C7 + C8$$

$$T(n, m) = O(C1 + C2 + C3 + C4*n + C5*n + C6*n*m + C7 + C8)$$

$$T(n, m) = O(n + n*m)$$

$$T(n, m) = O(n*m)$$

Adicionalmente, por ser un arreglo de arreglos, la operación acceder tiene complejidad $O(1)$.

| MÉTODO | COMPLEJIDAD |
|---------|-------------|
| Llenar | $O(n*m)$ |
| Acceder | $O(1)$ |

Tabla 1 - Complejidad de la estructura de datos

4.4 Tiempos de Ejecución

| CONJUNTO DE DATOS | NÚMERO DE FILAS | TIEMPO DE EJECUCIÓN [segundos] |
|-------------------|-----------------|--------------------------------|
| Train 0 | 15000 | 0.153814634000355 |
| Train 1 | 45000 | 0.4339690319998226 |
| Train 5 | 57765 | 0.5300589880002917 |
| Train 2 | 75000 | 0.8745048579999093 |
| Train 3 | 105000 | 1.3179701559997739 |
| Train 4 | 135000 | 2.362833066999883 |

Tabla 2 - Tiempos de ejecución

4.5 Memoria

| CONJUNTO DE DATOS | NÚMERO DE FILAS | CONSUMO DE MEMORIA [MB] |
|-------------------|-----------------|-------------------------|
| Train 0 | 15000 | 61.44 |
| Train 1 | 45000 | 184.32 |
| Train 5 | 57765 | 245.76 |
| Train 2 | 75000 | 317.44 |
| Train 3 | 105000 | 440.32 |
| Train 4 | 135000 | 563.2 |

Tabla 3 - Consumo de memoria

4.6 Análisis de los resultados

El resumen de datos calculados se puede observar en la siguiente tabla:

| CONJUNTO DE DATOS | NÚMERO DE FILAS | TIEMPO DE EJECUCIÓN [segundos] | CONSUMO DE MEMORIA [MB] |
|-------------------|-----------------|--------------------------------|-------------------------|
| Train 0 | 15000 | 0.153814634000355 | 61.44 |
| Train 1 | 45000 | 0.4339690319998226 | 184.32 |
| Train 5 | 57765 | 0.5300589880002917 | 245.76 |
| Train 2 | 75000 | 0.8745048579999093 | 317.44 |
| Train 3 | 105000 | 1.3179701559997739 | 440.32 |
| Train 4 | 135000 | 2.362833066999883 | 563.2 |

Tabla 4 - Resumen datos calculados

Adicionalmente, se puede observar el comportamiento de estas dos variables calculadas según el número de filas ingresadas, como se muestra en la siguiente figura:

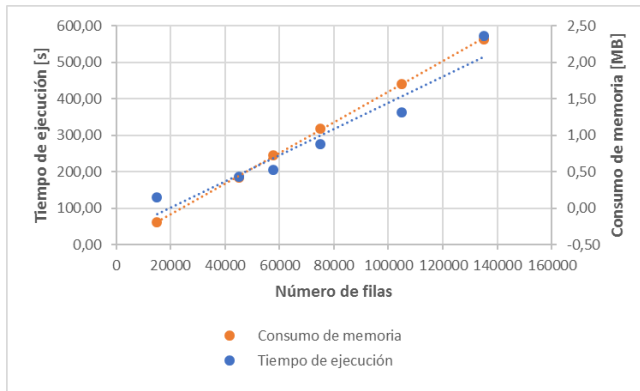


Figura 10 – Resumen comportamiento

REFERENCIAS

1. ACM. (2012). ACM Computing Classification System. [online] Available at: <https://dl.acm.org/ccs>
2. Wikipedia. (2019). Algoritmo ID3. [online] Available at: https://es.wikipedia.org/wiki/Algoritmo_ID3
3. Wikipedia. (2020). C4.5. [online] Available at: <https://es.wikipedia.org/wiki/C4.5>
4. López, B. (2005). Algoritmo C4.5. [ebook] Available at: [http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/C4.5/C4.5\(2005-II-B\).pdf](http://www.itnuevolaredo.edu.mx/takeyas/Apuntes/Inteligencia%20Artificial/Apuntes/tareas_alumnos/C4.5/C4.5(2005-II-B).pdf)
5. Yiu, T. (2019). Understanding Random Forest. [online] Towards data science. Available at: <https://towardsdatascience.com/understanding-random-forest-58381e0602d2>
6. Serna, S. (2009). Comparación de árboles de regresión y clasificación y regresión logística. Maestría. Universidad Nacional de Colombia.