Paper – Proyecto final sobre árboles de decisión

Juan Manuel Muñoz (202010100010), Juan Miguel Castro (202010077010) y Juan Manuel Zapata (202010091010).

Resumen.

Este trabajo tiene como tema principal los árboles de decisión, enfocándose en el análisis de los resultados de las pruebas Saber. El objetivo es identificar los factores que más influyen en un estudiante a la hora de presentar sus pruebas Saber pro mediante los árboles de decisión. La pregunta problematizadora del escrito es: ¿qué factores influyen en el resultado de las pruebas Saber pro en base a los resultados de las pruebas Saber 11?

Abstract.

This paper has as its main theme the decision trees, focusing on the analysis of the results of the Saber tests. The objective is to identify the factors that most influence a student when presenting their Saber pro tests through decision trees. The problematizing question of the paper is: what factors influence the result of the Saber pro tests based on the results of the Saber 11 tests?

Palabras clave:

Machine Learning, árboles de decisión, bosques aleatorios, Gini, Icfes, impureza.

Introducción

Uno de los momentos más importantes de un estudiante que está cursando la educación superior es cuando debe de hacer frente a las pruebas Saber pro. Estas pueden ser incluso más importantes que las pruebas Saber que se presentan en grado 11. Pero, ¿hay alguna forma de predecir si un estudiante tendrá éxito en las pruebas saber Pro?

Estas cuestiones han llevado al equipo de trabajo, conformado por Juan Manuel Muñoz, Juan Miguel Castro Y Juan Manuel Zapata, a recurrir al emergente mundo del Machine Learning para así conocer cuáles son los factores que más pueden influir en el resultado de las pruebas saber pro en base a sus resultados en la prueba ICFES de grado 11. El grupo hará uso de uno de los algoritmos de Machine Learning más conocidos de todos, el árbol de decisión.

¿Por qué escoger un árbol de decisión?

Se ha escogido la implementación de los árboles de decisión debido a su fácil interpretación a la hora de leer sus resultados y la rapidez con que trabaja a comparación de otros algoritmos de predicción. Pero no existe un solo tipo árbol de decisión, existen distintos tipos de árboles que se rigen sobre distintos criterios y formas de operar. Dos de sus más grandes exponentes son los algoritmos CART y los algoritmos C4.5, la principal diferencia entre estas dos formas de crear árboles es su criterio de separación. Mientras que CART se rige en el índice de Gini, el algoritmo de C4.5 se rige por la ganancia de información y entropía (Orellana Alvear, 2018).

Dejando de lado los algoritmos, es preciso hablar de los dos tipos de árboles, los de regresión y clasificación. Mientras que los de clasificación permiten predecir valores de verdadero o falso

(conocidos como valores discretos) y sus variables son categorías, los de regresión predicen valores lineales, como los precios de un hogar, y sus variables son continuas (Sitiobigdata, 2019).

Todo el grupo se ha decantado por implementar el algoritmo CART de clasificación. No se tomó en cuenta el algoritmo CART de regresión ya que su objetivo es el de producir valores continuos o lineales.

Como en este caso, se debe predecir si un estudiante aprobará o no aprobará la prueba Saber Pro en base a sus puntajes previos y factores socio económicos, se puede ver que lo que se busca predecir son valores discretos, éxito o no éxito. Motivo que apoya la decisión de implementar una clasificación.

¿Qué es el índice de Gini y el porqué de su uso?

[EI] El índice de Gini es una medida que surge con el objetivo de medir la desigualdad en los ingresos de la población. Fue ideado por el italiano Corrado Gini y es ahora uno de los criterios más usados en los árboles de decisión de Machine Learning (Coeficiente de Gini, s.f).

Este índice consiste en un número que va entre el 0 y el 1. Si el número se acerca a 0, significa que la información es pura. Mientras que, si el número se acerca a 1, significa que la información es impura.

El grupo ha escogido utilizar el algoritmo CART ya que sus separaciones se rigen sobre el índice de Gini, pues funciona muy bien a la hora de predecir valores discretos (verdadero o falso), también porque este índice solamente funciona con separaciones binarias.

¿Cómo calcular el índice Gini?

Es algo complejo de explicar, por lo que primero se verán las fórmulas a aplicar y luego se expondrá un ejemplo.

Cuando te encuentras en un nodo, lo primero que se debe de calcular es el porcentaje de fracaso que se generaría en los nodos hijos al separar los datos en base a un atributo. Por ejemplo, ver cuánto porcentaje de fracaso generarían los nodos hijos si la información se separará en base a la pregunta "Puntaje de inglés menor a 50". Para obtener este porcentaje, se aplica la fórmula (n datos fracaso / n total de datos).

Después de obtener el porcentaje de fracaso, se debe buscar la impureza de cada nodo hijo, la fórmula para obtener este índice es *(p^2+q^2)* en donde *p* significa el porcentaje éxito, mientras que *q* significa el porcentaje de fracaso. Si por ejemplo se cuentan con 10 datos, en donde 7 poseen un resultado positivo y 3 poseen un resultado negativo, se dividen por el número de datos ambos valores, obteniendo *7/10 = 0.7* y *3/10 = 0.3*. Se agregan a la fórmula, quedando así: *(0.7^2 + 0.3^2)*.

Este proceso se debe hacer tanto en el nodo izquierdo como en el derecho. Una vez se tengan las impurezas de los dos nodos hijos, se debe aplicar la siguiente fórmula para ahora si calcular el índice Gini del nodo padre:

*G = 1 - ((n datos der * impureza derecha + n datos izq *impureza izquierda) / (n total de datos))*

Ahora, un ejemplo. Se tiene un salón de 30 personas, y se quiere predecir qué personas, en base a ciertos criterios, juegan cricket. Como nodo padre se tiene el conjunto de los 30 estudiantes, y se desea ver qué atributo separa de mejor forma a las personas que juegan cricket de las que no lo hacen.

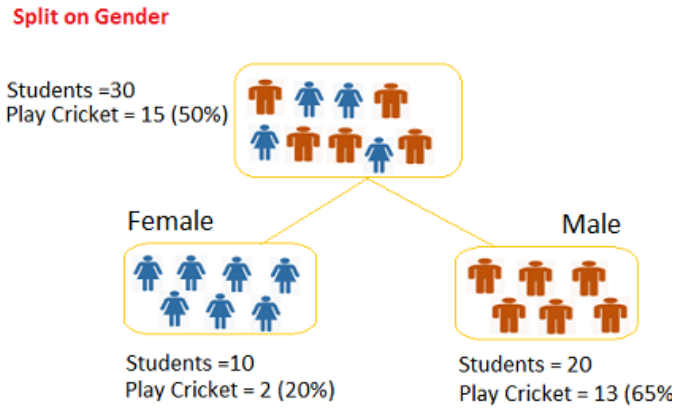


Figura 1.

El primer criterio que se utiliza es el de separar el grupo por género, en el nodo derecho se encuentran los hombres, y en el izquierdo las mujeres. En el nodo derecho se encuentran 20 personas, de las cuales juegan 13. Por lo que el porcentaje de personas que juegan es *13/20*, por lo que la variable p será igual a *0.65*, mientras que la variable q es del porcentaje restante, el 0.35. Se busca la impureza del nodo derecho *con (0.65^2 + 0.35^2)* dando como resultado *0.55*.

Ahora se repite el proceso en el nodo izquierdo, en el cual hay 10 personas, de las cuales 2 juegan cricket. El porcentaje de personas que juegan es *2/10*, que es igual a *0.2*. El porcentaje de fracaso es igual a 0.8. Se busca la impureza del nodo izquierdo con *(0.2^2+0.8^2)* dando como resultado *0.68*.

Ahora es momento de encontrar la impureza del nodo padre aplicando la fórmula, quedando lo siguiente: *1 - ((10*0.68 + 20*0.55) /30)* dando como resultado una impureza del *0.4*.

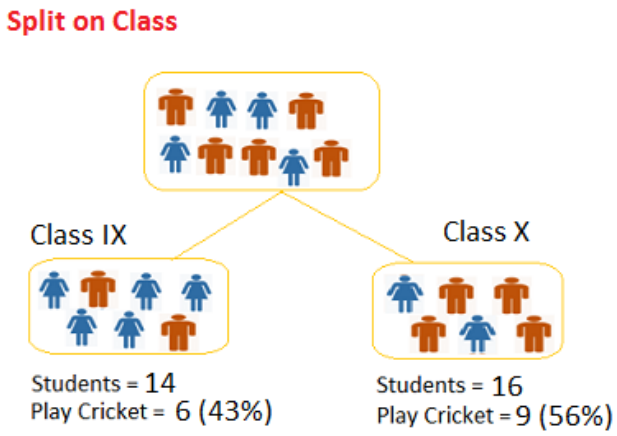


Figura 2.

Una vez obtenida la impureza de Gini en base al género de los estudiantes, se propone ahora separarlos por el criterio del grado que están cursando.

Se tiene el nodo padre con los 30 estudiantes, en el nodo hijo de la derecha se encuentran los estudiantes del grado X y en la izquierda los estudiantes del grado IX. En el nodo derecho hay 16 estudiantes de los que 9 juegan cricket, por lo que el porcentaje de jugadores es *9/16* y el porcentaje de no jugadores es *7/16*. Se tiene entonces que *p* es igual a *0.56* y *q* a *0.44*. Aplicando la fórmula de impureza se tiene *(0.56^2+0.44^2)* es igual a *0.51*.

Se repite con el nodo izquierdo en donde se encuentran 14 personas de las cuales 6 juegan, por lo que el porcentaje de jugadores es del *6/14* y el de no jugadores es del *8/14*. El valor de *p* es *0.43* y de *q* es de *0.57*, reemplazando en la fórmula se obtiene *(0.43^2+0.57^2)* es igual también a *0.51*.

Con las dos impurezas de los nodos hijos ya obtenidas, es solo cuestión de reemplazar en la fórmula principal. *1-((14*0.51+16*0.51) /30)* dando como resultado una impureza del *0.49.*

Ahora que se han obtenido los dos índices de Gini, en base al género (*0.4*) y al grado (*0.49*), se puede concluir que la mejor separación es la que se basa en el género ya que es la que genera una impureza de Gini más cercana a 0. Y es así, que se realizan las separaciones en el árbol de decisión CART por medio del índice de Gini.

Construcción del árbol.

Ya explicado el funcionamiento del índice de Gini, que es la parte más importante de todas, es necesario ver la implementación del árbol de decisiones, comenzando con la lectura y procesamiento de los datos.

Lectura de datos.

Se han realizado tres tipos de lectura y procesamiento de datos. En el primer tipo de lectura solamente se tienen en cuenta los puntajes de cada estudiante. Por ejemplo, puntaje de inglés, de español, matemáticas, etc. Para este tipo de lectura no es necesario realizar limpieza de datos vacíos ya que en los puntajes no se encuentra ningún vacío.

En el segundo tipo de lectura, solamente se leen los datos socioeconómicos del estudiante. Por ejemplo, si este tiene nevera, microondas, internet, el salario de sus padres, el nivel de educación de los padres, etc. Para esta lectura sí que es necesario hacer barrido de las filas que contengan datos vacíos, ya que un estudiante pudo haber dejado en blanco esos espacios.

En el tercer tipo de lectura, se leen tanto los puntajes como los datos socioeconómicos de cada estudiante. También es necesario barrer los datos vacíos ya que se está tratando con datos socioeconómicos.

Los nodos del árbol.

Debido a que se están tratando árboles, es necesario crear nodos. Como se debe manejar una mayor información, los nodos tienen una mayor cantidad de atributos, los cuales son:

- **Pregunta:** se tiene un atributo en el cual se debe almacenar la pregunta de la cual dependerá la partición del árbol.

- **Gini:** este atributo almacena el valor del índice de Gini que va desde el 0 al 1.
- **Número de elementos:** Es un contador que indica la cantidad de datos que tiene el nodo.
- **Respuesta:** debido a que la construcción del árbol de forma que a los nodos izquierdos van los valores de falso y a la derecha los valores de verdadero, este atributo indica si el nodo es verdadero o falso. Es necesario este atributo a la hora de recorrer el árbol.
- **Número del nodo:** es una especie de identificador para el nodo, es diferente para cada nodo. Este es supremamente necesario a la hora de la exportación y visualización del árbol.
- **Nodo izquierdo:** apuntador al nodo hijo izquierdo.
- **Nodo derecho:** apuntador al nodo hijo derecho.

La clase pregunta.

Para la pregunta, se tiene una clase en donde se almacena el atributo y el valor o respuesta al que se debe parecer. Por ejemplo, si se almacena un atributo de puntaje de matemáticas, la pregunta que se va a crear debe ser menor o igual a un determinado número *(<=)*, mientras que si la categoría es si el estudiante tiene internet, la pregunta debe ser igual a una respuesta *(==)*.

La clase de árbol de decisión.

Ya cimentados los nodos, se puede crear el árbol. El árbol recibe en su constructor dos atributos, la profundidad y la cantidad máxima de nodos. Estos dos atributos permiten ajustar el crecimiento del árbol para evitar que caiga en el sobreajuste y sea poco efectivo. El atributo profundidad limita el número de niveles hasta dónde crecerá, mientras que el otro atributo limita el número máximo de nodos que se crearan.

Para crear el árbol, se llama a la función **create tree**. Se debe de entrar por parámetros el nombre del archivo csv y el árbol procesará los datos. Una vez ingresados los datos, el árbol internamente realizará el proceso de separación de los datos, evaluará cada pregunta y su índice Gini, con absolutamente todos los atributos. Se aplica la misma idea explicada en el apartado de cómo calcular el índice Gini, pero a una mayor escala. Una vez obtenido el menor índice Gini, se usa la pregunta que lo acompaña y se parten los datos en base a dicha pregunta, a la izquierda se van todos los datos que retornen falsa la pregunta y a la derecha los que la retornen verdadera.

Ya con los datos de la izquierda y derecha separados, el índice Gini y la pregunta, se miran las condiciones de parada. Si el nodo es nulo, significa que se está en la raíz y se crea el nodo raíz con el índice Gini, la pregunta y el número de datos que pasan por dicho nodo. En la siguiente condición de parada se mira si la pregunta es nula. Si es así, se puede considerar a este nodo como un nodo vacío, ya que no hay más datos a los cuales realizarle un análisis. Se llega a la última condición de parada, se verifica si el índice Gini es 0 o si ya se alcanzó la profundidad máxima o la máxima cantidad de nodos, si se cumple alguna de las tres condiciones, significa que se ha llegado al final y se retorna el nodo.

Si no se entra en ninguna condición de parada, se aplica la teoría de los árboles binarios. Se toman los datos partidos de la izquierda y se llama al nodo hijo de la izquierda usando nuevamente la función **create tree** recursivamente, pero ahora entrando por parámetros a los datos partidos de la izquierda. Se realiza igualmente lo mismo con los datos de la derecha.

Una vez terminado este proceso recursivo que, nuevamente se menciona, es la misma aplicación de los árboles binarios, el árbol estará creado. Ahora es solamente hacer las pruebas que se requieran con él, como evaluar su precisión, exportarlo, visualizarlo, etc.

Visualización del árbol.

Ya con el árbol creado, se puede llamar a la función **export tree** que creará un archivo en formato *.dot*. Dentro del archivo se creará un pequeño programa que es capaz de leer la herramienta de modelado Graphviz. Para esto es necesario el atributo **número del nodo** ya que, para hacer las relaciones entre nodos en Graphviz, es necesario que cada nodo tenga un identificador único o el árbol será imposible de graficar. Para convertir el archivo exportado en una imagen, existen dos formas. La primera es tomar el código de Graphviz y pegarlo en una página web. La segunda opción es, una vez ya creado el archivo, usar funciones de librerías externas como Pydot plus que permiten la gestión de archivos con esta extensión y exportar el archivo en png, jpg o pdf.

Recorrer el árbol.

Para recorrer el árbol, se toman los datos de un solo estudiante y, en base a las preguntas de cada nodo, se avanza al nodo izquierdo o derecho de un nodo padre hasta llegar a un nodo raíz que es un nodo que no tiene hijos y se retorna el atributo **Respuesta** de dicho nodo raíz.

Precisión del árbol.

Para medir la precisión del árbol, se debe tomar un conjunto distinto de datos a los que se usaron para crear el árbol. De ahí se escogen todas las variables y se dejan por fuera las variables objetivo, por medio de la lectura de datos. Se realiza el proceso descrito para recorrer el árbol, pero ahora no con un solo estudiante, sino con todos los datos disponibles. Se creará una lista donde irá guardando los resultados de predicción que retorne el recorrido del árbol en base a cada estudiante.

Realizado este proceso, debe quedar una lista de valores booleanos de igual longitud a la de los valores objetivo que fueron mencionados al principio. Lo siguiente es comparar ambas listas y ver cuántas veces coinciden sus resultados. Ese número de veces se divide por la cantidad total de datos y se multiplica por 100, así obteniendo el porcentaje de aciertos del árbol.

Matriz de confusión.

Para visualizar la matriz de confusión del árbol, se toman, como en la precisión del árbol, los datos de predicción creados al recorrer el árbol y los datos objetivo. Se declaran cuatro variables que serán contadores, falsos positivos (cuando el árbol arroja un resultado falso y acierta), falsos negativos (cuando arroja un resultado falso y falla), verdaderos positivos (cuando arroja un resultado verdadero y acierta) y verdaderos negativos (cuando arroja un resultado verdadero y falla).

Se comparan los resultados de predicción con los objetivos, si el resultado de predicción da verdadero y el objetivo da verdadero, se incrementa en 1 la variable verdaderos positivos, si la predicción da falso y el objetivo también da falso, se incrementa la variable falsos positivos, si predice falso y el

objetivo arroja verdadero, se incrementa verdaderos y negativos, y si no ocurre ninguna de las anteriores, se incrementa falsos negativos.

Una vez se terminen de recorrer las listas, se imprime por pantalla la matriz de confusión en el siguiente orden.

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 3.

Bosques aleatorios.

Los bosques aleatorios, en pocas palabras, son una basta colección de árboles de decisión en donde a la hora de tomar una decisión se miran los resultados que arroja cada árbol y el resultado que más se repita será el que retorne el bloque aleatorio.

A la hora de la creación de los árboles, el bosque de decisión toma aleatoriamente ciertas variables que contenga el set de datos y en base a estas variables aleatorias crea cada árbol. Por lo que siempre se tendrá un bosque distinto e imposible de replicar. El gran problema de los bosques aleatorios es su gran carga computacional que los vuelve demasiado pesados, pero son de los algoritmos más efectivos de todos.

El bosque, en vez de tener una raíz como los árboles, tiene una lista en donde se almacenarán los árboles. En su constructor recibe tres atributos, la cantidad de árboles que se quiere crear, la profundidad que tendrán y la cantidad de atributos aleatorios que se desea que cada árbol tenga.

Construcción del bosque.

Para construir el bosque, se llama la función **create forest** y se ingresa por parámetros el archivo csv del cual se crearán los árboles. Dependiendo de la cantidad de árboles que se desean crear, los datos se partirán esa cantidad de veces, de forma de que queden uniformes. Por ejemplo, si se tiene un conjunto de 1000 datos y se desean crear 25 árboles, se harán conjuntos de datos distintos con una longitud de 40 cada uno.

Luego de que estén fragmentados los datos, se mira la variable que indica cuantos atributos debe manejar cada árbol, indicado en el constructor de la clase. En base a este número, se parte cada conjunto de datos en esa cantidad de atributos, pero escogidos de una forma aleatoria. Se repite este proceso con todos los datos que fueron fragmentados y en base a estos se crean los árboles de decisión.

Precisión del bosque.

Se realiza el mismo proceso que el explicado previamente con los árboles de decisión, la diferencia radica en que se crean dos variables, el número de votos de falso y el número de votos de verdadero, también se crea una lista en donde se almacenan las predicciones.

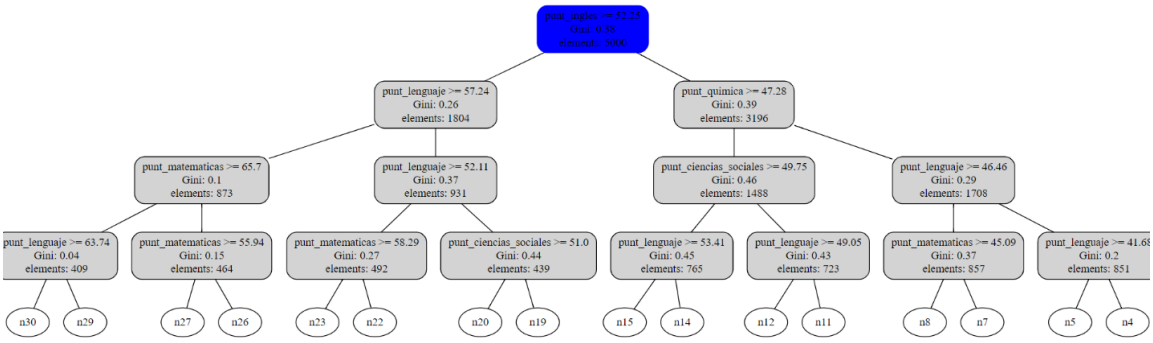
Se ingresan por parámetros los datos donde están los atributos y se evalúan todos los árboles. Si un árbol retorna un resultado verdadero, se incrementan los votos de verdadero, de lo contrario incrementan los votos de false. Una vez recorridos todos los árboles, se mira qué variable tiene más votos y dependiendo de quién tiene más votos, se agrega verdadero o falso en la lista de predicciones. Se repite este proceso hasta que termine la lista de los atributos y se retorna la lista con las predicciones.

Para evaluar el modelo, se realiza exactamente igual que con los árboles. Se toma la lista de predicción y se compara con la variable objetivo, y de ahí se obtiene el resultado.

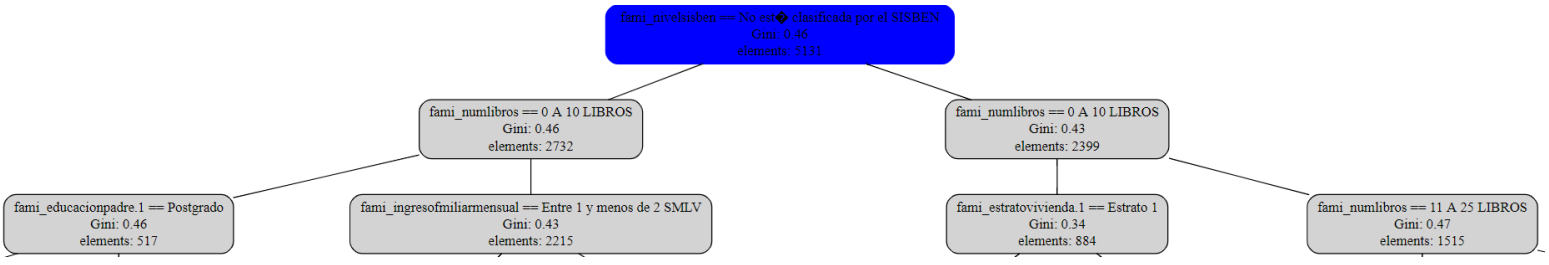
Estudio de los datos de los ICFES.

Ya explicados los árboles de decisión y brevemente los bosques aleatorios, se puede hablar de los resultados obtenidos en el estudio de las pruebas del ICFES.

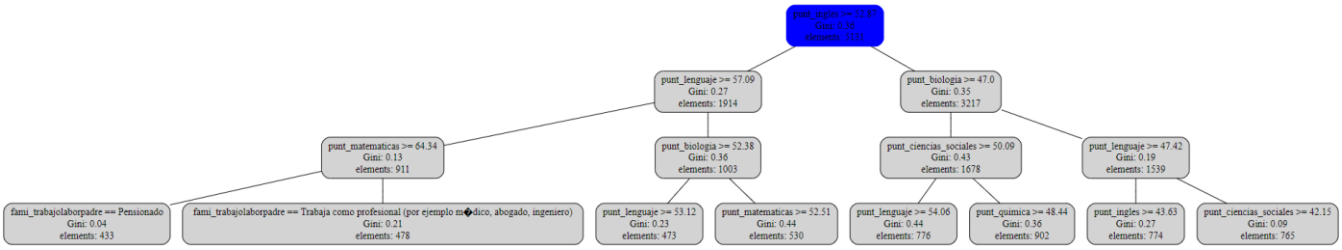
Para comenzar, se leyeron los datos de las tres formas posibles, tomando puntajes, tomando información socioeconómica y tomando tanto la información socioeconómica como los puntajes. Posteriormente se crearon los árboles y se exportaron, obteniendo lo siguiente:



Árbol creado con los puntajes



Árbol creado con los datos socioeconómicos



Árbol creado con los datos socioeconómicos y los puntajes.

Para comprender los árboles y ver qué factores influyen más, se miran los primeros nodos del árbol. Entre más cerca estén de la raíz, más importantes serán los atributos. Si es la misma raíz, el atributo de la raíz será más importante y más influyente en el resultado.

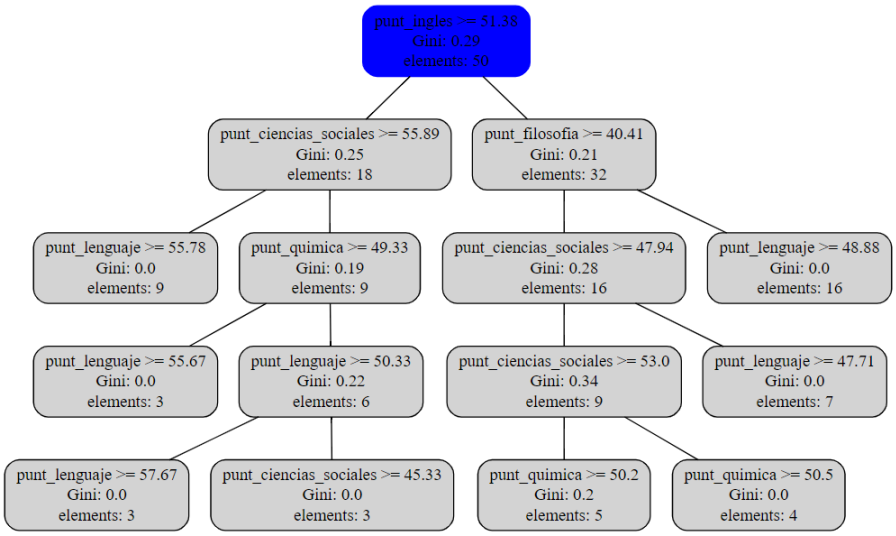
En el primer árbol, en donde se evalúan solamente los puntajes, se puede ver que en la raíz aparece el puntaje de inglés, por lo que se puede concluir que el atributo más influyente en cuestión de puntajes es la prueba de inglés. De ahí le siguen la prueba de lenguaje, que aparece varias veces, la prueba de química, de matemáticas y ciencias sociales. Por lo que se puede determinar que estas son las pruebas más importantes de todas y que más influyen, sobresaliendo entre todas estas la prueba de inglés y lenguaje.

En el segundo árbol, en donde se evalúa la socioeconomía, se puede ver que el atributo de la raíz es referente al Sisbén familiar, por lo que es el atributo más importante. De ahí le sigue la cantidad de libros que se encuentran en la casa. Si se mira el resto del árbol, se observa que todas las preguntas están relacionadas con los ingresos, el trabajo de los padres y el estrato de la vivienda. Es así que se puede determinar que estos atributos, socioeconómicamente hablando, son los más influyentes sobre un estudiante que presenta las pruebas Saber.

En el tercer árbol, en donde se evalúa la socioeconomía y los puntajes, el árbol presenta un comportamiento similar al del primer árbol. El elemento de la raíz sigue siendo igualmente la prueba de inglés, adelante continúa apareciendo la prueba de lenguaje y, por primera vez, aparece la prueba de biología.

De lo anterior se puede concluir que los factores más influyentes sobre el resultado son los puntajes, en donde destaca la prueba de inglés y lenguaje, y se deja de un lado totalmente a los aspectos socioeconómicos.

Árbol con mejor precisión.



Después de mucho tiempo de análisis de datos, el mejor árbol que fue posible crear fue este. El cual posee una precisión del 77% y se creó solamente en base a los puntajes. Se sigue observando que el elemento más importante es la prueba de inglés, aunque no se quedan atrás las pruebas de ciencias sociales y filosofía.

Conclusiones.

En lo referente a los árboles, se pueden concluir varias cosas. Es muy importante crear árboles, por decirlo así, “flexibles” para evitar que se caiga en el sobreajuste y se genere un modelo con baja precisión. Otro tema de alta importancia a la hora de crear el árbol es haber limpiado correctamente los datos vacíos para que el modelo no presente fallas. El factor que más influye a la hora de crear el árbol es escoger correctamente los atributos variables que se analizarán para predecir el modelo, si se escogen atributos innecesarios, se creará un modelo pobre.

En el apartado del manejo de árboles hay varias cosas que se pudieron hallar. Los árboles con baja profundidad, aproximadamente de 5 hasta 10, son los modelos que mejor predicen los resultados. Los árboles con una profundidad mayor o menor a este rango presentan resultados demasiado pobres. Aparte de que ayuda a la precisión, hace que los modelos sean más livianos computacionalmente, evitando que se desborde la memoria y se pierda tiempo valioso.

También se encontró que es más beneficioso para el modelo trabajar con conjuntos de datos pequeños (Entre 300 a 1500 datos) que con conjuntos de datos excesivamente grandes (10000 datos en adelante), ayuda a que el modelo sea flexible y versátil, y evita así el temido sobreajuste.

En el estudio de los resultados se pueden realizar las siguientes conclusiones. Los datos socioeconómicos generan más impureza que los puntajes, por lo que el puntaje es el factor más influyente en el resultado de un estudiante. Los atributos que hacen la diferencia son la prueba de inglés y la prueba de lenguaje. El puntaje de inglés es el atributo más importante e influyente de todos los atributos posibles.

Referencias.

Coeficiente de Gini. (Sin fecha). En Wikipedia. Recuperado el 2 de diciembre de 2020 de https://es.wikipedia.org/wiki/Coeficiente_de_Gini

Orellana Alvear, J. (2018). Árboles de decisión y Random Forest [en línea]. Recuperado de https://bookdown.org/content/2031/arboles-de-decision-parte-i.html

Sitiobigdata. (2019). Árbol de decisión en Machine Learning (Parte 1) [en línea]. Recuperado de https://sitiobigdata.com/2019/12/14/arbol-de-decision-en-machine-learning-parte-1/

Figuras.

[1, 2] Orellana Alvear, J. (2018). Árboles de decisión y Random Forest [Figura]. Recuperado de https://bookdown.org/content/2031/arboles-de-decision-parte-i.html

[3] Zelada, C. (2017). Evaluación de modelos de clasificación [Figura]. Recuperado de https://rpubs.com/chzelada/275494