


Inspira Crea Transforma

PREDICCIÓN DEL ÉXITO ACADÉMICO EN EDUCACIÓN SUPERIOR UTILIZANDO ÁRBOLES DE DECISIÓN

Marlon Pérez Ríos
Miguel Ángel Sarmiento Aguiar

DISEÑO ESTRUCTURA DE DATOS

	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



	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

	Atributo 1	Atributo 2	...	Atributo m
Estudiante 1	1-1	1-2	...	1-m
Estudiante 2	2-1	2-2	...	2-m
...
Estudiante n	n-1	n-2	...	n-m



	Atributo 1	Atributo 2	...	Atributo m
Estudiante 1	1-1	1-2	...	1-m
Estudiante 2	2-1	2-2	...	2-m
...
Estudiante n	n-1	n-2	...	n-m

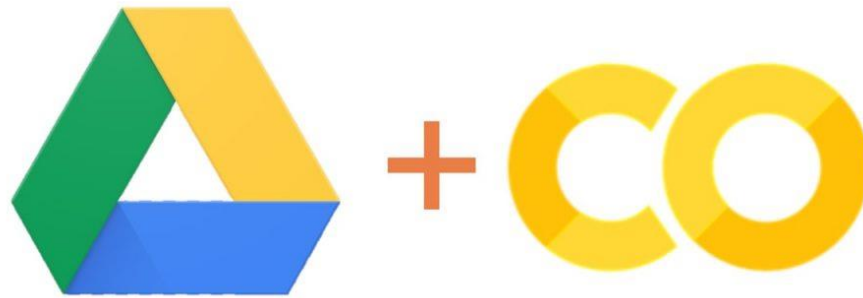


	Atributo 1	Atributo 2	...	Atributo m
Estudiante 1	1-1	1-2	...	1-m
Estudiante 2	2-1	2-2	...	2-m
...
Estudiante n	n-1	n-2	...	n-m

DISEÑO ESTRUCTURA DE DATOS

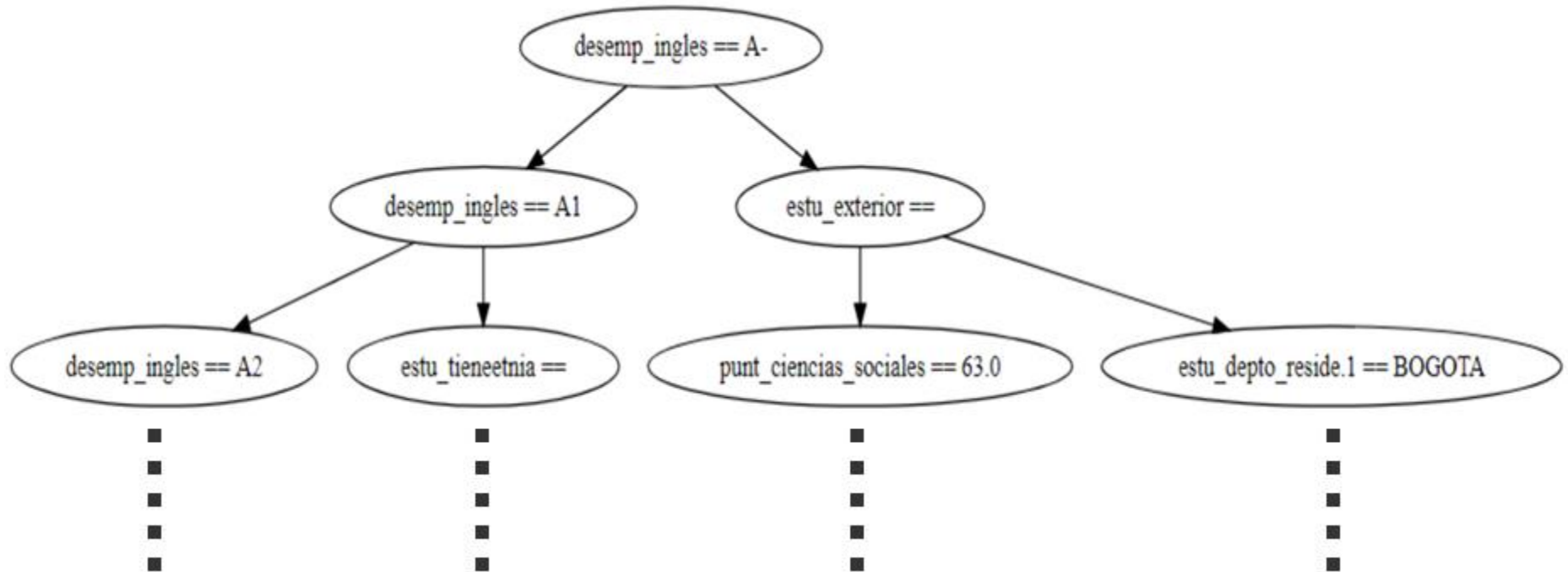
1. Se crea estructura de datos Train
2. Se crea estructura de datos Test
3. Se crea el árbol binario
4. Se crea código para dibujar árbol en WebGraphviz
5. Se calcula éxito real
6. Se calcula pronóstico de éxito mediante probabilidad
7. Se calcula error entre valor real y pronóstico
8. Se imprimen resultados

CÓDIGO



https://colab.research.google.com/drive/1eblzVhNQ_UbezrSfg4Hfxz_DT2hYF79K#scrollTo=5XRS9TEGTcLM

CÓDIGO



OPERACIONES Y COMPLEJIDAD

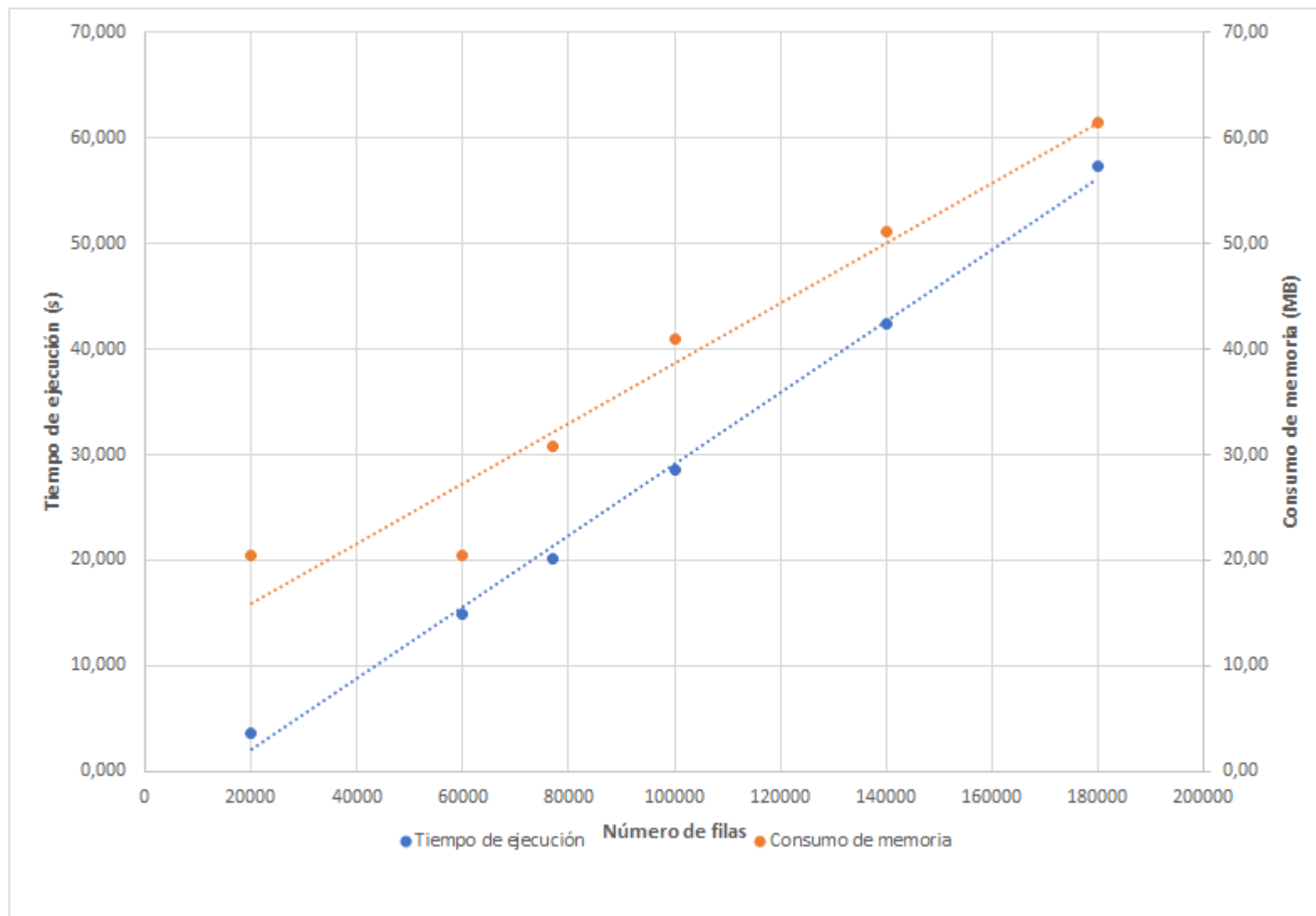
FUNCIÓN	COMPLEJIDAD
Crear_Estructura_Datos(Direccion_csv)	$O(n*m)$
Dividir_Matriz_En_Dos(Matriz, Pos_Variable, Valor)	$O(n)$
Calcular_Valores_Variable(Matriz, Pos_Variable)	$O(n*\log(n))$
Calcular_Mejor_Variable(Matriz)	$O(n*m*\log(n))$
Arbol_Binario(Matriz, Nivel_Maximo)	$O(n*m*\log(n)*2^m)$
Calcular_Probabilidad_Exito(Matriz)	$O(n)$
Calcular_Pronostico_Exito(Fila, Arbol)	$O(m)$
Calcular_Exito_Colectivo(Matriz, Arbol)	$O(n*m)$
Calcular_Exito_Individual(Matriz, Arbol)	$O(m)$
main(Train, Test, Nivel_Arbol, Fila)	$O(n*m*\log(n)*2^m)$

¿POR QUÉ LA ESTRUCTURA DE DATOS ES UNA BUENA SOLUCIÓN?

Es una buena solución ya que se obtienen buenos tiempos de ejecución (menores a 1 minuto) trabajando con arreglos de arreglos en Python, además se requiere de poca memoria RAM (61.44MB para el archivo más grande). Por último, los errores calculados entre los datos reales y el pronóstico del algoritmo están en su mayoría por debajo del 10%, trabajando con árbol binario de búsqueda hasta nivel 6.

TIEMPO Y MEMORIA

CONJUNTO DE DATOS	NÚMERO DE FILAS	TIEMPO DE EJECUCIÓN [Segundos]	CONSUMO DE MEMORIA [MB]
Train 0 - Test 0	20000	3,698	20,48
Train 1 - Test 1	60000	14,954	20,48
Train 5 - Test 5	77020	20,137	30,72
Train 2 - Test 2	100000	28,569	40,96
Train 3 - Test 3	140000	42,354	51,20
Train 4 - Test 4	180000	57,242	61,44



¿PREGUNTAS?

