Exámen 2T
IA

Orihuela	Gil	Richard Hector	9064877 LP
PATERNO	MATERNO	NOMBRES	CI

DATASET

cpu.arff

Convertir datos a CSV

Abrimos el archivo con weka y guardamos en formato csv

Datos

MYCT: tiempo de ciclo de la máquina en nanosegundos (entero)
MMIN: memoria principal mínima en kilobytes (entero)
MMAX: memoria principal máxima en kilobytes (entero)
CACH: memoria caché en kilobytes (entero)
CHMIN: canales mínimos en unidades (entero)
CHMAX: máximo de canales en unidades (entero)
clase: rendimiento relativo publicado (entero)

Leer con python

Estamos trabajando con datos supervisados.

$$f(x) = y$$

Data Preprocessing

No faltan valores en las columnas, por lo que *NO* es necesario *IMPUTACION*

HISTOGRAMAS

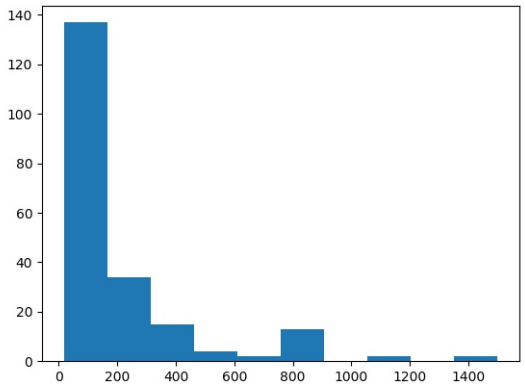

Figura 1: MYCT

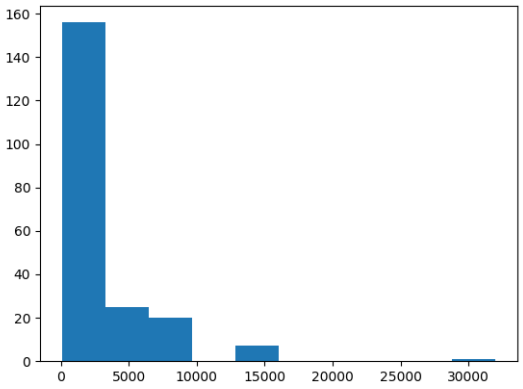

Figura 2: MMIN

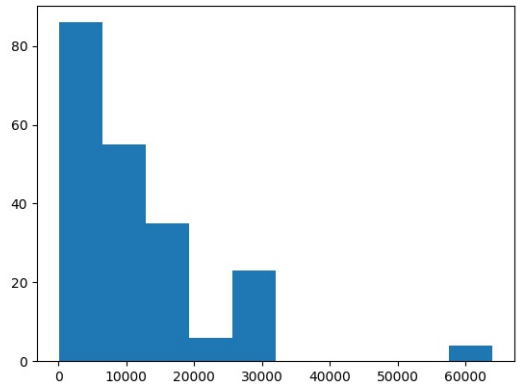

Figura 3: MMAX

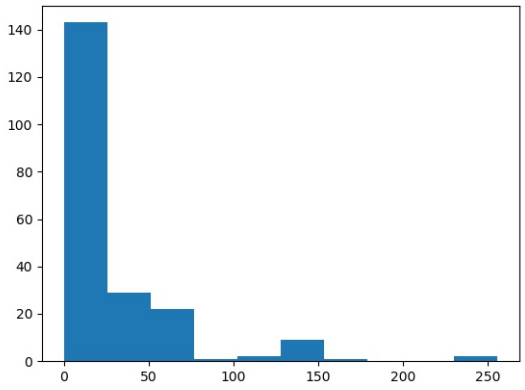

Figura 4: CACH

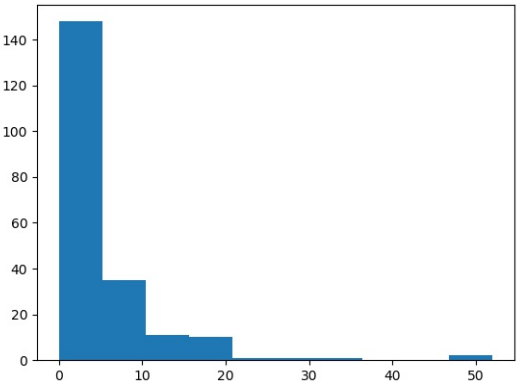

Figura 5: CHMIN

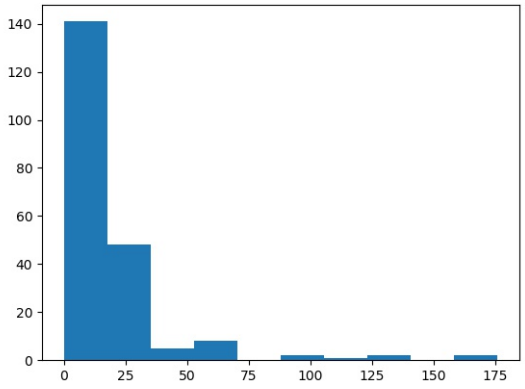

Figura 6: CHMAX

NO UTILIZAMOS DATA REDUCTION por la pequeña cantidad de datos.

LOS DATOS PUEDEN SER USADOS

Árbol de decisión C4.5

Es un árbol de decisión, ampliamente utilizado; la idea subyacente de este tipo de método es realizar particiones al conjunto de datos repetidamente, con la finalidad de que las particiones sean cada vez más puras, es decir, que contengan elementos de una sola clase.
Cada partición o separación de un conjunto de datos (S), se realiza probando todos los posibles valores de las instancias en cada dimensión o atributo(A), y después se selecciona a la mejor partición de acuerdo a algún criterio. Este proceso se realiza de manera recursiva.

Entropía(medida del desorden de un sistema)

$$E(S) = -\sum_{i=1}^{N} p_i \; log(p_i)$$

El C4.5 construye un árbol de decisión mediante el algoritmo "divide y vencerás" y evalúa la información en cada caso utilizando los criterios de Entropía, Ganancia o proporción de ganancia, según sea el caso.

Realice una red neuronal XOR mediante con python simple.

Los pasos básicos involucrados en el entrenamiento de una red neuronal se dan a continuación:
1-Inicialización aleatoria de pesos.
2-Iterando sobre datos completos.
a) Feed Forward: calcula la salida de la red neuronal con los pesos inicializados aleatoriamente y calcula la función de pérdida entre la salida real y la salida prevista.
b) Propagación hacia atrás: calcule la función de gradiente de pérdida con respecto a los pesos y actualice los pesos utilizando el descenso del gradiente.
4-Repite hasta que la pérdida sea mínima.

DataSet

Tabla: XOR

X1	X2	y
0	0	0
1	0	1
0	1	1
1	1	0

Separable no linealmente

Implementación de Python

Implementaremos en Python usando la biblioteca numpy. Para nuestro caso, ejecutaremos el algoritmo para 10000 iteraciones con tres valores diferentes de tasas de aprendizaje 1.

Github: https://github.com/richardorihuela/2TxorPythonSimple