**1. Introducción**

* En este ejercicio trabajamos con un **problema de clasificación binaria**.
* El dataset (Classification\_Data.csv) contiene dos variables de entrada: **x** y **y**, y una variable de salida llamada **label** que puede ser 0 o 1.
* El objetivo es entrenar un modelo que aprenda a separar correctamente los datos en esas dos clases.

**2. Preparación de los datos**

* Se cargó el dataset en un **DataFrame de Pandas** para inspeccionarlo.
* Se dividió en:
  + X = variables de entrada (x y y)
  + y = variable objetivo (label)
* Se aplicó **escalado estándar** (StandardScaler) para que ambas variables tengan la misma magnitud y el modelo entrene mejor.
* Se realizó una **división entrenamiento/prueba** (80% / 20%) usando train\_test\_split.

**3. Definición del modelo**

* Se utilizó un **modelo de red neuronal artificial secuencial** con la librería **TensorFlow / Keras**.
* La arquitectura es la siguiente:
  + Capa de entrada: **64 neuronas, activación ReLU**.
  + Segunda capa: **32 neuronas, ReLU**.
  + Tercera capa: **16 neuronas, ReLU**.
  + Una capa de **Dropout (0.2)** para evitar sobreajuste.
  + Cuarta capa: **8 neuronas, ReLU**.
  + Capa de salida: **1 neurona con activación sigmoide**, que devuelve un valor entre 0 y 1 (probabilidad de pertenecer a la clase 1).
* El modelo se compiló con:
  + Optimizador **Adam** con tasa de aprendizaje 0.001.
  + Función de pérdida **binary\_crossentropy** (adecuada para clasificación binaria).
  + Métrica de evaluación principal: **accuracy**.

**4. Entrenamiento**

* El modelo se entrenó durante **50 épocas** con lotes de 32 ejemplos.
* Durante el proceso, se monitoreó tanto el rendimiento en el conjunto de entrenamiento como en el conjunto de validación.
* El objetivo era que el modelo aprendiera a separar correctamente las dos clases de datos.

**5. Resultados**

* Al evaluar el modelo en el conjunto de prueba, se obtuvo una **accuracy final de aproximadamente XX%** (menciona el valor real que viste en consola).
* Además, se generó una **matriz de confusión** que muestra cuántos elementos de cada clase fueron clasificados correctamente y cuáles se confundieron.
* Finalmente, se generó un **reporte de clasificación** con tres métricas importantes:
  + **Precisión (precision):** mide qué tan confiable es el modelo cuando predice una clase positiva.
  + **Recall (sensibilidad):** mide cuántos de los casos positivos reales logra identificar el modelo.
  + **F1-score:** es el balance entre precisión y recall.

**6. Conclusión**

* Se utilizó una **red neuronal profunda con varias capas densas** para resolver un problema de clasificación binaria.
* El modelo logró una precisión bastante buena, aunque se puede mejorar ajustando hiperparámetros como el número de épocas, neuronas o la tasa de aprendizaje.
* Las métricas de evaluación, junto con la matriz de confusión, muestran cómo de bien el modelo separa las dos clases.