Lo primero realizado fue analizar el artículo del dataset para tener contexto y entender los datos presentados, luego, se realizó la descarga de los datos.



Después, se realizó el análisis para determinar la forma de transformar los datos pensando en obtener el mejor rendimiento al ser procesados por la red neuronal.

Se decidió realizar categorización de datos para convertir variables categóricas en variables numéricas usando el método one-hot, debido a que este tipo de red neuronal requiere entradas numéricas. Normalización de datos numéricos para evitar variables con escalas muy mayores y disminuir el impacto proporcionado en el modelo si se compara con otras variables. Mapeo de variable objetivo para ayudar al modelo a comprender las relaciones entre las clases.

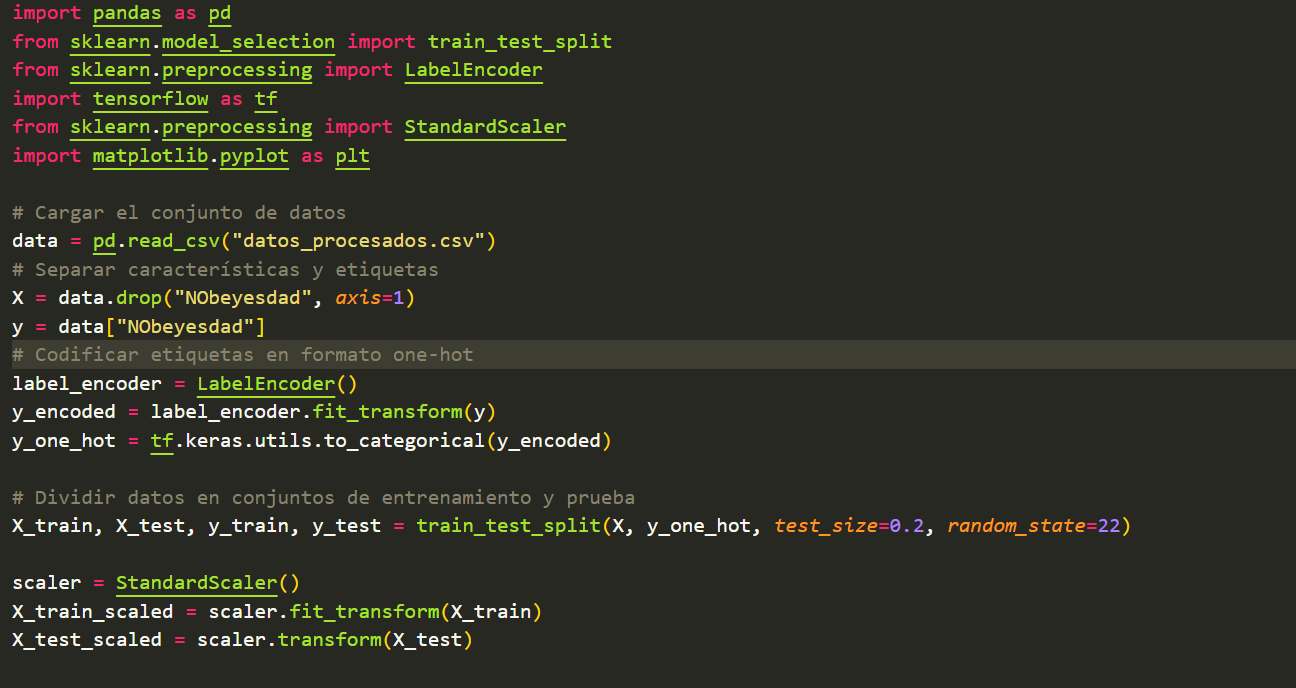


Con este procesamiento de datos, se obtuvo un nuevo dataset llamado “datos\_procesados.csv” el cuál es el que se utilizó para el entrenamiento de la red neuronal.

Para la construcción de la red neuronal se utilizaron las librerías sklearn, tensorflow y para graficar se utilizó matplotlib.

Primero se leen los datos, luego, se dividen los datos de entrada en la variable “X” y los de salida en la variable “Y”.

Se realiza la división del conjunto de entrenamiento y prueba, dejando un 80% de los datos para entrenamiento y el 20% restante para pruebas.



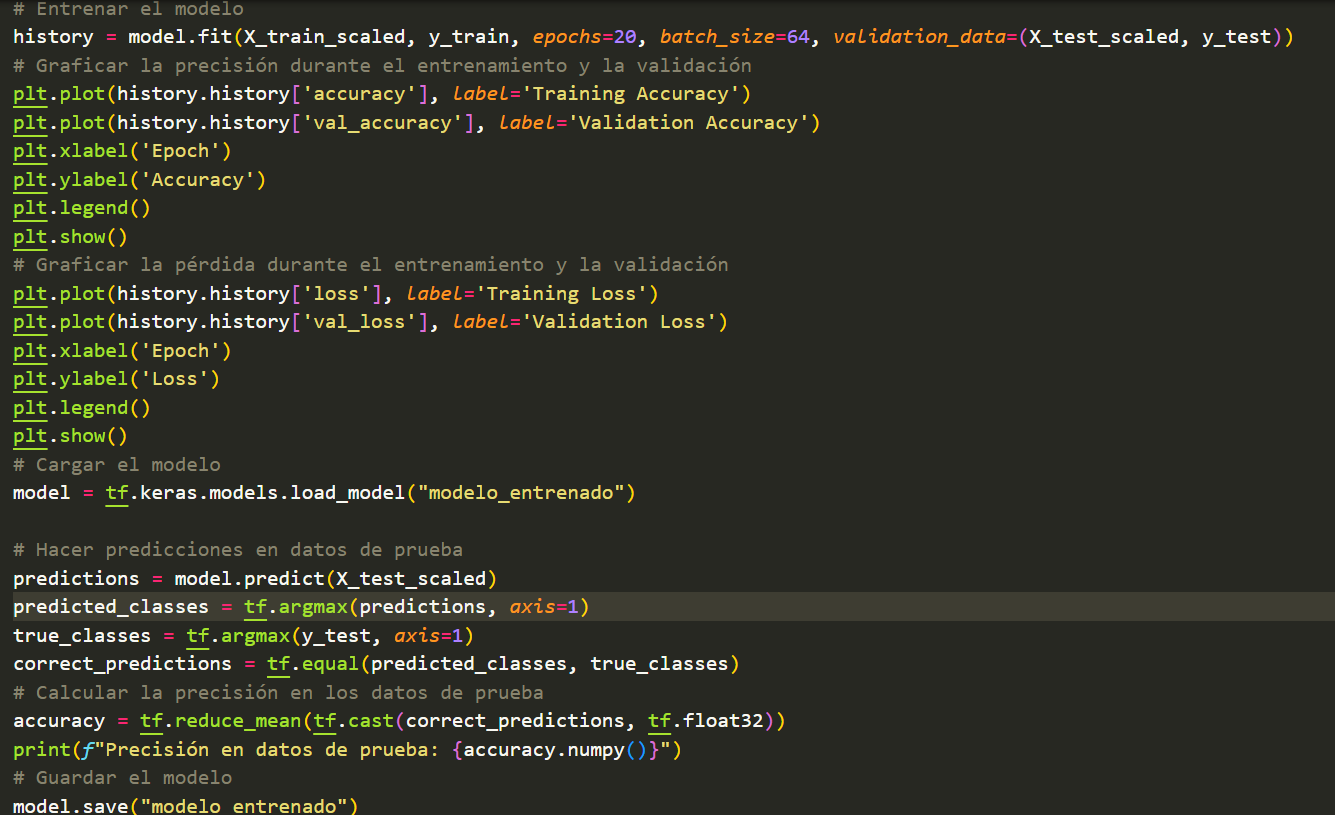
Lo siguiente es la creación del modelo con su arquitectura. Se realizaron varias pruebas con diferentes cantidades de números de neuronas en la capa oculta, también, se probó con las funciones de activación ReLU, Tanh y LeakyReLU. La fusión sigmoide se descartó debido a que no era un problema de clasificación binario.

Texto

Descripción generada automáticamente

Se decide mantener la configuración de 128 neuronas en la capa oculta, función de activación LeakyReLU para la capa inicial y oculta, y par la capa de salida, la función de activación SoftMax debido la mejora en los resultados. La función softmax se determina para la capa de salida porque tiene ventajas a la hora de realizar la clasificación de clases. La tasa de aprendizaje se mantiene en 0.0095 ya que dicho valor ayudó a la generación de los mejores resultados.

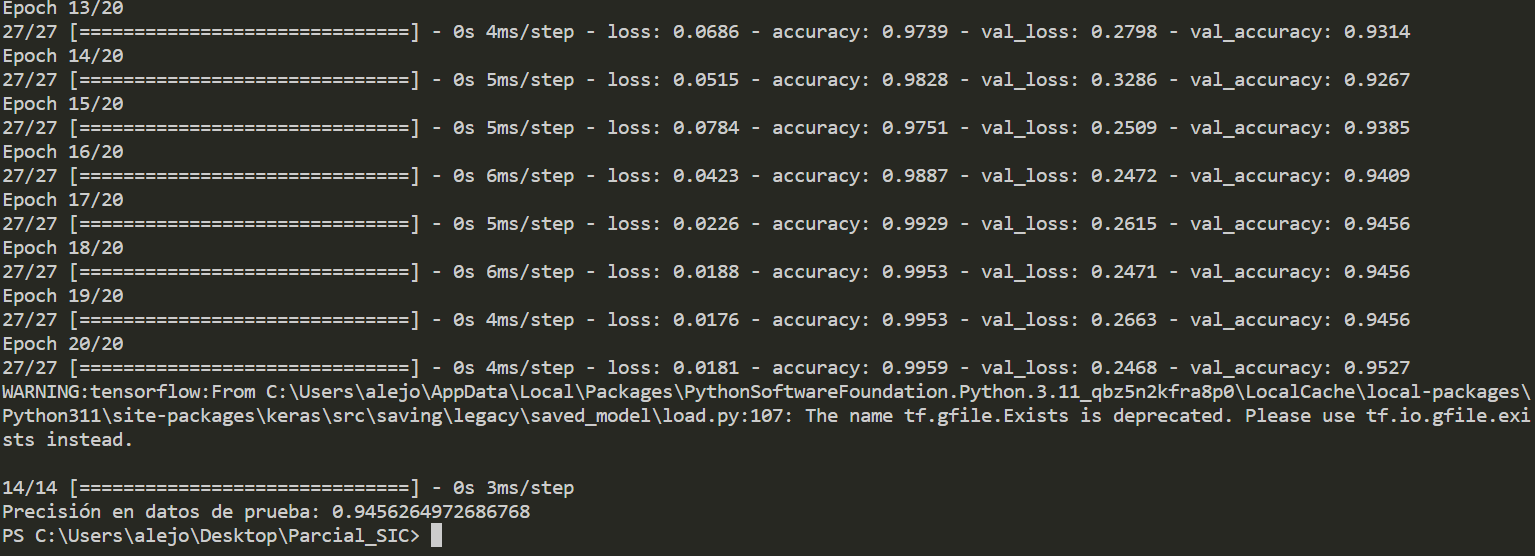
Por último, se realiza el entrenamiento del modelo, se grafica y se pone a prueba.



Se empezó probando con 10 épocas y con una división de lotes de tamaño 32, pero se obtuvo 85% de precisión, esto combinado con la demás configuración de la arquitectura. Luego de varias pruebas modificando dichos valores, el mejor resultado se obtuvo al tener 20 épocas y una divisón de lotes de 64, generando una precisión del 94%.

Al entrenar el modelo con los datos de entrenamiento separados al principio, se obtuvo una precisión también del 94%.

Se guarda el modelo.



Gráfico, Gráfico de líneas

Descripción generada automáticamenteGráfico, Gráfico de líneas

Descripción generada automáticamente