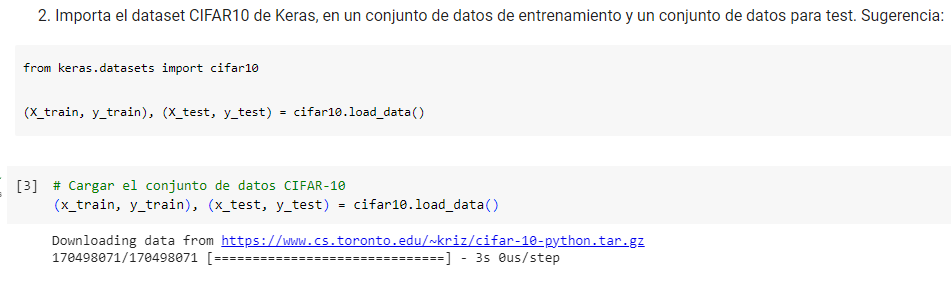
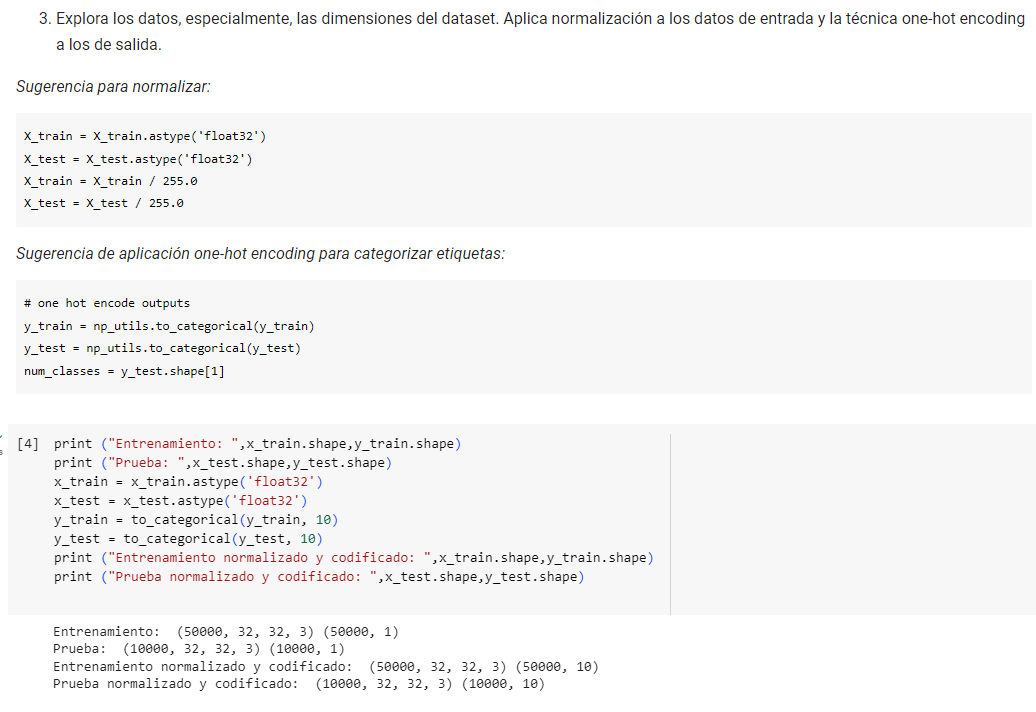


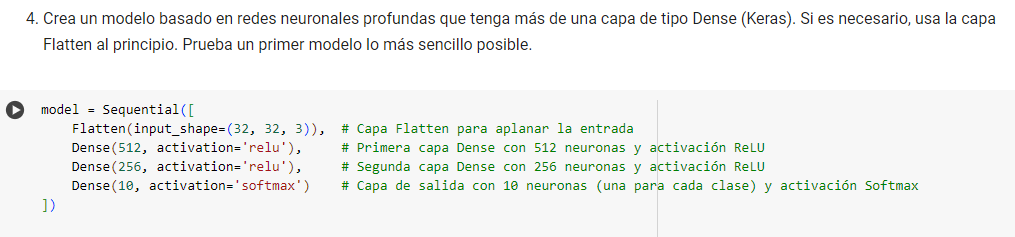
Aquí importo todos los módulos necesarios como numpy, reducido cómo np, el dataset, útiles, modelos y el tipo de red.



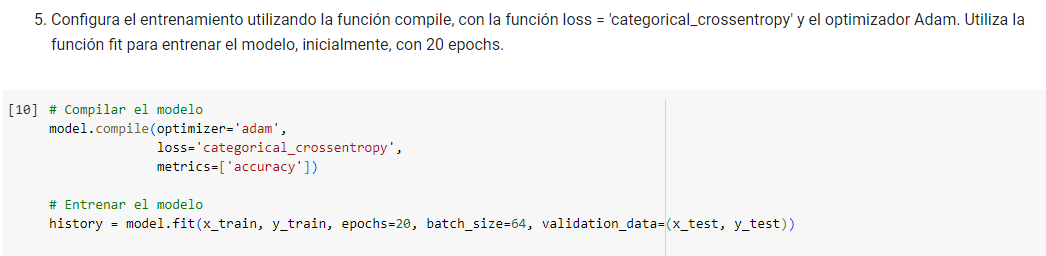
Aquí importo el dataset con los datos de entrenamientos.



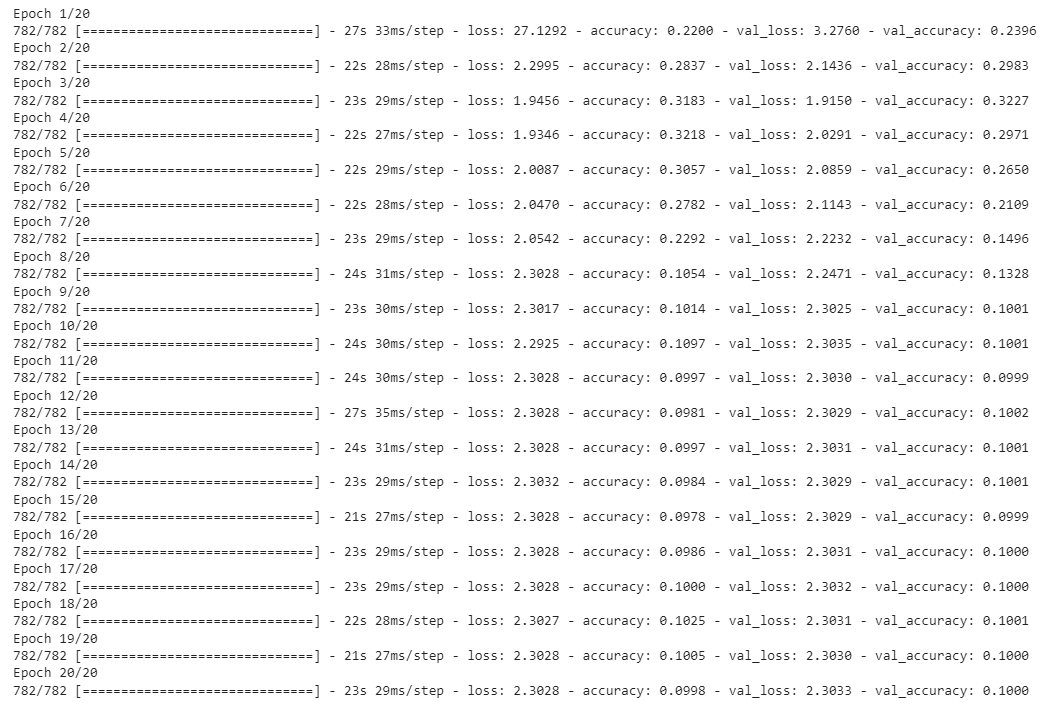
Aquí los datos normales imprimidos de entrenamiento y de prueba, y luego normalizado y codificado con hot encoding.



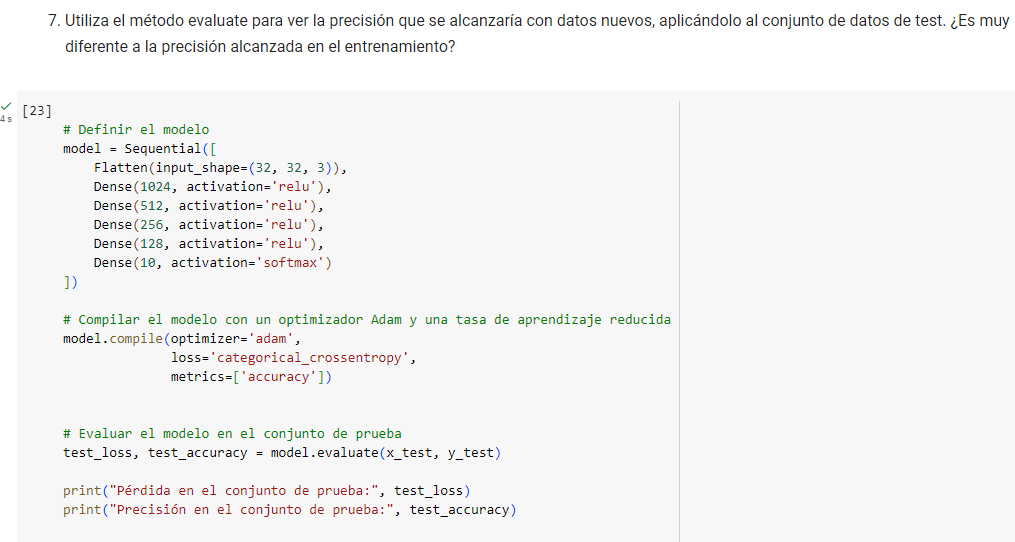
Aquí creo en la entrada una flatten con 32 x 32 pixeles y 3 canales, una capa dense 512 neuronas por ReLu, una segunda capa 256 po ReLu y una última dense con activación softmax.



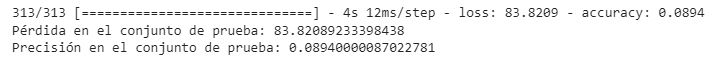
Aquí compilo con optimizador Adam, con la función loss y con accurazy y luego el fit con 20 epochs el tamaño y los datos de validación.



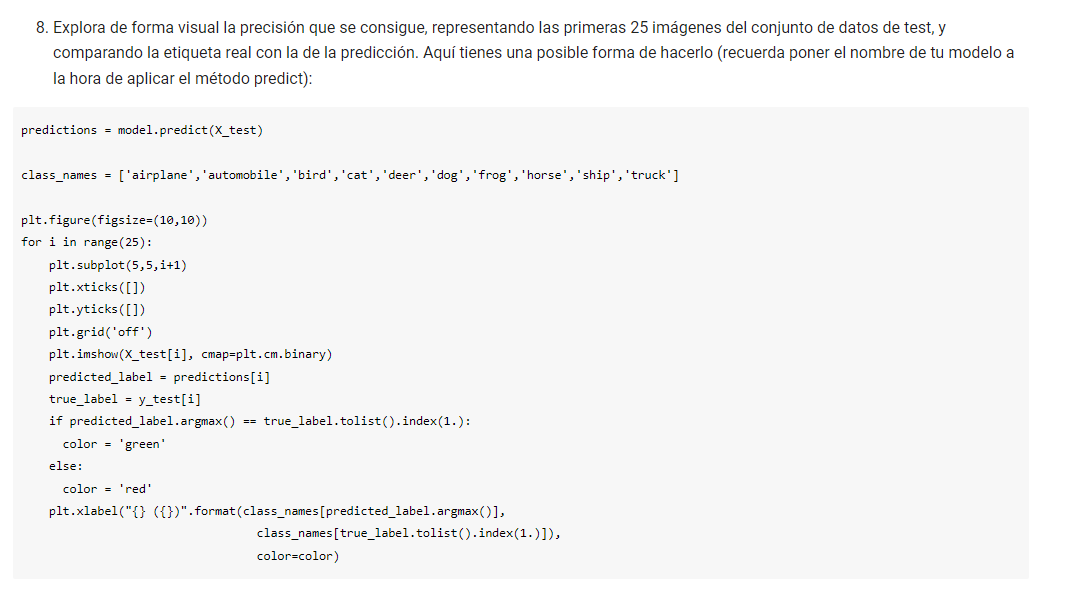
Y los datos con mucha pérdida y una precisión 25%

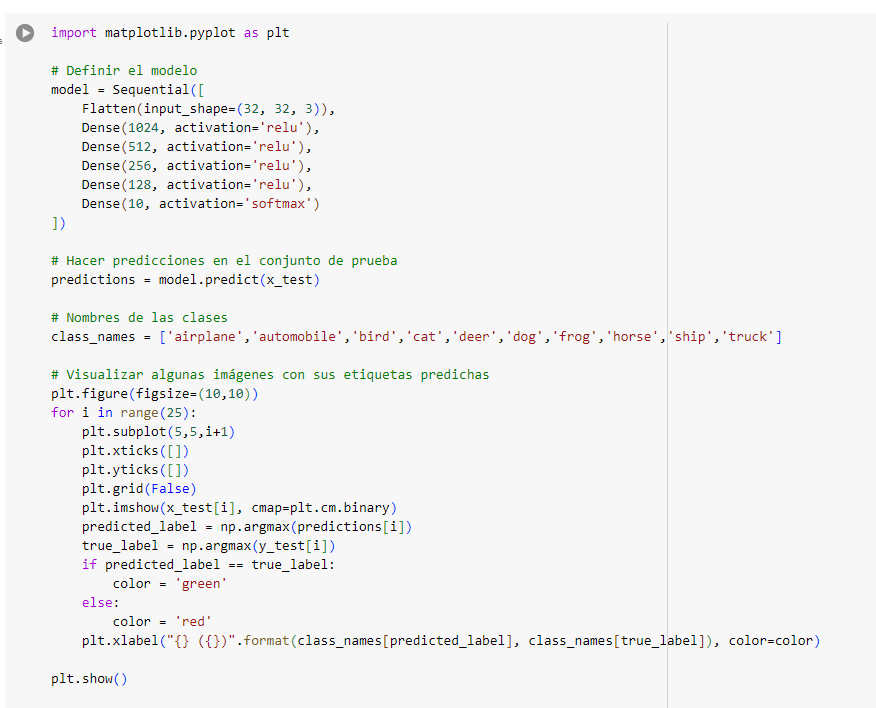


Y luego pongo 2 capas más relu, con el modelo.evaluate que he notado que pierde bastante precisión respecto a model.fit, aunque es muchísimo más rápido



Como se puede ver tiene mucha menos precisión.





Aquí como se puede ver defino el modelo con la capa principal flatten y las demás dense, hacemos predicciones y ponemos nombres a las clases.

Luego predecimos sus etiquetas y predecimos los colores en la que salen el nombre predicho, el nombre real y el color

Aquí se ve la salida del código:

