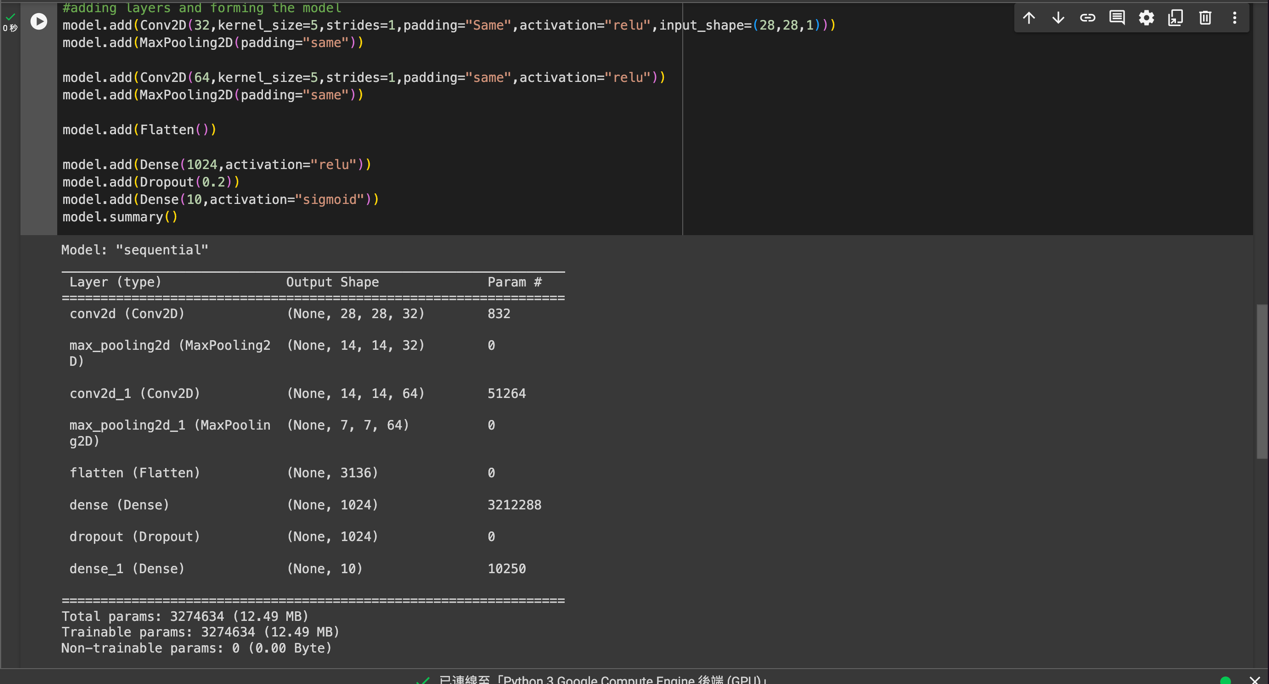
原始資料、數據



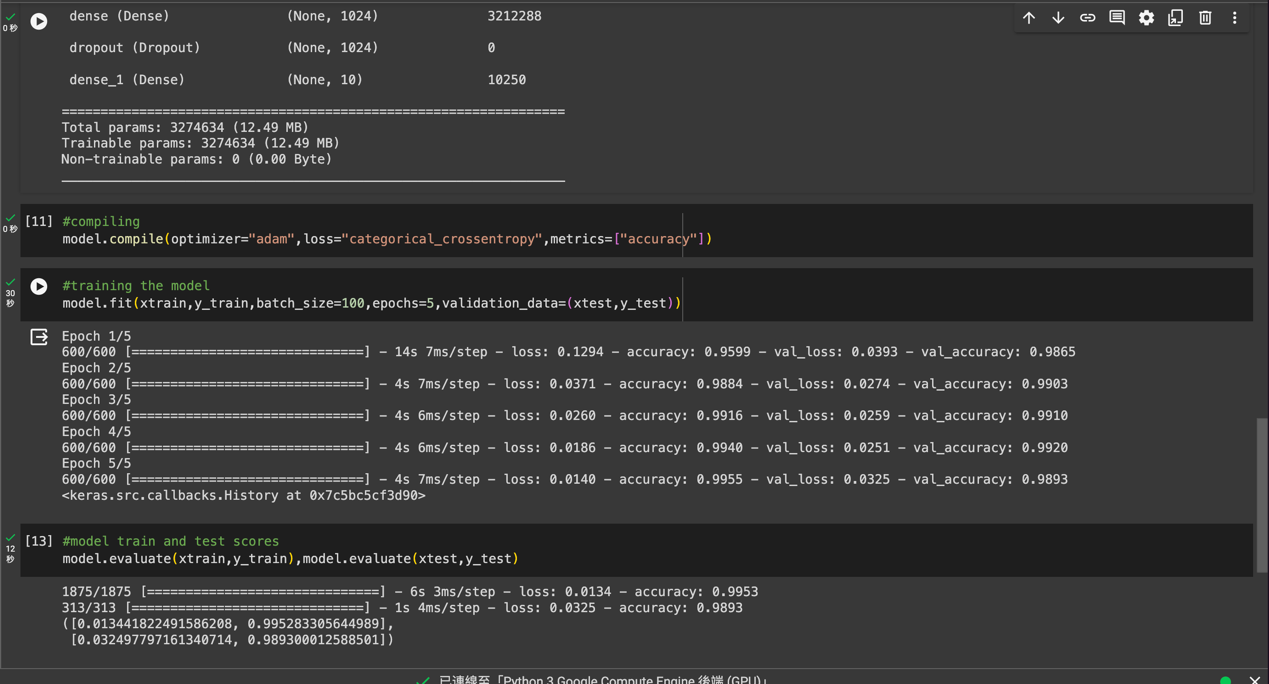
val\_loss: 0.0393 – val\_accuracy: 0.9865

val\_loss: 0.0274 – val\_accuracy: 0.9903

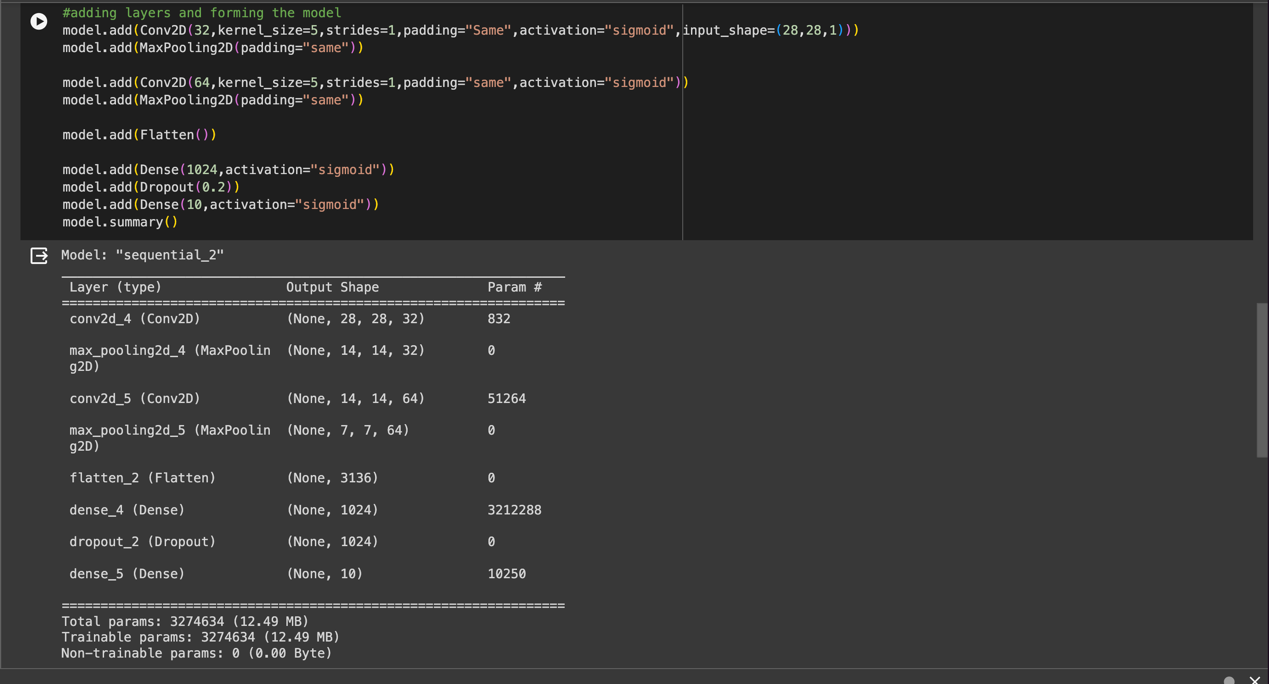
val\_loss: 0.0259 – val\_accuracy: 0.9910

val\_loss: 0.0251 – val\_accuracy: 0.9920

val\_loss: 0.0325 – val\_accuracy: 0.9893



把所有activation換成sigmoid



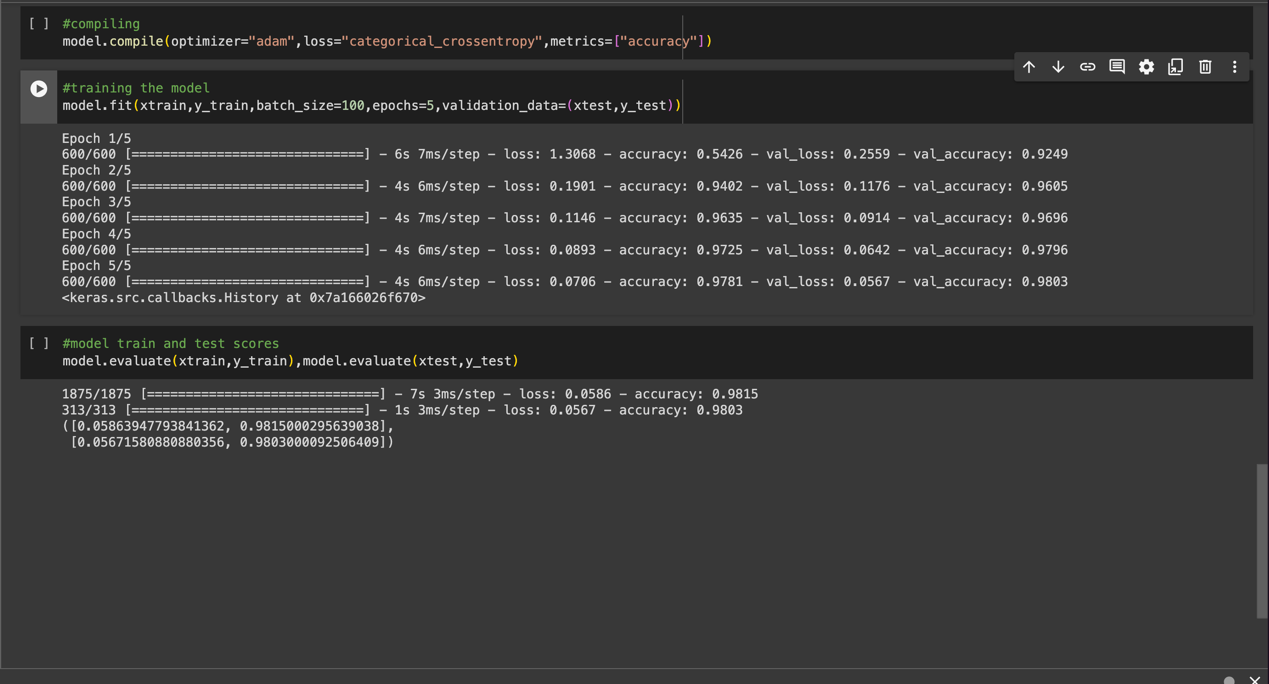
val\_loss: 0.2559 - val\_accuracy: 0.9249

val\_loss: 0.1176 - val\_accuracy: 0.9605

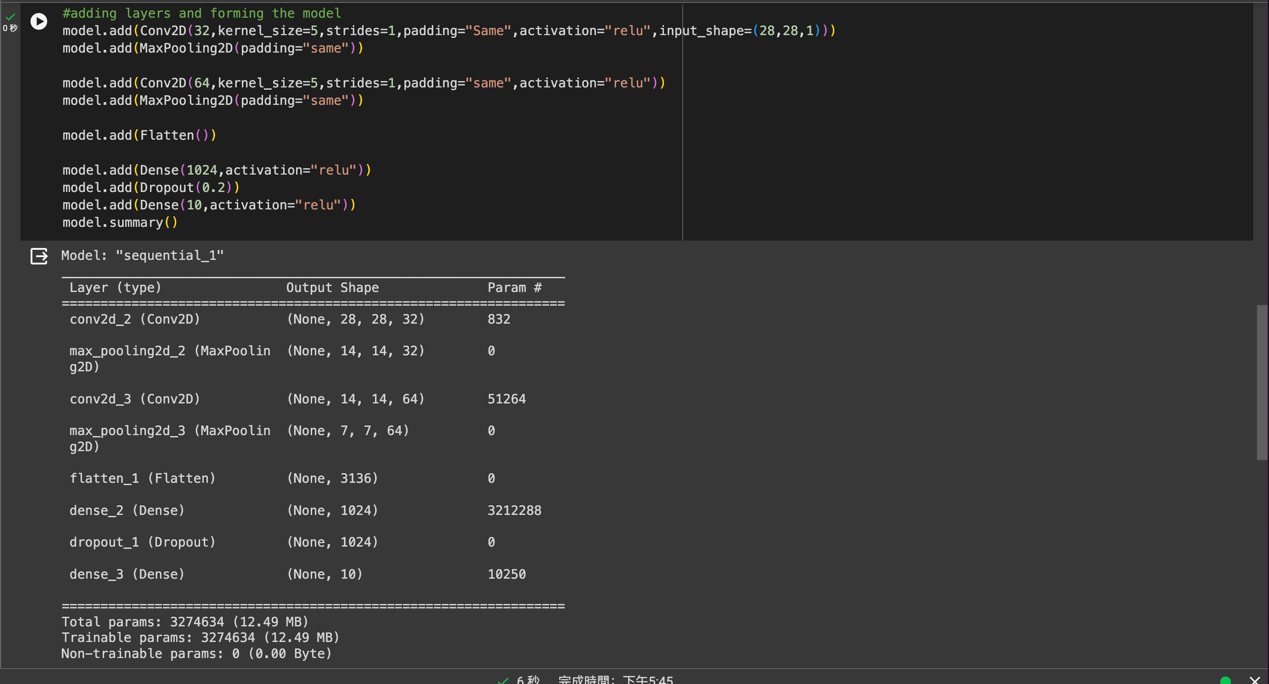
val\_loss: 0.0914 - val\_accuracy: 0.9696

val\_loss: 0.0642 - val\_accuracy: 0.9796

val\_loss: 0.0567 - val\_accuracy: 0.9803



把所有activation換成relu



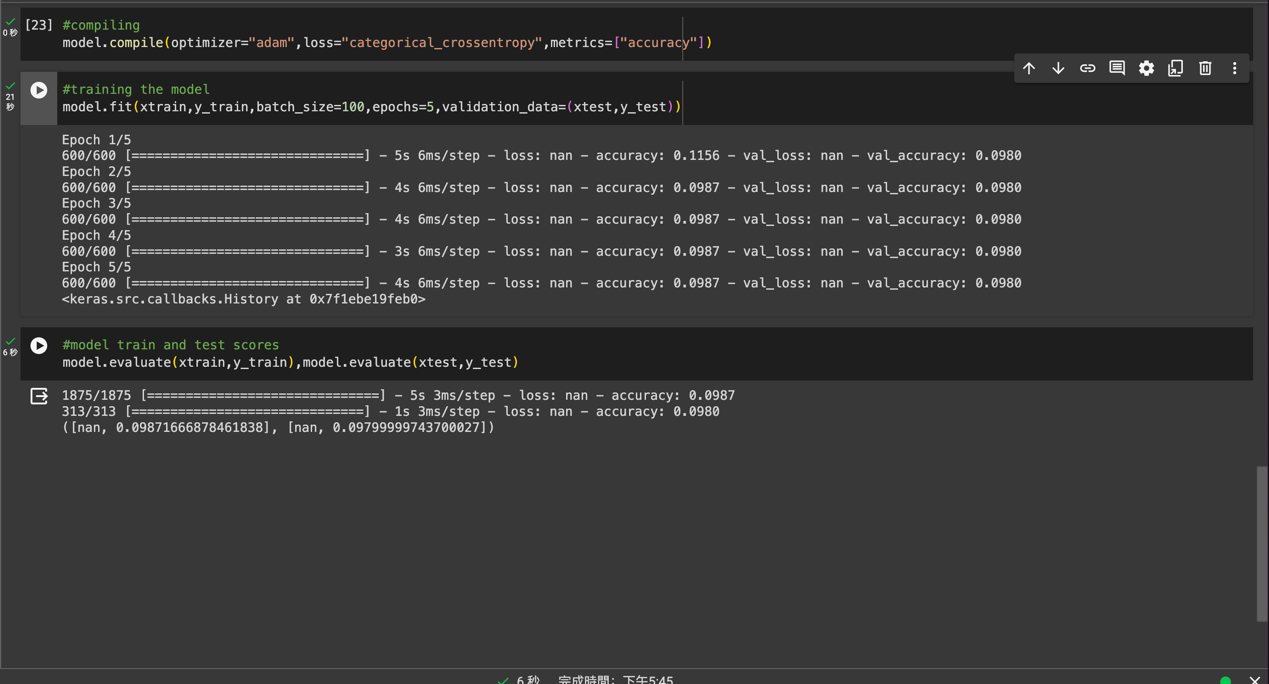
val\_loss: nan - val\_accuracy: 0.0980

val\_loss: nan - val\_accuracy: 0.0980

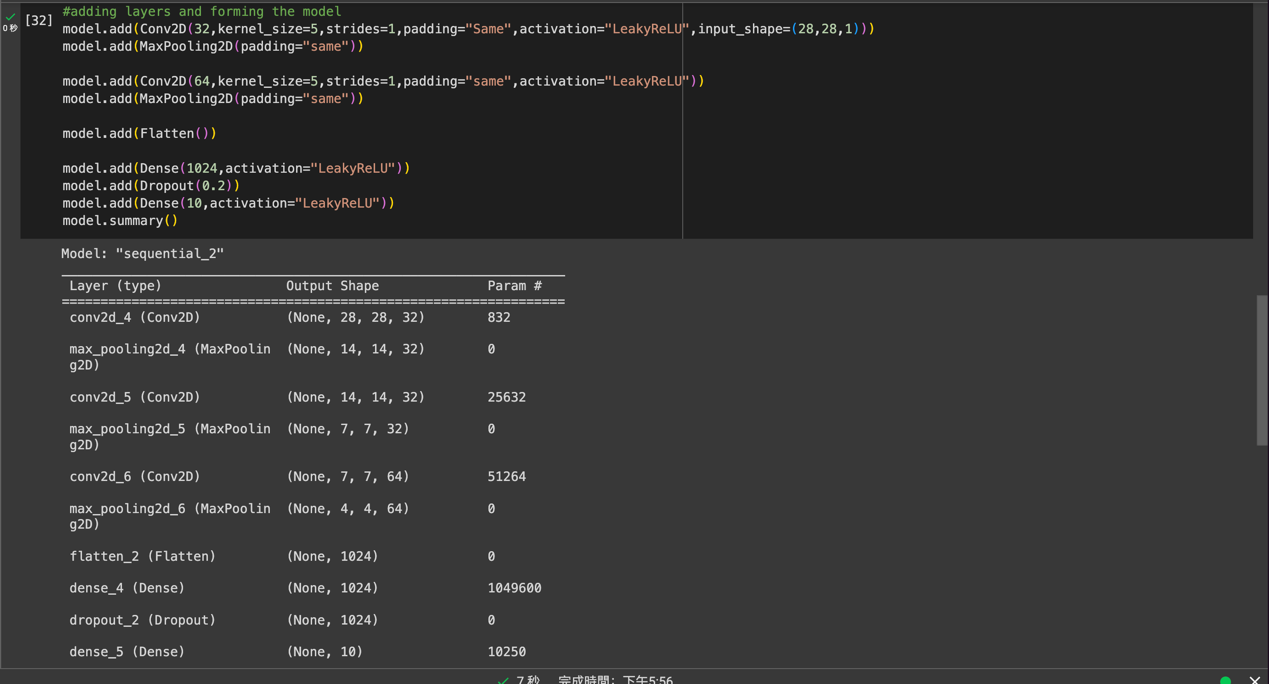
val\_loss: nan - val\_accuracy: 0.0980

val\_loss: nan - val\_accuracy: 0.0980

val\_loss: nan - val\_accuracy: 0.0980



把所有activation換成LeakyReLU



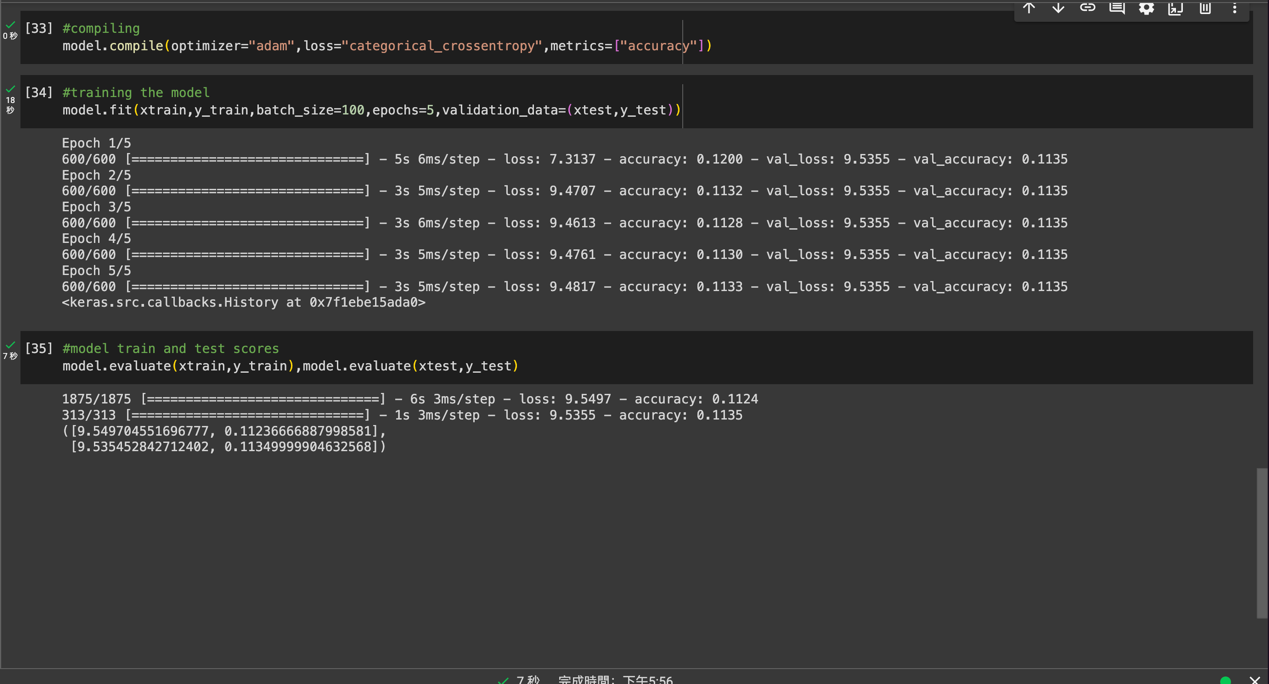
val\_loss: 9.5355 - val\_accuracy: 0.1135

val\_loss: 9.5355 - val\_accuracy: 0.1135

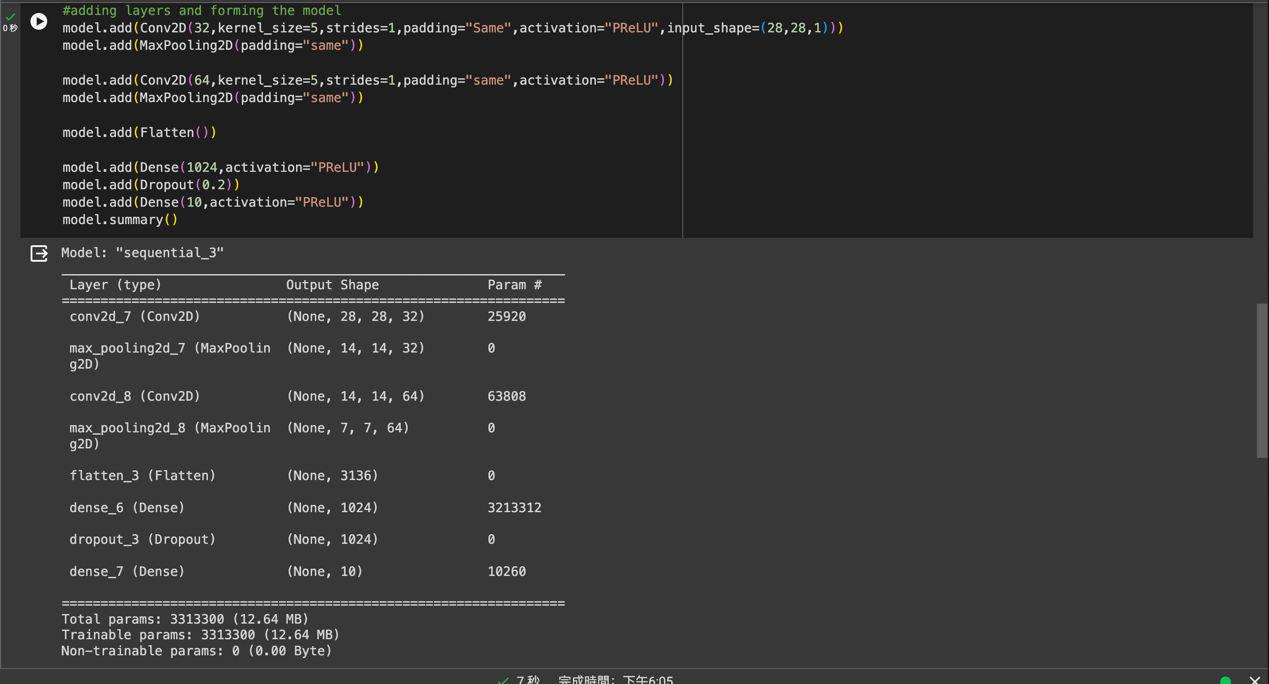
val\_loss: 9.5355 - val\_accuracy: 0.1135

val\_loss: 9.5355 - val\_accuracy: 0.1135

val\_loss: 9.5355 - val\_accuracy: 0.1135



把所有activation換成PReLU



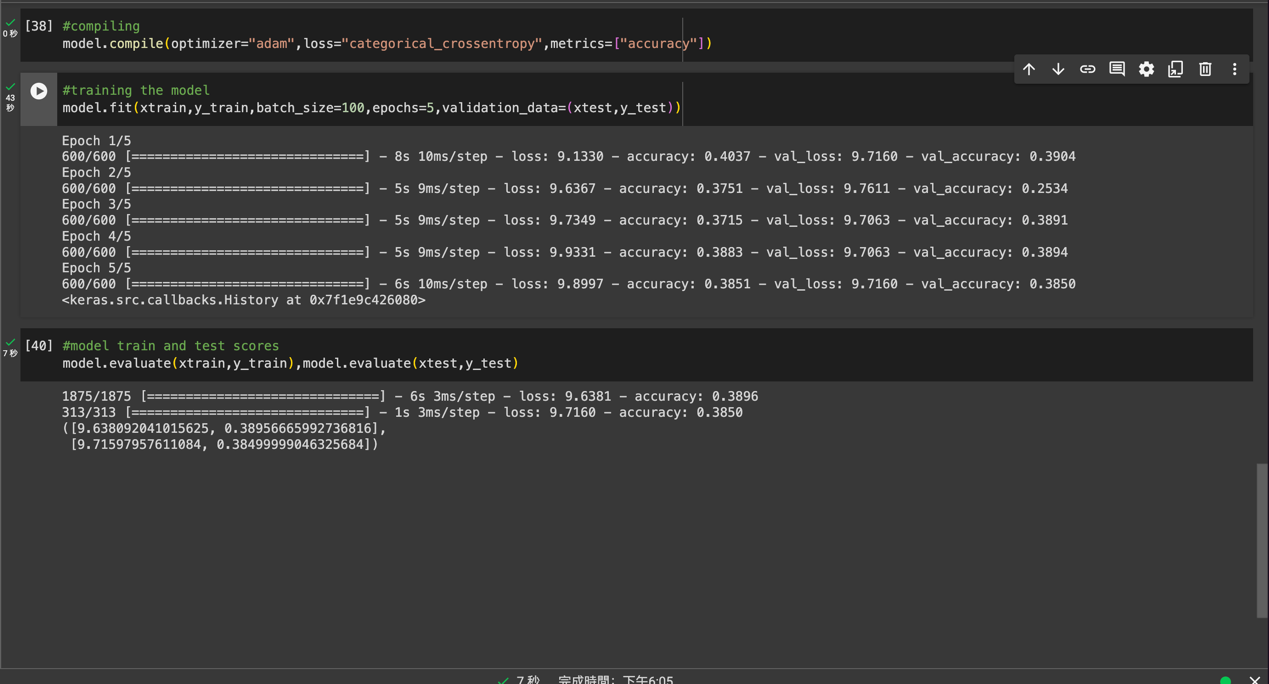
val\_loss: 9.7160 - val\_accuracy: 0.3904

val\_loss: 9.7611 - val\_accuracy: 0.2534

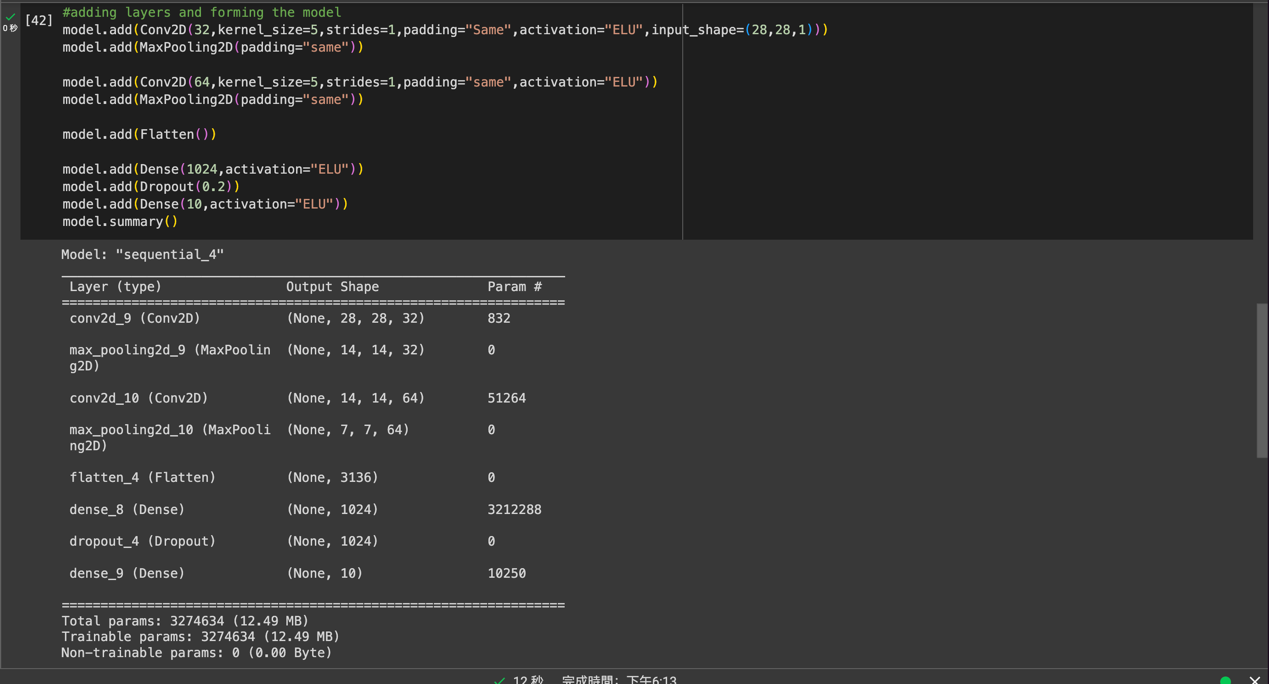
val\_loss: 9.7063 - val\_accuracy: 0.3891

val\_loss: 9.7063 - val\_accuracy: 0.3894

val\_loss: 9.7160 - val\_accuracy: 0.3850



把所有activation換成ELU



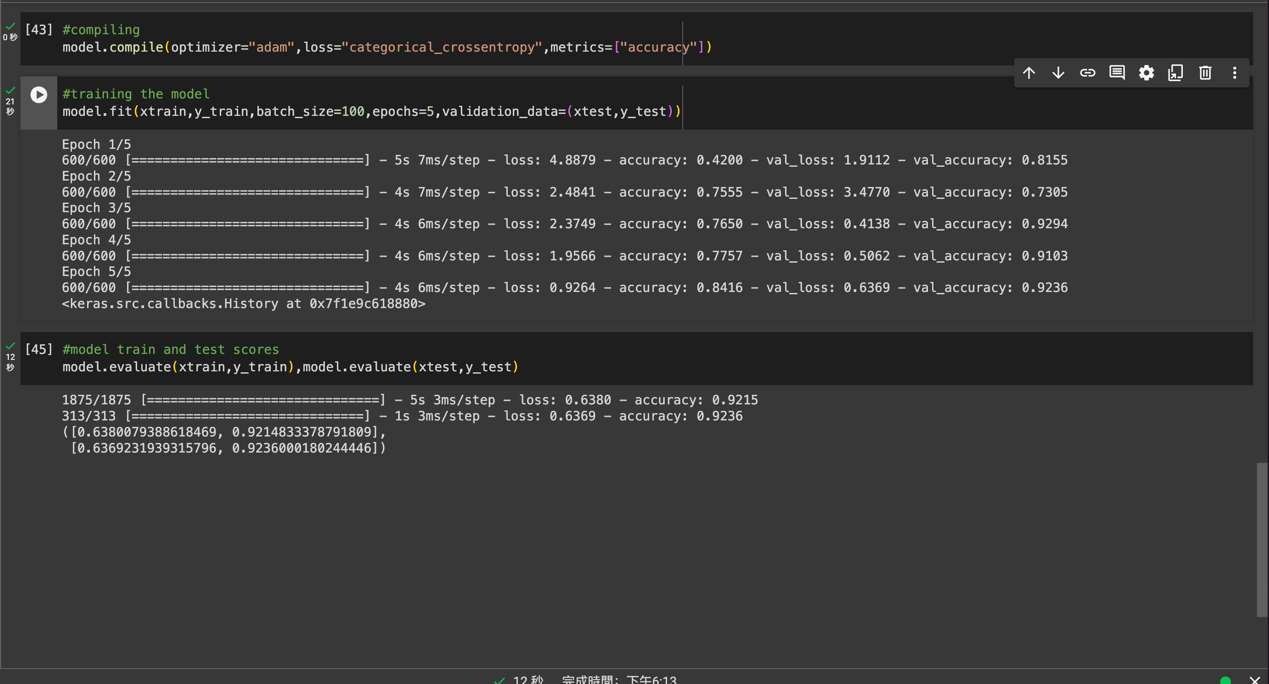
val\_loss: 1.9112 - val\_accuracy: 0.8155

val\_loss: 3.4770 - val\_accuracy: 0.7305

val\_loss: 0.4138 - val\_accuracy: 0.9294

val\_loss: 0.5062 - val\_accuracy: 0.9103

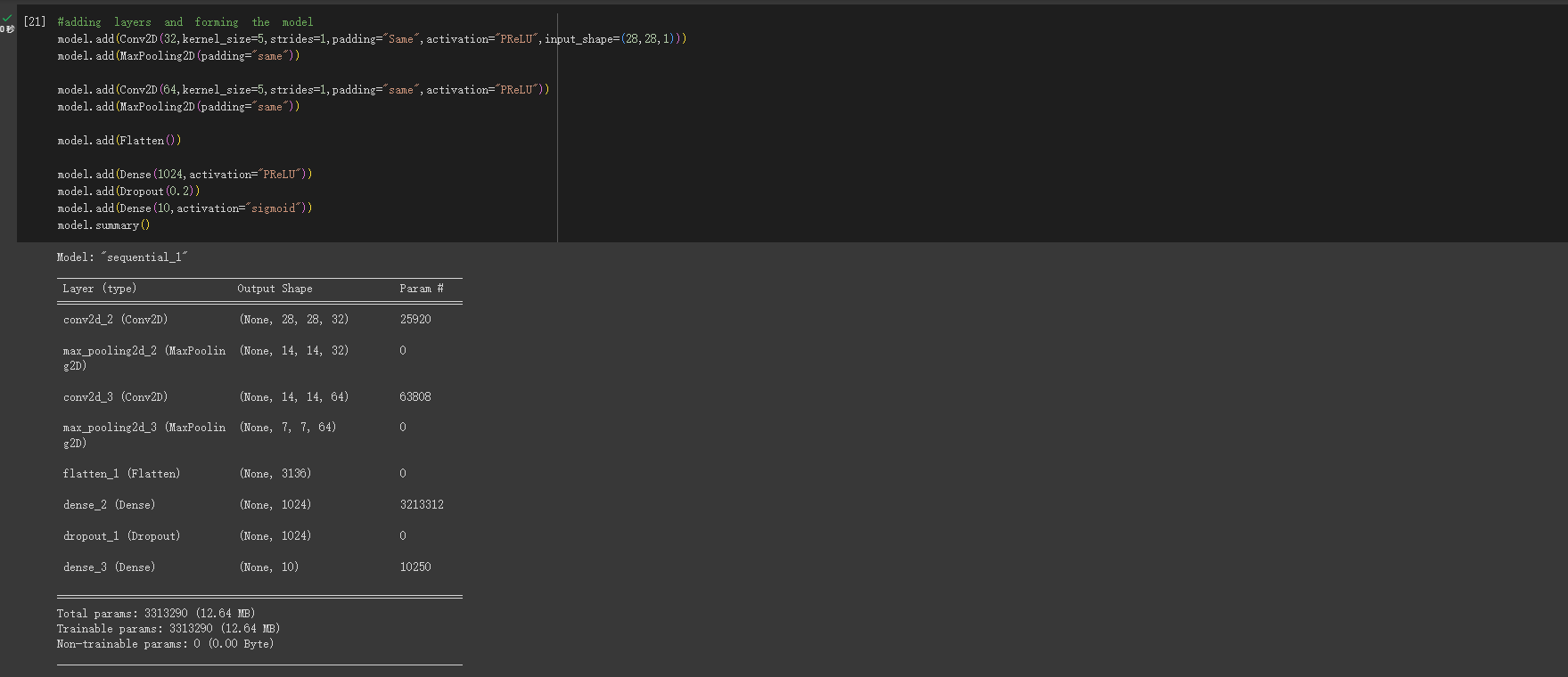
val\_loss: 0.6369 - val\_accuracy: 0.9236



重新編譯太多次我發現好像會禁止使用T4GPU連線禁止編譯，還想使用的話需要額外在購買T4GPU的使用權限，不然只能用非常緩慢的CPU來進行編譯，幾乎每次編譯都要耗時20分鐘以上

有關於activation的部分，把所有activation改成同一種函數後的結果如上述所示，表現最佳的是sigmoid，有點差強人意的是ELU，因為起初值誤差有點大，後面學習過後的值也沒有sigmoid來的好，整體穩定度也是沒有sigmoid來的佳

嘗試將activation用類似於起始資料的方式將relu改成PReLU



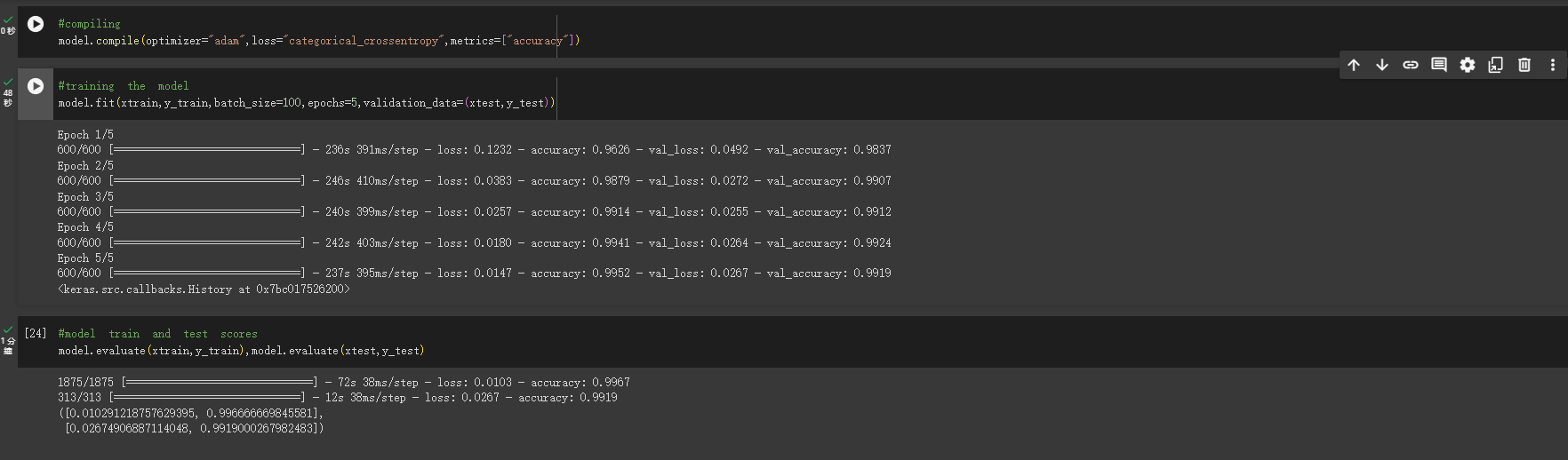
val\_loss: 0.0492 - val\_accuracy: 0.9837

val\_loss: 0.0272 - val\_accuracy: 0.9907

val\_loss: 0.0255 - val\_accuracy: 0.9912

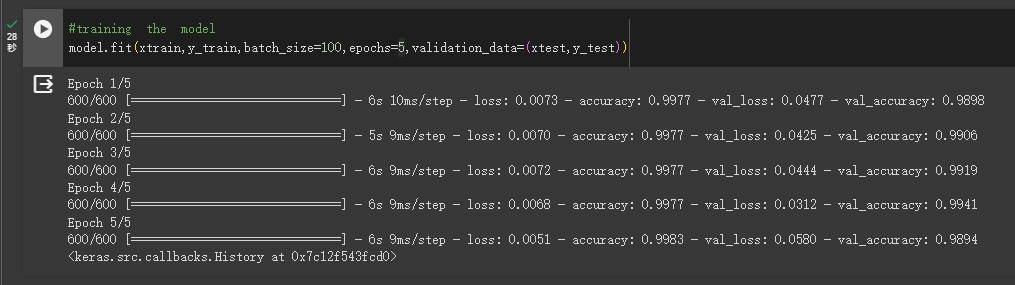
val\_loss: 0.0264 - val\_accuracy: 0.9924

val\_loss: 0.0267 - val\_accuracy: 0.9919

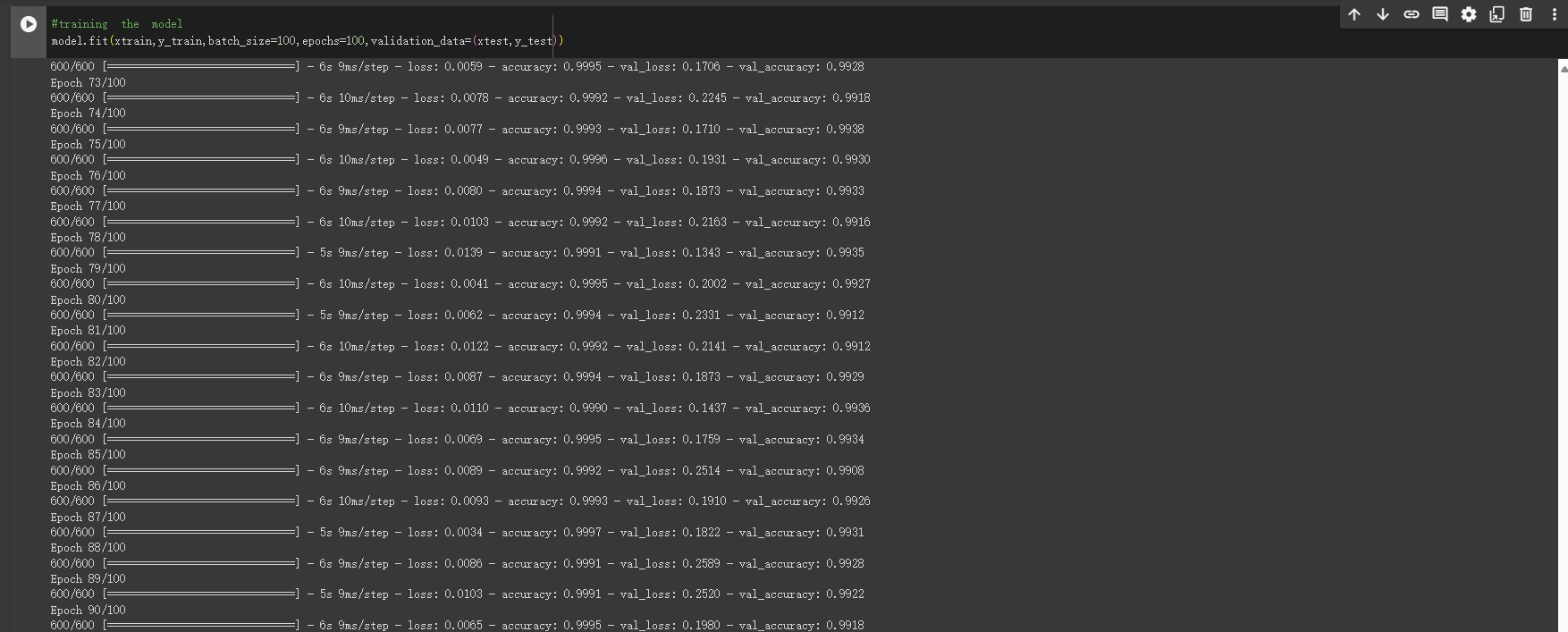


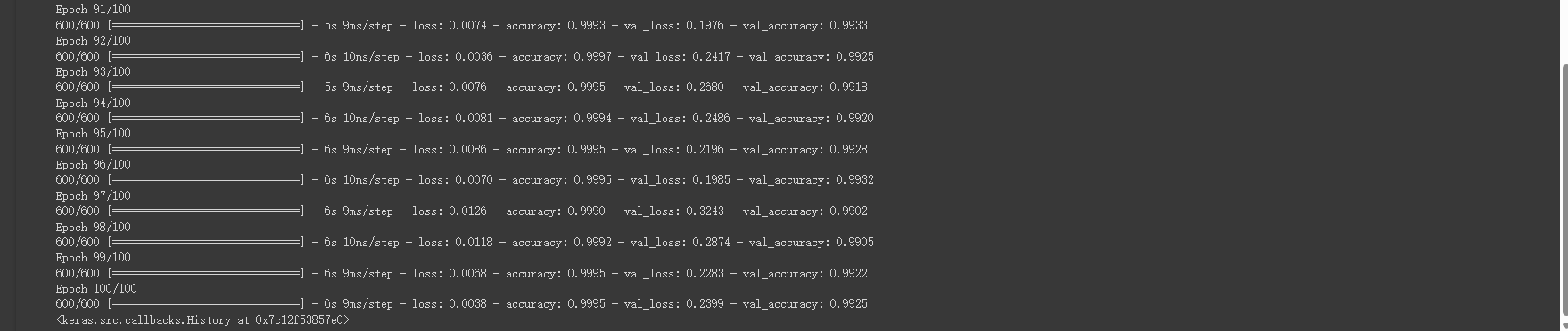
雖然說最後數值和原始的沒有差很多，但是這個的最後幾個都是99.多，原始的最後一個又變回了98.多，所以目前這個應該是最佳解

Epoch原始資料(Epoch = 5)，用於調整AI總體演算個數的數量

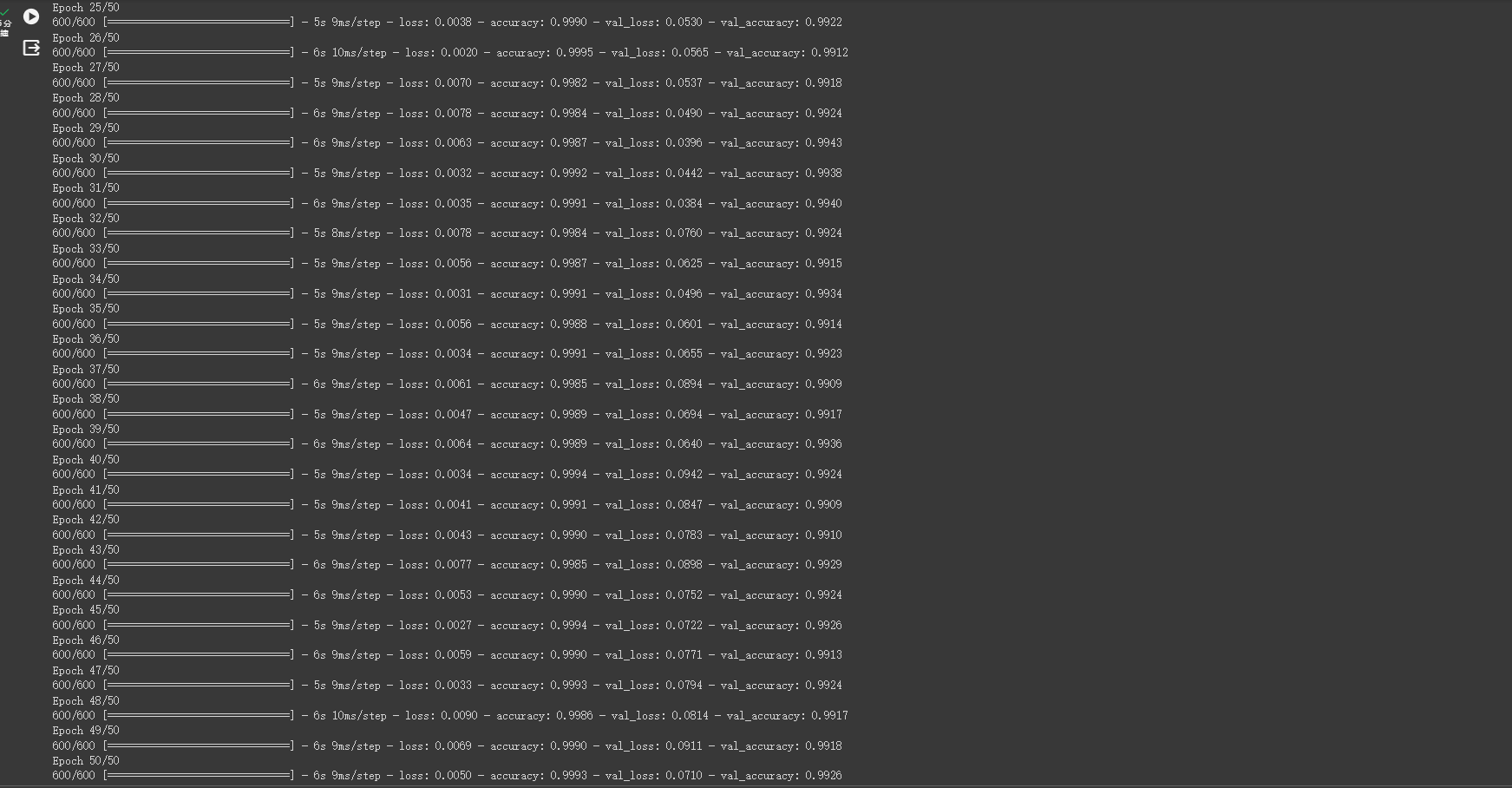
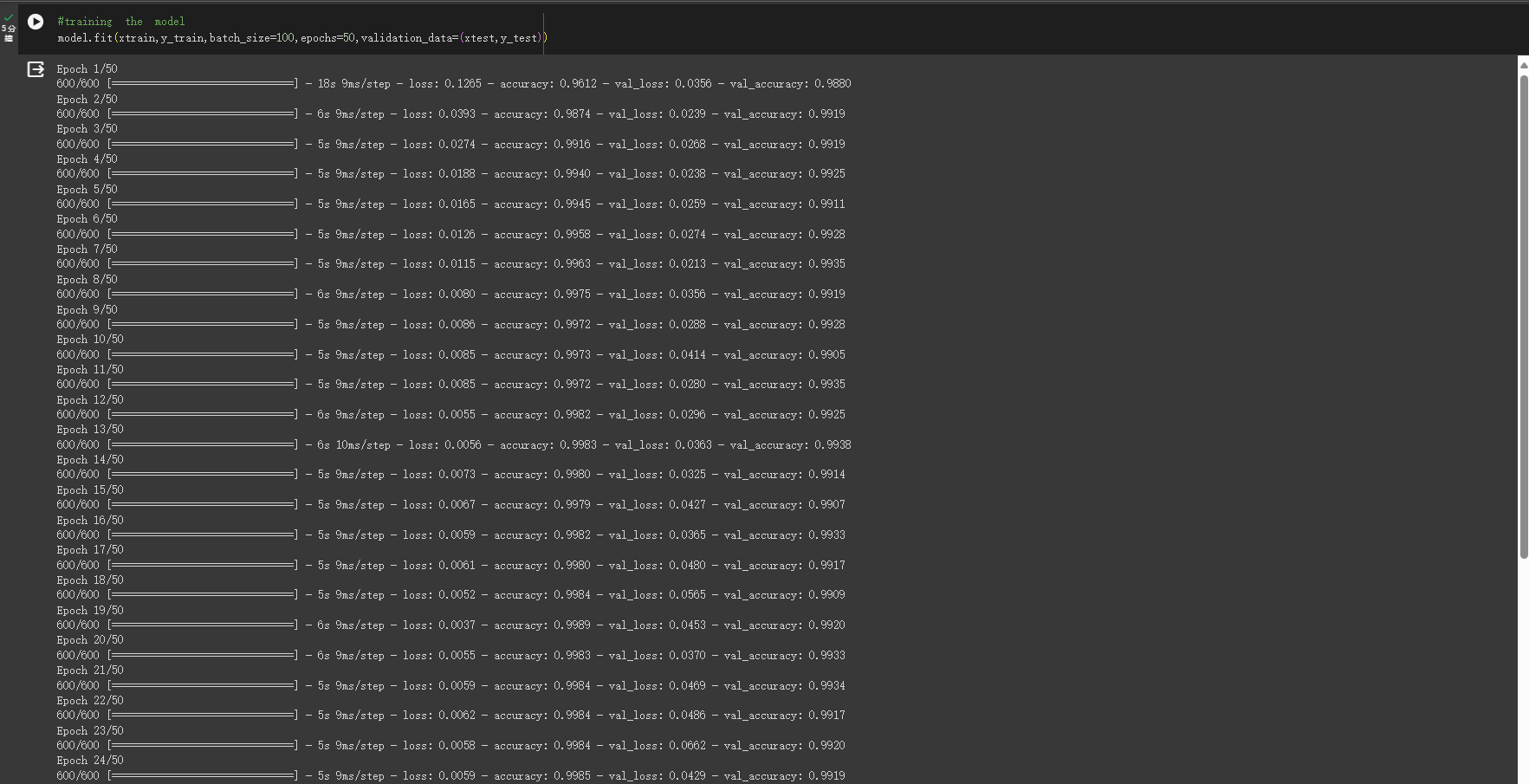


Epoch = 100

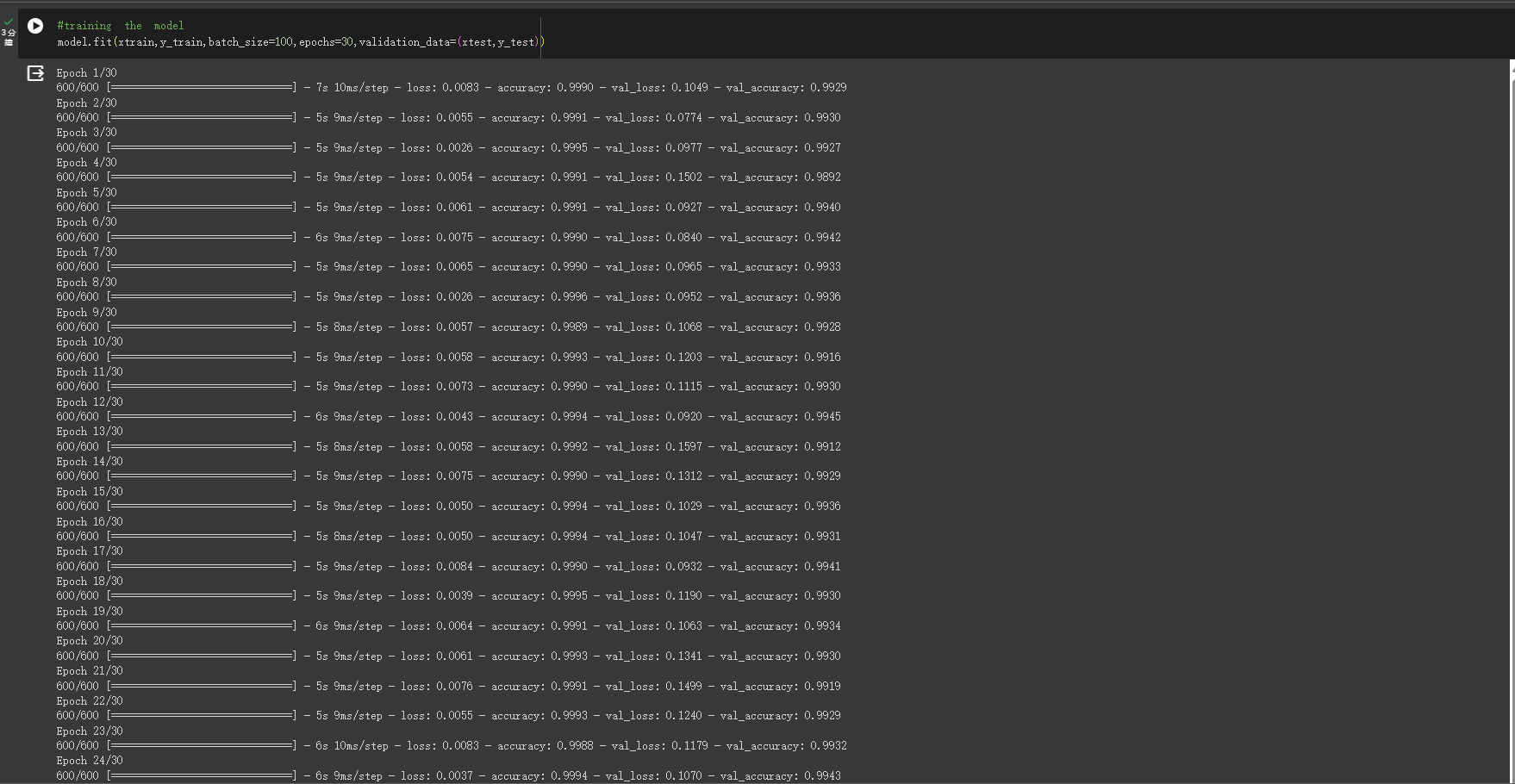


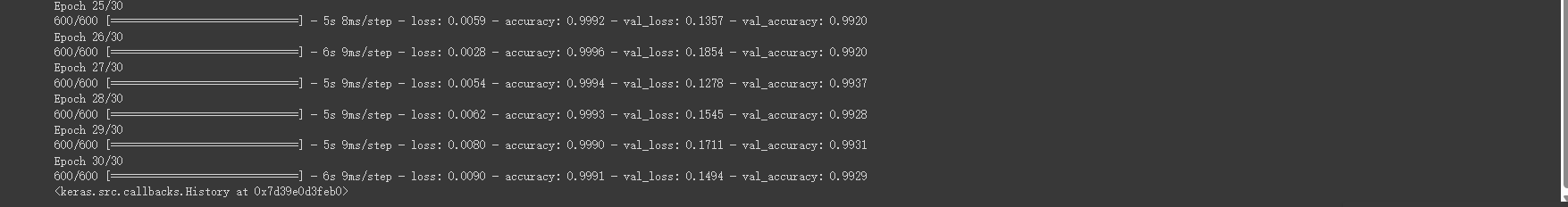


Epoch = 50

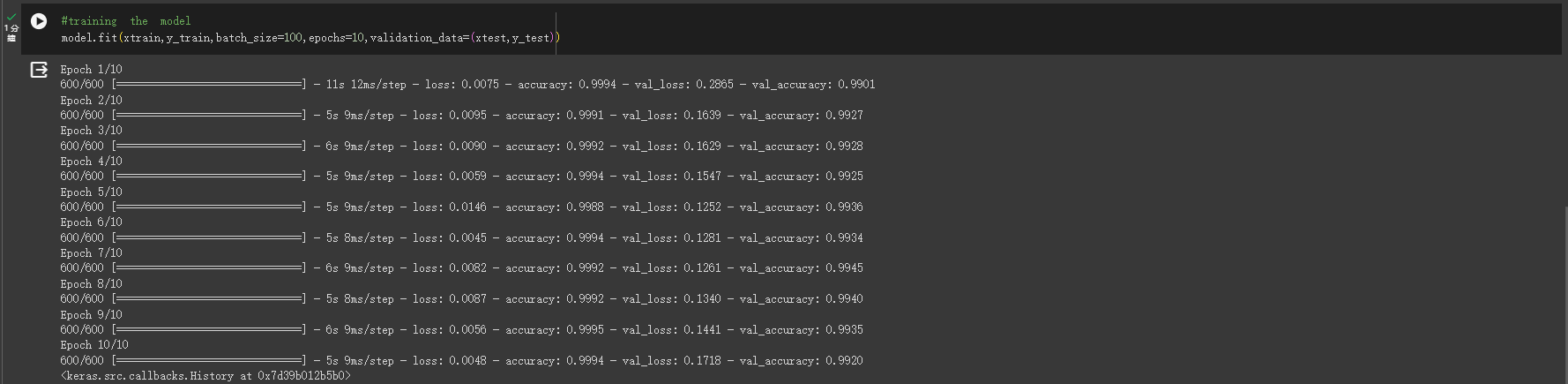


Epoch = 30

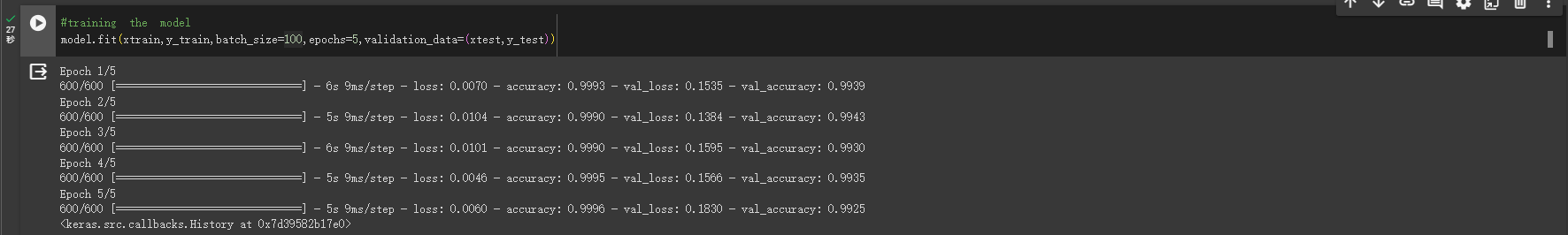




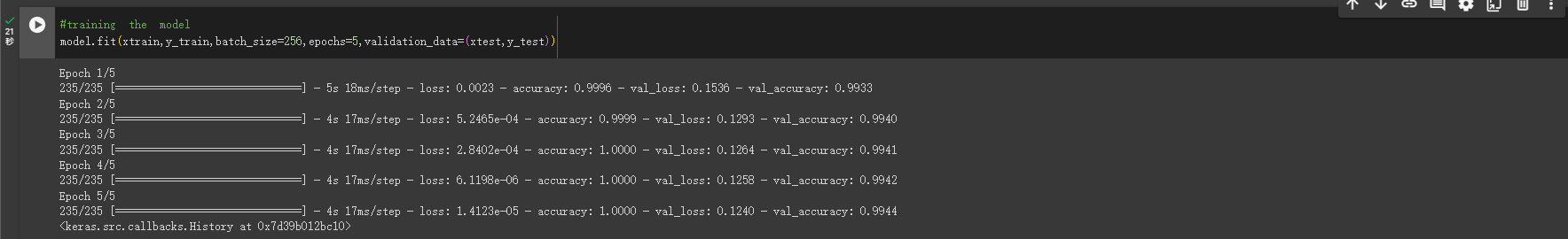
Epoch = 10



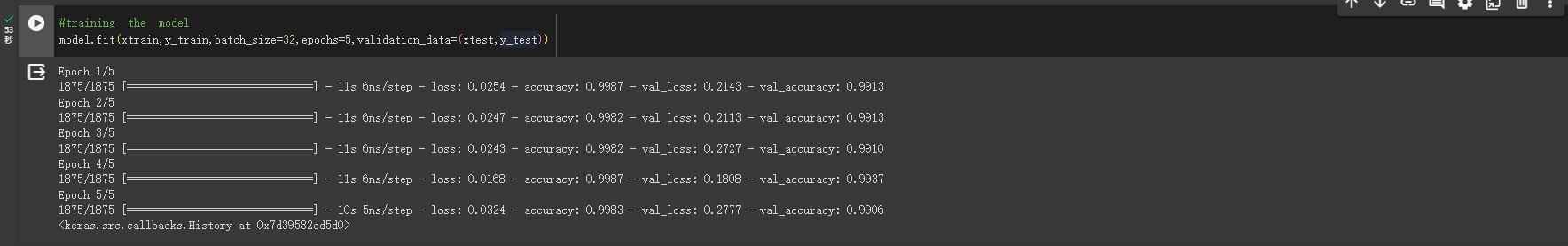
Batch\_size原始資料(Batch\_size = 100)，用於調整一次餵給AI的資料數量大小



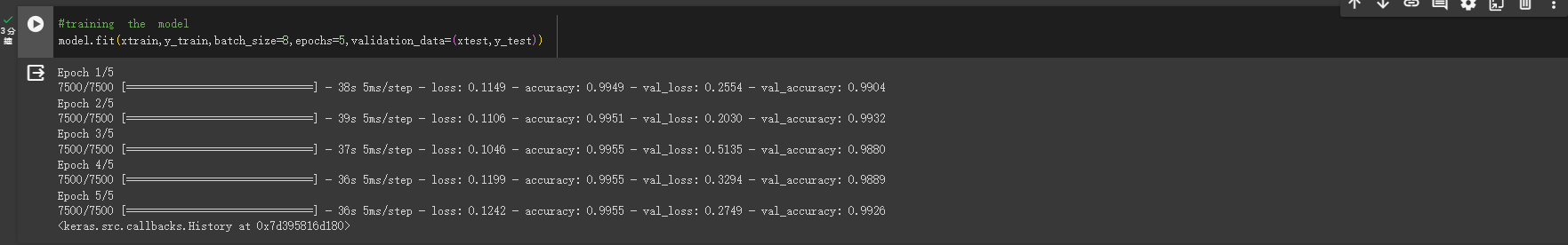
Batch\_size = 256



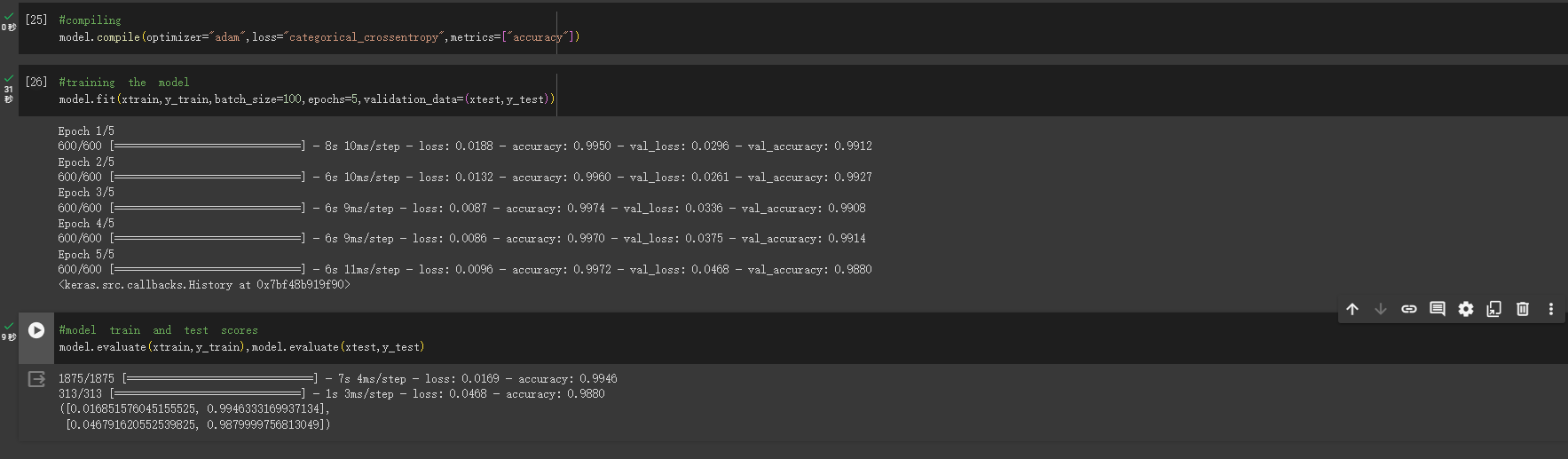
Batch\_size = 32



Batch\_size = 8



Loss function原始資料(Loss function = Cross-Entropy)，選擇要用的損失函數



Loss function = MSE(mean\_squared\_error)

