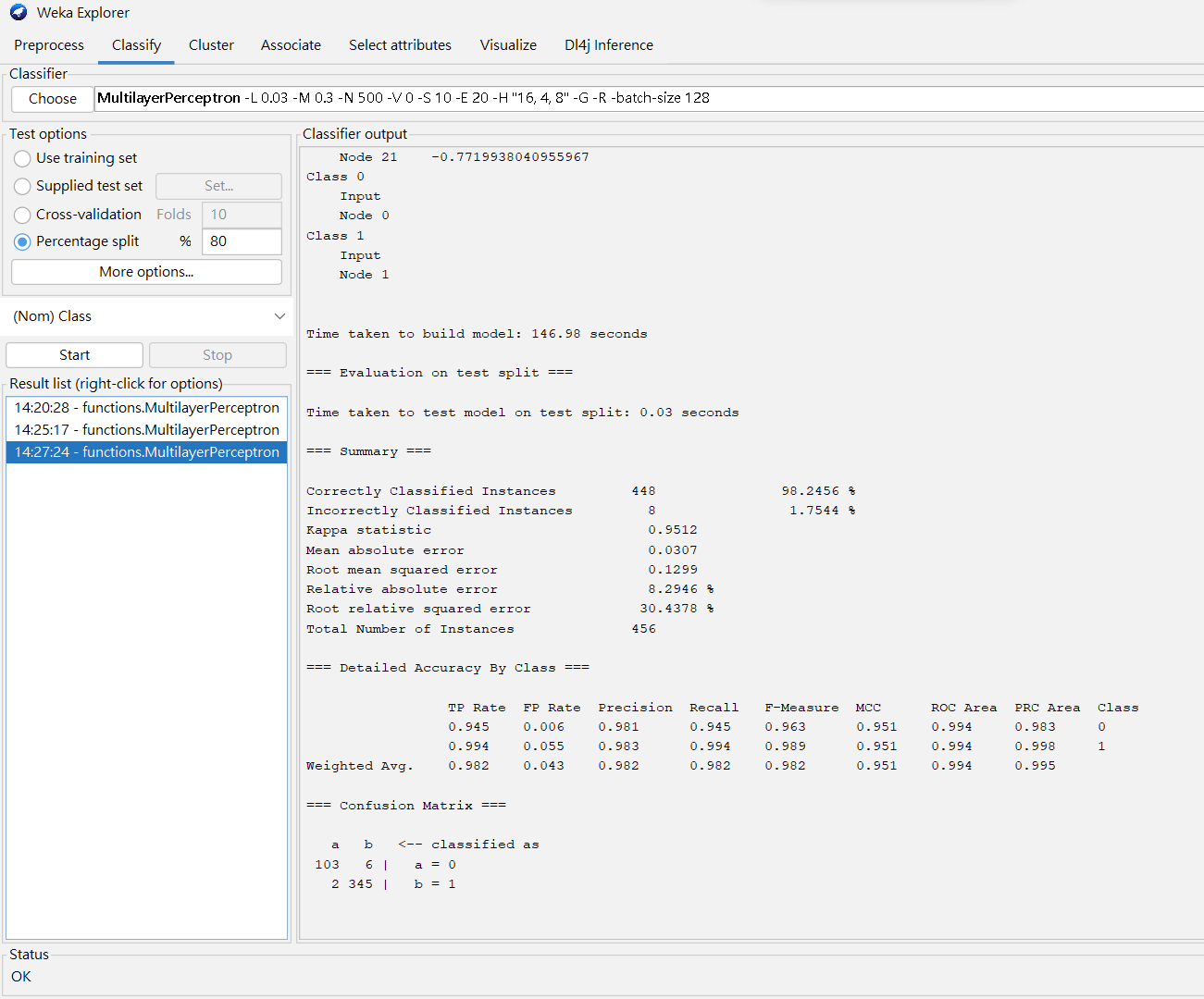
**HW8\_109403021**

**步驟2: 建立模型、**

**MLP:**

****

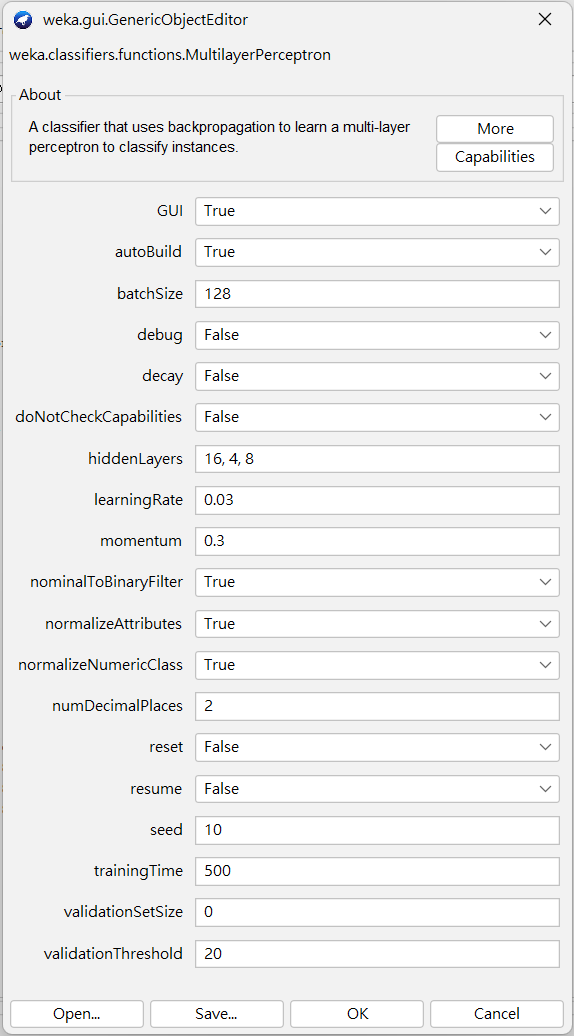
**準確率**

**選此建立MLP模型**

**切割80%為訓練集**

**預測Class**

使用我用python前處理完的output.csv來建立模型並訓練，以上是得到的結果

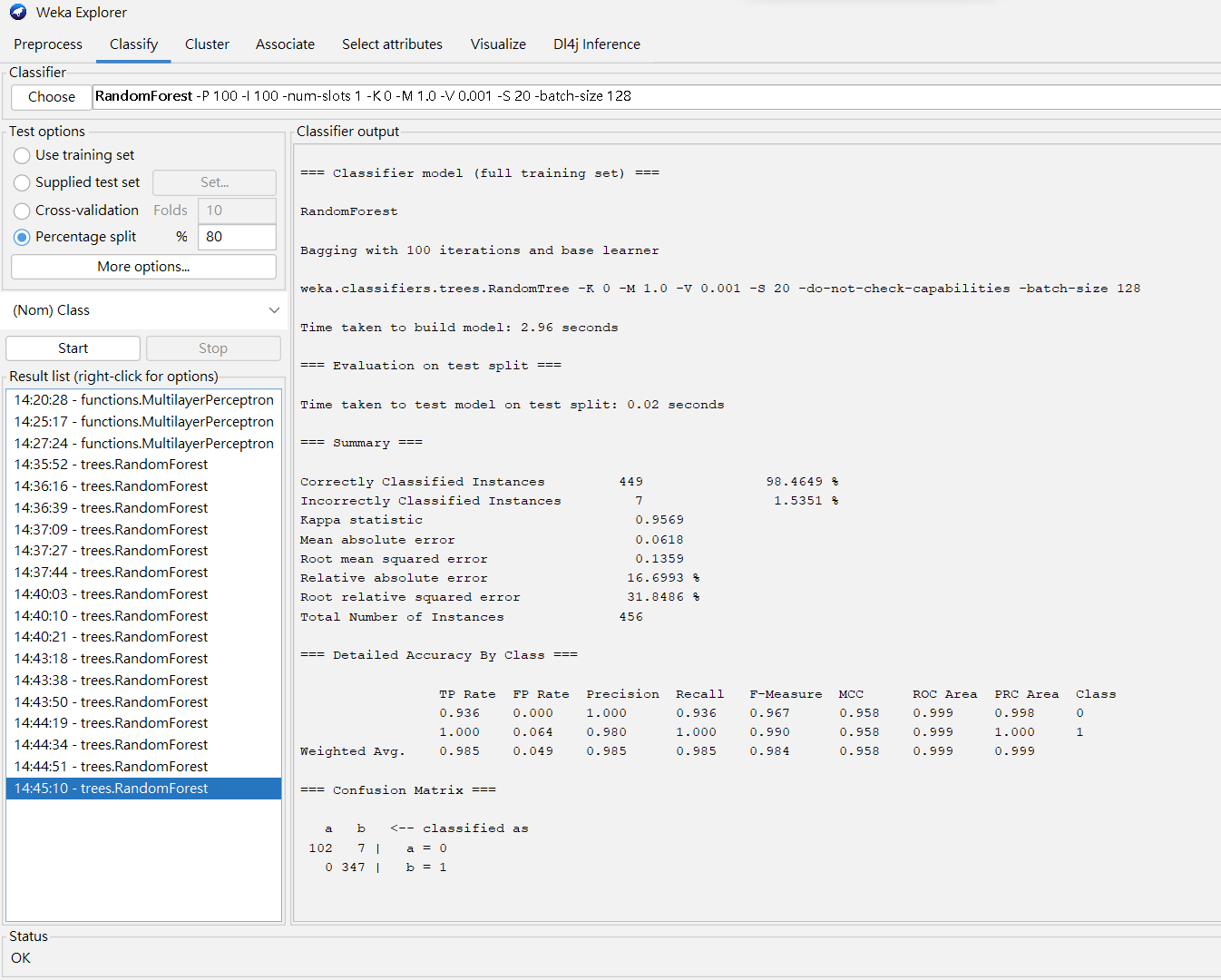
****

這裡主要調整了:

超參數:學習率(learningRate)和動量(momentum)，把學習率調得比默認值更小、動量調得比默認值更大

設置了三層隱藏層: 16,4,8

**RandomForest:**

****

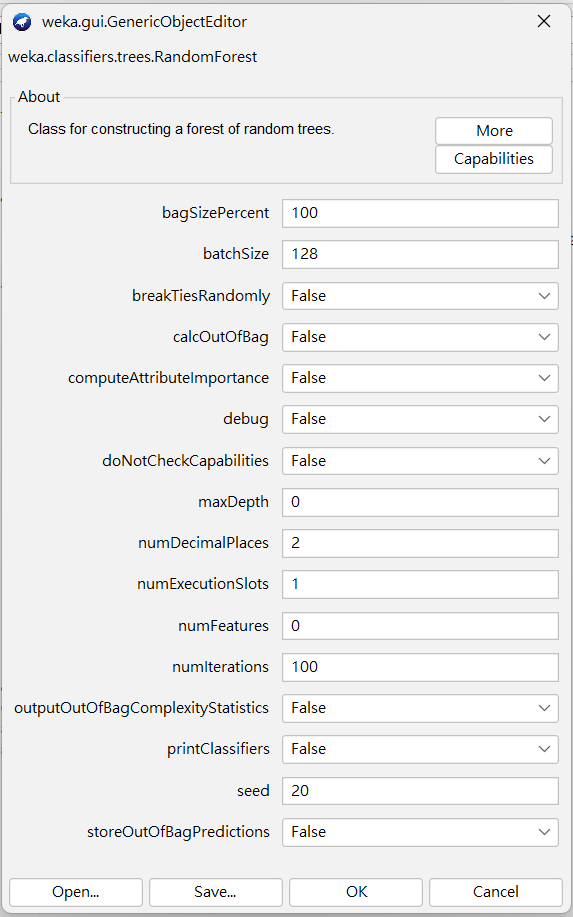
**選此建立Random Forest模型**

**準確率**

**預測Class**

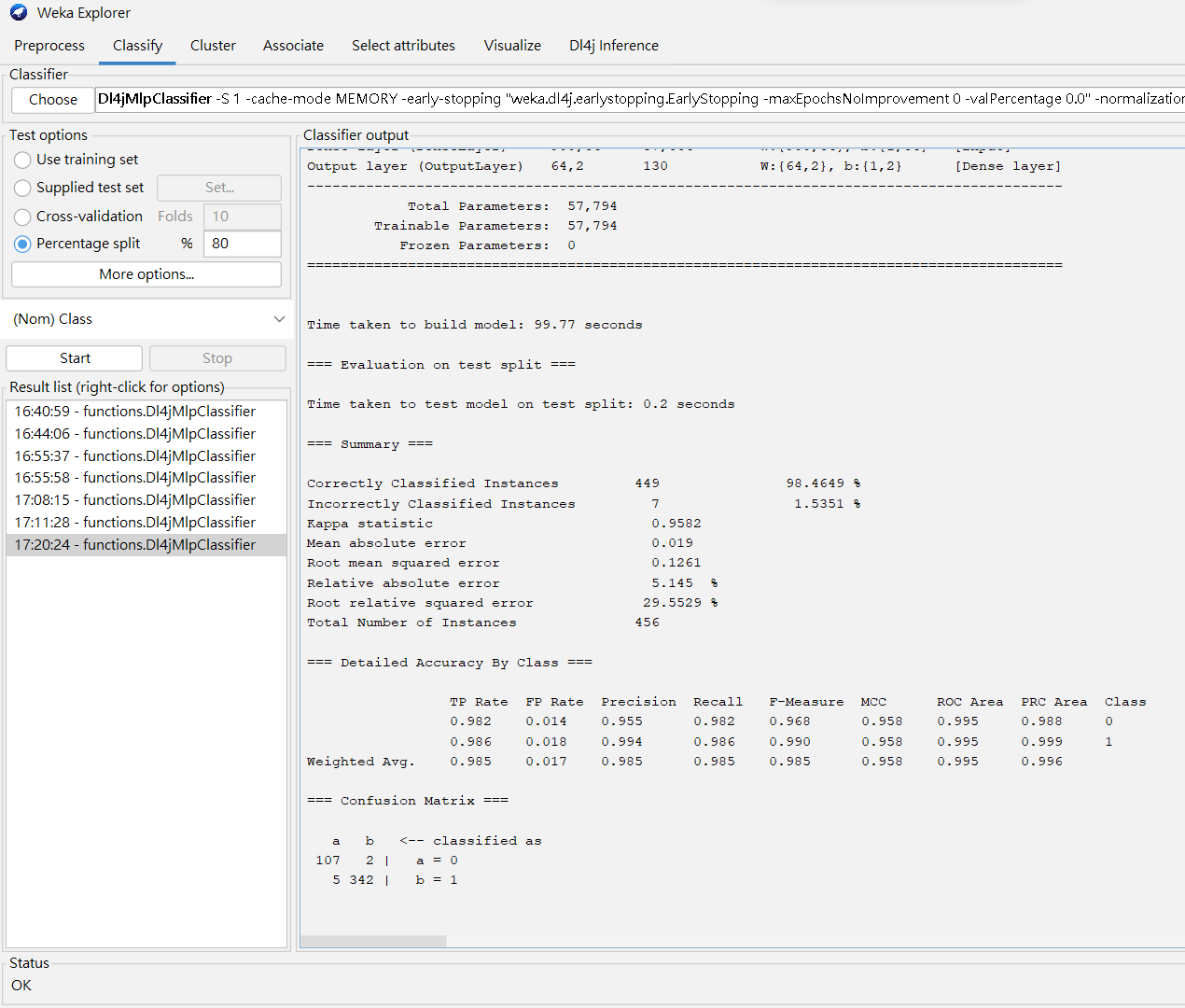
**切割80%為訓練集**

使用我用python前處理完的output.csv來建立模型並訓練，以上是得到的結果

****

這裡的超參數與模型架構大部分皆使用默認值，僅稍微調整了batchSize

**CNN:**

****

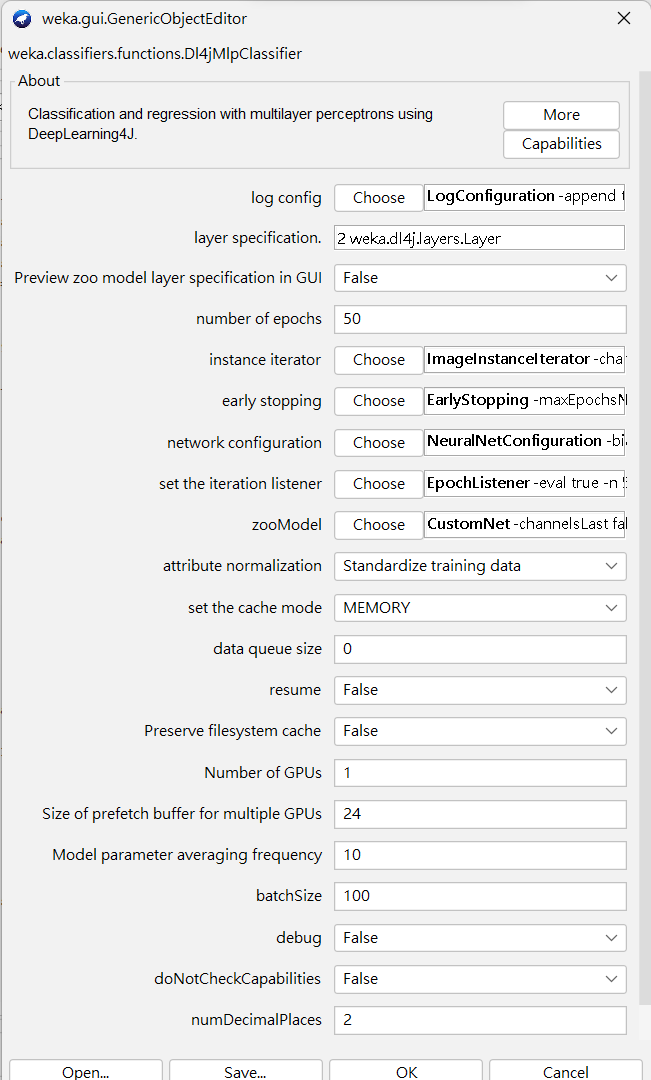
**準確率**

**選此建立CNN模型**

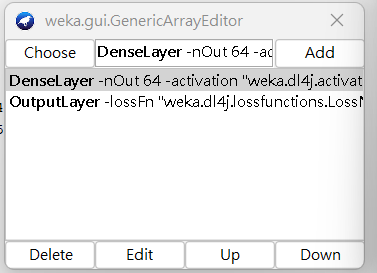
**切割80%為訓練集**

**預測Class**

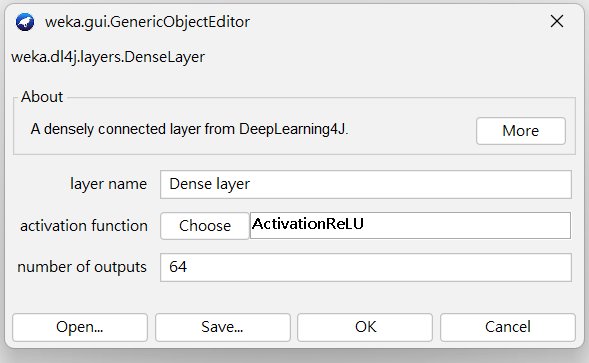
使用我用python前處理完的output.csv和輸出的圖檔來建立模型並訓練，以上是得到的結果

****

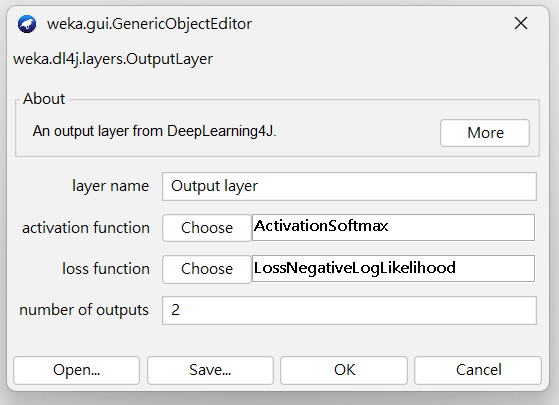
將epochs從默認值10提升到50，instance iterator選取ImageInstanceIterator並選到圖檔資料夾

****

模型這邊疊了一層Dense Layer和一層Output Layer

****

Dense Layer的參數設置

****

Output Layer的參數設置

**步驟3: 模型大比較、**

1. **各模型不同超參數表現差異**

以下各模型改變某一超參數值其他不變

|  |  |  |
| --- | --- | --- |
| **模型與比較之超參數** | **超參數值** | **準確率** |
| MLP-比較**learningRate** | 0.3 |  |
| 0.03 |  |
| RandomForest-比較**MaxDepth** | 0 |  |
| 1 |  |
| CNN-比較  **batchSize** | 100 |  |
| 500 |  |

1. **模型優缺點**

|  |  |  |
| --- | --- | --- |
|  | **優點** | **缺點** |
| **MLP** | 靈活、易使用 | 易陷入局部最小值 |
| **RandomForest** | 不太會過擬合 | 內存消耗大 |
| **CNN** | 善於處理視覺數據 | 計算成本大 |

1. **改進之處**

**MLP:**

可以用正則化技術減少過擬合風險、可以選擇更好的優化算法(Adam等等)

**RandomForest:**

調整森林大小以平衡準確性和預測速度

**CNN:**

增加卷積層和全連接層來捕捉更多圖像特徵

1. **最後選擇**

此次任務我會選擇用RandomForest模型，因為它的準確度表現最好而且速度最快