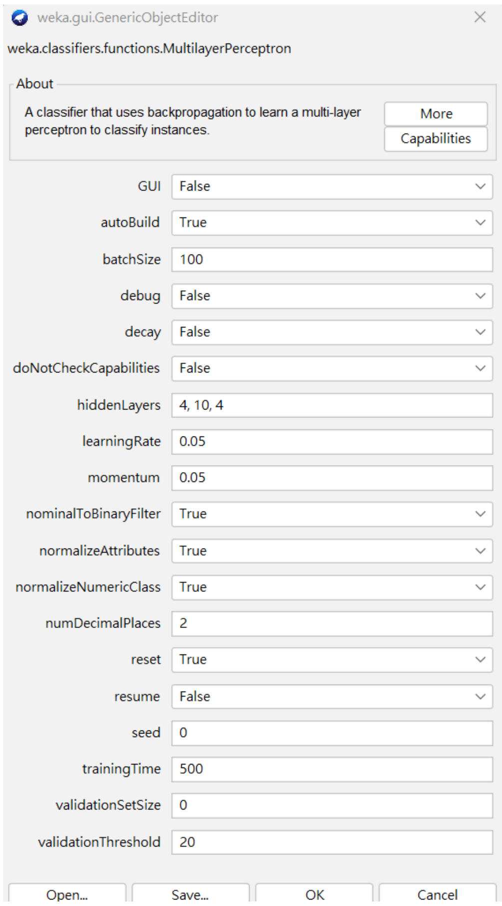
建立模型

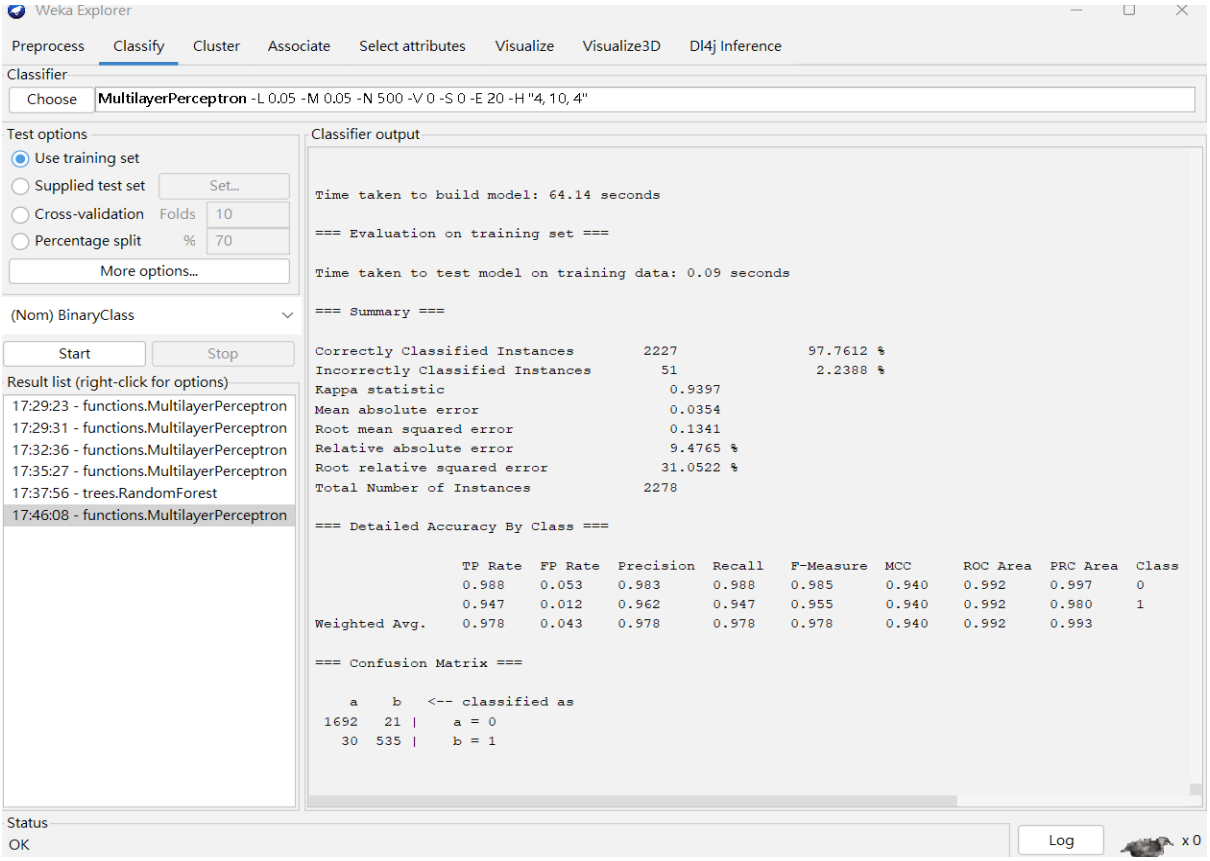
MLP RandomForest 兩個模型都是用這個 csv(由 class 和 one hot 過的 permission)

BinaryCla	ANDROI	ANDROI	ANDROI	adnroid.pe	ae.gov.mo	air.br.com	air.com.ea	air.com.la	androd.per	android.M	android.ha	android.ha	android.ha	androi
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	

MLP

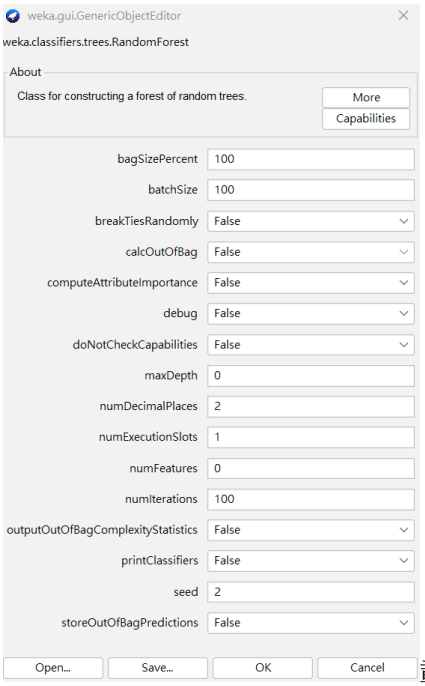


載入 csv 後將 class 轉成 nominal，即可訓練

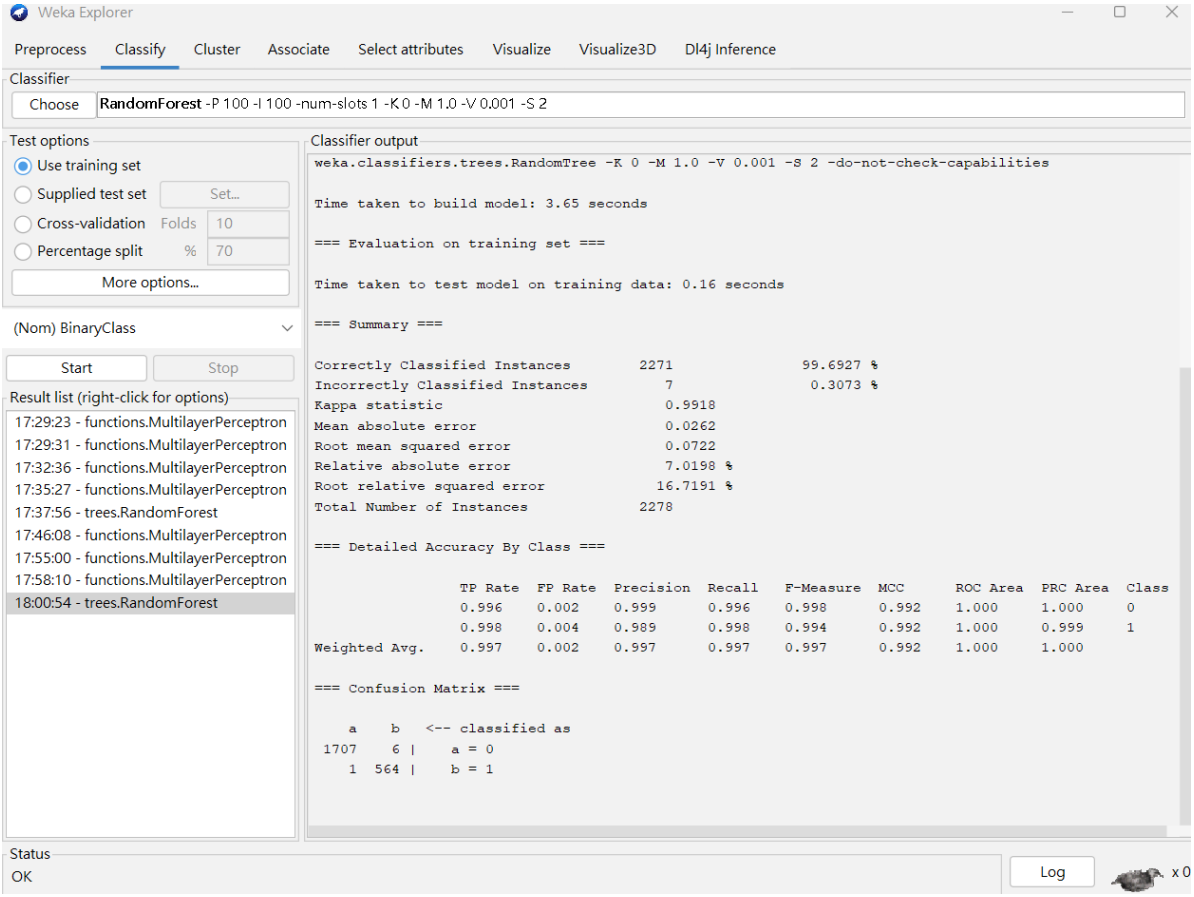


最後是 97%，參數調 hidden layer ,learning rate ,momentum

RandomForest



載入 csv 後將 class 轉成 nominal，即可訓練



最後為 99% 參數調 seed 為 2

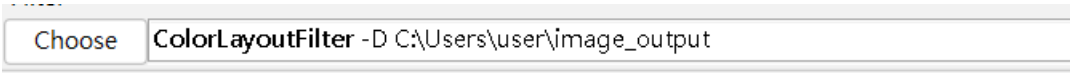
CNN 處理

載入 csv(有 class、檔名)

Class	Image
0	image_0.png
0	image_1.png
0	image_2.png
0	image_3.png
0	image_4.png
0	image_5.png
0	image_6.png
0	image_7.png
0	image_8.png
0	image_9.png
0	image_10.png
0	image_11.png
0	image_12.png
0	image_13.png
0	image_14.png
0	image_15.png
0	image_16.png
0	image_17.png
0	image_18.png

將 csv 的檔名轉成 string

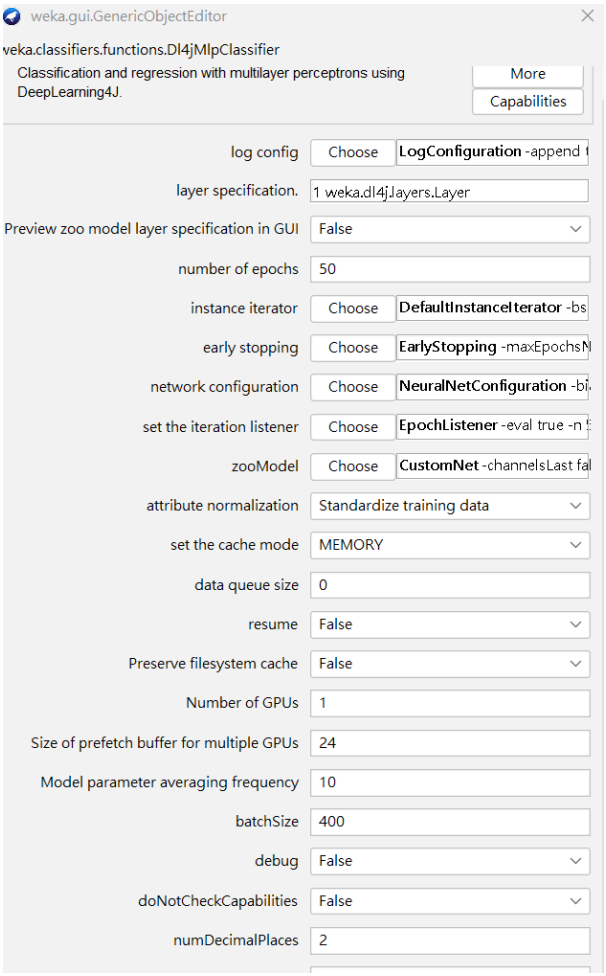
使用 image filter 的 ColorLayoutFilter 載入圖片

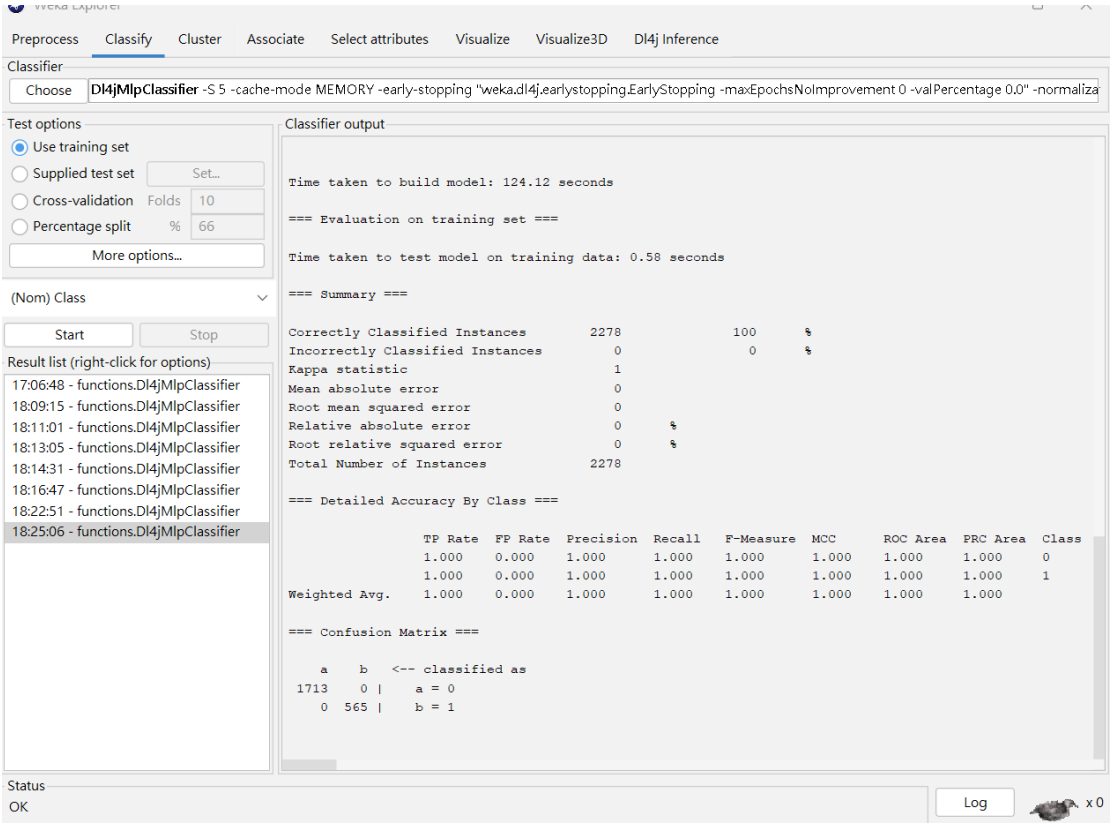


將檔名再轉回 nominal

使用 classify 的 Dl4

參數調整:Batch size 200 seed 5 epoch25


weka.gui.GenericObjectEditor
veka.classifiers.functions.Dl4jMlpClassifier
Classification and regression with multilayer perceptrons using DeepLearning4J.
More
Capabilities
log config
Choose
LogConfiguration -append
layer specification.
1 weka.dl4j.layers.Layer
Preview zoo model layer specification in GUI
False
number of epochs
50
instance iterator
Choose
DefaultInstanceIterator -bs
early stopping
Choose
EarlyStopping -maxEpochsN
network configuration
Choose
NeuralNetConfiguration -bi
set the iteration listener
Choose
EpochListener -eval true -n
zooModel
Choose
CustomNet -channelsLast fal
attribute normalization
Standardize training data
set the cache mode
MEMORY
data queue size
0
resume
False
Preserve filesystem cache
False
Number of GPUs
1
Size of prefetch buffer for multiple GPUs
24
Model parameter averaging frequency
10
batchSize
400
debug
False
doNotCheckCapabilities
False
numDecimalPlaces
2


Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Visualize3D
Dl4j Inference
Classifier
Choose
Dl4jMlpClassifier -S 5 -cache-mode MEMORY -early-stopping "weka.dl4j.earlystopping.EarlyStopping -maxEpochsNoImprovement 0 -valPercentage 0.0" -normaliza
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) Class
Start
Stop
Result list (right-click for options)
17:06:48 - functions.Dl4jMlpClassifier
18:09:15 - functions.Dl4jMlpClassifier
18:11:01 - functions.Dl4jMlpClassifier
18:13:05 - functions.Dl4jMlpClassifier
18:14:31 - functions.Dl4jMlpClassifier
18:16:47 - functions.Dl4jMlpClassifier
18:22:51 - functions.Dl4jMlpClassifier
18:25:06 - functions.Dl4jMlpClassifier
Classifier output
Time taken to build model: 124.12 seconds
=== Evaluation on training set ===
Time taken to test model on training data: 0.58 seconds
=== Summary ===
Correctly Classified Instances 2278 100 %
Incorrectly Classified Instances 0 0 %
Kappa statistic 1
Mean absolute error 0
Root mean squared error 0
Relative absolute error 0 %
Root relative squared error 0 %
Total Number of Instances 2278
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000 0
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000 1
Weighted Avg. 1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000
=== Confusion Matrix ===
a b <-- classified as
1713 0 | a = 0
0 565 | b = 1
Status
OK
Log
x 0

大統整

Mlp

Hidden layer	Learning rate	momentum	準確率
4,5,4	0.2	0.3	75.426%
4,10,4	0.2	0.3	75.1915
4,10,4	0.1	0.1	97.454%
4,10,4	0.05	0.05	97.7612%最高

Random forest

seed	準確率
1	97.2181%
2	99.6927%
3	99.6927% 最高

Cnn

epoch	Batch size	seed	準確率
10	100	1	94.732%
15	100	1	94.7761%
20	100	5	95.3029%
25	100	5	95.9614%
25	200	5	95.9614%
30	400	5	97.8929%
50	400	5	100% 最高

使用模型優缺點

多層感知器（MLP）的優缺點

優點：

1. 靈活性高：MLP 是一種多層神經網絡模型，具有多個隱藏層，可以學習非線性關係，適用於各種複雜的問題。
2. 可以處理多類別問題：MLP 可以輕鬆處理多類別分類問題，通過適當的激活函數和輸出層設計。
3. 可以進行端對端學習：MLP 可以通過反向傳播算法進行端對端學習，自動學習特徵表示和模式識別。

缺點：

1. 需要大量的數據：MLP 通常需要大量的訓練數據才能獲得良好的性能，尤其在複雜的問題上。
2. 容易過擬合：MLP 模型的參數量較多，容易在訓練過程中出現過擬合現象，需要適當的正則化和調優。
3. 對初始值敏感：MLP 的性能可能受到初始權重和偏差的影響，不同的初始值可能導致不同的結果，需要謹慎調整。

隨機森林（Random Forest）的優缺點

優點：

1. 優秀的預測性能：隨機森林通過集成多個決策樹，可以獲得較高的預測準確性，並且對於處理高維度數據效果較好。
2. 防止過擬合：隨機森林使用自助抽樣和特徵隨機選擇，有效地減少了模型的方差，降低了過擬合的風險。
3. 能夠處理大數據集：隨機森林可以並行處理大型數據集，加速了模型的訓練過程。

缺點：

1. 較大的存儲空間：隨機森林需要存儲多個決策樹，佔用較大的存儲空間，尤其是在樹的數量很大時。
2. 參數調優較為困難：隨機森林具有多個超參數，需要進行參數調優以獲得最佳的性能。
3. 預測速度較慢：隨機森林的預測速度較慢，特別是當樹的數量很大時，需要遍歷多個決策樹進行預測。

卷積神經網絡（CNN）的優缺點

優點：

1. 對圖像等結構化數據的處理效果好：CNN 在處理圖像等結構化數據方面具有優勢，能夠自動學習圖像中的特徵和模式。
2. 參數共享和平移不變性：CNN 中的卷積層和池化層具有參數共享和平移不變性的特點，減少了模型的參數量，同時能夠捕捉局部特徵。
3. 適用於大規模數據：CNN 能夠有效處理大規模數據集，並且通過 GPU 加速可以實現高效的訓練和預測。

缺點：

1. 需要大量的訓練數據：CNN 通常需要大量的訓練數據才能獲得良好的性能，特別是在複雜的問題上。
2. 訓練時間較長：由於 CNN 具有較大的模型容量，訓練時間通常較長，尤其是在更深的網絡結構中。
3. 可解釋性較差：CNN 由於其複雜的結構和內部特性，其模型的可解釋性相對較差，難以理解模型內部的決策過程。

改進之處

多層感知器（MLP）的改進方法：

1. 正則化技術：使用 L1 或 L2 正則化，或者使用 Dropout 等方法來減少過擬合問題。
2. 優化算法的選擇：嘗試不同的優化算法，如 Adam、RMSprop 等，以提高收斂速度和性能。
3. 調整模型結構：增加或減少隱藏層的數量，調整隱藏層的神經元數量，以找到更適合問題的模型結構。

隨機森林（Random Forest）的改進方法：

1. 調整樹的數量：增加或減少森林中樹的數量，通過調整樹的數量可以控制模型的複雜度和預測性能。
2. 特徵選擇：使用特徵選擇算法，如信息增益、基尼指數等，選擇最具有區分能力的特徵，以提高模型的準確性和泛化能力。
3. 集成學習技術：嘗試使用其他集成學習方法，如梯度提升樹（Gradient Boosting Tree），以進一步提升模型的性能。

卷積神經網絡（CNN）的改進方法：

1. 增加深度：增加網絡的深度可以提高模型的表示能力和學習能力，但需要注意避免過度擬合問題。
2. 使用預訓練模型：可以使用預訓練的模型，如 VGG、ResNet 等，在大規模數據集上進行微調，以提高性能和加速訓練過程。
3. 優化算法的選擇：與 MLP 相似，選擇合適的優化算法，調整學習率和批次大小等參數，以提高訓練效率和性能。

選擇哪個最好

我認為是 CNN，畢竟是處理圖像，mlp 跟 random forest 比較適合用來處理非線性問題，在 weka 做的時候也是 CNN 表現較好