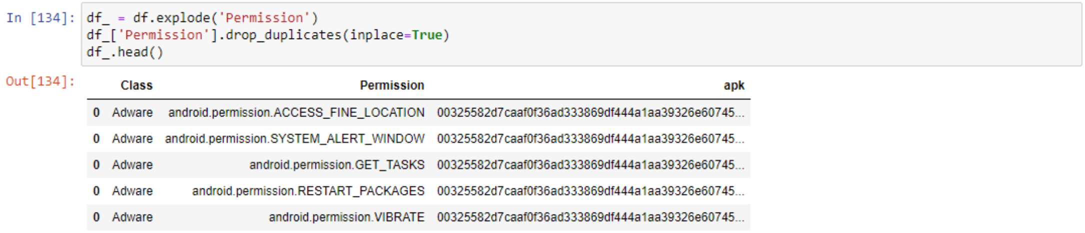
# ETC_HW8_107403020 李泳輝

## 資料前處理

1. 使用 explode 將 Permision 的 list 展開

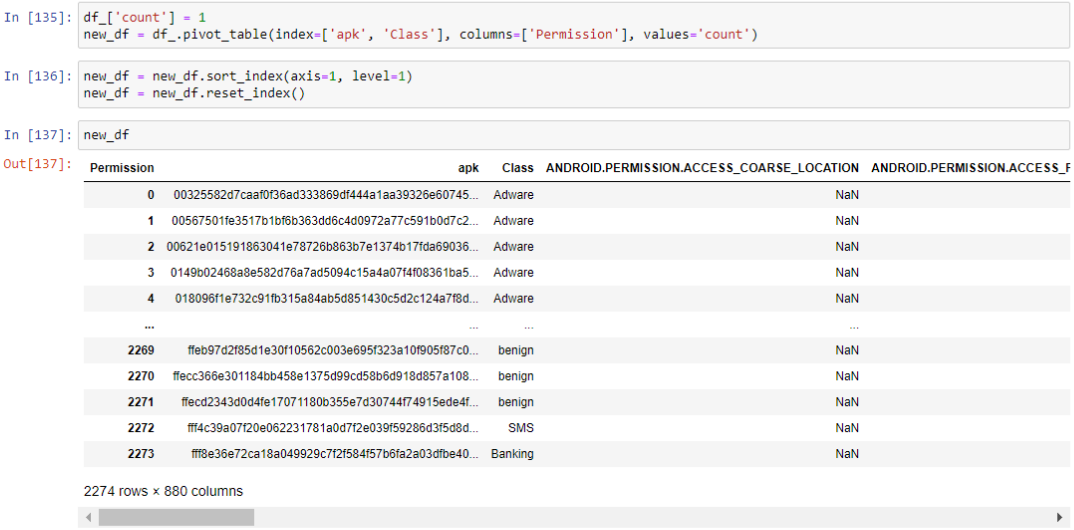


```
In [134]: df_ = df.explode('Permission')
          df_['Permission'].drop_duplicates(inplace=True)
          df_.head()
```

Out[134]:

| | Class | Permission | apk |
|---|---|---|---|
| 0 | Adware | android.permission.ACCESS_FINE_LOCATION | 00325582d7caaf0f36ad333869df444a1aa39326e60745... |
| 0 | Adware | android.permission.SYSTEM_ALERT_WINDOW | 00325582d7caaf0f36ad333869df444a1aa39326e60745... |
| 0 | Adware | android.permission.GET_TASKS | 00325582d7caaf0f36ad333869df444a1aa39326e60745... |
| 0 | Adware | android.permission.RESTART_PACKAGES | 00325582d7caaf0f36ad333869df444a1aa39326e60745... |
| 0 | Adware | android.permission.VIBRATE | 00325582d7caaf0f36ad333869df444a1aa39326e60745... |

2. 根據 apk 和 class 將展開後的 Permission 重新分配到相應的欄位。



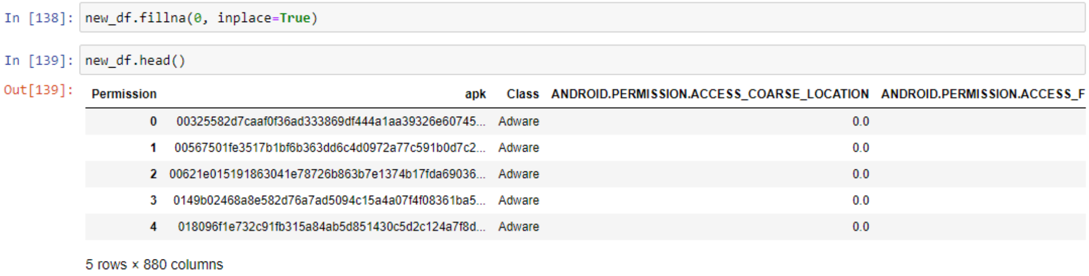
```
In [135]: df_['count'] = 1
          new_df = df_.pivot_table(index=['apk', 'Class'], columns=['Permission'], values='count')

In [136]: new_df = new_df.sort_index(axis=1, level=1)
          new_df = new_df.reset_index()

In [137]: new_df
```

Out[137]:

| Permission | apk | Class | ANDROID.PERMISSION.ACCESS_COARSE_LOCATION | ANDROID.PERMISSION.ACCESS_F |
|---|---|---|---|---|
| 0 | 00325582d7caaf0f36ad333869df444a1aa39326e60745... | Adware | NaN | |
| 1 | 00567501fe3517b1bf6b363dd6c4d0972a77c591b0d7c2... | Adware | NaN | |
| 2 | 00621e015191863041e78726b863b7e1374b17fda69036... | Adware | NaN | |
| 3 | 0149b02468a8e582d76a7ad5094c15a4a07f4f08361ba5... | Adware | NaN | |
| 4 | 018096f1e732c91fb315a84ab5d851430c5d2c124a7f8d... | Adware | NaN | |
| ... | ... | ... | ... | |
| 2269 | ffeb97d2f85d1e30f10562c003e695f323a10f905f87c0... | benign | NaN | |
| 2270 | ffecc366e301184bb458e1375d99cd58b6d918d857a108... | benign | NaN | |
| 2271 | ffecd2343d0d4fe17071180b355e7d30744f74915ede4f... | benign | NaN | |
| 2272 | fff4c39a07f20e062231781a0d7f2e039f59286d3f5d8d... | SMS | NaN | |
| 2273 | fff8e36e72ca18a049929c7f2f584f57b6fa2a03dfbe40... | Banking | NaN | |

2274 rows × 880 columns

3. 填補空值



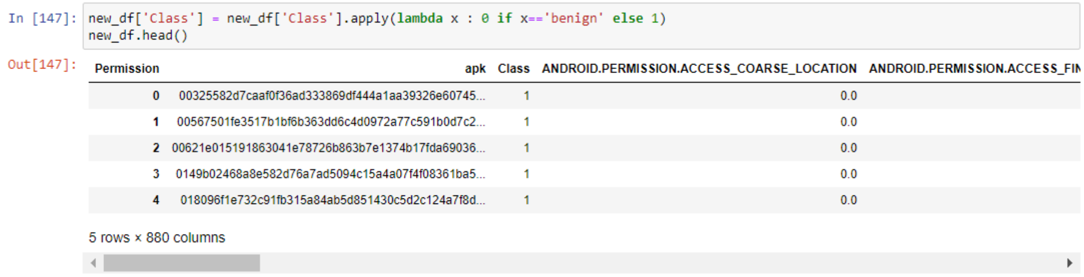
```
In [138]: new_df.fillna(0, inplace=True)

In [139]: new_df.head()
```

Out[139]:

| Permission | apk | Class | ANDROID.PERMISSION.ACCESS_COARSE_LOCATION | ANDROID.PERMISSION.ACCESS_F |
|---|---|---|---|---|
| 0 | 00325582d7caaf0f36ad333869df444a1aa39326e60745... | Adware | 0.0 | |
| 1 | 00567501fe3517b1bf6b363dd6c4d0972a77c591b0d7c2... | Adware | 0.0 | |
| 2 | 00621e015191863041e78726b863b7e1374b17fda69036... | Adware | 0.0 | |
| 3 | 0149b02468a8e582d76a7ad5094c15a4a07f4f08361ba5... | Adware | 0.0 | |
| 4 | 018096f1e732c91fb315a84ab5d851430c5d2c124a7f8d... | Adware | 0.0 | |

5 rows × 880 columns

4. 將 Class 改成數值，benign 為 0，其餘為 1



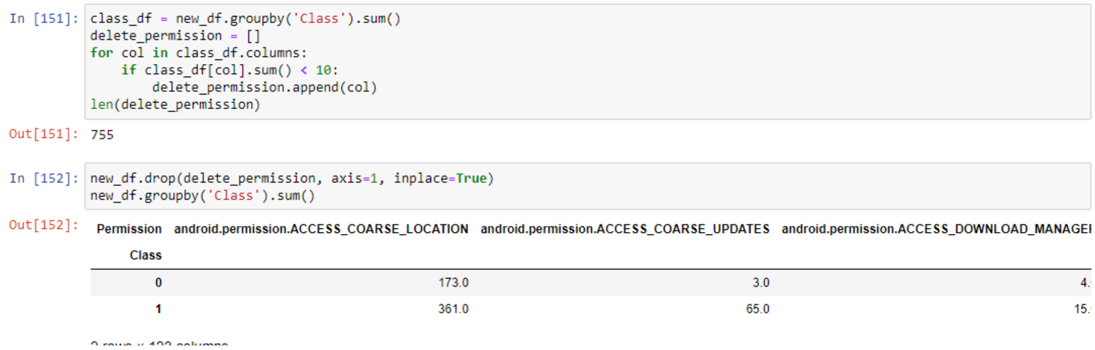
In [147]:
```python
new_df['Class'] = new_df['Class'].apply(lambda x : 0 if x=='benign' else 1)
new_df.head()
```

Out[147]:

| Permission | apk | Class | ANDROID.PERMISSION.ACCESS_COARSE_LOCATION | ANDROID.PERMISSION.ACCESS_FIN |
|---|---|---|---|---|
| **0** | 00325582d7caaf0f36ad333869df444a1aa39326e60745... | 1 | 0.0 | |
| **1** | 00567501fe3517b1bf6b363dd6c4d0972a77c591b0d7c2... | 1 | 0.0 | |
| **2** | 00621e015191863041e78726b863b7e1374b17fda69036... | 1 | 0.0 | |
| **3** | 0149b02468a8e582d76a7ad5094c15a4a07f4f08361ba5... | 1 | 0.0 | |
| **4** | 018096f1e732c91fb315a84ab5d851430c5d2c124a7f8d... | 1 | 0.0 | |

5 rows × 880 columns

## 5. 因為欄位數太多，留下 permission 出現次數超過 5 的欄位



In [151]:
```python
class_df = new_df.groupby('Class').sum()
delete_permission = []
for col in class_df.columns:
    if class_df[col].sum() < 10:
        delete_permission.append(col)
len(delete_permission)
```

Out[151]:
```
755
```

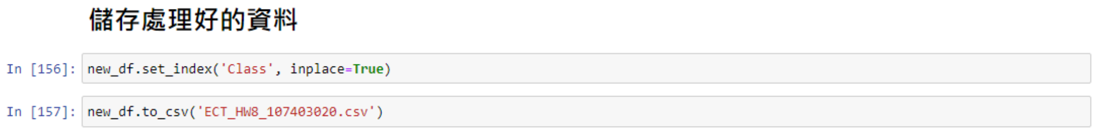
In [152]:
```python
new_df.drop(delete_permission, axis=1, inplace=True)
new_df.groupby('Class').sum()
```

Out[152]:

| Permission | android.permission.ACCESS_COARSE_LOCATION | android.permission.ACCESS_COARSE_UPDATES | android.permission.ACCESS_DOWNLOAD_MANAGEI |
|---|---|---|---|
| **Class** | | | |
| **0** | 173.0 | 3.0 | 4. |
| **1** | 361.0 | 65.0 | 15. |

2 rows × 123 columns

## 6. 將處理好的資料儲存

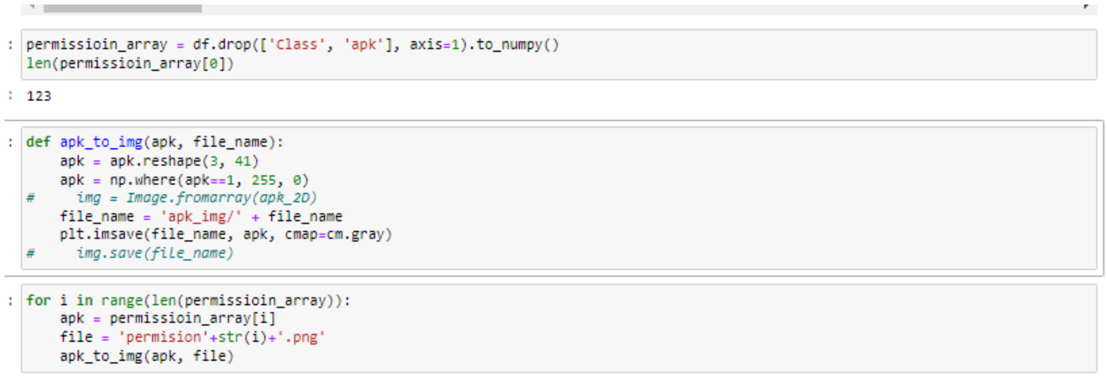


### 儲存處理好的資料

In [156]:
```python
new_df.set_index('Class', inplace=True)
```

In [157]:
```python
new_df.to_csv('ECT_HW8_107403020.csv')
```

## 7. 將 apk 轉圖片



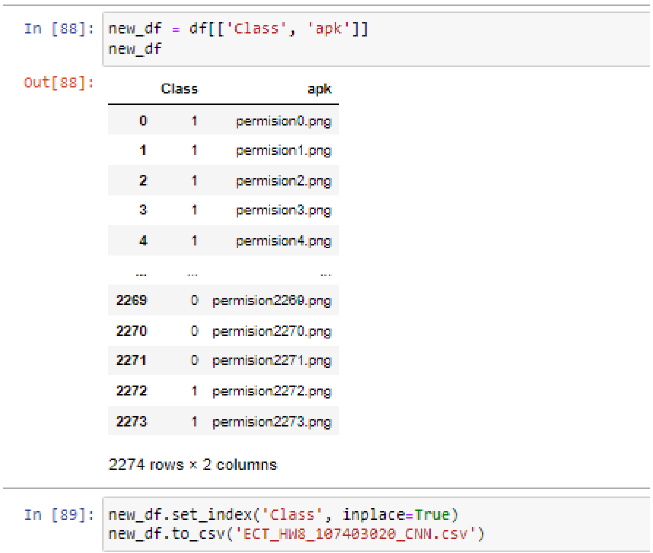
```python
permissioin_array = df.drop(['Class', 'apk'], axis=1).to_numpy()
len(permissioin_array[0])
```

```
123
```

```python
def apk_to_img(apk, file_name):
    apk = apk.reshape(3, 41)
    apk = np.where(apk==1, 255, 0)
#    img = Image.fromarray(apk_2D)
    file_name = 'apk_img/' + file_name
    plt.imsave(file_name, apk, cmap=cm.gray)
#    img.save(file_name)
```

```python
for i in range(len(permissioin_array)):
    apk = permissioin_array[i]
    file = 'permision'+str(i)+'.png'
    apk_to_img(apk, file)
```

8. 將處理好的資料儲存



In [88]:

```
new_df = df[['Class', 'apk']]
new_df
```

Out[88]:

| | Class | apk |
|---|---|---|
| **0** | 1 | permision0.png |
| **1** | 1 | permision1.png |
| **2** | 1 | permision2.png |
| **3** | 1 | permision3.png |
| **4** | 1 | permision4.png |
| **...** | ... | ... |
| **2269** | 0 | permision2269.png |
| **2270** | 0 | permision2270.png |
| **2271** | 0 | permision2271.png |
| **2272** | 1 | permision2272.png |
| **2273** | 1 | permision2273.png |

2274 rows × 2 columns

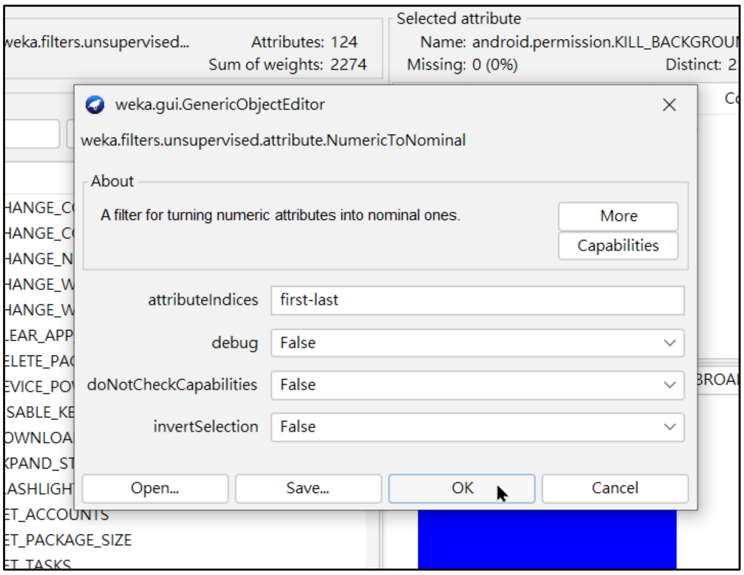
In [89]:

```
new_df.set_index('Class', inplace=True)
new_df.to_csv('ECT_HW8_107403020_CNN.csv')
```

# 訓練模型

1. 第一類資料，用來訓練 RandomForest、MLP

numeric_to_nominal



weka.filters.unsupervised... Attributes: 124
Sum of weights: 2274

Selected attribute
Name: android.permission.KILL_BACKGROUI
Missing: 0 (0%) Distinct: 2

weka.gui.GenericObjectEditor

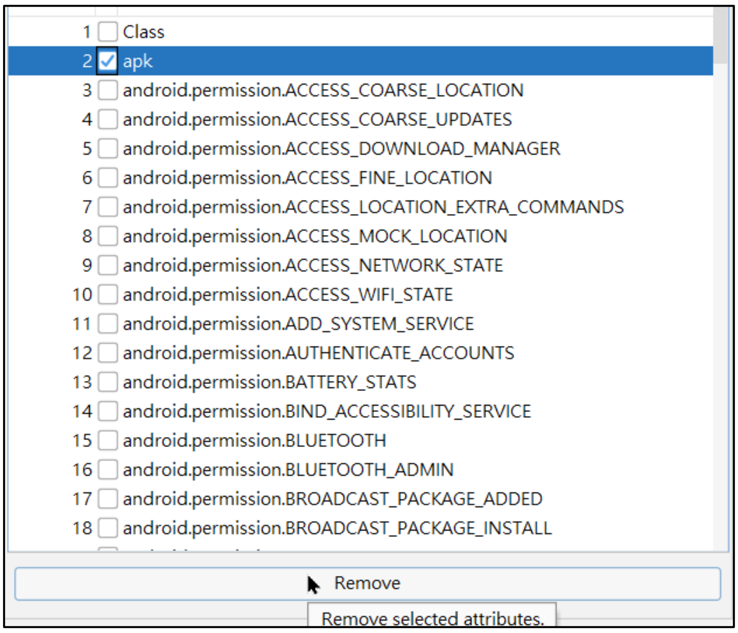
weka.filters.unsupervised.attribute.NumericToNominal

About

A filter for turning numeric attributes into nominal ones.

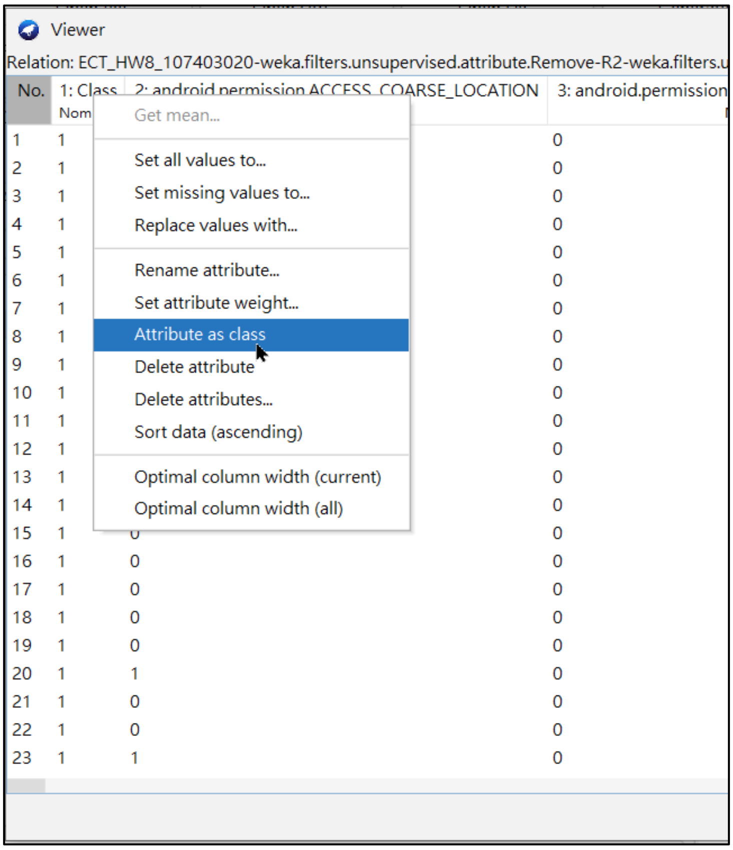
| | |
|---|---|
| attributeIndices | first-last |
| debug | False |
| doNotCheckCapabilities | False |
| invertSelection | False |

去除 apk 欄位



1 □ Class
2 ☑ apk
3 □ android.permission.ACCESS_COARSE_LOCATION
4 □ android.permission.ACCESS_COARSE_UPDATES
5 □ android.permission.ACCESS_DOWNLOAD_MANAGER
6 □ android.permission.ACCESS_FINE_LOCATION
7 □ android.permission.ACCESS_LOCATION_EXTRA_COMMANDS
8 □ android.permission.ACCESS_MOCK_LOCATION
9 □ android.permission.ACCESS_NETWORK_STATE
10 □ android.permission.ACCESS_WIFI_STATE
11 □ android.permission.ADD_SYSTEM_SERVICE
12 □ android.permission.AUTHENTICATE_ACCOUNTS
13 □ android.permission.BATTERY_STATS
14 □ android.permission.BIND_ACCESSIBILITY_SERVICE
15 □ android.permission.BLUETOOTH
16 □ android.permission.BLUETOOTH_ADMIN
17 □ android.permission.BROADCAST_PACKAGE_ADDED
18 □ android.permission.BROADCAST_PACKAGE_INSTALL

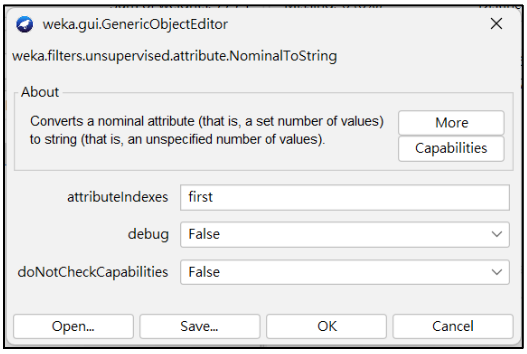
設定 Class



Relation: ECT_HW8_107403020-weka.filters.unsupervised.attribute.Remove-R2-weka.filters.u

| No. | 1: Class<br>Nom | 2: android.permission.ACCESS_COARSE_LOCATION | 3: android.permission |
|---|---|---|---|
| 1 | 1 | | 0 |
| 2 | 1 | | 0 |
| 3 | 1 | | 0 |
| 4 | 1 | | 0 |
| 5 | 1 | | 0 |
| 6 | 1 | | 0 |
| 7 | 1 | | 0 |
| 8 | 1 | | 0 |
| 9 | 1 | | 0 |
| 10 | 1 | | 0 |
| 11 | 1 | | 0 |
| 12 | 1 | | 0 |
| 13 | 1 | | 0 |
| 14 | 1 | | 0 |
| 15 | 1 | 0 | 0 |
| 16 | 1 | 0 | 0 |
| 17 | 1 | 0 | 0 |
| 18 | 1 | 0 | 0 |
| 19 | 1 | 0 | 0 |
| 20 | 1 | 1 | 0 |
| 21 | 1 | 0 | 0 |
| 22 | 1 | 0 | 0 |
| 23 | 1 | 1 | 0 |

2. 第二類資料 apk, Class (apk 放的是圖片位置)，用來訓練 CNN
將 apk 轉 string



weka.filters.unsupervised.attribute.NominalToString

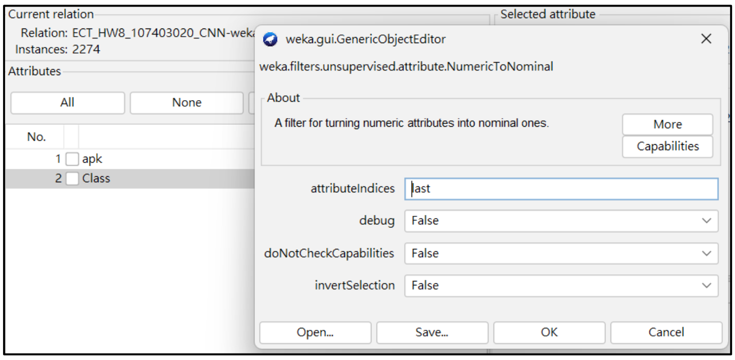
Converts a nominal attribute (that is, a set number of values) to string (that is, an unspecified number of values).

attributeIndexes: first

debug: False

doNotCheckCapabilities: False

numeric_to_nominal



Relation: ECT_HW8_107403020_CNN-weka

Instances: 2274

1 □ apk

2 □ Class

weka.filters.unsupervised.attribute.NumericToNominal

A filter for turning numeric attributes into nominal ones.

attributeIndices: last

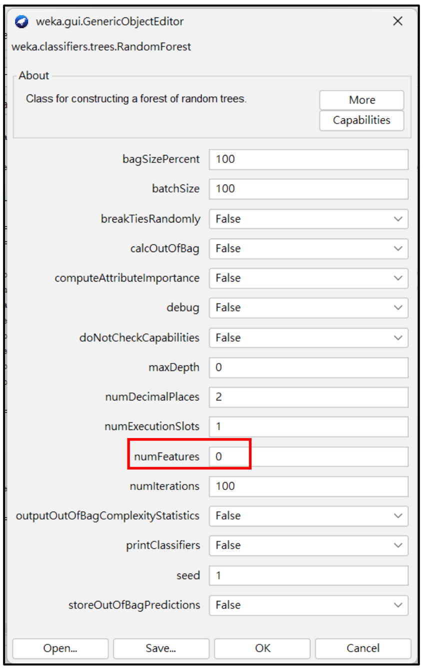
debug: False

doNotCheckCapabilities: False

invertSelection: False

1 . RandomForest

超參數 採用系統預設值。使用 10 Cross-validation 來進行訓練。

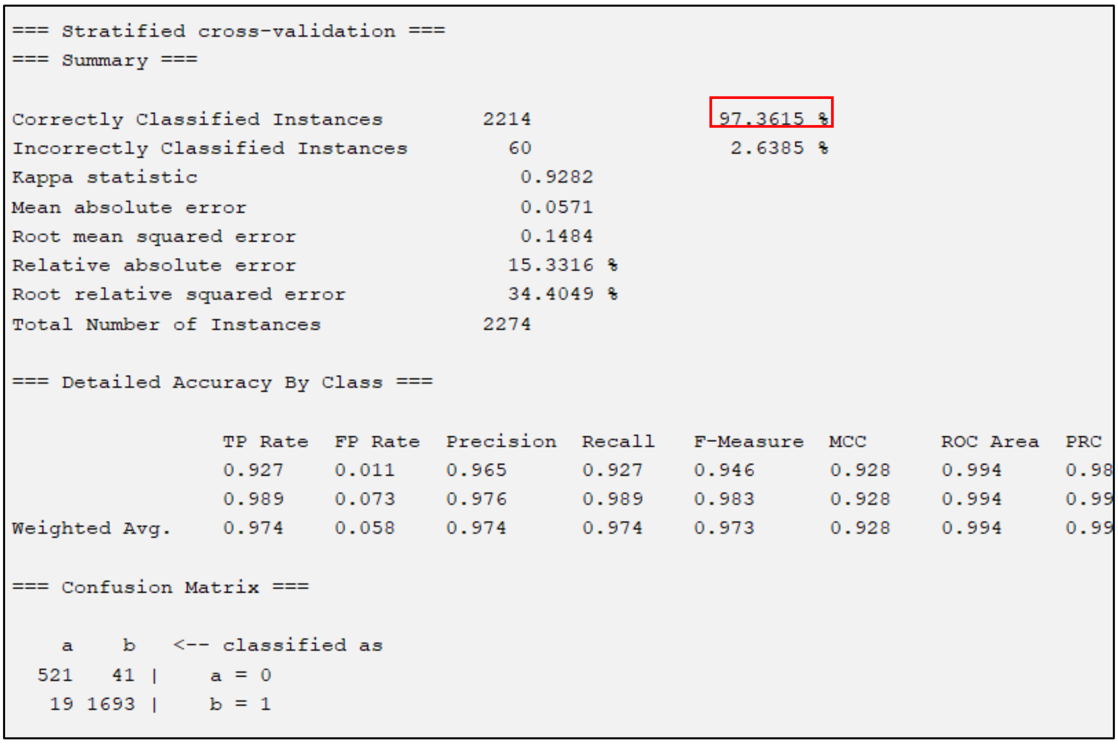


weka.classifiers.trees.RandomForest

About

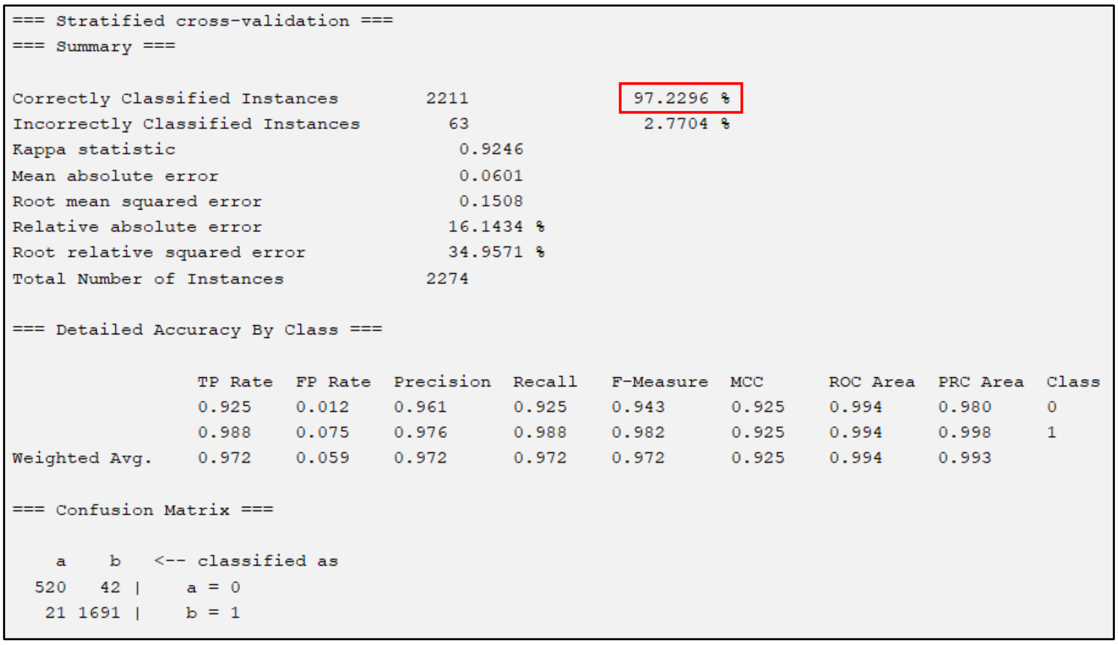
Class for constructing a forest of random trees.

| | |
|---|---|
| bagSizePercent | 100 |
| batchSize | 100 |
| breakTiesRandomly | False |
| calcOutOfBag | False |
| computeAttributeImportance | False |
| debug | False |
| doNotCheckCapabilities | False |
| maxDepth | 0 |
| numDecimalPlaces | 2 |
| numExecutionSlots | 1 |
| numFeatures | 0 |
| numIterations | 100 |
| outputOutOfBagComplexityStatistics | False |
| printClassifiers | False |
| seed | 1 |
| storeOutOfBagPredictions | False |



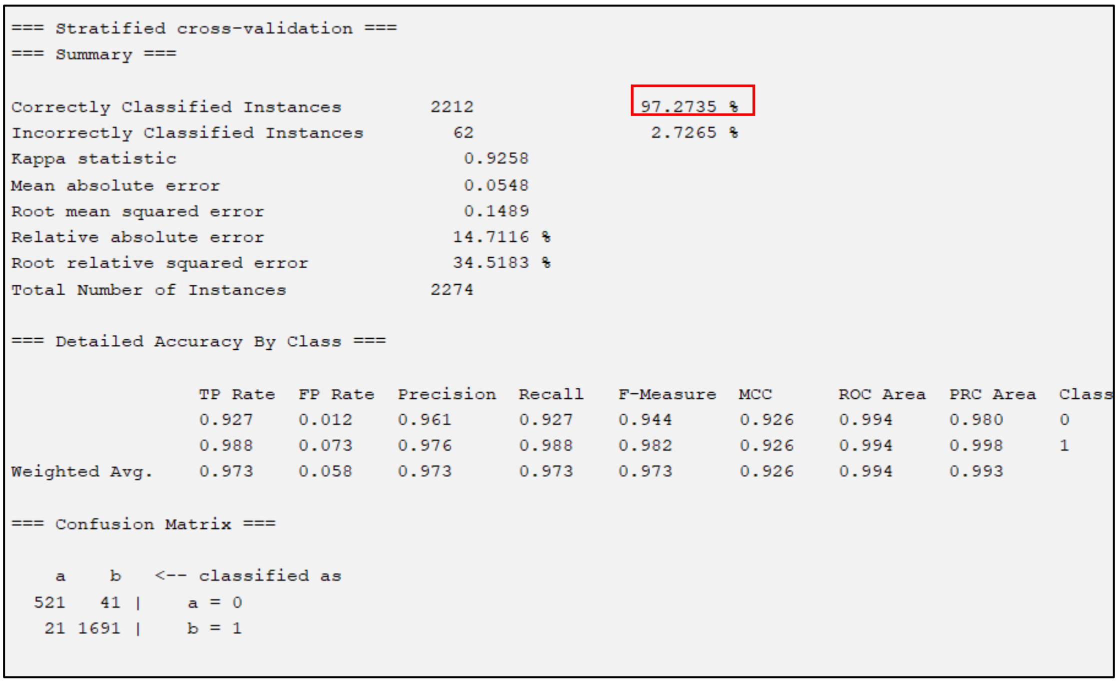
```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        2214               97.3615 %
Incorrectly Classified Instances        60                2.6385 %
Kappa statistic                          0.9282
Mean absolute error                      0.0571
Root mean squared error                  0.1484
Relative absolute error                 15.3316 %
Root relative squared error             34.4049 %
Total Number of Instances             2274

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC
                 0.927    0.011    0.965      0.927    0.946      0.928    0.994     0.98
                 0.989    0.073    0.976      0.989    0.983      0.928    0.994     0.99
Weighted Avg.    0.974    0.058    0.974      0.974    0.973      0.928    0.994     0.99

=== Confusion Matrix ===

    a    b   <-- classified as
  521   41 |    a = 0
   19 1693 |    b = 1
```

2. 將 numFeatures 設為 5，numFeatures 是設定有多少的 features 用於 random selection。



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        2211               97.2296 %
Incorrectly Classified Instances        63                2.7704 %
Kappa statistic                          0.9246
Mean absolute error                      0.0601
Root mean squared error                  0.1508
Relative absolute error                 16.1434 %
Root relative squared error             34.9571 %
Total Number of Instances             2274

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.925    0.012    0.961      0.925    0.943      0.925    0.994     0.980     0
                 0.988    0.075    0.976      0.988    0.982      0.925    0.994     0.998     1
Weighted Avg.    0.972    0.059    0.972      0.972    0.972      0.925    0.994     0.993

=== Confusion Matrix ===

    a    b   <-- classified as
  520   42 |    a = 0
   21 1691 |    b = 1
```

3. 將 numFeatures 設為 11

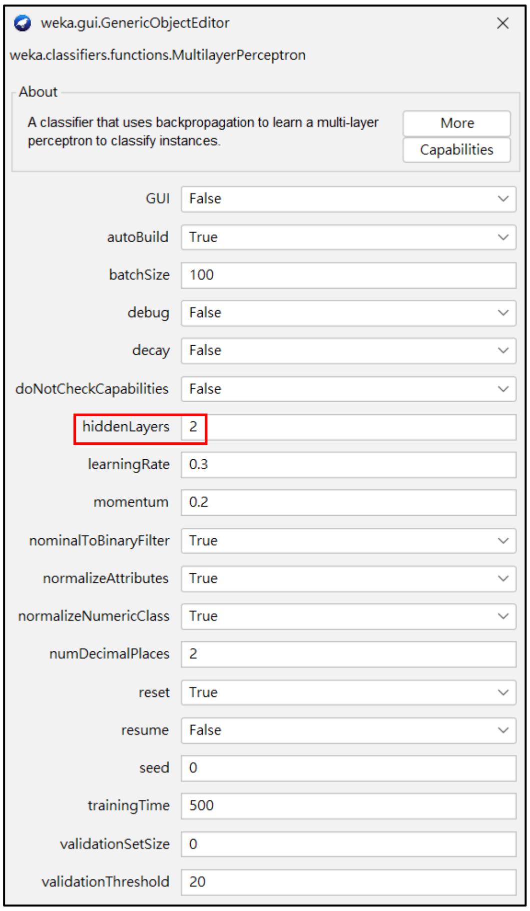


```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        2212               97.2735 %
Incorrectly Classified Instances        62                2.7265 %
Kappa statistic                          0.9258
Mean absolute error                      0.0548
Root mean squared error                  0.1489
Relative absolute error                 14.7116 %
Root relative squared error             34.5183 %
Total Number of Instances             2274

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.927    0.012    0.961      0.927    0.944      0.926    0.994     0.980     0
                 0.988    0.073    0.976      0.988    0.982      0.926    0.994     0.998     1
Weighted Avg.    0.973    0.058    0.973      0.973    0.973      0.926    0.994     0.993

=== Confusion Matrix ===

    a    b   <-- classified as
  521   41 |    a = 0
   21 1691 |    b = 1
```

2. MLP

使用 10 Cross-validation 來進行訓練。

超參數 ： 因為預設的模型 hidden layer 層數過多，訓練時間過久。將 hidden layer 改成 2，也就是指有一層 hidden layer，node 個數為 2。



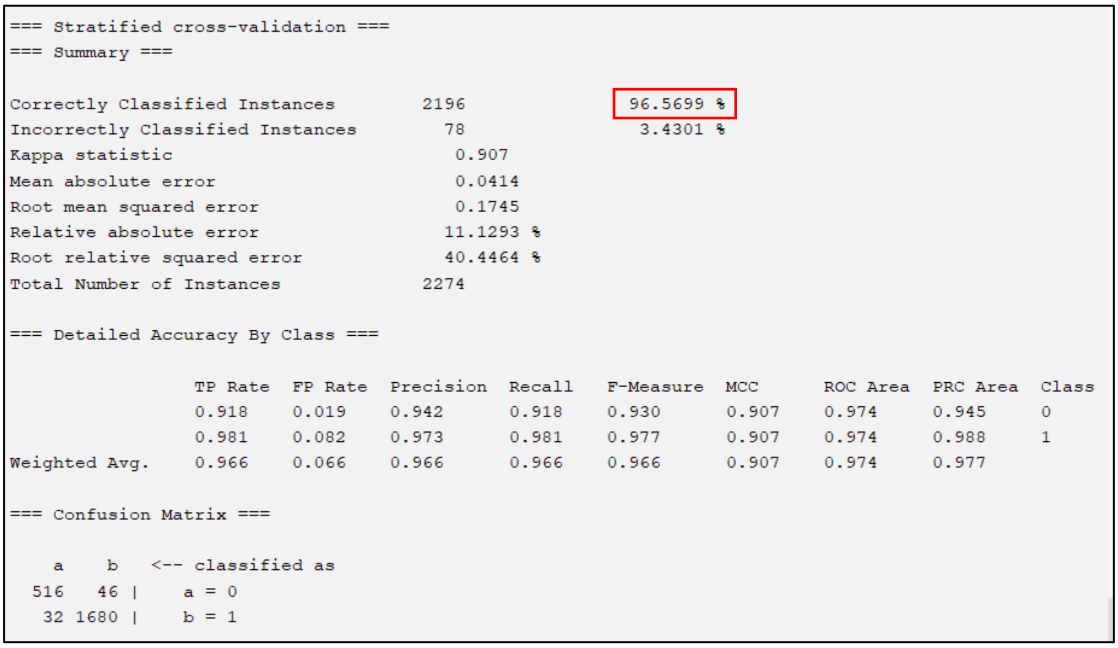
weka.gui.GenericObjectEditor

weka.classifiers.functions.MultilayerPerceptron

About

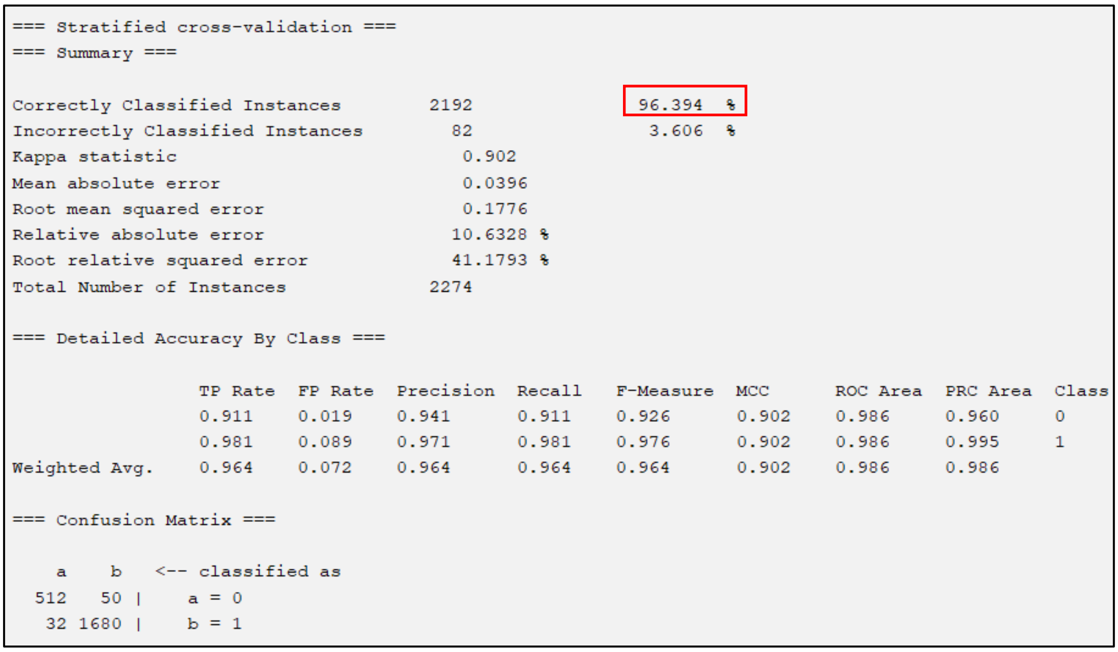
A classifier that uses backpropagation to learn a multi-layer perceptron to classify instances.

| | |
|---|---|
| GUI | False |
| autoBuild | True |
| batchSize | 100 |
| debug | False |
| decay | False |
| doNotCheckCapabilities | False |
| hiddenLayers | 2 |
| learningRate | 0.3 |
| momentum | 0.2 |
| nominalToBinaryFilter | True |
| normalizeAttributes | True |
| normalizeNumericClass | True |
| numDecimalPlaces | 2 |
| reset | True |
| resume | False |
| seed | 0 |
| trainingTime | 500 |
| validationSetSize | 0 |
| validationThreshold | 20 |



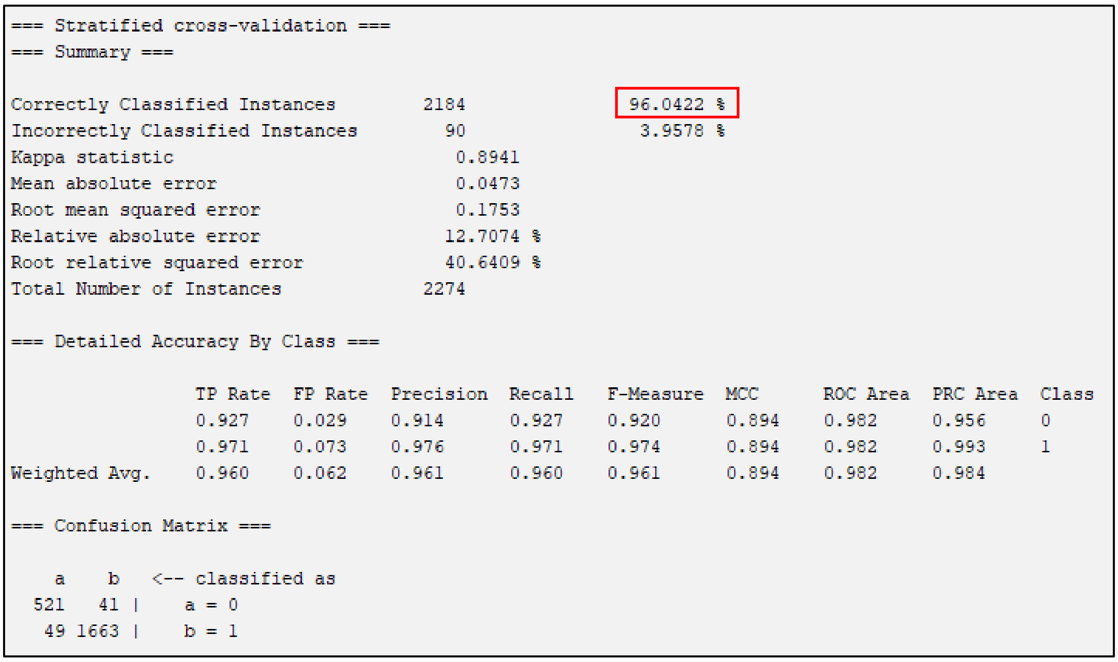
```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        2196               96.5699 %
Incorrectly Classified Instances        78                3.4301 %
Kappa statistic                          0.907
Mean absolute error                      0.0414
Root mean squared error                  0.1745
Relative absolute error                 11.1293 %
Root relative squared error             40.4464 %
Total Number of Instances             2274

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.918    0.019    0.942      0.918    0.930      0.907    0.974     0.945     0
                 0.981    0.082    0.973      0.981    0.977      0.907    0.974     0.988     1
Weighted Avg.    0.966    0.066    0.966      0.966    0.966      0.907    0.974     0.977

=== Confusion Matrix ===

    a    b   <-- classified as
  516   46 |    a = 0
   32 1680 |    b = 1
```

超參數 ：hidden layer 設為 10，有一層 hidden layer，node 個數為 10



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        2192               96.394  %
Incorrectly Classified Instances        82                3.606  %
Kappa statistic                          0.902
Mean absolute error                      0.0396
Root mean squared error                  0.1776
Relative absolute error                 10.6328 %
Root relative squared error             41.1793 %
Total Number of Instances             2274

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.911    0.019    0.941      0.911    0.926      0.902    0.986     0.960     0
                 0.981    0.089    0.971      0.981    0.976      0.902    0.986     0.995     1
Weighted Avg.    0.964    0.072    0.964      0.964    0.964      0.902    0.986     0.986

=== Confusion Matrix ===

    a    b   <-- classified as
  512   50 |    a = 0
   32 1680 |    b = 1
```

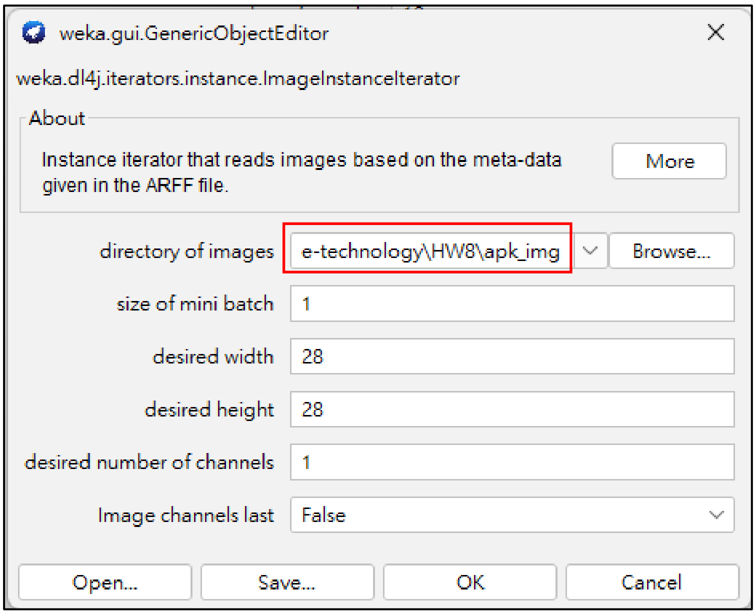
超參數 ：hidden layer 設為 10, 2，代表有兩層 hidden layer，第一層 node 個數為 10，第二層為 2



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        2184               96.0422 %
Incorrectly Classified Instances        90                3.9578 %
Kappa statistic                          0.8941
Mean absolute error                      0.0473
Root mean squared error                  0.1753
Relative absolute error                 12.7074 %
Root relative squared error             40.6409 %
Total Number of Instances             2274

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.927    0.029    0.914      0.927    0.920      0.894    0.982     0.956     0
                 0.971    0.073    0.976      0.971    0.974      0.894    0.982     0.993     1
Weighted Avg.    0.960    0.062    0.961      0.960    0.961      0.894    0.982     0.984

=== Confusion Matrix ===

    a    b   <-- classified as
  521   41 |    a = 0
   49 1663 |    b = 1
```

3. CNN

最大的問題是電腦效能，太複雜的模型電腦跑不動。CNN 考慮到時間因素，因此使用 percentage split 來將資料拆成訓練集、測試集，其中訓練集佔 66%。instance iterator 選擇 ImageInstanceIterator，讓 weka 會根據 apk 中放的檔名，到選定的資料夾讀取圖片 。



weka.gui.GenericObjectEditor

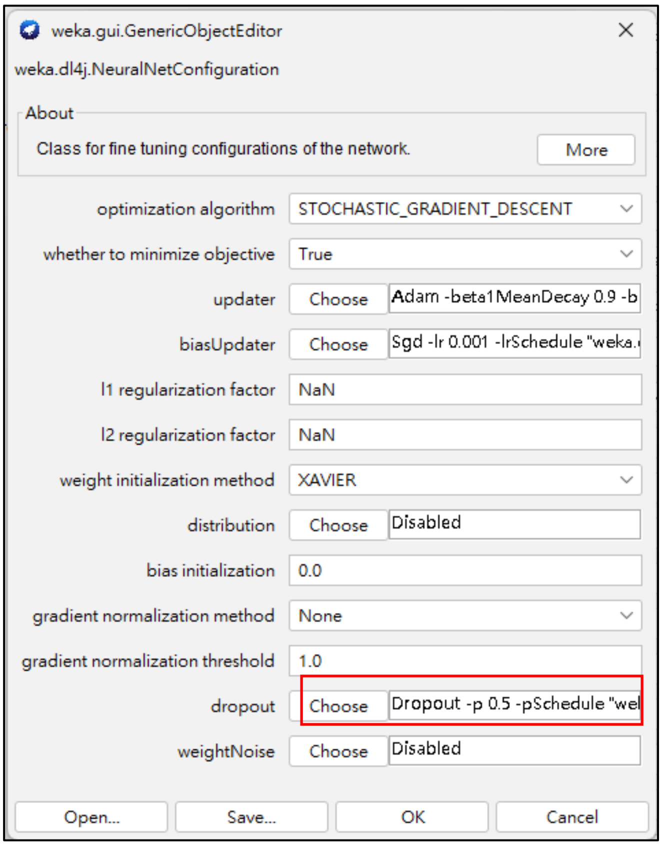
weka.dl4j.iterators.instance.ImageInstanceIterator

About

Instance iterator that reads images based on the meta-data given in the ARFF file.

| | |
|---|---|
| directory of images | e-technology\HW8\apk_img |
| size of mini batch | 1 |
| desired width | 28 |
| desired height | 28 |
| desired number of channels | 1 |
| Image channels last | False |

設定 dropout  訓練時每次會隨機丟棄 0.5%的神經節點。



weka.gui.GenericObjectEditor

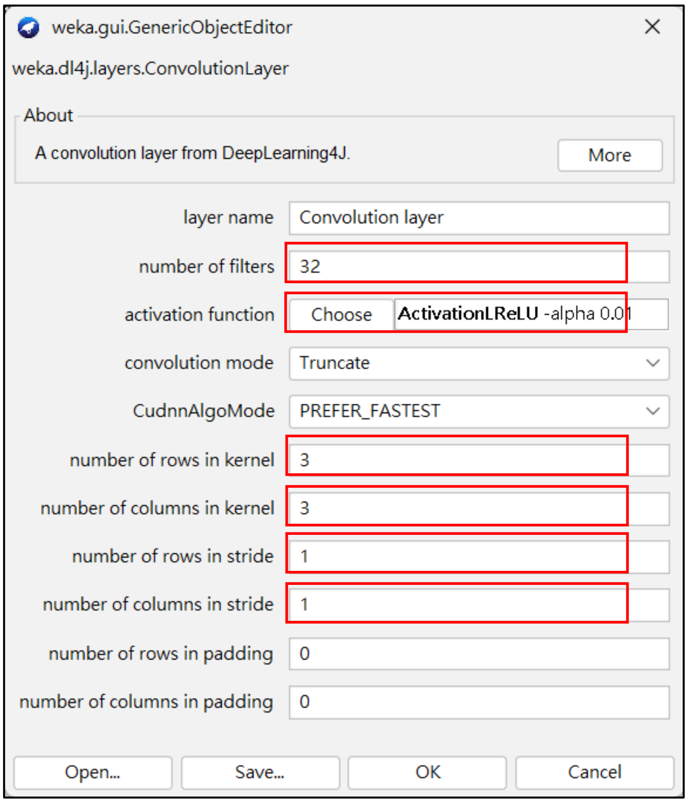
weka.dl4j.NeuralNetConfiguration

About

Class for fine tuning configurations of the network.

| | |
|---|---|
| optimization algorithm | STOCHASTIC_GRADIENT_DESCENT |
| whether to minimize objective | True |
| updater | Adam -beta1MeanDecay 0.9 -b |
| biasUpdater | Sgd -lr 0.001 -lrSchedule "weka.( |
| l1 regularization factor | NaN |
| l2 regularization factor | NaN |
| weight initialization method | XAVIER |
| distribution | Disabled |
| bias initialization | 0.0 |
| gradient normalization method | None |
| gradient normalization threshold | 1.0 |
| dropout | Dropout -p 0.5 -pSchedule "wel |
| weightNoise | Disabled |

- convolution layer 的 nOut 是 filiter 的個數，activation 設定成 ReLU。在此的捲積層 kernel 均是 3X3，stride 均設為 1。



weka.gui.GenericObjectEditor

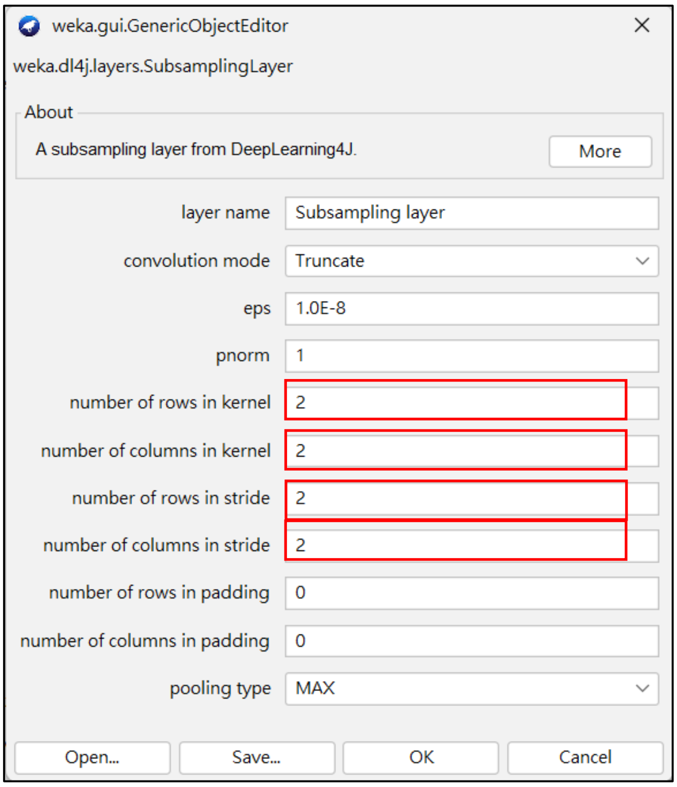
weka.dl4j.layers.ConvolutionLayer

About

A convolution layer from DeepLearning4J.

| | |
|---|---|
| layer name | Convolution layer |
| number of filters | 32 |
| activation function | Choose ActivationLReLU -alpha 0.01 |
| convolution mode | Truncate |
| CudnnAlgoMode | PREFER_FASTEST |
| number of rows in kernel | 3 |
| number of columns in kernel | 3 |
| number of rows in stride | 1 |
| number of columns in stride | 1 |
| number of rows in padding | 0 |
| number of columns in padding | 0 |

- subsampling 設定為 Max，會將 filiter 得到的 feature map 做壓縮。kernel 均設為 2X2，stride 為 2。



weka.gui.GenericObjectEditor

weka.dl4j.layers.SubsamplingLayer

About

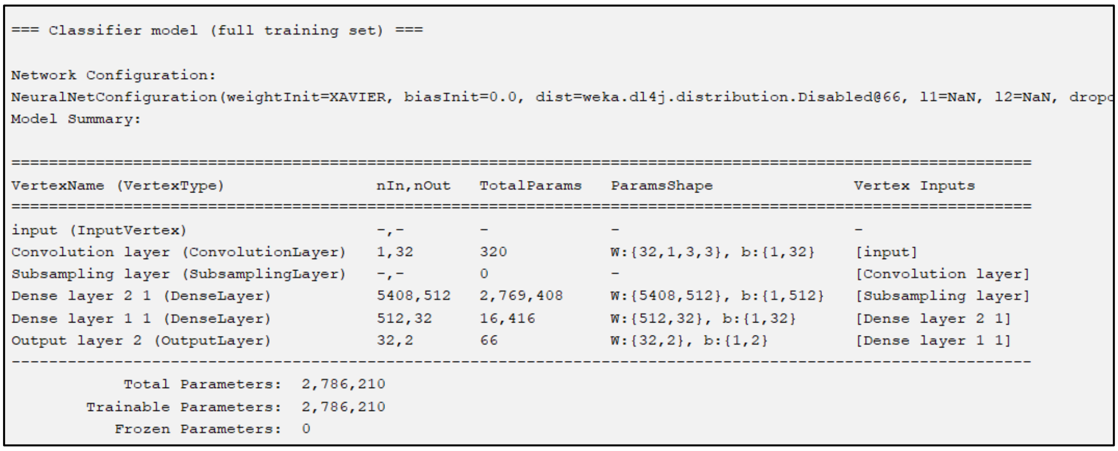
A subsampling layer from DeepLearning4J.

| | |
|---|---|
| layer name | Subsampling layer |
| convolution mode | Truncate |
| eps | 1.0E-8 |
| pnorm | 1 |
| number of rows in kernel | 2 |
| number of columns in kernel | 2 |
| number of rows in stride | 2 |
| number of columns in stride | 2 |
| number of rows in padding | 0 |
| number of columns in padding | 0 |
| pooling type | MAX |

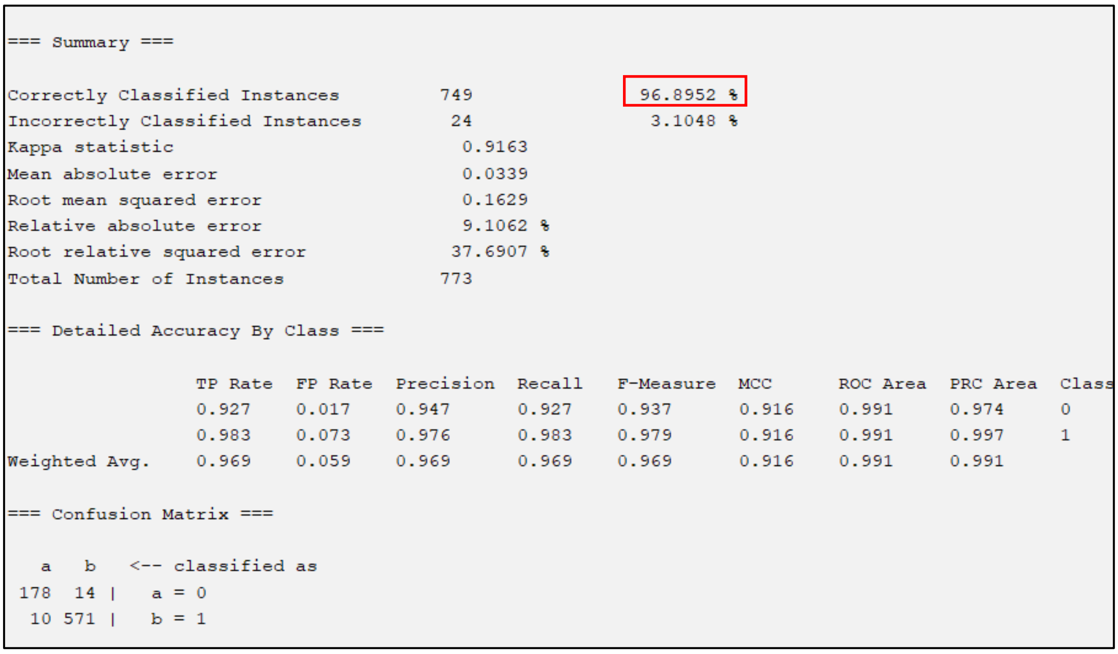
- 第一層的 Dense 會把上一層所有的 feature map 展開，排成一列。activation 設定成 ReLU。
- 最後一層的 output layer 的 Loss function 設定成 LossNegativeLogLikelihood。

模型一:

一層 convlution:



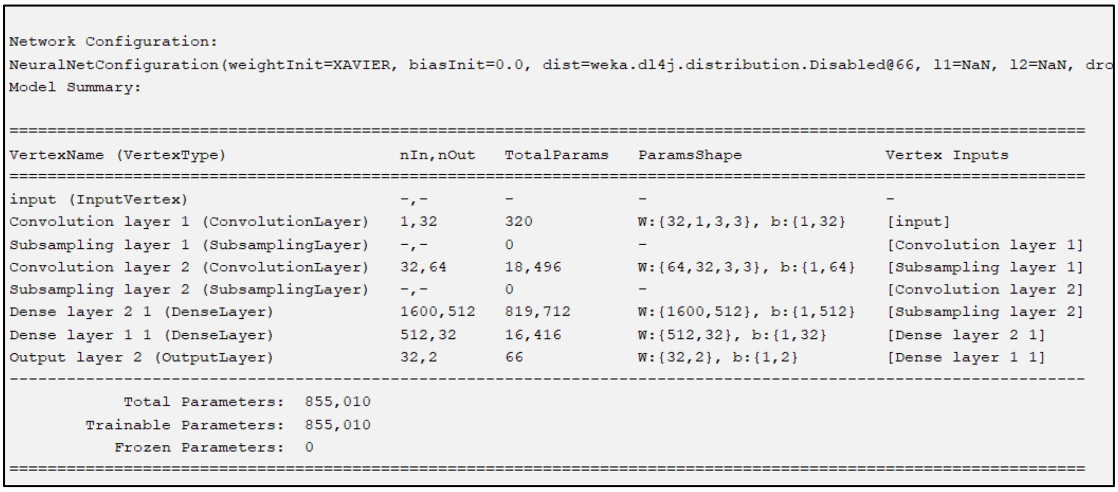
```
=== Classifier model (full training set) ===

Network Configuration:
NeuralNetConfiguration(weightInit=XAVIER, biasInit=0.0, dist=weka.dl4j.distribution.Disabled@66, l1=NaN, l2=NaN, dropo
Model Summary:

=============================================================================================================
VertexName (VertexType)               nIn,nOut   TotalParams   ParamsShape                Vertex Inputs
=============================================================================================================
input (InputVertex)                   -,-        -             -                          -
Convolution layer (ConvolutionLayer)  1,32       320           W:{32,1,3,3}, b:{1,32}     [input]
Subsampling layer (SubsamplingLayer)  -,-        0             -                          [Convolution layer]
Dense layer 2 1 (DenseLayer)          5408,512   2,769,408     W:{5408,512}, b:{1,512}    [Subsampling layer]
Dense layer 1 1 (DenseLayer)          512,32     16,416        W:{512,32}, b:{1,32}       [Dense layer 2 1]
Output layer 2 (OutputLayer)          32,2       66            W:{32,2}, b:{1,2}          [Dense layer 1 1]
-------------------------------------------------------------------------------------------------------------
            Total Parameters:  2,786,210
        Trainable Parameters:  2,786,210
           Frozen Parameters:  0
```



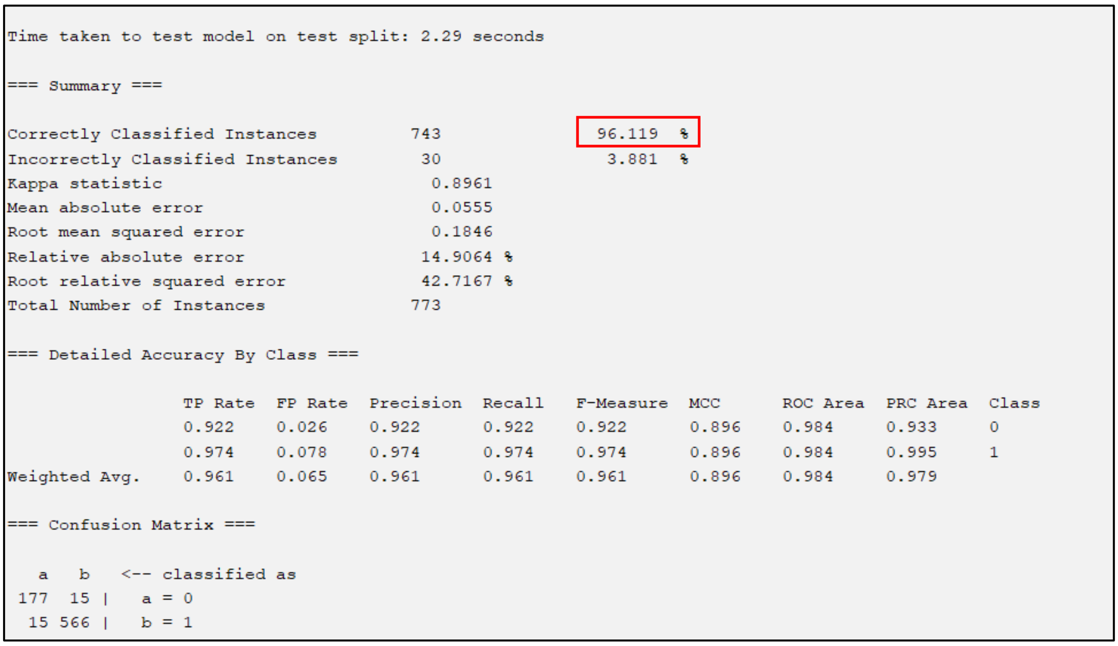
```
=== Summary ===

Correctly Classified Instances         749               96.8952 %
Incorrectly Classified Instances        24                3.1048 %
Kappa statistic                          0.9163
Mean absolute error                      0.0339
Root mean squared error                  0.1629
Relative absolute error                  9.1062 %
Root relative squared error             37.6907 %
Total Number of Instances              773

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.927    0.017    0.947      0.927    0.937      0.916    0.991     0.974     0
                 0.983    0.073    0.976      0.983    0.979      0.916    0.991     0.997     1
Weighted Avg.    0.969    0.059    0.969      0.969    0.969      0.916    0.991     0.991

=== Confusion Matrix ===

   a   b   <-- classified as
 178  14 |   a = 0
  10 571 |   b = 1
```

模型二:

兩層 convolution

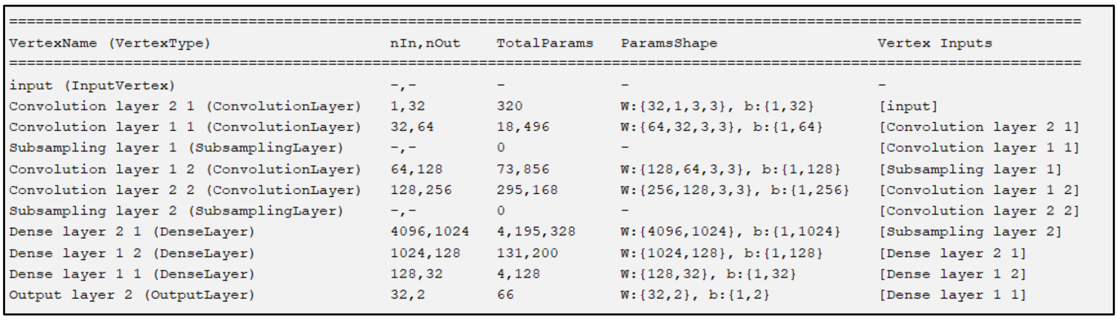


```
Network Configuration:
NeuralNetConfiguration(weightInit=XAVIER, biasInit=0.0, dist=weka.dl4j.distribution.Disabled@66, l1=NaN, l2=NaN, dro
Model Summary:

==============================================================================================================
VertexName (VertexType)                 nIn,nOut   TotalParams   ParamsShape                Vertex Inputs
==============================================================================================================
input (InputVertex)                     -,-        -             -                          -
Convolution layer 1 (ConvolutionLayer)  1,32       320           W:{32,1,3,3}, b:{1,32}     [input]
Subsampling layer 1 (SubsamplingLayer)  -,-        0             -                          [Convolution layer 1]
Convolution layer 2 (ConvolutionLayer)  32,64      18,496        W:{64,32,3,3}, b:{1,64}    [Subsampling layer 1]
Subsampling layer 2 (SubsamplingLayer)  -,-        0             -                          [Convolution layer 2]
Dense layer 2 1 (DenseLayer)            1600,512   819,712       W:{1600,512}, b:{1,512}    [Subsampling layer 2]
Dense layer 1 1 (DenseLayer)            512,32     16,416        W:{512,32}, b:{1,32}       [Dense layer 2 1]
Output layer 2 (OutputLayer)            32,2       66            W:{32,2}, b:{1,2}          [Dense layer 1 1]
--------------------------------------------------------------------------------------------------------------
            Total Parameters:  855,010
        Trainable Parameters:  855,010
           Frozen Parameters:  0
==============================================================================================================
```

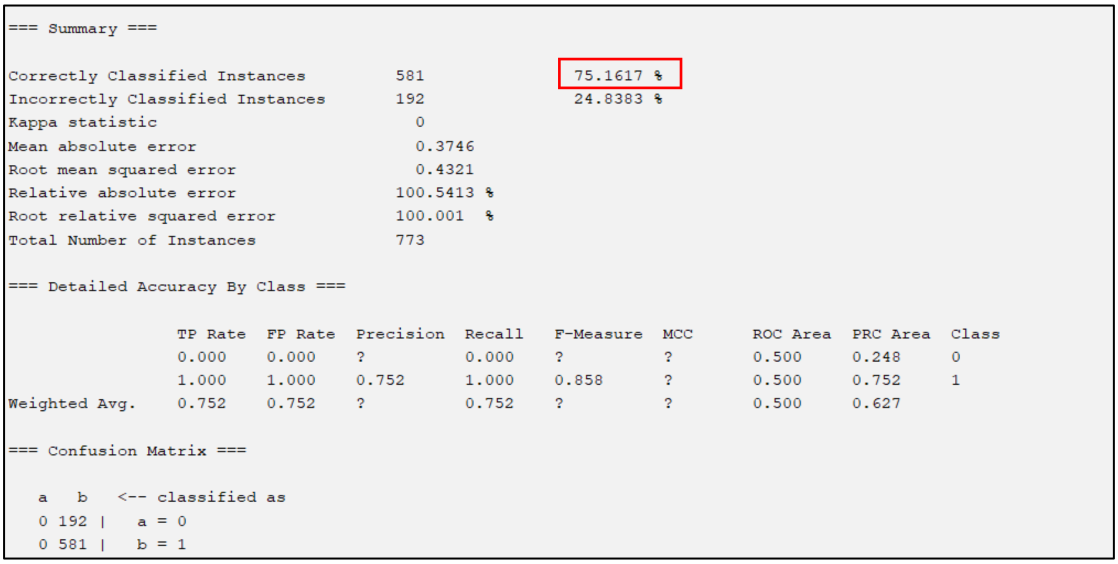


```
Time taken to test model on test split: 2.29 seconds

=== Summary ===

Correctly Classified Instances         743               96.119  %
Incorrectly Classified Instances        30                3.881  %
Kappa statistic                          0.8961
Mean absolute error                      0.0555
Root mean squared error                  0.1846
Relative absolute error                 14.9064 %
Root relative squared error             42.7167 %
Total Number of Instances              773

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.922    0.026    0.922      0.922    0.922      0.896    0.984     0.933     0
                 0.974    0.078    0.974      0.974    0.974      0.896    0.984     0.995     1
Weighted Avg.    0.961    0.065    0.961      0.961    0.961      0.896    0.984     0.979

=== Confusion Matrix ===

   a   b   <-- classified as
 177  15 |   a = 0
  15 566 |   b = 1
```

模型三:



```
==============================================================================================================
VertexName (VertexType)                   nIn,nOut    TotalParams   ParamsShape                  Vertex Inputs
==============================================================================================================
input (InputVertex)                       -,-         -             -                            -
Convolution layer 2 1 (ConvolutionLayer)  1,32        320           W:{32,1,3,3}, b:{1,32}       [input]
Convolution layer 1 1 (ConvolutionLayer)  32,64       18,496        W:{64,32,3,3}, b:{1,64}      [Convolution layer 2 1]
Subsampling layer 1 (SubsamplingLayer)    -,-         0             -                            [Convolution layer 1 1]
Convolution layer 1 2 (ConvolutionLayer)  64,128      73,856        W:{128,64,3,3}, b:{1,128}    [Subsampling layer 1]
Convolution layer 2 2 (ConvolutionLayer)  128,256     295,168       W:{256,128,3,3}, b:{1,256}   [Convolution layer 1 2]
Subsampling layer 2 (SubsamplingLayer)    -,-         0             -                            [Convolution layer 2 2]
Dense layer 2 1 (DenseLayer)              4096,1024   4,195,328     W:{4096,1024}, b:{1,1024}    [Subsampling layer 2]
Dense layer 1 2 (DenseLayer)              1024,128    131,200       W:{1024,128}, b:{1,128}      [Dense layer 2 1]
Dense layer 1 1 (DenseLayer)              128,32      4,128         W:{128,32}, b:{1,32}         [Dense layer 1 2]
Output layer 2 (OutputLayer)              32,2        66            W:{32,2}, b:{1,2}            [Dense layer 1 1]
```



```
=== Summary ===

Correctly Classified Instances         581               75.1617 %
Incorrectly Classified Instances       192               24.8383 %
Kappa statistic                          0
Mean absolute error                      0.3746
Root mean squared error                  0.4321
Relative absolute error                100.5413 %
Root relative squared error            100.001  %
Total Number of Instances              773

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.248     0
                 1.000    1.000    0.752      1.000    0.858      ?        0.500     0.752     1
Weighted Avg.    0.752    0.752    ?          0.752    ?          ?        0.500     0.627

=== Confusion Matrix ===

   a   b   <-- classified as
   0 192 |   a = 0
   0 581 |   b = 1
```

# 評估報告

分析結果，需有圖表統整同一個模型不同超參數的表現以及各模型的表現

1. 模型比較

| | RandomForest | MLP | CNN |
|---|---|---|---|
| accuracy | 97.3615% | 96.5699% | 96.8952 |

2. Random Forest

| numberFeatures | accuracy |
|---|---|
| 0 | 97.2615% |
| 5 | 97.2296% |
| 11 | 97.2735% |

3. MLP

| hidden Layer | accuracy |
|---|---|
| 2 | 96.5699% |
| 10 | 96.394% |
| 10, 2 | 96.0422% |

4. CNN

| | accuracy |
|---|---|
| 模型一 | 96.8952% |
| 模型二 | 96.119% |
| 模型三 | 75.1617% |

- 使用的方法的優缺點

1. RandomForest 好處是執行快速且準確率高，但可讀性差。
2. MLP 則是簡單好懂，雖然 hidden layer 沒設很高，但執行的效果還算不錯，但 hidden layer 設太高，電腦執行時間久。
3. CNN 準確率不錯、參數共享，但執行的時間久，且容易 overfitting。

- 是否有改進之處

前處理的部分，可以嘗試刪除多個欄位，嘗試找到最適合的組合。
雖然這次實驗 CNN 模型複雜，對準確度沒有太大的幫助，反而造成準確度下降的問題。推測這可能跟資料量和 overfitting 有關。因此推測可以使用 Dropout 或 regulization 等技術，來避免 overfitting。

- 最後會選擇哪一個模型

最後選擇 RandomForest。考慮到電腦的執行時間，以及準確度，RandomForest 訓練出的結果明顯優於 MLP 和 CNN。