A Practical activity Report submitted

ELC ASSIGNMENT
Handwritten Digit Recognition

By

Adrija Sengupta

102203509

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

COMPUTER SCIENCE ENGINEERING DEPARTMENT - CSED

THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY, (A DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB

INDIA

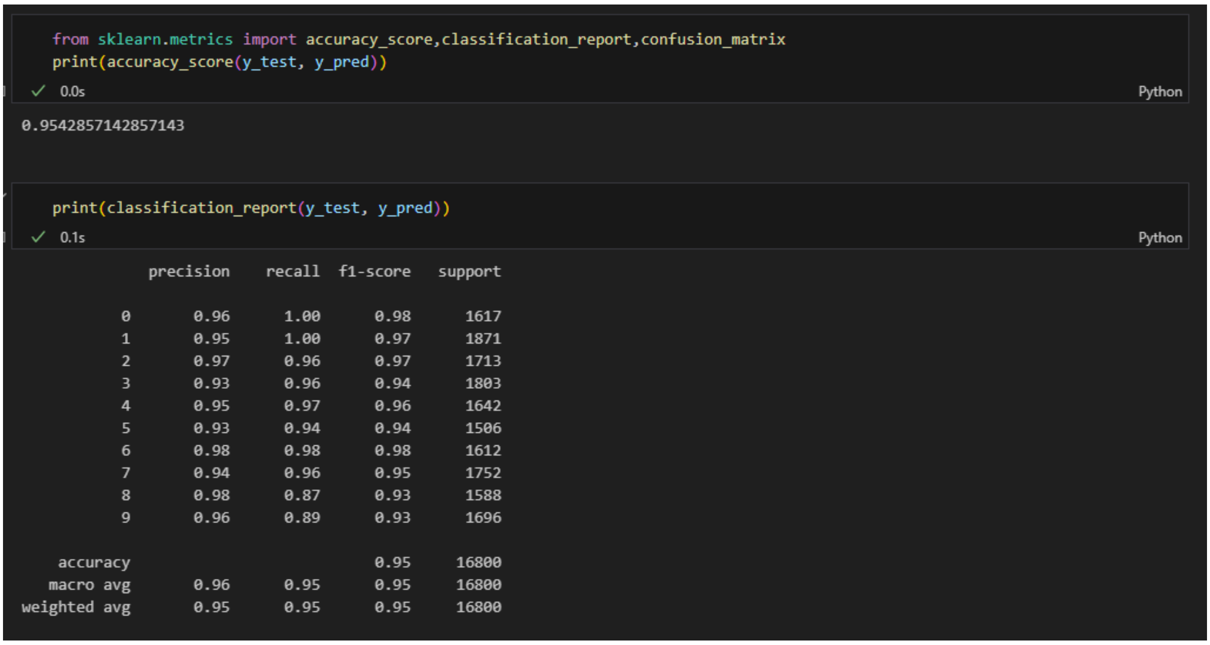
EXPERIMENT

Train your model using K-Nearest Neighbour Algorithm with having values of K as {2,4,5,6,7,10}, over data.csv file provided. The Train and Test split of the data should be in the ratio of 60:40, 70:30, 75:25, 80:20, 90:10, 95:5. Evaluate the performance of the model over test data for all these scenarios (36 cases), and submit the single jupyter notebook, having one of the scenario implemented (and rest in comments), and a single pdf file containing the results of all these scenarios (Accuracy, and Confusion Matrix), also your analysis regarding the dependency of the performance of model over training testing split and k value.

FOR K = 2

1) RATIO OF TRAIN AND TEST DATA - 60:40

ACCURACY – 0.95



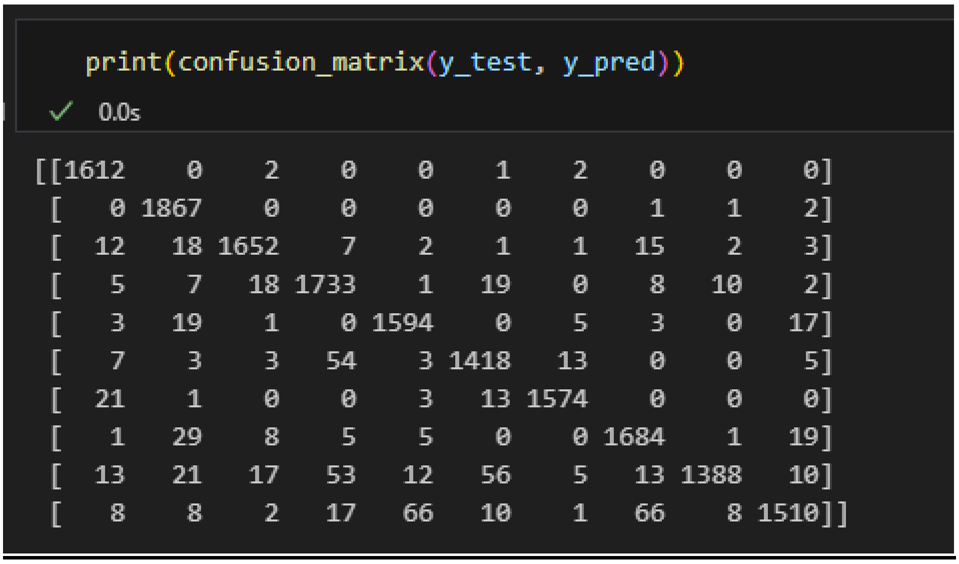
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.9542857142857143
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      1.00      0.98      1617
           1       0.95      1.00      0.97      1871
           2       0.97      0.96      0.97      1713
           3       0.93      0.96      0.94      1803
           4       0.95      0.97      0.96      1642
           5       0.93      0.94      0.94      1506
           6       0.98      0.98      0.98      1612
           7       0.94      0.96      0.95      1752
           8       0.98      0.87      0.93      1588
           9       0.96      0.89      0.93      1696

    accuracy                           0.95     16800
   macro avg       0.96      0.95      0.95     16800
weighted avg       0.95      0.95      0.95     16800
```

CONFUSION MATRIX

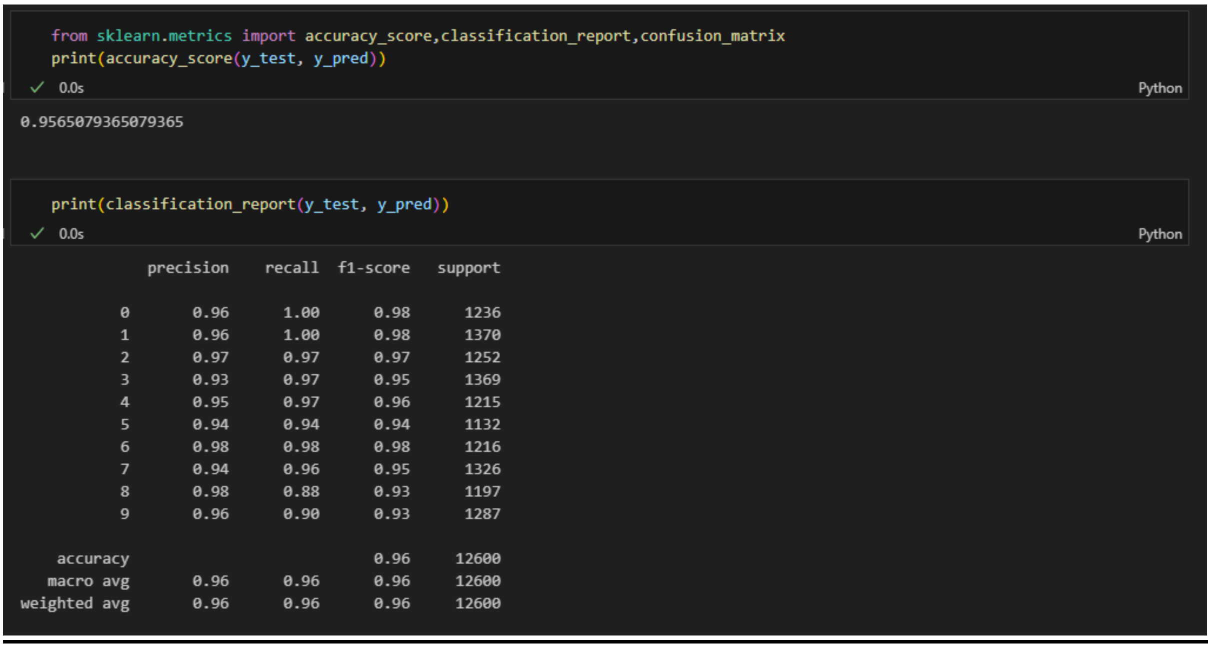


```
print(confusion_matrix(y_test, y_pred))
✓ 0.0s
```

```
[[1612    0    2    0    0    1    2    0    0    0]
 [   0 1867    0    0    0    0    0    1    1    2]
 [  12   18 1652    7    2    1    1   15    2    3]
 [   5    7   18 1733    1   19    0    8   10    2]
 [   3   19    1    0 1594    0    5    3    0   17]
 [   7    3    3   54    3 1418   13    0    0    5]
 [  21    1    0    0    3   13 1574    0    0    0]
 [   1   29    8    5    5    0    0 1684    1   19]
 [  13   21   17   53   12   56    5   13 1388   10]
 [   8    8    2   17   66   10    1   66    8 1510]]
```

2) RATIO OF TRAIN AND TEST DATA - 70:30

ACCURACY –0.95



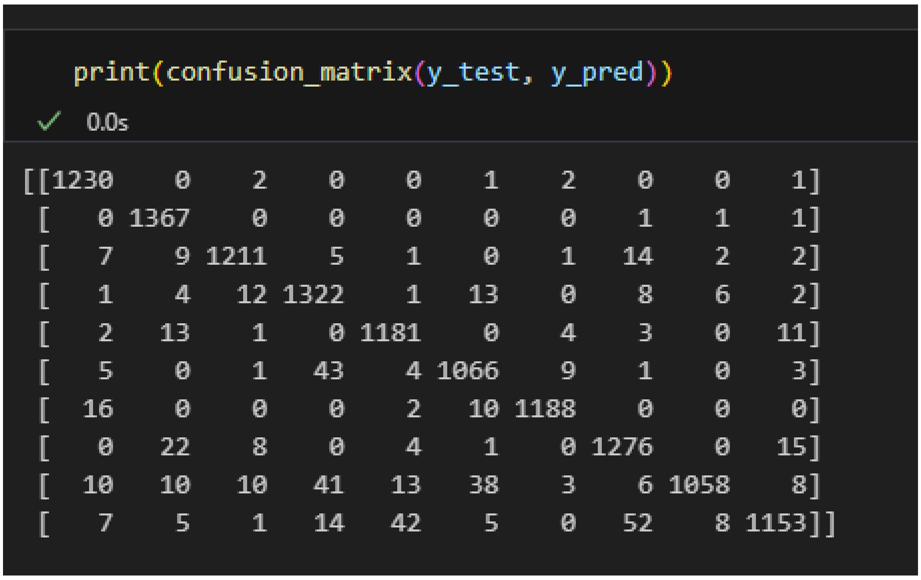
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
✓ 0.0s                                                                    Python
```

```
0.9565079365079365
```

```
print(classification_report(y_test, y_pred))
✓ 0.0s                                                                    Python
```

```
              precision    recall  f1-score   support

           0       0.96      1.00      0.98      1236
           1       0.96      1.00      0.98      1370
           2       0.97      0.97      0.97      1252
           3       0.93      0.97      0.95      1369
           4       0.95      0.97      0.96      1215
           5       0.94      0.94      0.94      1132
           6       0.98      0.98      0.98      1216
           7       0.94      0.96      0.95      1326
           8       0.98      0.88      0.93      1197
           9       0.96      0.90      0.93      1287

    accuracy                           0.96     12600
   macro avg       0.96      0.96      0.96     12600
weighted avg       0.96      0.96      0.96     12600
```

CONFUSION MATRIX



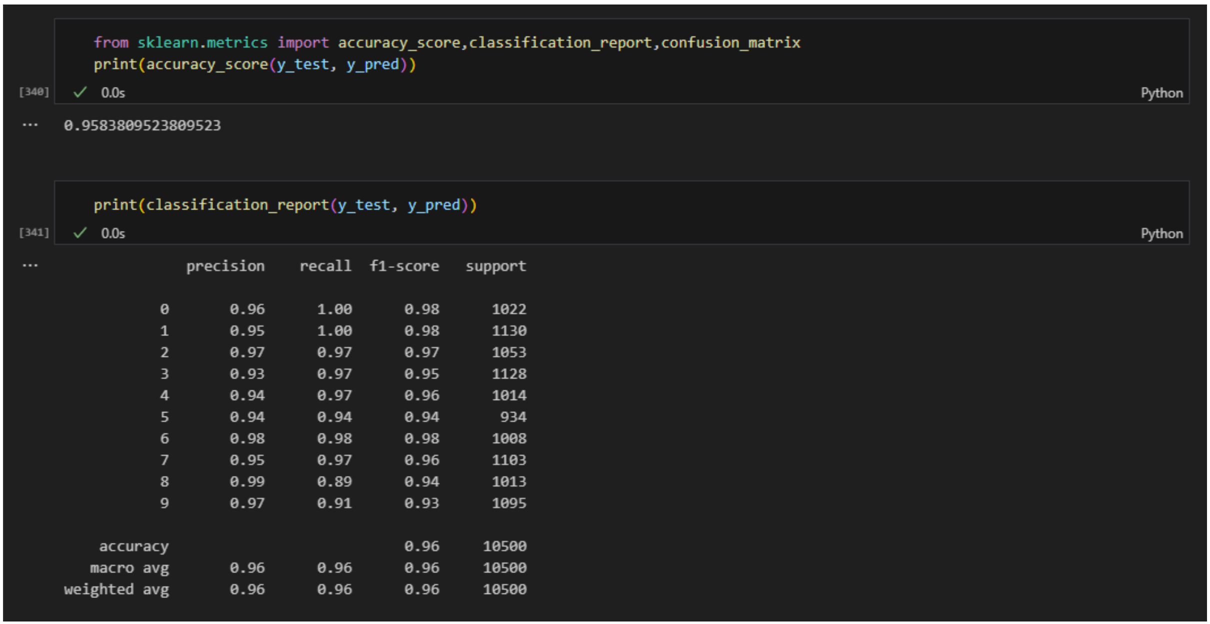
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1230    0    2    0    0    1    2    0    0    1]
 [   0 1367    0    0    0    0    0    1    1    1]
 [   7    9 1211    5    1    0    1   14    2    2]
 [   1    4   12 1322    1   13    0    8    6    2]
 [   2   13    1    0 1181    0    4    3    0   11]
 [   5    0    1   43    4 1066    9    1    0    3]
 [  16    0    0    0    2   10 1188    0    0    0]
 [   0   22    8    0    4    1    0 1276    0   15]
 [  10   10   10   41   13   38    3    6 1058    8]
 [   7    5    1   14   42    5    0   52    8 1153]]
```

3) RATIO OF TRAIN AND TEST DATA - 75:25

ACCURACY –0.95



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

[340] ✓ 0.0s Python

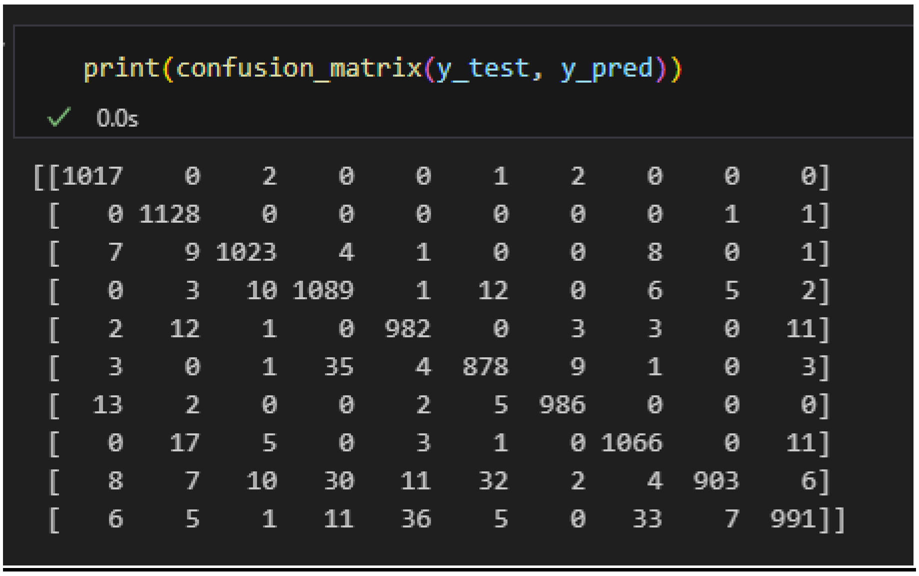
```
0.9583809523809523
```

```
print(classification_report(y_test, y_pred))
```

[341] ✓ 0.0s Python

```
              precision    recall  f1-score   support

           0       0.96      1.00      0.98      1022
           1       0.95      1.00      0.98      1130
           2       0.97      0.97      0.97      1053
           3       0.93      0.97      0.95      1128
           4       0.94      0.97      0.96      1014
           5       0.94      0.94      0.94       934
           6       0.98      0.98      0.98      1008
           7       0.95      0.97      0.96      1103
           8       0.99      0.89      0.94      1013
           9       0.97      0.91      0.93      1095

    accuracy                           0.96     10500
   macro avg       0.96      0.96      0.96     10500
weighted avg       0.96      0.96      0.96     10500
```

CONFUSION MATRIX

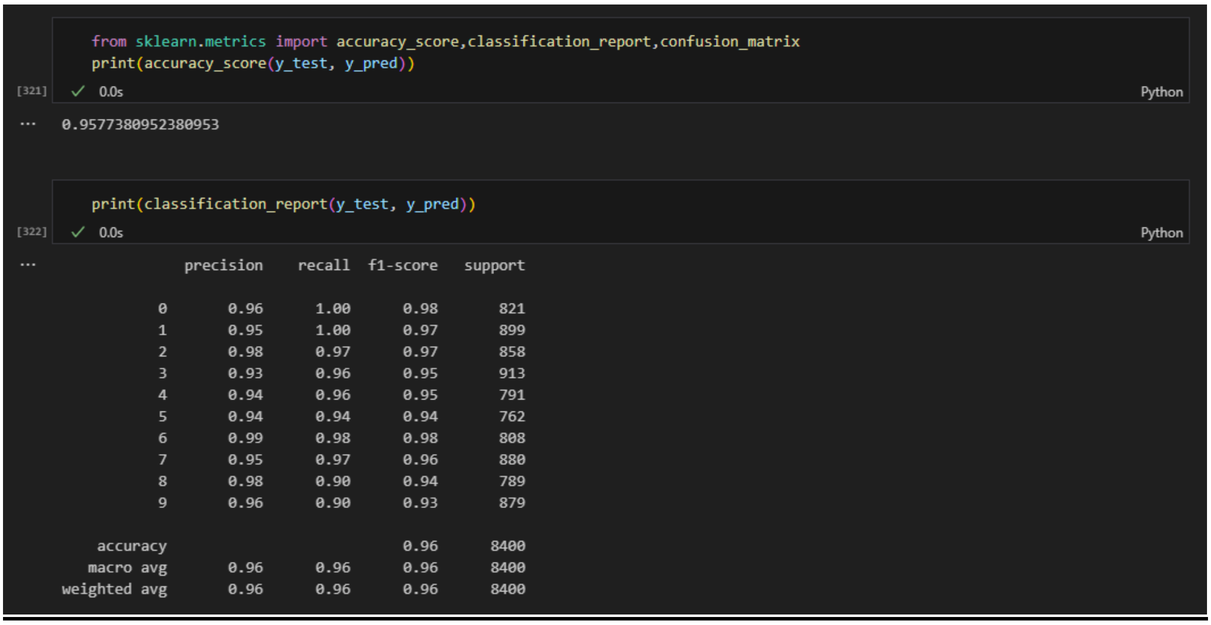


```
print(confusion_matrix(y_test, y_pred))
```

```
[[1017    0    2    0    0    1    2    0    0    0]
 [   0 1128    0    0    0    0    0    0    1    1]
 [   7    9 1023    4    1    0    0    8    0    1]
 [   0    3   10 1089    1   12    0    6    5    2]
 [   2   12    1    0  982    0    3    3    0   11]
 [   3    0    1   35    4  878    9    1    0    3]
 [  13    2    0    0    2    5  986    0    0    0]
 [   0   17    5    0    3    1    0 1066    0   11]
 [   8    7   10   30   11   32    2    4  903    6]
 [   6    5    1   11   36    5    0   33    7  991]]
```

4) RATIO OF TRAIN AND TEST DATA – 80:20

ACCURACY –0.95



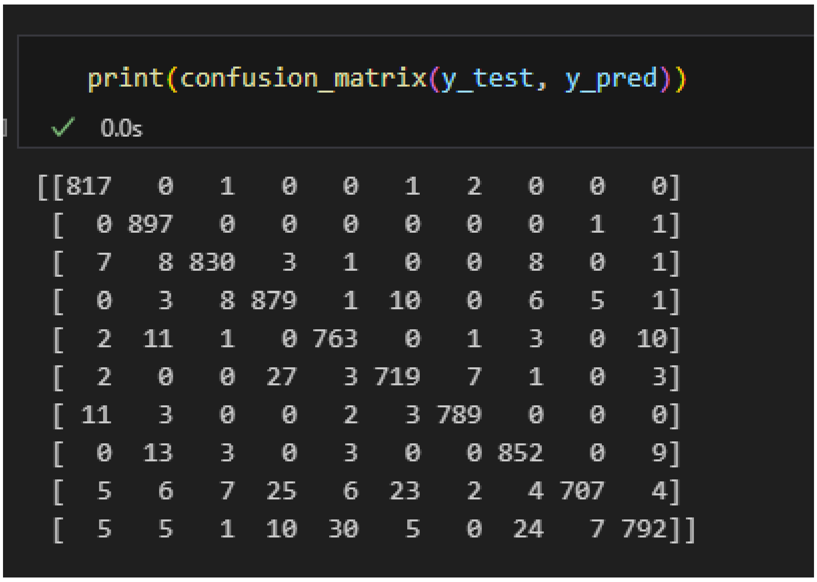
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.9577380952380953
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.96      1.00      0.98       821
           1       0.95      1.00      0.97       899
           2       0.98      0.97      0.97       858
           3       0.93      0.96      0.95       913
           4       0.94      0.96      0.95       791
           5       0.94      0.94      0.94       762
           6       0.99      0.98      0.98       808
           7       0.95      0.97      0.96       880
           8       0.98      0.90      0.94       789
           9       0.96      0.90      0.93       879

    accuracy                           0.96      8400
   macro avg       0.96      0.96      0.96      8400
weighted avg       0.96      0.96      0.96      8400
```

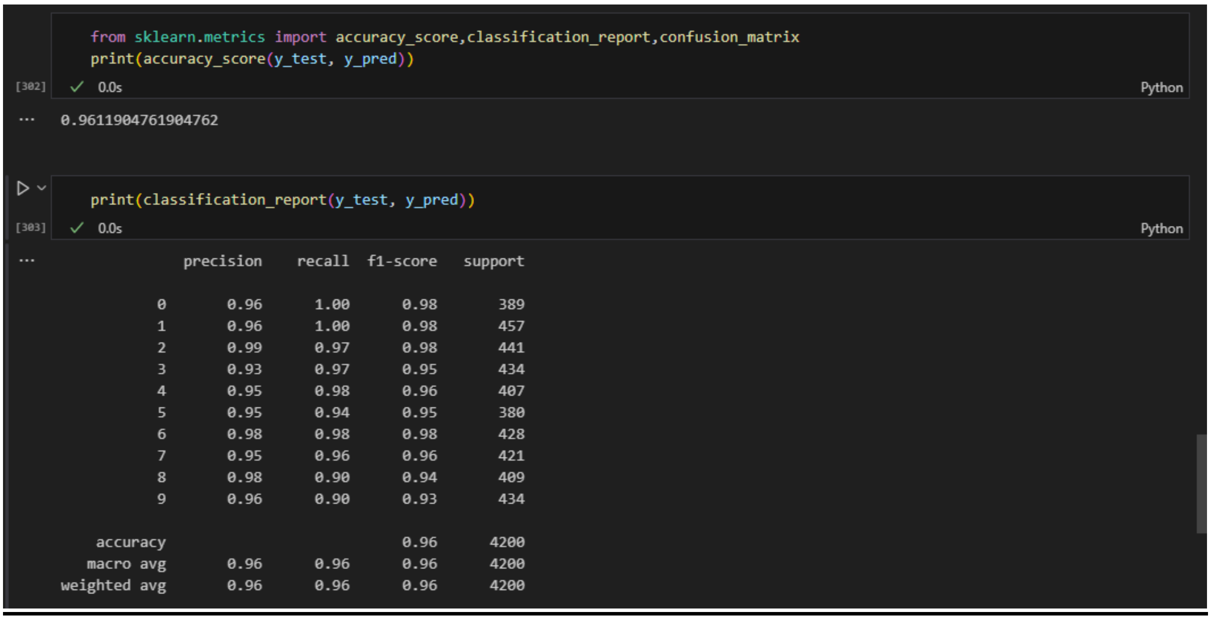
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
✓ 0.0s

[[817   0   1   0   0   1   2   0   0   0]
 [  0 897   0   0   0   0   0   0   1   1]
 [  7   8 830   3   1   0   0   8   0   1]
 [  0   3   8 879   1  10   0   6   5   1]
 [  2  11   1   0 763   0   1   3   0  10]
 [  2   0   0  27   3 719   7   1   0   3]
 [ 11   3   0   0   2   3 789   0   0   0]
 [  0  13   3   0   3   0   0 852   0   9]
 [  5   6   7  25   6  23   2   4 707   4]
 [  5   5   1  10  30   5   0  24   7 792]]
```

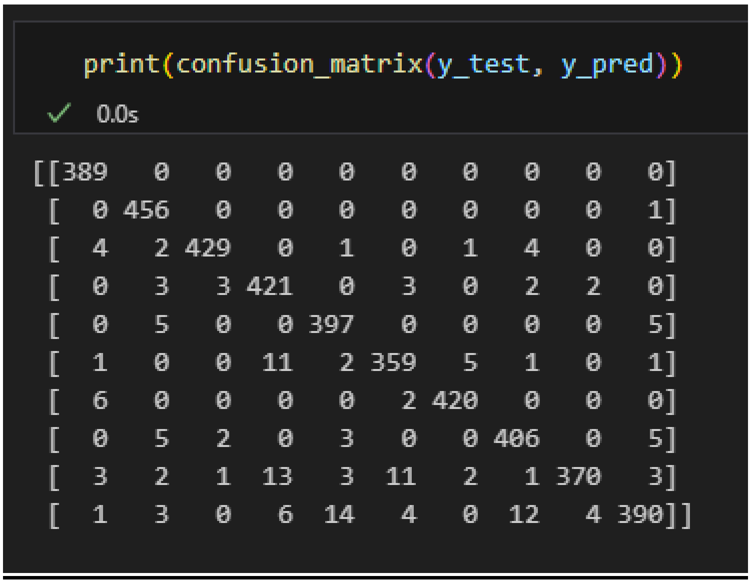
5) RATIO OF TRAIN AND TEST DATA – 90:10

ACCURACY –0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
[302] ✓ 0.0s                                                        Python

... 0.9611904761904762

print(classification_report(y_test, y_pred))
[303] ✓ 0.0s                                                        Python

...              precision    recall  f1-score   support

           0       0.96      1.00      0.98       389
           1       0.96      1.00      0.98       457
           2       0.99      0.97      0.98       441
           3       0.93      0.97      0.95       434
           4       0.95      0.98      0.96       407
           5       0.95      0.94      0.95       380
           6       0.98      0.98      0.98       428
           7       0.95      0.96      0.96       421
           8       0.98      0.90      0.94       409
           9       0.96      0.90      0.93       434

    accuracy                           0.96      4200
   macro avg       0.96      0.96      0.96      4200
weighted avg       0.96      0.96      0.96      4200
```

CONFUSION MATRIX

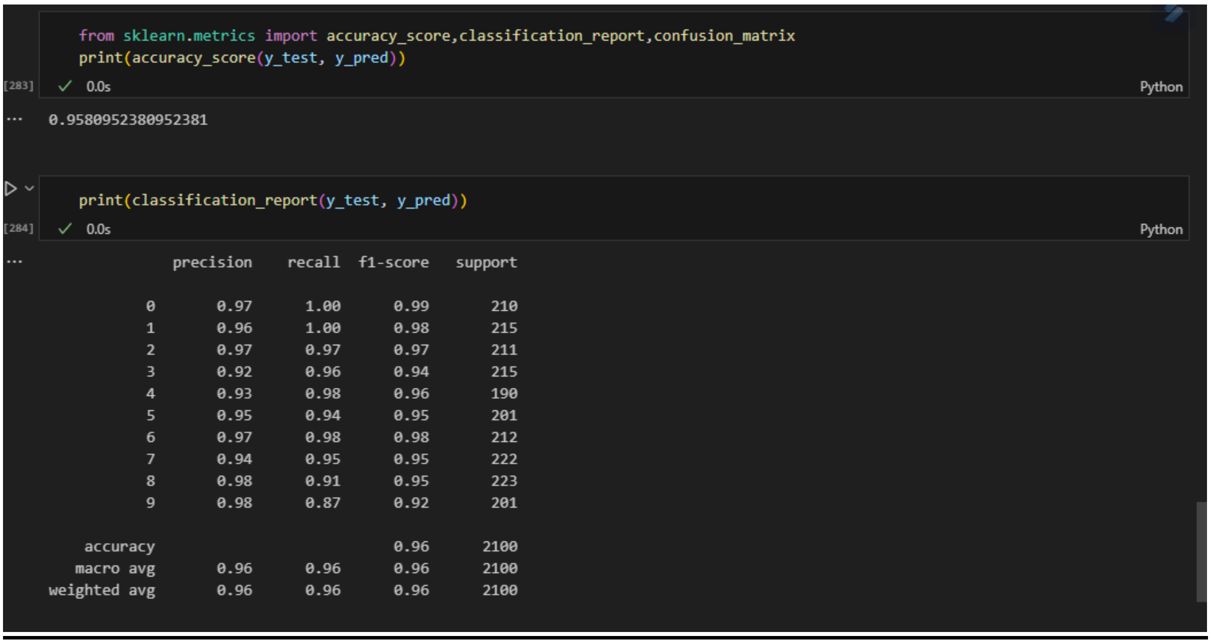


```
print(confusion_matrix(y_test, y_pred))
```

```
[[389   0   0   0   0   0   0   0   0   0]
 [  0 456   0   0   0   0   0   0   0   1]
 [  4   2 429   0   1   0   1   4   0   0]
 [  0   3   3 421   0   3   0   2   2   0]
 [  0   5   0   0 397   0   0   0   0   5]
 [  1   0   0  11   2 359   5   1   0   1]
 [  6   0   0   0   0   2 420   0   0   0]
 [  0   5   2   0   3   0   0 406   0   5]
 [  3   2   1  13   3  11   2   1 370   3]
 [  1   3   0   6  14   4   0  12   4 390]]
```

6) RATIO OF TRAIN AND TEST DATA – 95:5

ACCURACY –0.95



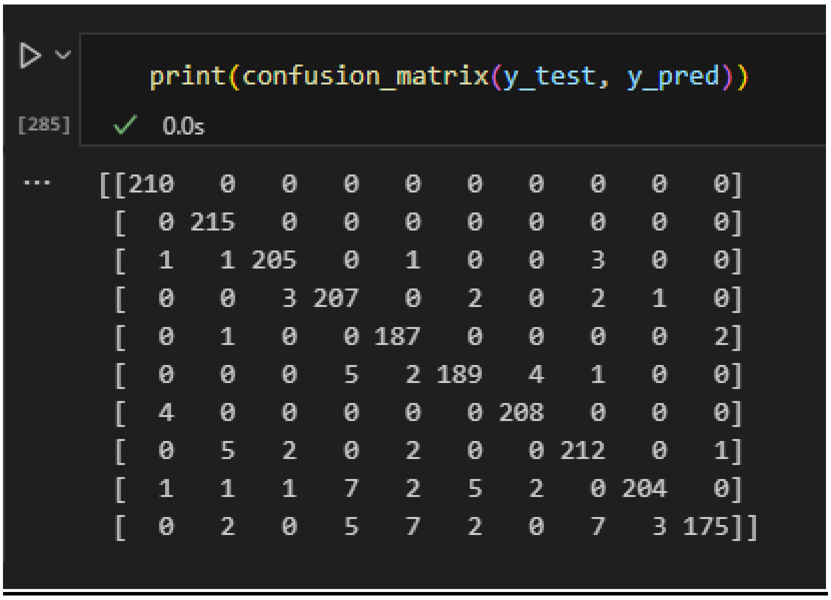
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.9580952380952381
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.99       210
           1       0.96      1.00      0.98       215
           2       0.97      0.97      0.97       211
           3       0.92      0.96      0.94       215
           4       0.93      0.98      0.96       190
           5       0.95      0.94      0.95       201
           6       0.97      0.98      0.98       212
           7       0.94      0.95      0.95       222
           8       0.98      0.91      0.95       223
           9       0.98      0.87      0.92       201

    accuracy                           0.96      2100
   macro avg       0.96      0.96      0.96      2100
weighted avg       0.96      0.96      0.96      2100
```

CONFUSION MATRIX



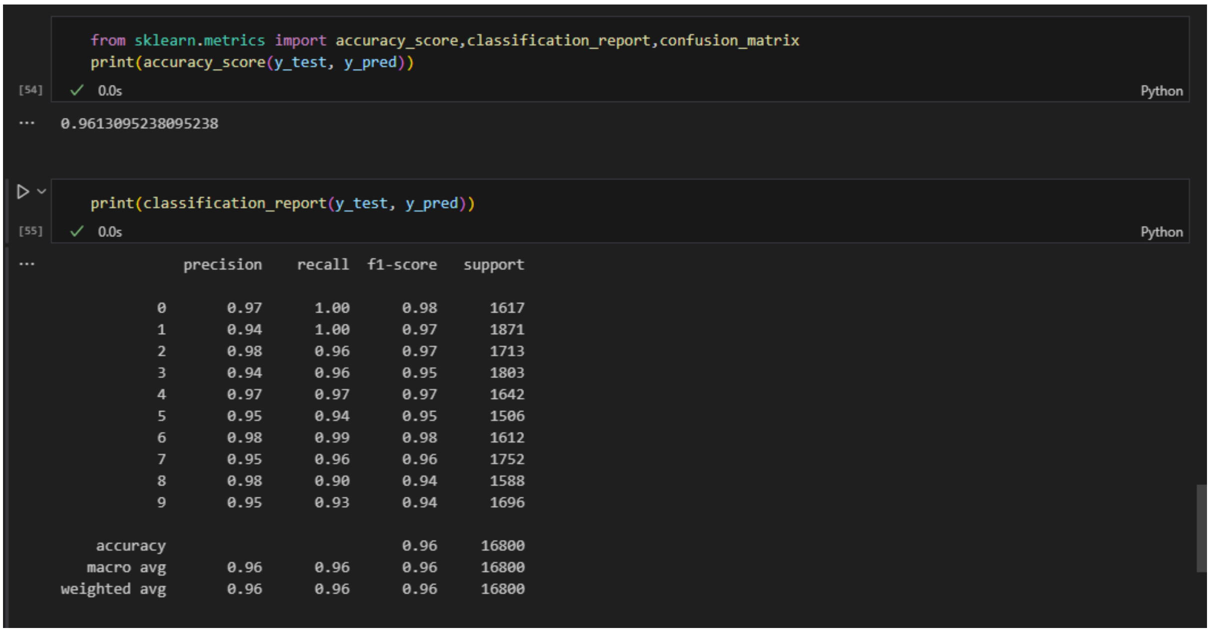
```
print(confusion_matrix(y_test, y_pred))
```

```
[[210   0   0   0   0   0   0   0   0   0]
 [  0 215   0   0   0   0   0   0   0   0]
 [  1   1 205   0   1   0   0   3   0   0]
 [  0   0   3 207   0   2   0   2   1   0]
 [  0   1   0   0 187   0   0   0   0   2]
 [  0   0   0   5   2 189   4   1   0   0]
 [  4   0   0   0   0   0 208   0   0   0]
 [  0   5   2   0   2   0   0 212   0   1]
 [  1   1   1   7   2   5   2   0 204   0]
 [  0   2   0   5   7   2   0   7   3 175]]
```

FOR K = 4

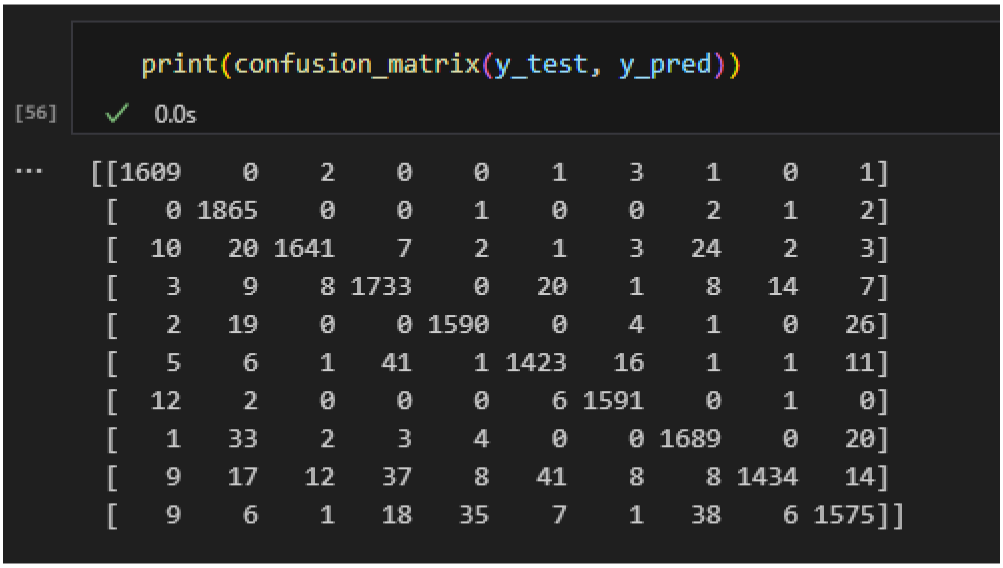
1) RATIO OF TRAIN AND TEST DATA - 60:40

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
[54] ✓ 0.0s                                                    Python
··· 0.9613095238095238

print(classification_report(y_test, y_pred))
[55] ✓ 0.0s                                                    Python
···               precision    recall  f1-score   support

           0       0.97      1.00      0.98      1617
           1       0.94      1.00      0.97      1871
           2       0.98      0.96      0.97      1713
           3       0.94      0.96      0.95      1803
           4       0.97      0.97      0.97      1642
           5       0.95      0.94      0.95      1506
           6       0.98      0.99      0.98      1612
           7       0.95      0.96      0.96      1752
           8       0.98      0.90      0.94      1588
           9       0.95      0.93      0.94      1696

    accuracy                           0.96     16800
   macro avg       0.96      0.96      0.96     16800
weighted avg       0.96      0.96      0.96     16800
```

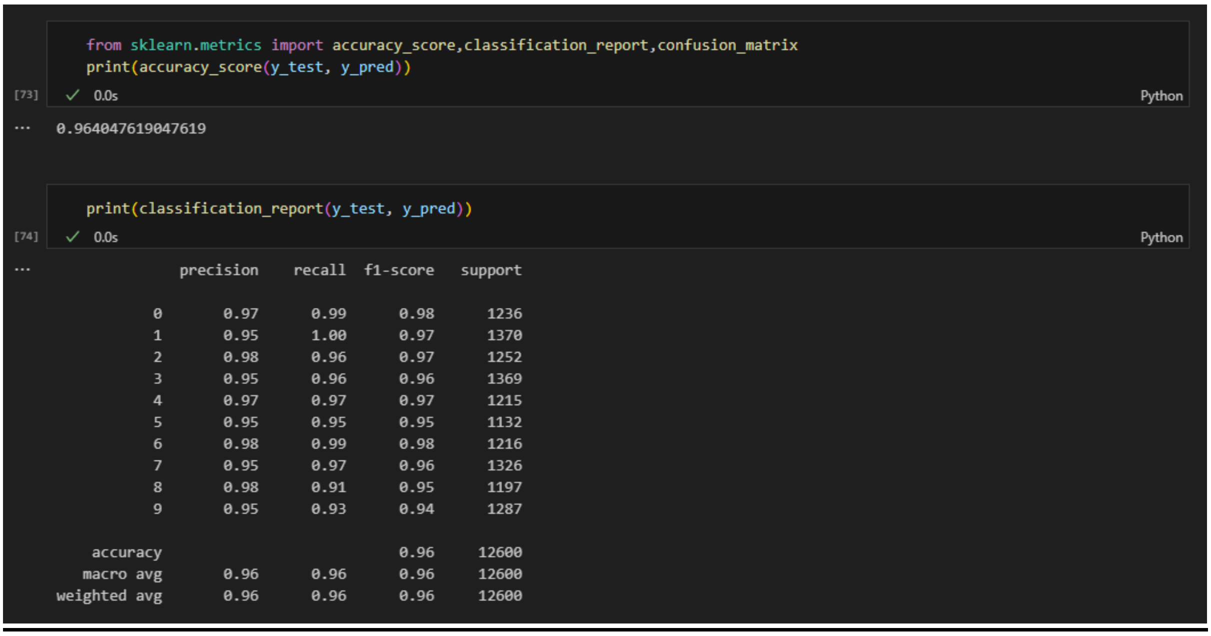
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
[56] ✓ 0.0s
··· [[1609    0    2    0    0    1    3    1    0    1]
     [   0 1865    0    0    1    0    0    2    1    2]
     [  10   20 1641    7    2    1    3   24    2    3]
     [   3    9    8 1733    0   20    1    8   14    7]
     [   2   19    0    0 1590    0    4    1    0   26]
     [   5    6    1   41    1 1423   16    1    1   11]
     [  12    2    0    0    0    6 1591    0    1    0]
     [   1   33    2    3    4    0    0 1689    0   20]
     [   9   17   12   37    8   41    8    8 1434   14]
     [   9    6    1   18   35    7    1   38    6 1575]]
```

2) RATIO OF TRAIN AND TEST DATA - 70:30

ACCURACY – 0.96



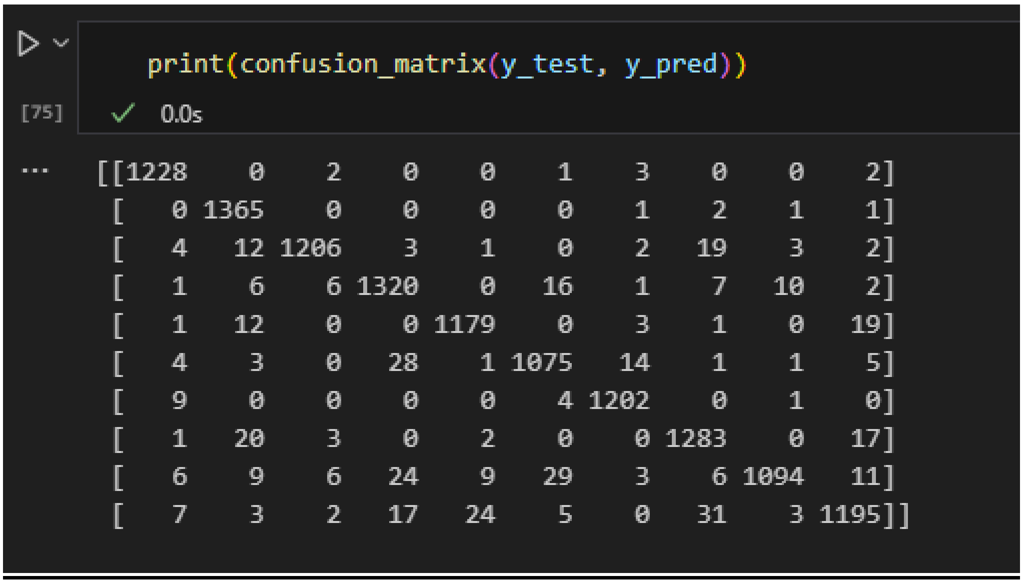
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.964047619047619
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1236
           1       0.95      1.00      0.97      1370
           2       0.98      0.96      0.97      1252
           3       0.95      0.96      0.96      1369
           4       0.97      0.97      0.97      1215
           5       0.95      0.95      0.95      1132
           6       0.98      0.99      0.98      1216
           7       0.95      0.97      0.96      1326
           8       0.98      0.91      0.95      1197
           9       0.95      0.93      0.94      1287

    accuracy                           0.96     12600
   macro avg       0.96      0.96      0.96     12600
weighted avg       0.96      0.96      0.96     12600
```

CONFUSION MATRIX

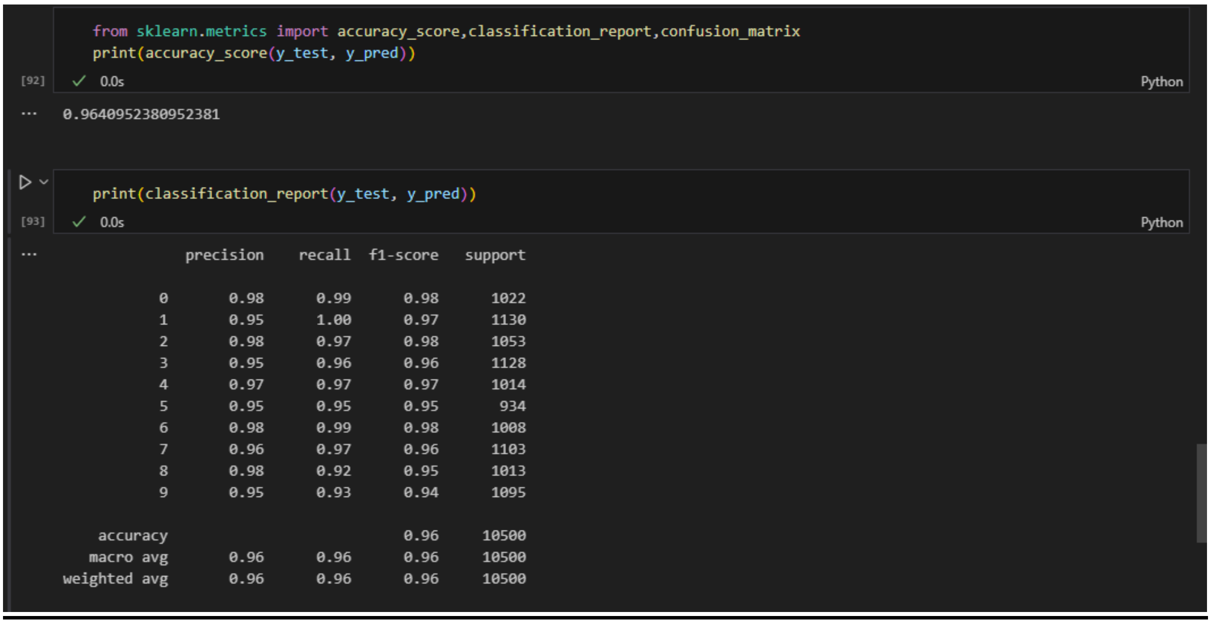


```
print(confusion_matrix(y_test, y_pred))
```

```
[[1228    0    2    0    0    1    3    0    0    2]
 [   0 1365    0    0    0    0    1    2    1    1]
 [   4   12 1206    3    1    0    2   19    3    2]
 [   1    6    6 1320    0   16    1    7   10    2]
 [   1   12    0    0 1179    0    3    1    0   19]
 [   4    3    0   28    1 1075   14    1    1    5]
 [   9    0    0    0    0    4 1202    0    1    0]
 [   1   20    3    0    2    0    0 1283    0   17]
 [   6    9    6   24    9   29    3    6 1094   11]
 [   7    3    2   17   24    5    0   31    3 1195]]
```

3) RATIO OF TRAIN AND TEST DATA - 75:25

ACCURACY – 0.96



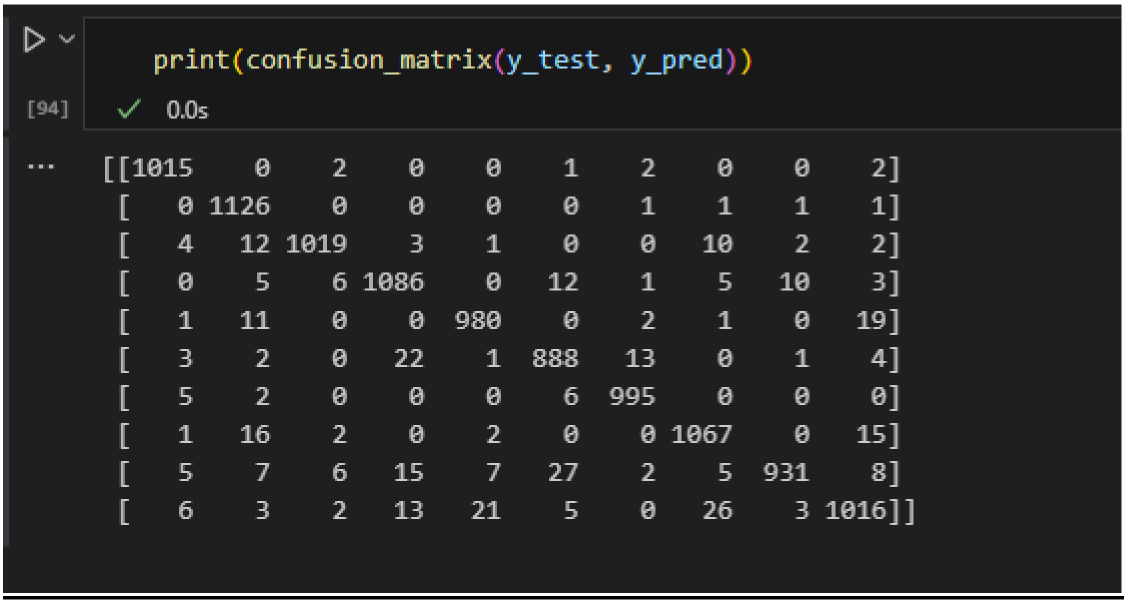
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.9640952380952381
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      1022
           1       0.95      1.00      0.97      1130
           2       0.98      0.97      0.98      1053
           3       0.95      0.96      0.96      1128
           4       0.97      0.97      0.97      1014
           5       0.95      0.95      0.95       934
           6       0.98      0.99      0.98      1008
           7       0.96      0.97      0.96      1103
           8       0.98      0.92      0.95      1013
           9       0.95      0.93      0.94      1095

    accuracy                           0.96     10500
   macro avg       0.96      0.96      0.96     10500
weighted avg       0.96      0.96      0.96     10500
```

CONFUSION MATRIX

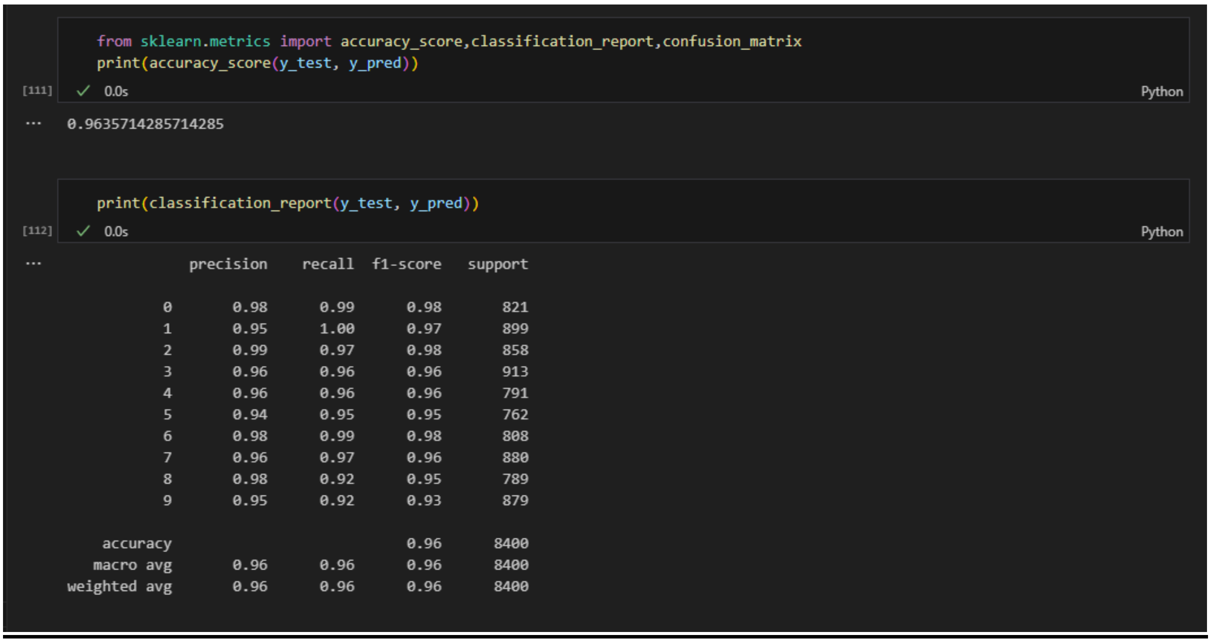


```
print(confusion_matrix(y_test, y_pred))
```

```
[[1015    0    2    0    0    1    2    0    0    2]
 [   0 1126    0    0    0    0    1    1    1    1]
 [   4   12 1019    3    1    0    0   10    2    2]
 [   0    5    6 1086    0   12    1    5   10    3]
 [   1   11    0    0  980    0    2    1    0   19]
 [   3    2    0   22    1  888   13    0    1    4]
 [   5    2    0    0    0    6  995    0    0    0]
 [   1   16    2    0    2    0    0 1067    0   15]
 [   5    7    6   15    7   27    2    5  931    8]
 [   6    3    2   13   21    5    0   26    3 1016]]
```

4) RATIO OF TRAIN AND TEST DATA - 80:20

ACCURACY – 0.96



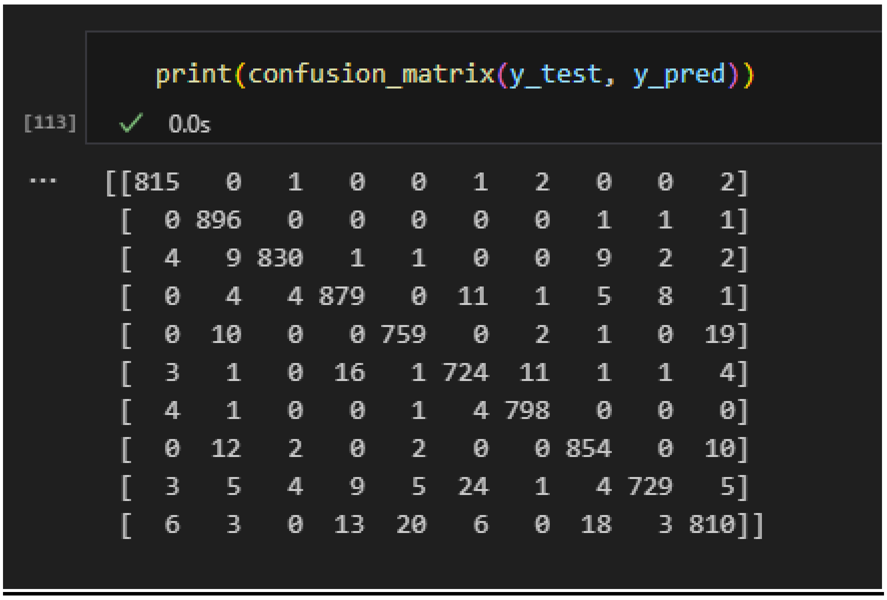
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.9635714285714285
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       821
           1       0.95      1.00      0.97       899
           2       0.99      0.97      0.98       858
           3       0.96      0.96      0.96       913
           4       0.96      0.96      0.96       791
           5       0.94      0.95      0.95       762
           6       0.98      0.99      0.98       808
           7       0.96      0.97      0.96       880
           8       0.98      0.92      0.95       789
           9       0.95      0.92      0.93       879

    accuracy                           0.96      8400
   macro avg       0.96      0.96      0.96      8400
weighted avg       0.96      0.96      0.96      8400
```

CONFUSION MATRIX

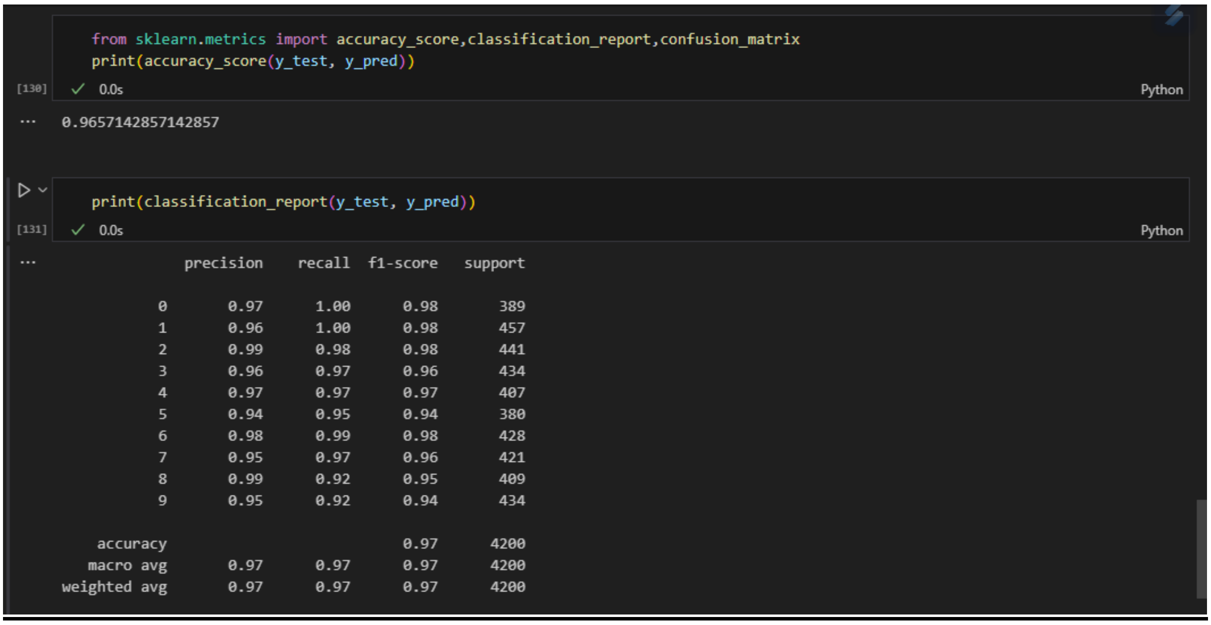


```
print(confusion_matrix(y_test, y_pred))
```

```
[[815   0   1   0   0   1   2   0   0   2]
 [  0 896   0   0   0   0   0   1   1   1]
 [  4   9 830   1   1   0   0   9   2   2]
 [  0   4   4 879   0  11   1   5   8   1]
 [  0  10   0   0 759   0   2   1   0  19]
 [  3   1   0  16   1 724  11   1   1   4]
 [  4   1   0   0   1   4 798   0   0   0]
 [  0  12   2   0   2   0   0 854   0  10]
 [  3   5   4   9   5  24   1   4 729   5]
 [  6   3   0  13  20   6   0  18   3 810]]
```

5) RATIO OF TRAIN AND TEST DATA - 90:10

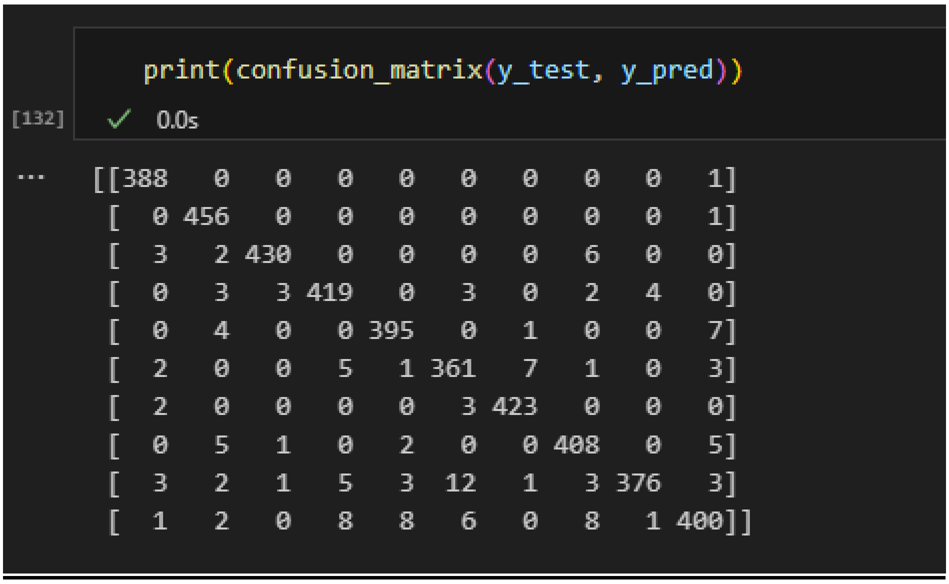
ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
[130]  ✓ 0.0s                                                      Python

0.9657142857142857
```

```
print(classification_report(y_test, y_pred))
[131]  ✓ 0.0s                                                      Python

              precision    recall  f1-score   support

           0       0.97      1.00      0.98       389
           1       0.96      1.00      0.98       457
           2       0.99      0.98      0.98       441
           3       0.96      0.97      0.96       434
           4       0.97      0.97      0.97       407
           5       0.94      0.95      0.94       380
           6       0.98      0.99      0.98       428
           7       0.95      0.97      0.96       421
           8       0.99      0.92      0.95       409
           9       0.95      0.92      0.94       434

    accuracy                           0.97      4200
   macro avg       0.97      0.97      0.97      4200
weighted avg       0.97      0.97      0.97      4200
```

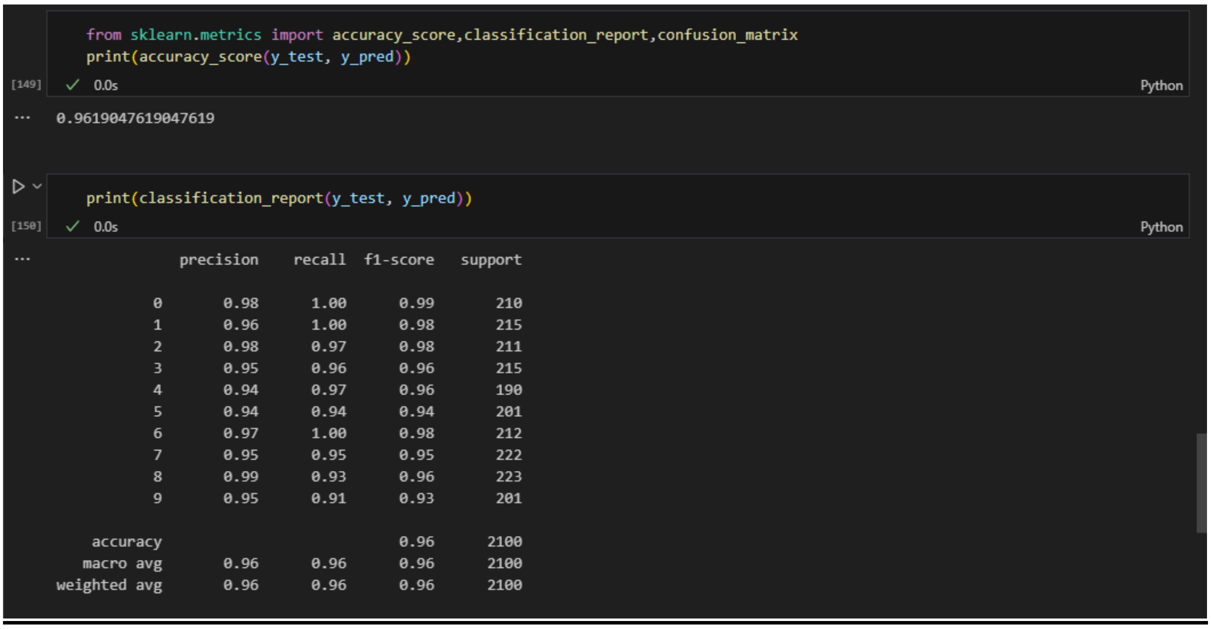
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
[132]  ✓ 0.0s

[[388   0   0   0   0   0   0   0   0   1]
 [  0 456   0   0   0   0   0   0   0   1]
 [  3   2 430   0   0   0   0   6   0   0]
 [  0   3   3 419   0   3   0   2   4   0]
 [  0   4   0   0 395   0   1   0   0   7]
 [  2   0   0   5   1 361   7   1   0   3]
 [  2   0   0   0   0   3 423   0   0   0]
 [  0   5   1   0   2   0   0 408   0   5]
 [  3   2   1   5   3  12   1   3 376   3]
 [  1   2   0   8   8   6   0   8   1 400]]
```

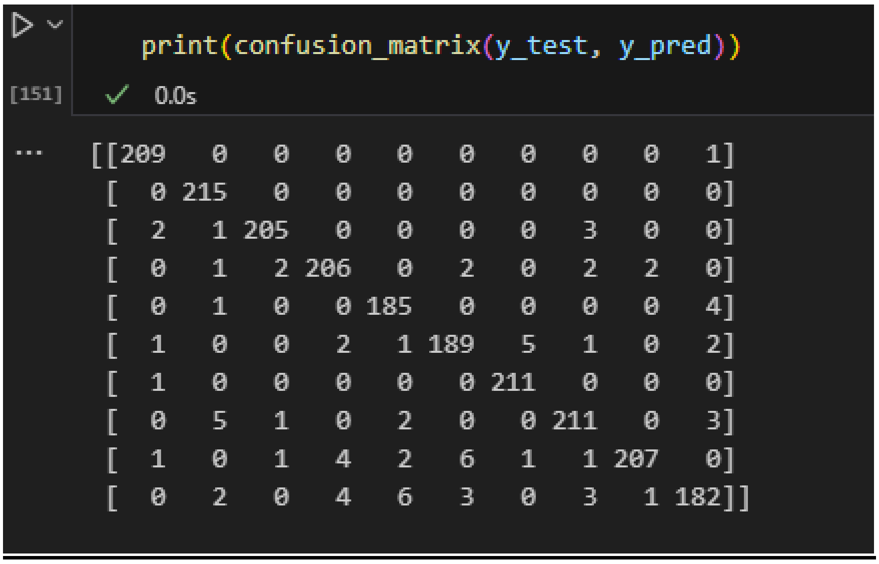
6) RATIO OF TRAIN AND TEST DATA - 95:05

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
[149] ✓ 0.0s                                                         Python
··· 0.9619047619047619

print(classification_report(y_test, y_pred))
[150] ✓ 0.0s                                                         Python
···              precision    recall  f1-score   support

           0       0.98      1.00      0.99       210
           1       0.96      1.00      0.98       215
           2       0.98      0.97      0.98       211
           3       0.95      0.96      0.96       215
           4       0.94      0.97      0.96       190
           5       0.94      0.94      0.94       201
           6       0.97      1.00      0.98       212
           7       0.95      0.95      0.95       222
           8       0.99      0.93      0.96       223
           9       0.95      0.91      0.93       201

    accuracy                           0.96      2100
   macro avg       0.96      0.96      0.96      2100
weighted avg       0.96      0.96      0.96      2100
```

CONFUSION MATRIX

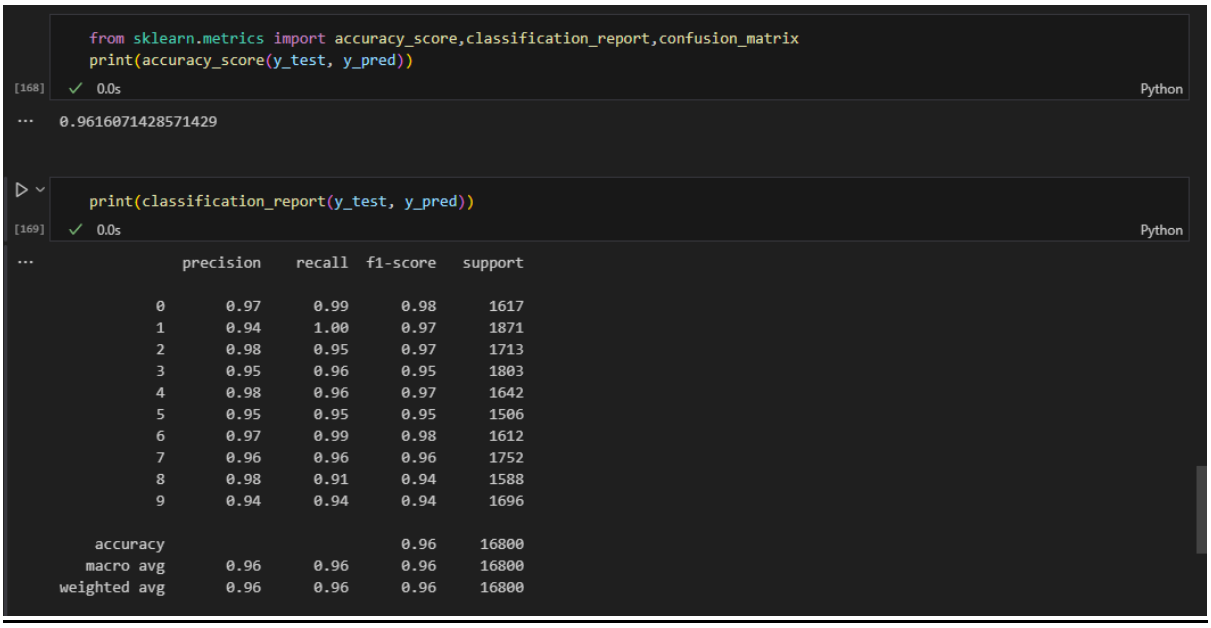


```
print(confusion_matrix(y_test, y_pred))
[151] ✓ 0.0s
··· [[209   0   0   0   0   0   0   0   0   1]
     [  0 215   0   0   0   0   0   0   0   0]
     [  2   1 205   0   0   0   0   3   0   0]
     [  0   1   2 206   0   2   0   2   2   0]
     [  0   1   0   0 185   0   0   0   0   4]
     [  1   0   0   2   1 189   5   1   0   2]
     [  1   0   0   0   0   0 211   0   0   0]
     [  0   5   1   0   2   0   0 211   0   3]
     [  1   0   1   4   2   6   1   1 207   0]
     [  0   2   0   4   6   3   0   3   1 182]]
```

FOR K = 5

1) RATIO OF TRAIN AND TEST DATA - 60:40

ACCURACY – 0.96



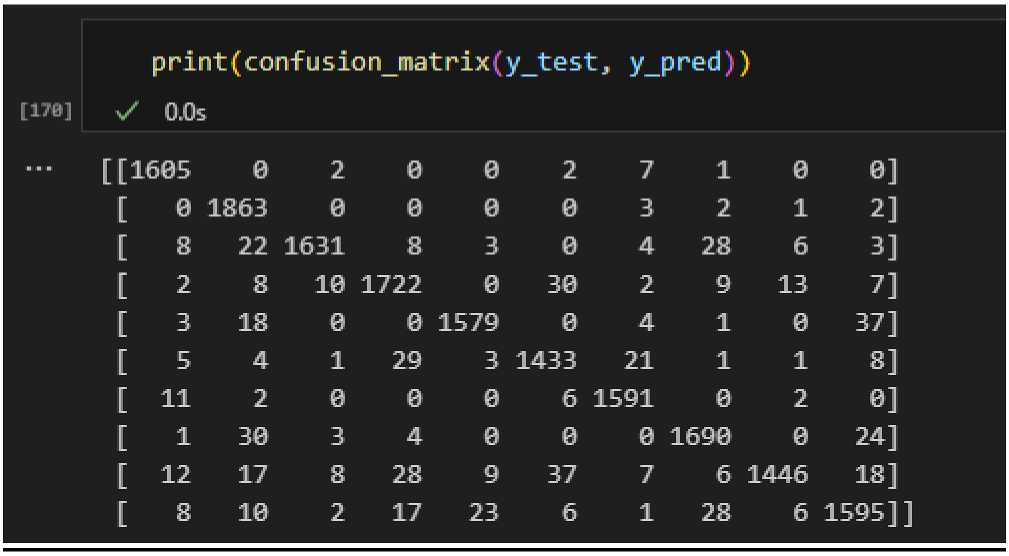
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```
[168] ✓ 0.0s Python

```
0.9616071428571429
```

```
print(classification_report(y_test, y_pred))
```
[169] ✓ 0.0s Python

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1617
           1       0.94      1.00      0.97      1871
           2       0.98      0.95      0.97      1713
           3       0.95      0.96      0.95      1803
           4       0.98      0.96      0.97      1642
           5       0.95      0.95      0.95      1506
           6       0.97      0.99      0.98      1612
           7       0.96      0.96      0.96      1752
           8       0.98      0.91      0.94      1588
           9       0.94      0.94      0.94      1696

    accuracy                           0.96     16800
   macro avg       0.96      0.96      0.96     16800
weighted avg       0.96      0.96      0.96     16800
```

CONFUSION MATRIX

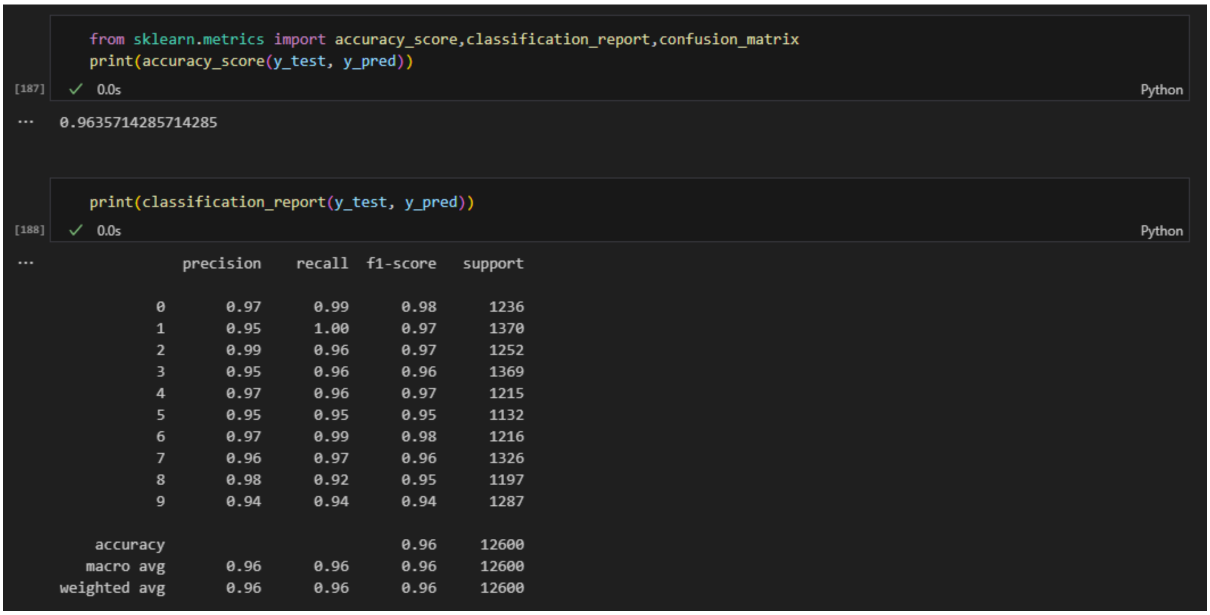


```
print(confusion_matrix(y_test, y_pred))
```
[170] ✓ 0.0s

```
[[1605    0    2    0    0    2    7    1    0    0]
 [   0 1863    0    0    0    0    3    2    1    2]
 [   8   22 1631    8    3    0    4   28    6    3]
 [   2    8   10 1722    0   30    2    9   13    7]
 [   3   18    0    0 1579    0    4    1    0   37]
 [   5    4    1   29    3 1433   21    1    1    8]
 [  11    2    0    0    0    6 1591    0    2    0]
 [   1   30    3    4    0    0    0 1690    0   24]
 [  12   17    8   28    9   37    7    6 1446   18]
 [   8   10    2   17   23    6    1   28    6 1595]]
```

2) RATIO OF TRAIN AND TEST DATA - 70:30

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

[187] ✓ 0.0s Python

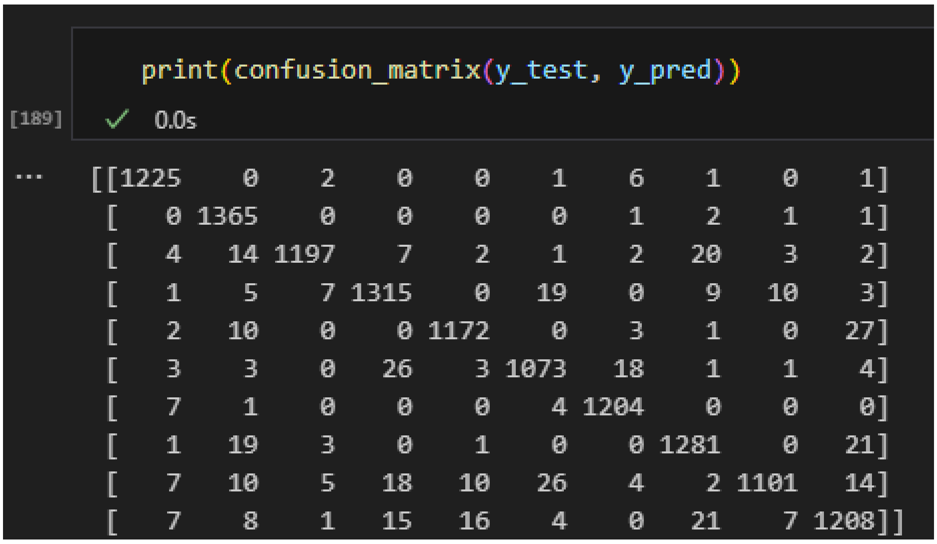
```
0.9635714285714285
```

```
print(classification_report(y_test, y_pred))
```

[188] ✓ 0.0s Python

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1236
           1       0.95      1.00      0.97      1370
           2       0.99      0.96      0.97      1252
           3       0.95      0.96      0.96      1369
           4       0.97      0.96      0.97      1215
           5       0.95      0.95      0.95      1132
           6       0.97      0.99      0.98      1216
           7       0.96      0.97      0.96      1326
           8       0.98      0.92      0.95      1197
           9       0.94      0.94      0.94      1287

    accuracy                           0.96     12600
   macro avg       0.96      0.96      0.96     12600
weighted avg       0.96      0.96      0.96     12600
```

CONFUSION MATRIX



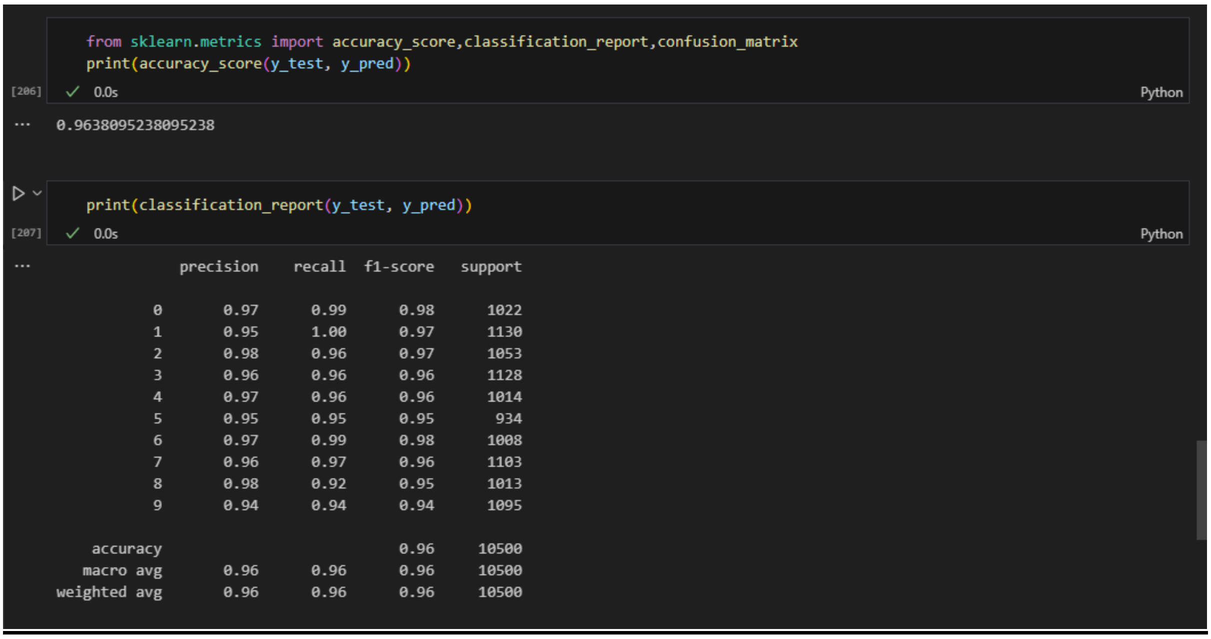
```
print(confusion_matrix(y_test, y_pred))
```

[189] ✓ 0.0s

```
[[1225    0    2    0    0    1    6    1    0    1]
 [   0 1365    0    0    0    0    1    2    1    1]
 [   4   14 1197    7    2    1    2   20    3    2]
 [   1    5    7 1315    0   19    0    9   10    3]
 [   2   10    0    0 1172    0    3    1    0   27]
 [   3    3    0   26    3 1073   18    1    1    4]
 [   7    1    0    0    0    4 1204    0    0    0]
 [   1   19    3    0    1    0    0 1281    0   21]
 [   7   10    5   18   10   26    4    2 1101   14]
 [   7    8    1   15   16    4    0   21    7 1208]]
```

3) RATIO OF TRAIN AND TEST DATA - 75:25

ACCURACY – 0.96



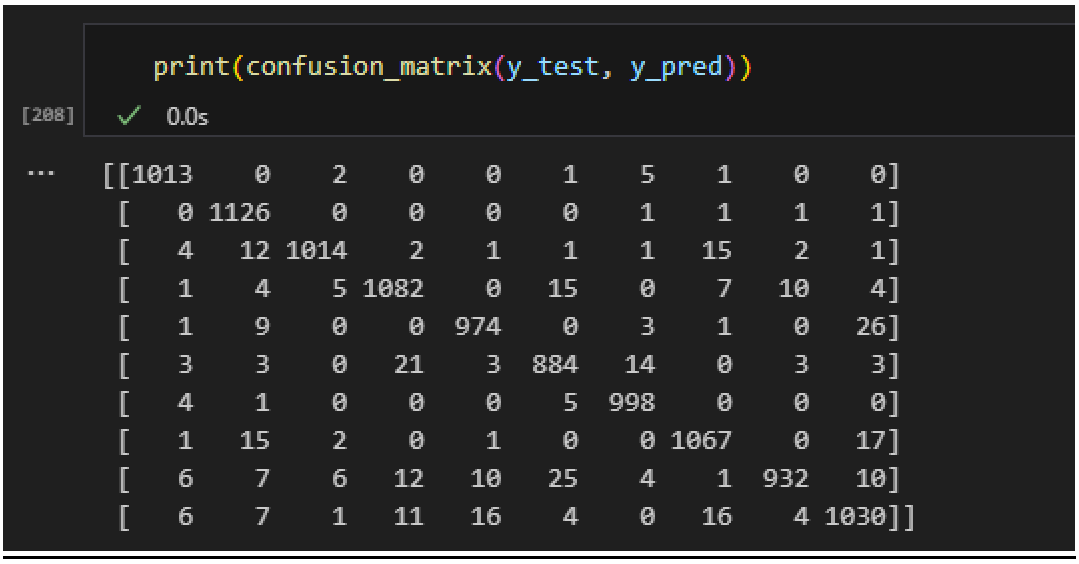
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```
[206] ✓ 0.0s Python

```
0.9638095238095238
```

```
print(classification_report(y_test, y_pred))
```
[207] ✓ 0.0s Python

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1022
           1       0.95      1.00      0.97      1130
           2       0.98      0.96      0.97      1053
           3       0.96      0.96      0.96      1128
           4       0.97      0.96      0.96      1014
           5       0.95      0.95      0.95       934
           6       0.97      0.99      0.98      1008
           7       0.96      0.97      0.96      1103
           8       0.98      0.92      0.95      1013
           9       0.94      0.94      0.94      1095

    accuracy                           0.96     10500
   macro avg       0.96      0.96      0.96     10500
weighted avg       0.96      0.96      0.96     10500
```

CONFUSION MATRIX

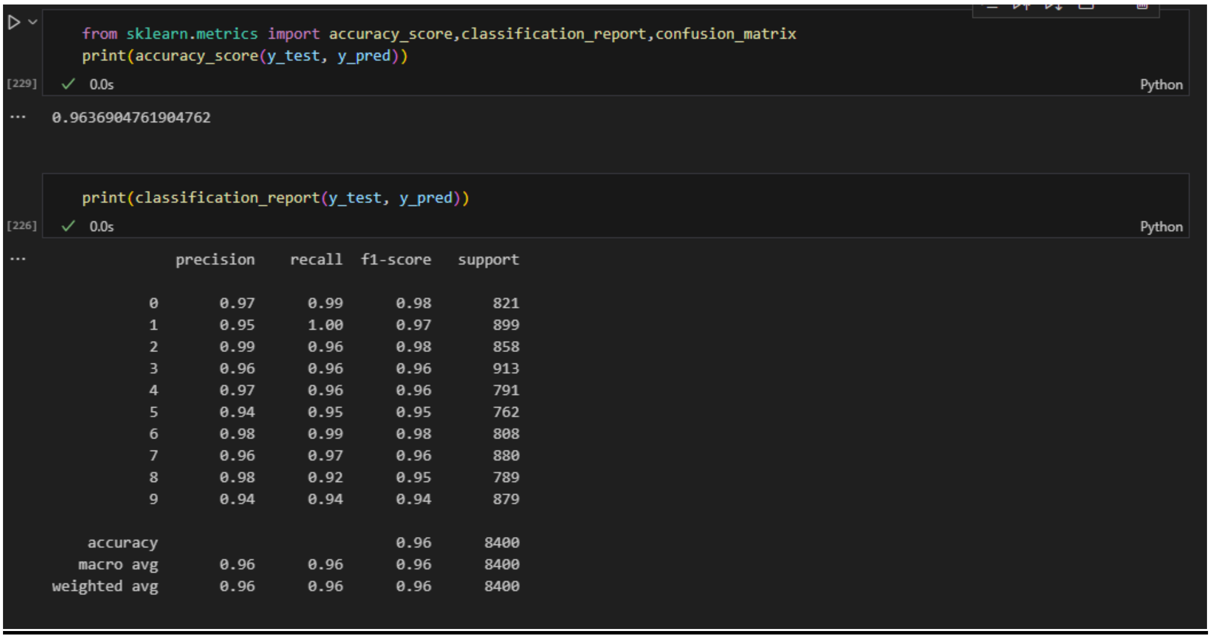


```
print(confusion_matrix(y_test, y_pred))
```
[208] ✓ 0.0s

```
[[1013    0    2    0    0    1    5    1    0    0]
 [   0 1126    0    0    0    0    1    1    1    1]
 [   4   12 1014    2    1    1    1   15    2    1]
 [   1    4    5 1082    0   15    0    7   10    4]
 [   1    9    0    0  974    0    3    1    0   26]
 [   3    3    0   21    3  884   14    0    3    3]
 [   4    1    0    0    0    5  998    0    0    0]
 [   1   15    2    0    1    0    0 1067    0   17]
 [   6    7    6   12   10   25    4    1  932   10]
 [   6    7    1   11   16    4    0   16    4 1030]]
```

4) RATIO OF TRAIN AND TEST DATA - 80:20

ACCURACY – 0.96



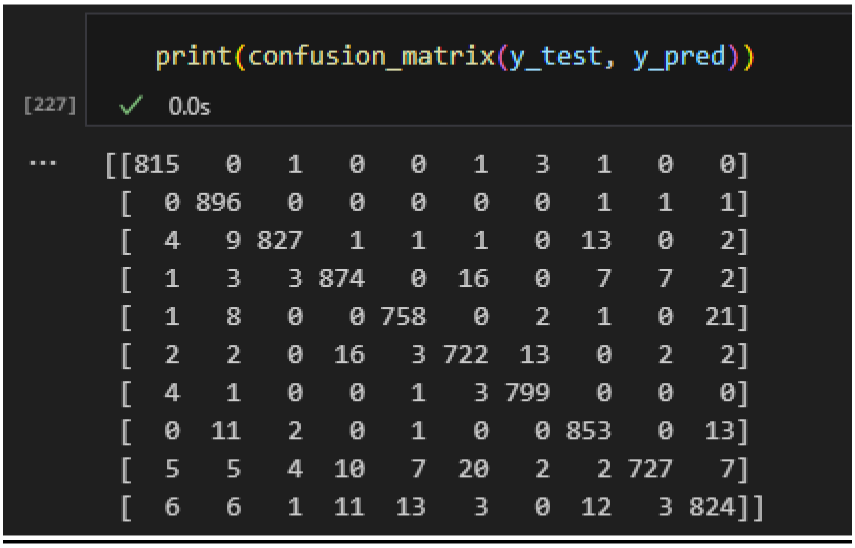
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```
[229] ✓ 0.0s Python

```
0.9636904761904762
```

```
print(classification_report(y_test, y_pred))
```
[226] ✓ 0.0s Python

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       821
           1       0.95      1.00      0.97       899
           2       0.99      0.96      0.98       858
           3       0.96      0.96      0.96       913
           4       0.97      0.96      0.96       791
           5       0.94      0.95      0.95       762
           6       0.98      0.99      0.98       808
           7       0.96      0.97      0.96       880
           8       0.98      0.92      0.95       789
           9       0.94      0.94      0.94       879

    accuracy                           0.96      8400
   macro avg       0.96      0.96      0.96      8400
weighted avg       0.96      0.96      0.96      8400
```

CONFUSION MATRIX

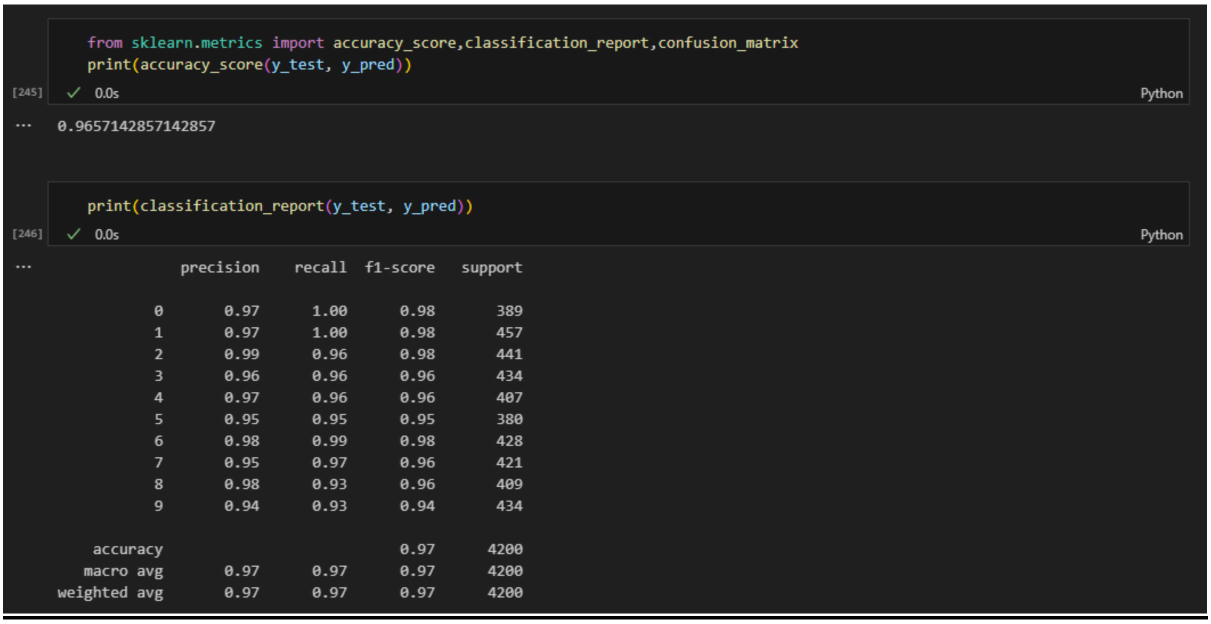


```
print(confusion_matrix(y_test, y_pred))
```
[227] ✓ 0.0s

```
[[815   0   1   0   0   1   3   1   0   0]
 [  0 896   0   0   0   0   0   1   1   1]
 [  4   9 827   1   1   1   0  13   0   2]
 [  1   3   3 874   0  16   0   7   7   2]
 [  1   8   0   0 758   0   2   1   0  21]
 [  2   2   0  16   3 722  13   0   2   2]
 [  4   1   0   0   1   3 799   0   0   0]
 [  0  11   2   0   1   0   0 853   0  13]
 [  5   5   4  10   7  20   2   2 727   7]
 [  6   6   1  11  13   3   0  12   3 824]]
```

5) RATIO OF TRAIN AND TEST DATA - 90:10

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

[245] ✓ 0.0s Python

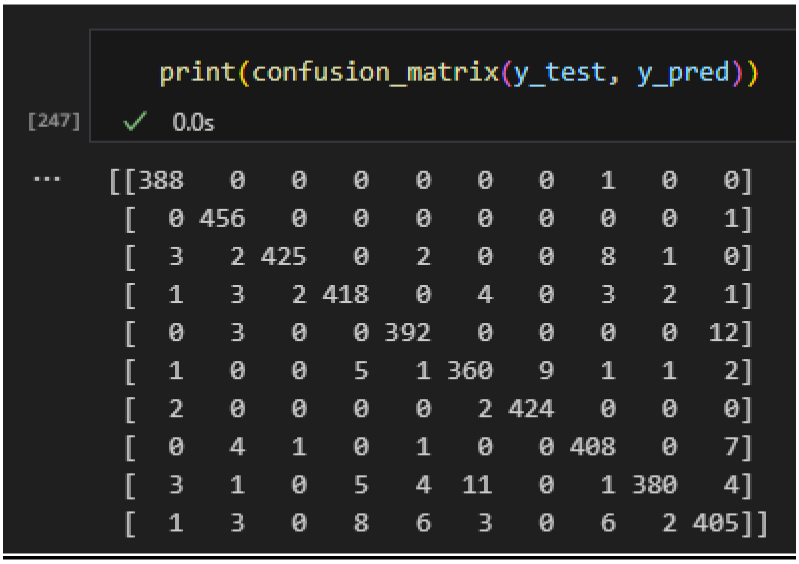
```
0.9657142857142857
```

```
print(classification_report(y_test, y_pred))
```

[246] ✓ 0.0s Python

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       389
           1       0.97      1.00      0.98       457
           2       0.99      0.96      0.98       441
           3       0.96      0.96      0.96       434
           4       0.97      0.96      0.96       407
           5       0.95      0.95      0.95       380
           6       0.98      0.99      0.98       428
           7       0.95      0.97      0.96       421
           8       0.98      0.93      0.96       409
           9       0.94      0.93      0.94       434

    accuracy                           0.97      4200
   macro avg       0.97      0.97      0.97      4200
weighted avg       0.97      0.97      0.97      4200
```

CONFUSION MATRIX



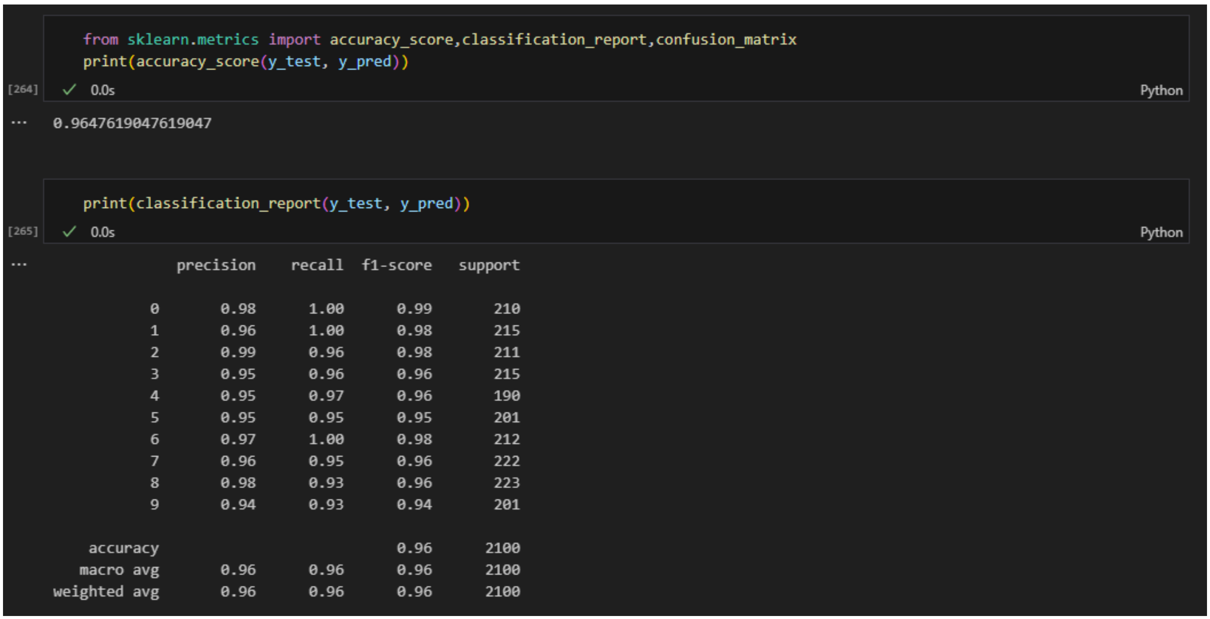
```
print(confusion_matrix(y_test, y_pred))
```

[247] ✓ 0.0s

```
[[388   0   0   0   0   0   0   1   0   0]
 [  0 456   0   0   0   0   0   0   0   1]
 [  3   2 425   0   2   0   0   8   1   0]
 [  1   3   2 418   0   4   0   3   2   1]
 [  0   3   0   0 392   0   0   0   0  12]
 [  1   0   0   5   1 360   9   1   1   2]
 [  2   0   0   0   0   2 424   0   0   0]
 [  0   4   1   0   1   0   0 408   0   7]
 [  3   1   0   5   4  11   0   1 380   4]
 [  1   3   0   8   6   3   0   6   2 405]]
```

6) RATIO OF TRAIN AND TEST DATA - 95:05

ACCURACY – 0.96



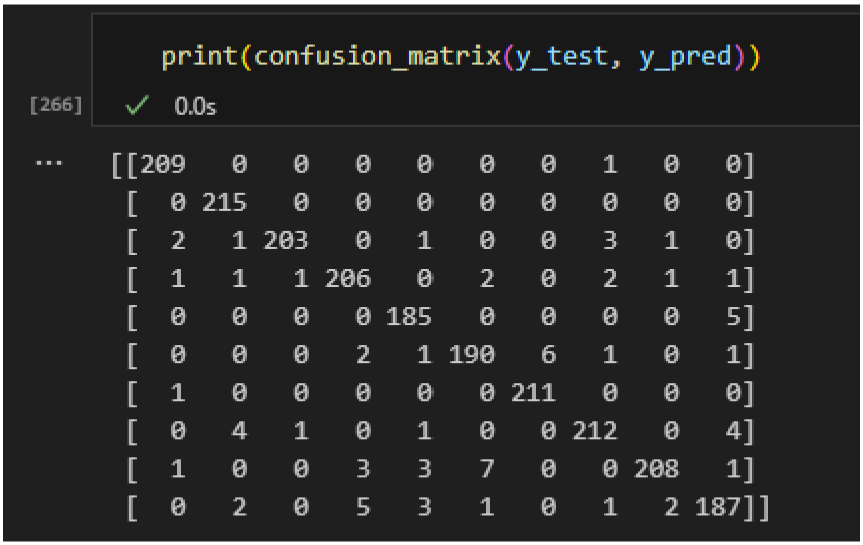
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

```
0.9647619047619047
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       210
           1       0.96      1.00      0.98       215
           2       0.99      0.96      0.98       211
           3       0.95      0.96      0.96       215
           4       0.95      0.97      0.96       190
           5       0.95      0.95      0.95       201
           6       0.97      1.00      0.98       212
           7       0.96      0.95      0.96       222
           8       0.98      0.93      0.96       223
           9       0.94      0.93      0.94       201

    accuracy                           0.96      2100
   macro avg       0.96      0.96      0.96      2100
weighted avg       0.96      0.96      0.96      2100
```

CONFUSION MATRIX



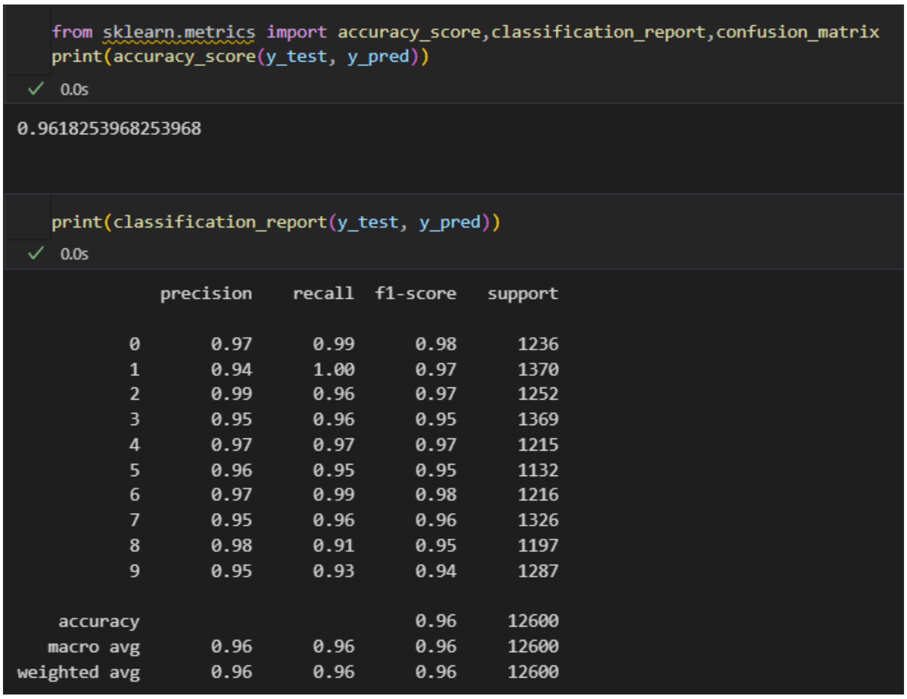
```
print(confusion_matrix(y_test, y_pred))
```

```
[[209   0   0   0   0   0   0   1   0   0]
 [  0 215   0   0   0   0   0   0   0   0]
 [  2   1 203   0   1   0   0   3   1   0]
 [  1   1   1 206   0   2   0   2   1   1]
 [  0   0   0   0 185   0   0   0   0   5]
 [  0   0   0   2   1 190   6   1   0   1]
 [  1   0   0   0   0   0 211   0   0   0]
 [  0   4   1   0   1   0   0 212   0   4]
 [  1   0   0   3   3   7   0   0 208   1]
 [  0   2   0   5   3   1   0   1   2 187]]
```

FOR K = 6

1) RATIO OF TRAIN AND TEST DATA - 70:30

ACCURACY – 0.96



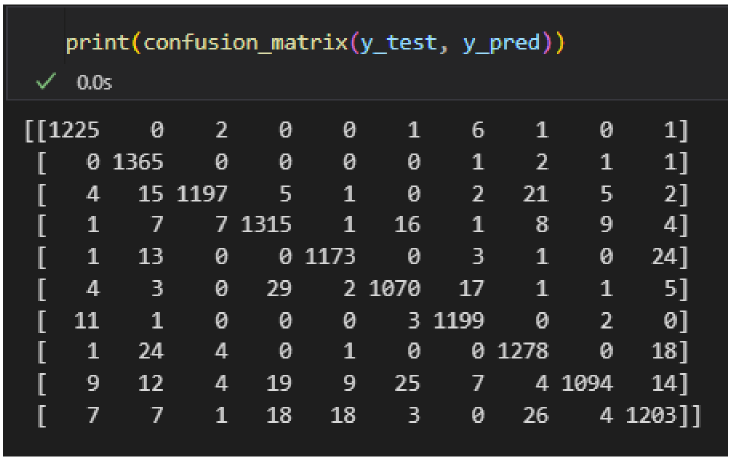
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```
✓ 0.0s

```
0.9618253968253968
```

```
print(classification_report(y_test, y_pred))
```
✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1236
           1       0.94      1.00      0.97      1370
           2       0.99      0.96      0.97      1252
           3       0.95      0.96      0.95      1369
           4       0.97      0.97      0.97      1215
           5       0.96      0.95      0.95      1132
           6       0.97      0.99      0.98      1216
           7       0.95      0.96      0.96      1326
           8       0.98      0.91      0.95      1197
           9       0.95      0.93      0.94      1287

    accuracy                           0.96     12600
   macro avg       0.96      0.96      0.96     12600
weighted avg       0.96      0.96      0.96     12600
```

CONFUSION MATRIX

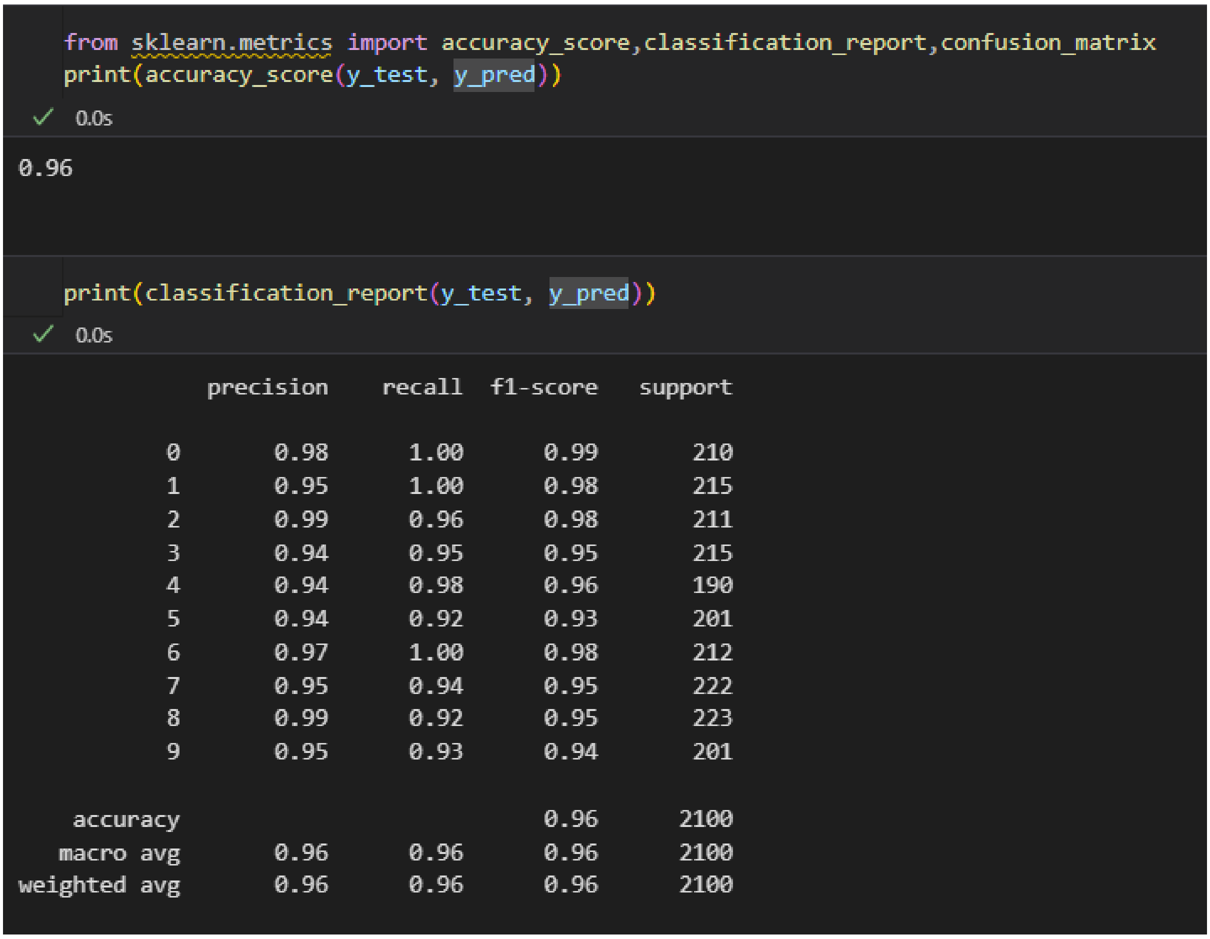


```
print(confusion_matrix(y_test, y_pred))
```
✓ 0.0s

```
[[1225    0    2    0    0    1    6    1    0    1]
 [   0 1365    0    0    0    0    1    2    1    1]
 [   4   15 1197    5    1    0    2   21    5    2]
 [   1    7    7 1315    1   16    1    8    9    4]
 [   1   13    0    0 1173    0    3    1    0   24]
 [   4    3    0   29    2 1070   17    1    1    5]
 [  11    1    0    0    0    3 1199    0    2    0]
 [   1   24    4    0    1    0    0 1278    0   18]
 [   9   12    4   19    9   25    7    4 1094   14]
 [   7    7    1   18   18    3    0   26    4 1203]]
```

2.) RATIO OF TRAIN AND TEST DATA - 95:05

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

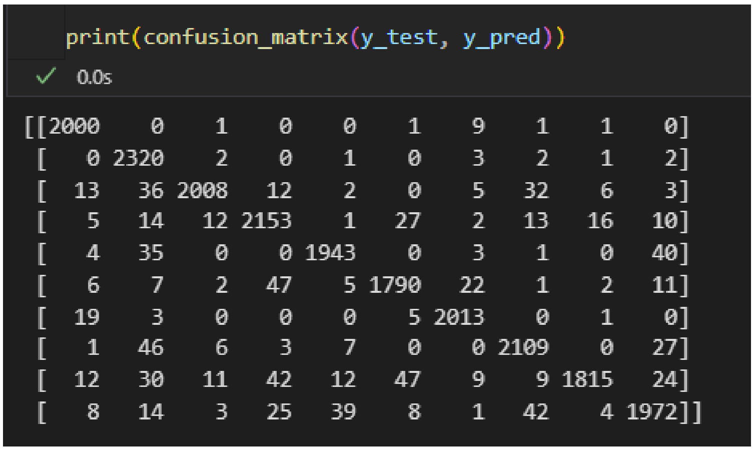
```
0.96
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99       210
           1       0.95      1.00      0.98       215
           2       0.99      0.96      0.98       211
           3       0.94      0.95      0.95       215
           4       0.94      0.98      0.96       190
           5       0.94      0.92      0.93       201
           6       0.97      1.00      0.98       212
           7       0.95      0.94      0.95       222
           8       0.99      0.92      0.95       223
           9       0.95      0.93      0.94       201

    accuracy                           0.96      2100
   macro avg       0.96      0.96      0.96      2100
weighted avg       0.96      0.96      0.96      2100
```

CONFUSION MATRIX



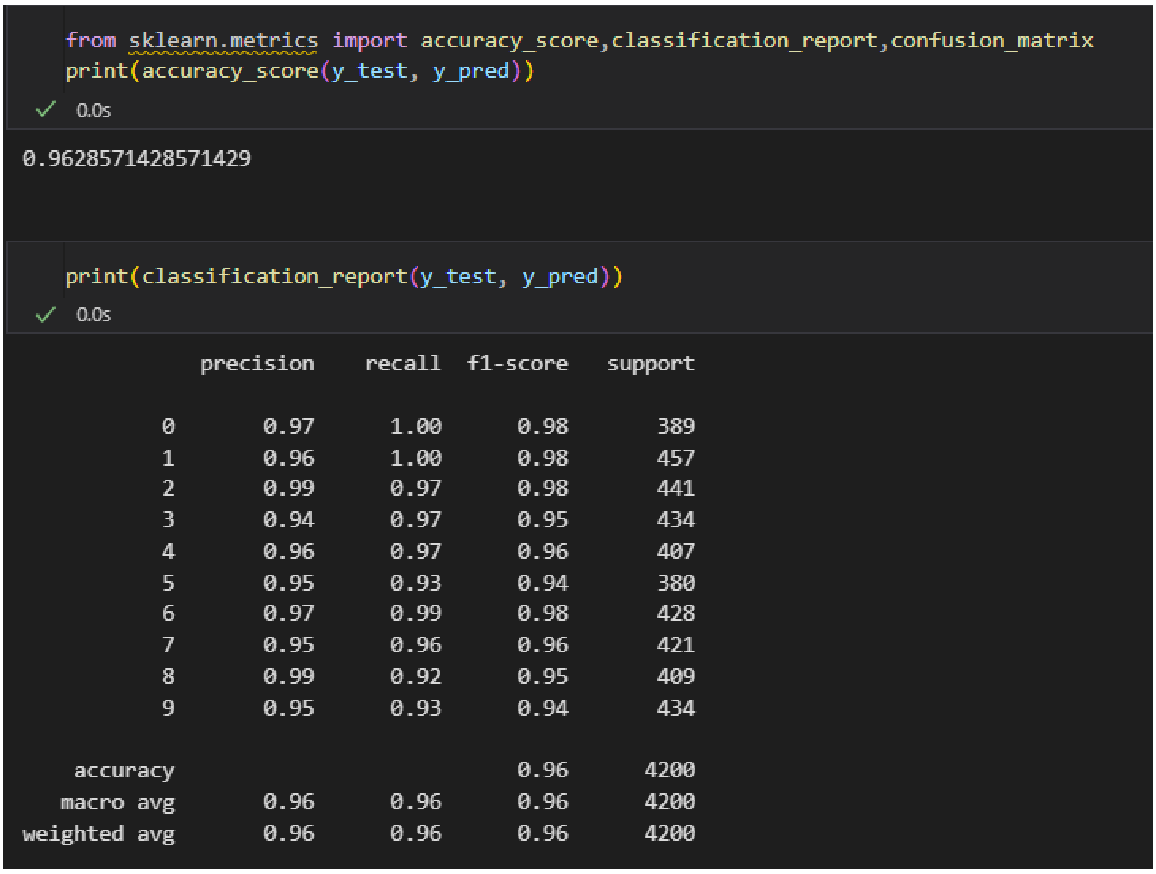
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[2000    0    1    0    0    1    9    1    1    0]
 [   0 2320    2    0    1    0    3    2    1    2]
 [  13   36 2008   12    2    0    5   32    6    3]
 [   5   14   12 2153    1   27    2   13   16   10]
 [   4   35    0    0 1943    0    3    1    0   40]
 [   6    7    2   47    5 1790   22    1    2   11]
 [  19    3    0    0    0    5 2013    0    1    0]
 [   1   46    6    3    7    0    0 2109    0   27]
 [  12   30   11   42   12   47    9    9 1815   24]
 [   8   14    3   25   39    8    1   42    4 1972]]
```

3.) RATIO OF TRAIN AND TEST DATA – 90:10

ACCURACY – 0.96



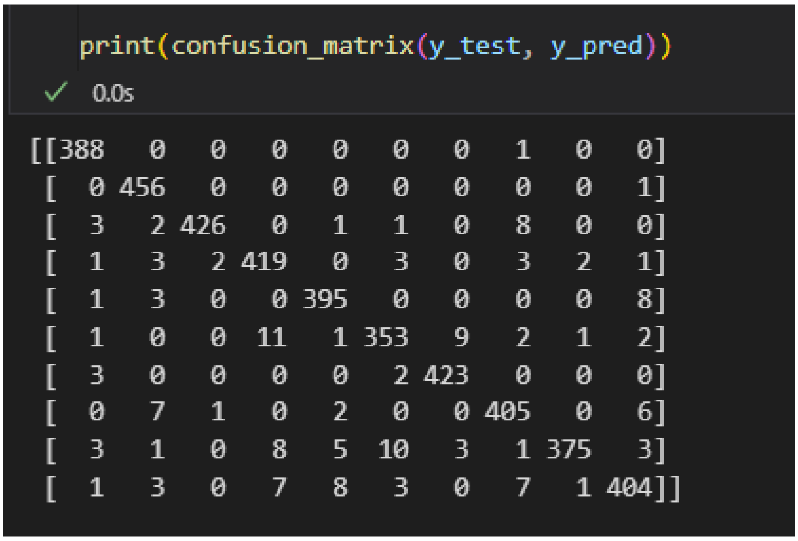
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```
✓ 0.0s

```
0.9628571428571429
```

```
print(classification_report(y_test, y_pred))
```
✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       389
           1       0.96      1.00      0.98       457
           2       0.99      0.97      0.98       441
           3       0.94      0.97      0.95       434
           4       0.96      0.97      0.96       407
           5       0.95      0.93      0.94       380
           6       0.97      0.99      0.98       428
           7       0.95      0.96      0.96       421
           8       0.99      0.92      0.95       409
           9       0.95      0.93      0.94       434

    accuracy                           0.96      4200
   macro avg       0.96      0.96      0.96      4200
weighted avg       0.96      0.96      0.96      4200
```

CONFUSION MATRIX

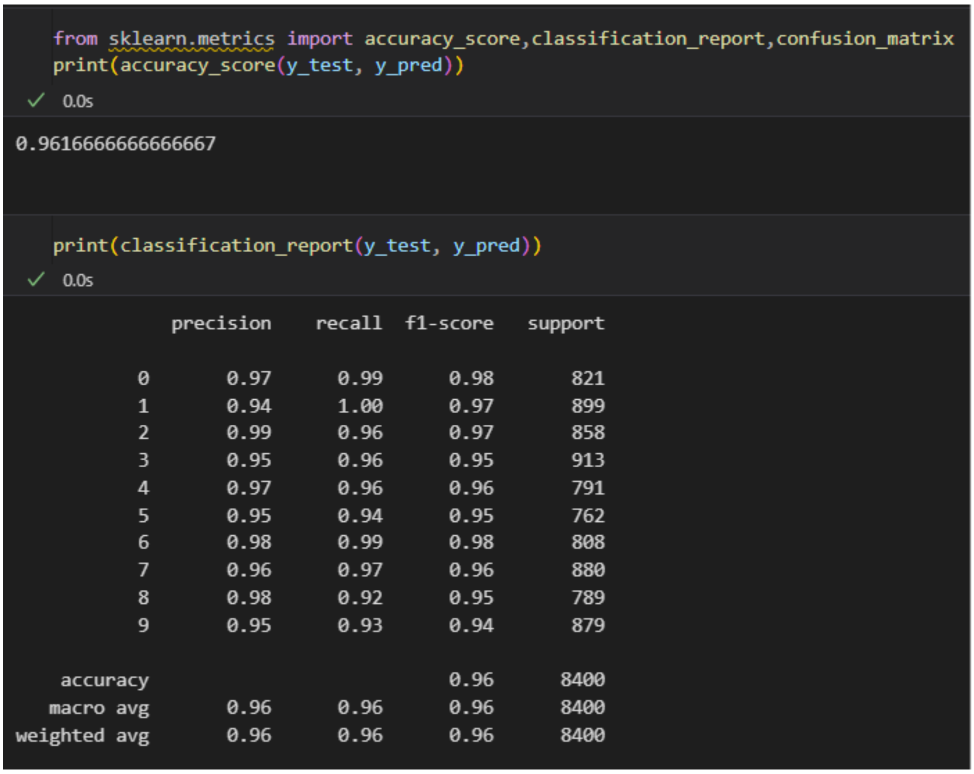


```
print(confusion_matrix(y_test, y_pred))
```
✓ 0.0s

```
[[388   0   0   0   0   0   0   1   0   0]
 [  0 456   0   0   0   0   0   0   0   1]
 [  3   2 426   0   1   1   0   8   0   0]
 [  1   3   2 419   0   3   0   3   2   1]
 [  1   3   0   0 395   0   0   0   0   8]
 [  1   0   0  11   1 353   9   2   1   2]
 [  3   0   0   0   0   2 423   0   0   0]
 [  0   7   1   0   2   0   0 405   0   6]
 [  3   1   0   8   5  10   3   1 375   3]
 [  1   3   0   7   8   3   0   7   1 404]]
```

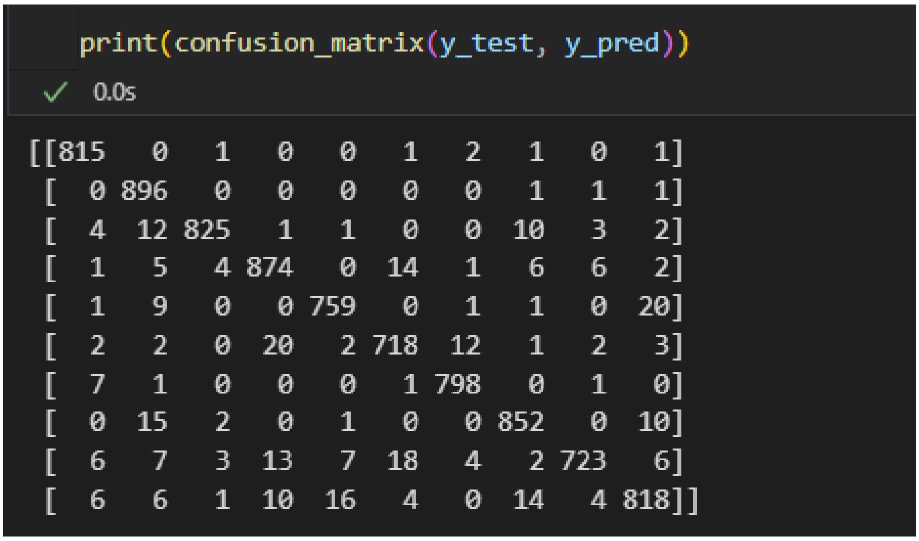
4) RATIO OF TRAIN AND TEST DATA – 80:20

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
✓ 0.0s

0.9616666666666667

print(classification_report(y_test, y_pred))
✓ 0.0s

              precision    recall  f1-score   support

           0       0.97      0.99      0.98       821
           1       0.94      1.00      0.97       899
           2       0.99      0.96      0.97       858
           3       0.95      0.96      0.95       913
           4       0.97      0.96      0.96       791
           5       0.95      0.94      0.95       762
           6       0.98      0.99      0.98       808
           7       0.96      0.97      0.96       880
           8       0.98      0.92      0.95       789
           9       0.95      0.93      0.94       879

    accuracy                           0.96      8400
   macro avg       0.96      0.96      0.96      8400
weighted avg       0.96      0.96      0.96      8400
```

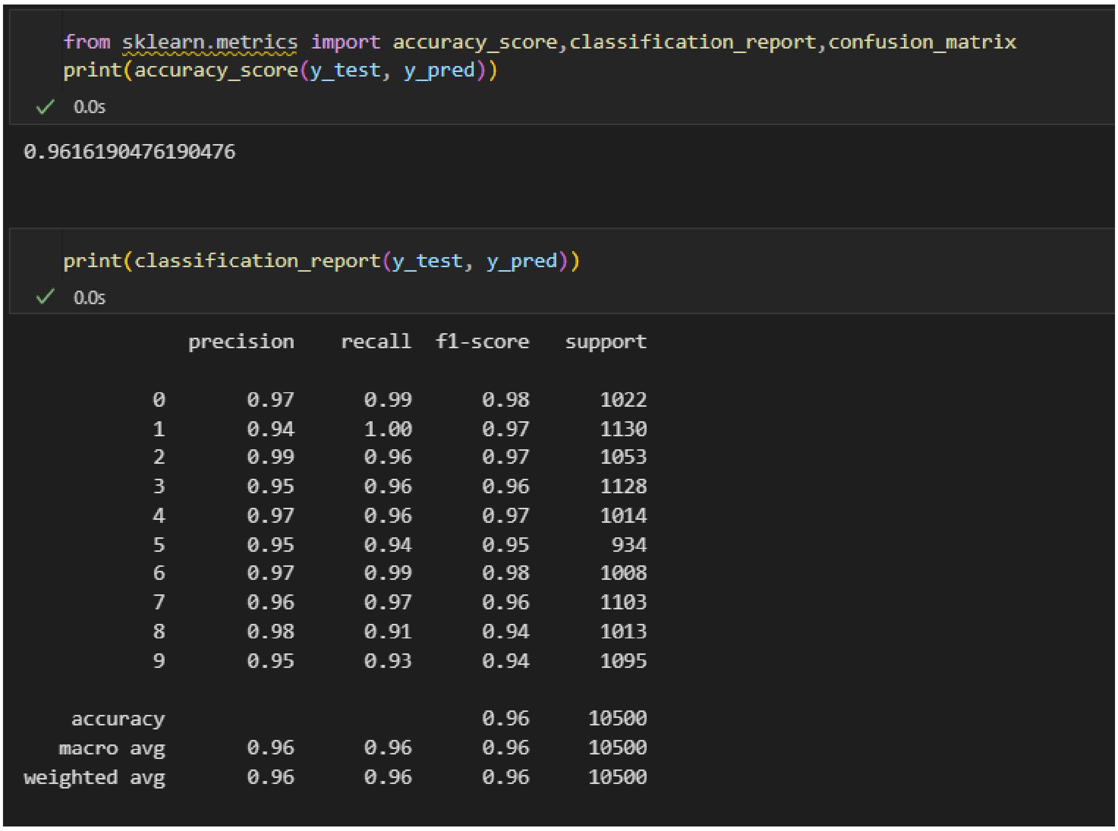
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
✓ 0.0s

[[815   0   1   0   0   1   2   1   0   1]
 [  0 896   0   0   0   0   0   1   1   1]
 [  4  12 825   1   1   0   0  10   3   2]
 [  1   5   4 874   0  14   1   6   6   2]
 [  1   9   0   0 759   0   1   1   0  20]
 [  2   2   0  20   2 718  12   1   2   3]
 [  7   1   0   0   0   1 798   0   1   0]
 [  0  15   2   0   1   0   0 852   0  10]
 [  6   7   3  13   7  18   4   2 723   6]
 [  6   6   1  10  16   4   0  14   4 818]]
```

5) RATIO OF TRAIN AND TEST DATA – 75:25

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

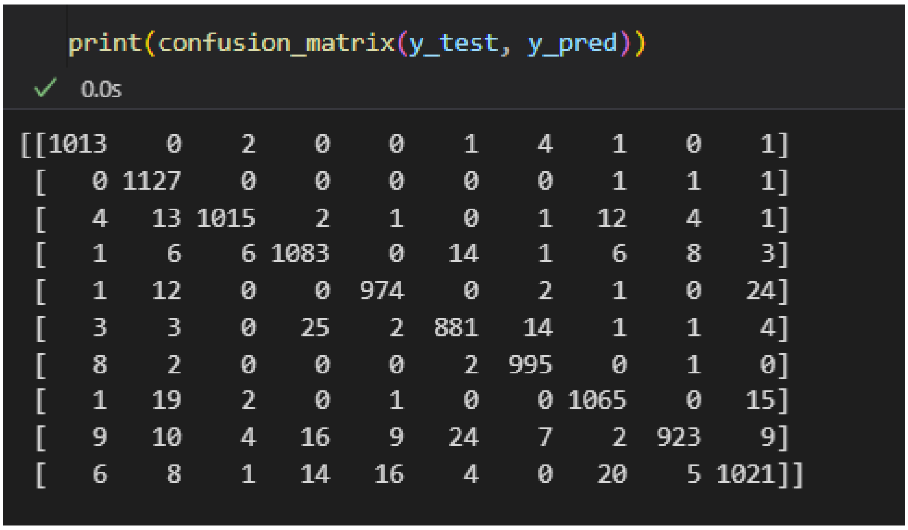
```
0.9616190476190476
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1022
           1       0.94      1.00      0.97      1130
           2       0.99      0.96      0.97      1053
           3       0.95      0.96      0.96      1128
           4       0.97      0.96      0.97      1014
           5       0.95      0.94      0.95       934
           6       0.97      0.99      0.98      1008
           7       0.96      0.97      0.96      1103
           8       0.98      0.91      0.94      1013
           9       0.95      0.93      0.94      1095

    accuracy                           0.96     10500
   macro avg       0.96      0.96      0.96     10500
weighted avg       0.96      0.96      0.96     10500
```

CONFUSION MATRIX



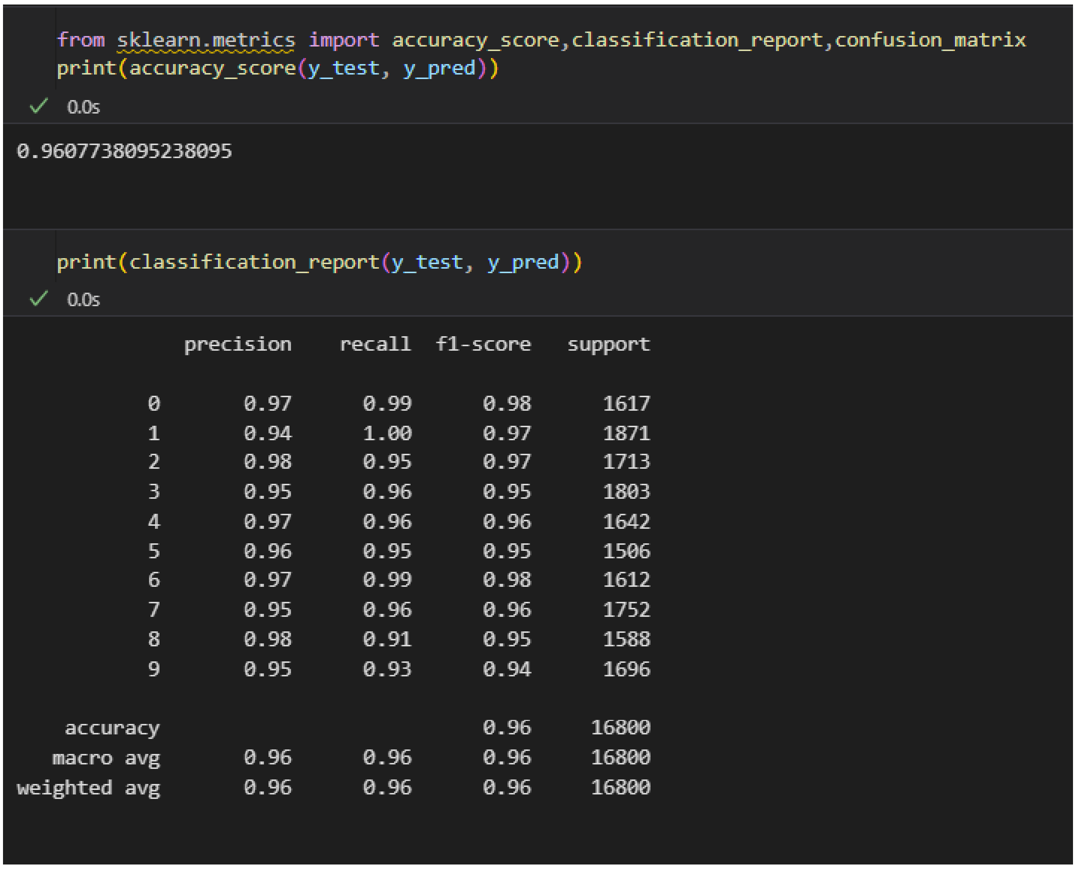
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1013    0    2    0    0    1    4    1    0    1]
 [   0 1127    0    0    0    0    0    1    1    1]
 [   4   13 1015    2    1    0    1   12    4    1]
 [   1    6    6 1083    0   14    1    6    8    3]
 [   1   12    0    0  974    0    2    1    0   24]
 [   3    3    0   25    2  881   14    1    1    4]
 [   8    2    0    0    0    2  995    0    1    0]
 [   1   19    2    0    1    0    0 1065    0   15]
 [   9   10    4   16    9   24    7    2  923    9]
 [   6    8    1   14   16    4    0   20    5 1021]]
```

6) RATIO OF TRAIN AND TEST DATA – 60:40

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

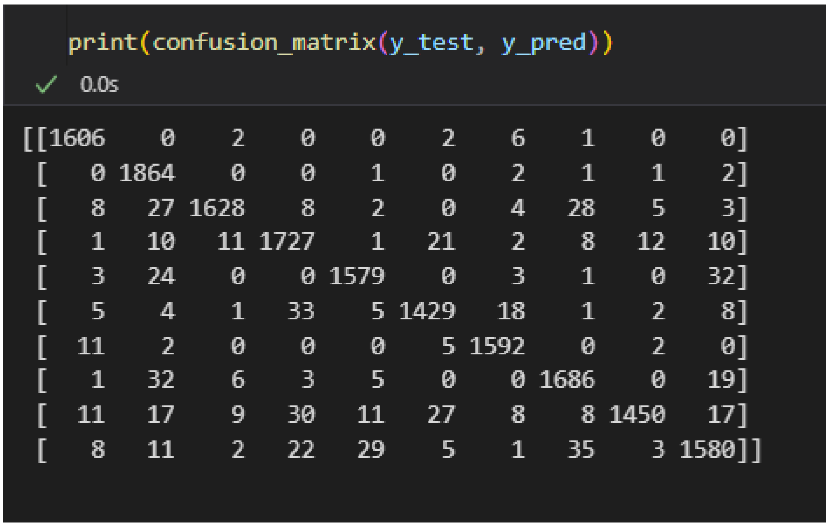
```
0.9607738095238095
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1617
           1       0.94      1.00      0.97      1871
           2       0.98      0.95      0.97      1713
           3       0.95      0.96      0.95      1803
           4       0.97      0.96      0.96      1642
           5       0.96      0.95      0.95      1506
           6       0.97      0.99      0.98      1612
           7       0.95      0.96      0.96      1752
           8       0.98      0.91      0.95      1588
           9       0.95      0.93      0.94      1696

    accuracy                           0.96     16800
   macro avg       0.96      0.96      0.96     16800
weighted avg       0.96      0.96      0.96     16800
```

CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
```

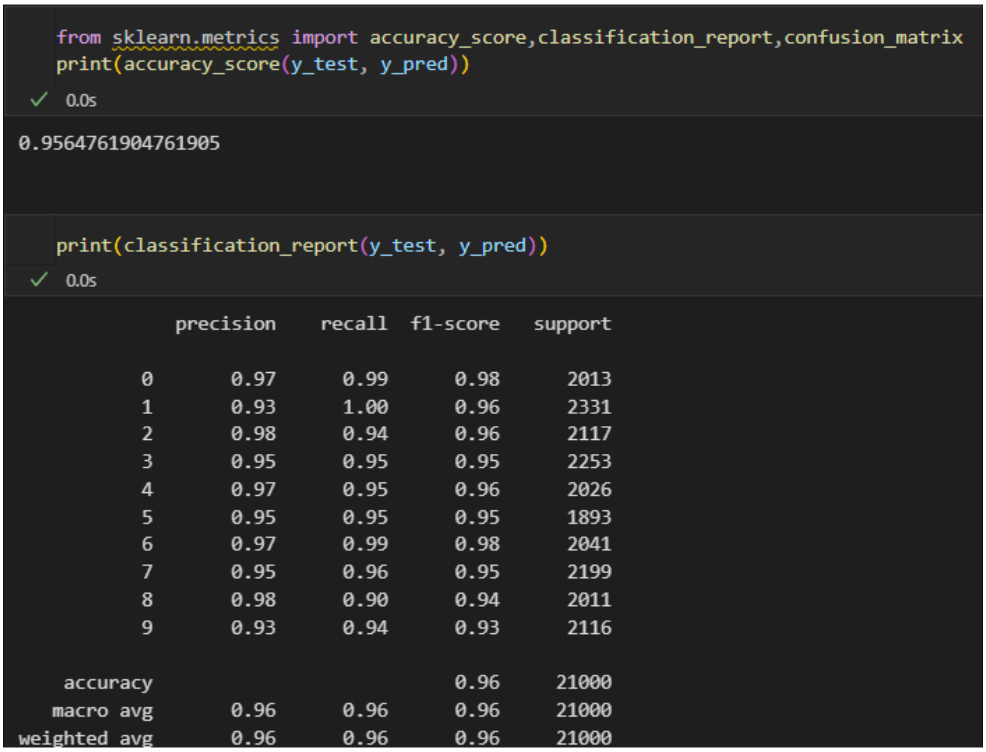
✓ 0.0s

```
[[1606    0    2    0    0    2    6    1    0    0]
 [   0 1864    0    0    1    0    2    1    1    2]
 [   8   27 1628    8    2    0    4   28    5    3]
 [   1   10   11 1727    1   21    2    8   12   10]
 [   3   24    0    0 1579    0    3    1    0   32]
 [   5    4    1   33    5 1429   18    1    2    8]
 [  11    2    0    0    0    5 1592    0    2    0]
 [   1   32    6    3    5    0    0 1686    0   19]
 [  11   17    9   30   11   27    8    8 1450   17]
 [   8   11    2   22   29    5    1   35    3 1580]]
```

FOR K = 7

1.) RATIO OF TRAIN AND TEST DATA - 95:05

ACCURACY – 0.95



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

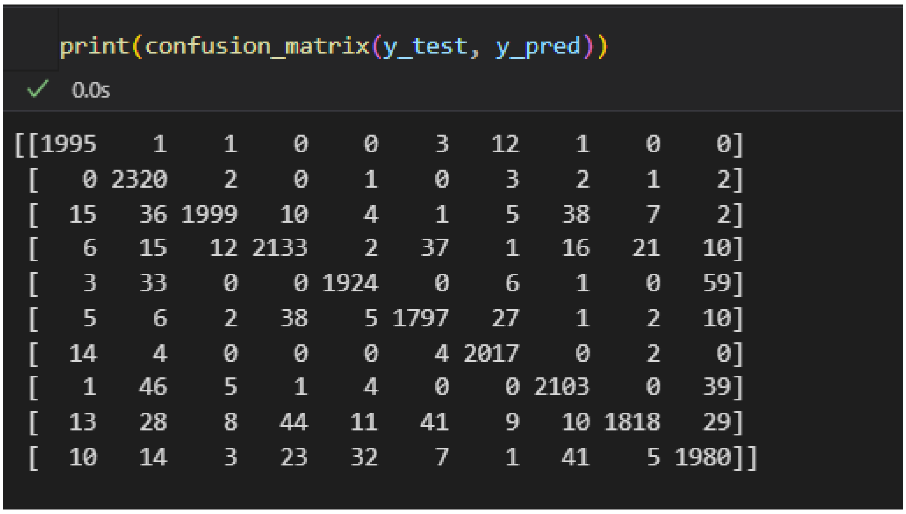
```
0.9564761904761905
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      2013
           1       0.93      1.00      0.96      2331
           2       0.98      0.94      0.96      2117
           3       0.95      0.95      0.95      2253
           4       0.97      0.95      0.96      2026
           5       0.95      0.95      0.95      1893
           6       0.97      0.99      0.98      2041
           7       0.95      0.96      0.95      2199
           8       0.98      0.90      0.94      2011
           9       0.93      0.94      0.93      2116

    accuracy                           0.96     21000
   macro avg       0.96      0.96      0.96     21000
weighted avg       0.96      0.96      0.96     21000
```

CONFUSION MATRIX



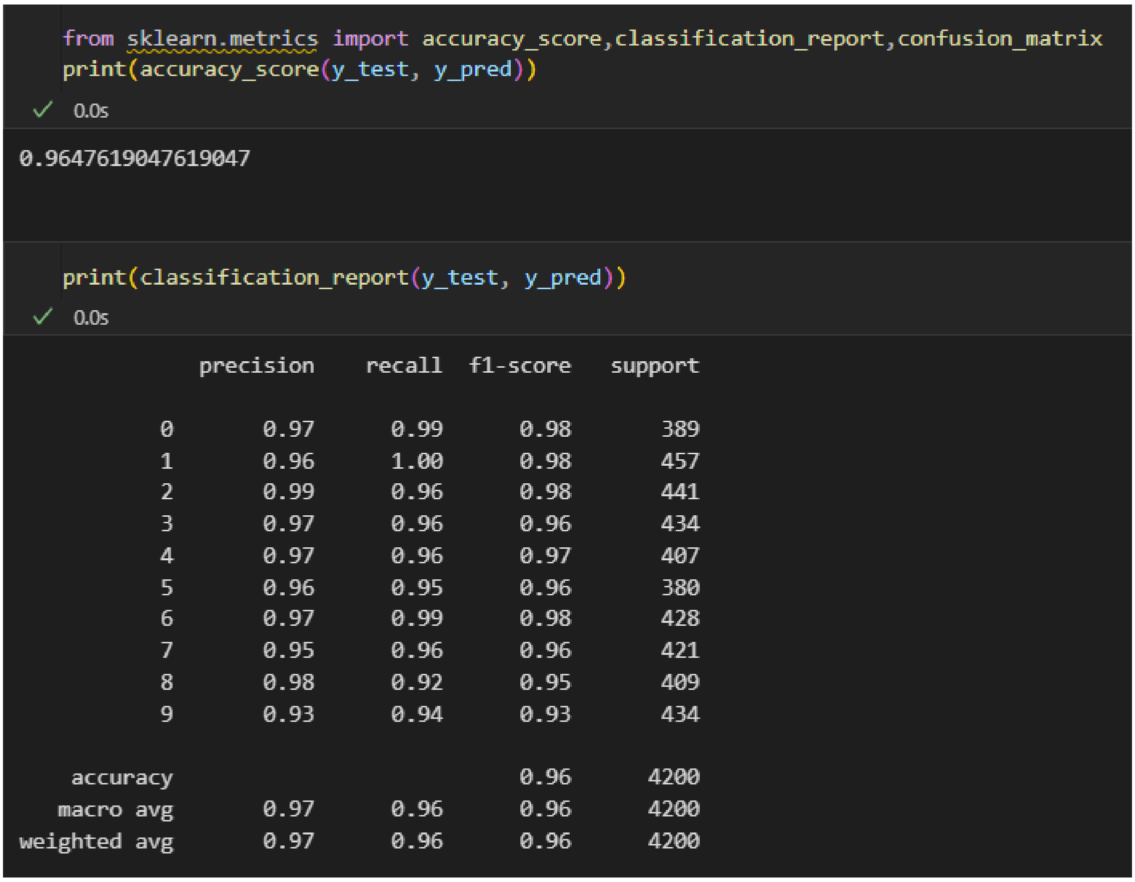
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1995    1    1    0    0    3   12    1    0    0]
 [   0 2320    2    0    1    0    3    2    1    2]
 [  15   36 1999   10    4    1    5   38    7    2]
 [   6   15   12 2133    2   37    1   16   21   10]
 [   3   33    0    0 1924    0    6    1    0   59]
 [   5    6    2   38    5 1797   27    1    2   10]
 [  14    4    0    0    0    4 2017    0    2    0]
 [   1   46    5    1    4    0    0 2103    0   39]
 [  13   28    8   44   11   41    9   10 1818   29]
 [  10   14    3   23   32    7    1   41    5 1980]]
```

2.) RATIO OF TRAIN AND TEST DATA - 90:10

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

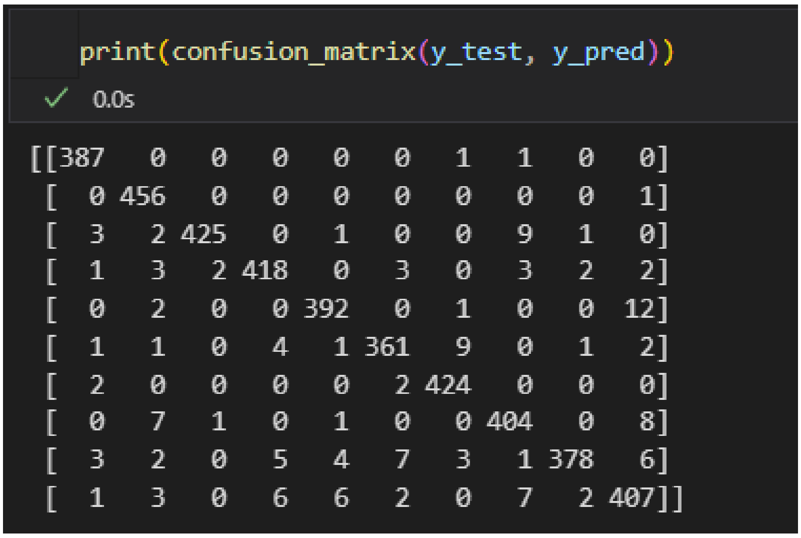
```
0.9647619047619047
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       389
           1       0.96      1.00      0.98       457
           2       0.99      0.96      0.98       441
           3       0.97      0.96      0.96       434
           4       0.97      0.96      0.97       407
           5       0.96      0.95      0.96       380
           6       0.97      0.99      0.98       428
           7       0.95      0.96      0.96       421
           8       0.98      0.92      0.95       409
           9       0.93      0.94      0.93       434

    accuracy                           0.96      4200
   macro avg       0.97      0.96      0.96      4200
weighted avg       0.97      0.96      0.96      4200
```

CONFUSION MATRIX



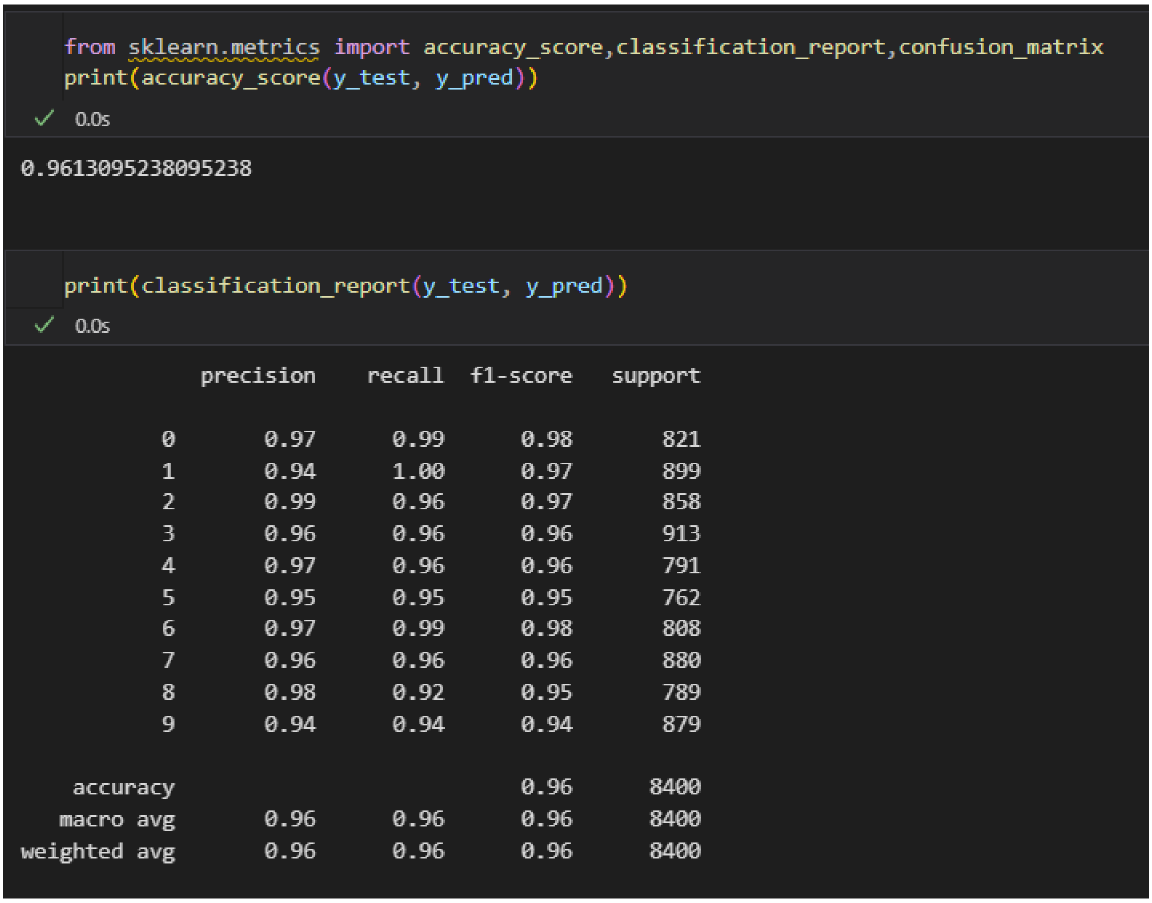
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[387   0   0   0   0   0   1   1   0   0]
 [  0 456   0   0   0   0   0   0   0   1]
 [  3   2 425   0   1   0   0   9   1   0]
 [  1   3   2 418   0   3   0   3   2   2]
 [  0   2   0   0 392   0   1   0   0  12]
 [  1   1   0   4   1 361   9   0   1   2]
 [  2   0   0   0   0   2 424   0   0   0]
 [  0   7   1   0   1   0   0 404   0   8]
 [  3   2   0   5   4   7   3   1 378   6]
 [  1   3   0   6   6   2   0   7   2 407]]
```

3.) RATIO OF TRAIN AND TEST DATA – 80:20

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

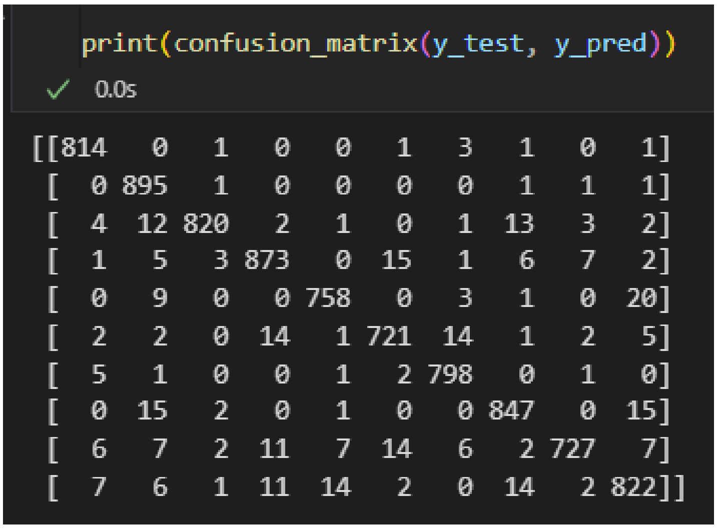
```
0.9613095238095238
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       821
           1       0.94      1.00      0.97       899
           2       0.99      0.96      0.97       858
           3       0.96      0.96      0.96       913
           4       0.97      0.96      0.96       791
           5       0.95      0.95      0.95       762
           6       0.97      0.99      0.98       808
           7       0.96      0.96      0.96       880
           8       0.98      0.92      0.95       789
           9       0.94      0.94      0.94       879

    accuracy                           0.96      8400
   macro avg       0.96      0.96      0.96      8400
weighted avg       0.96      0.96      0.96      8400
```

CONFUSION MATRIX



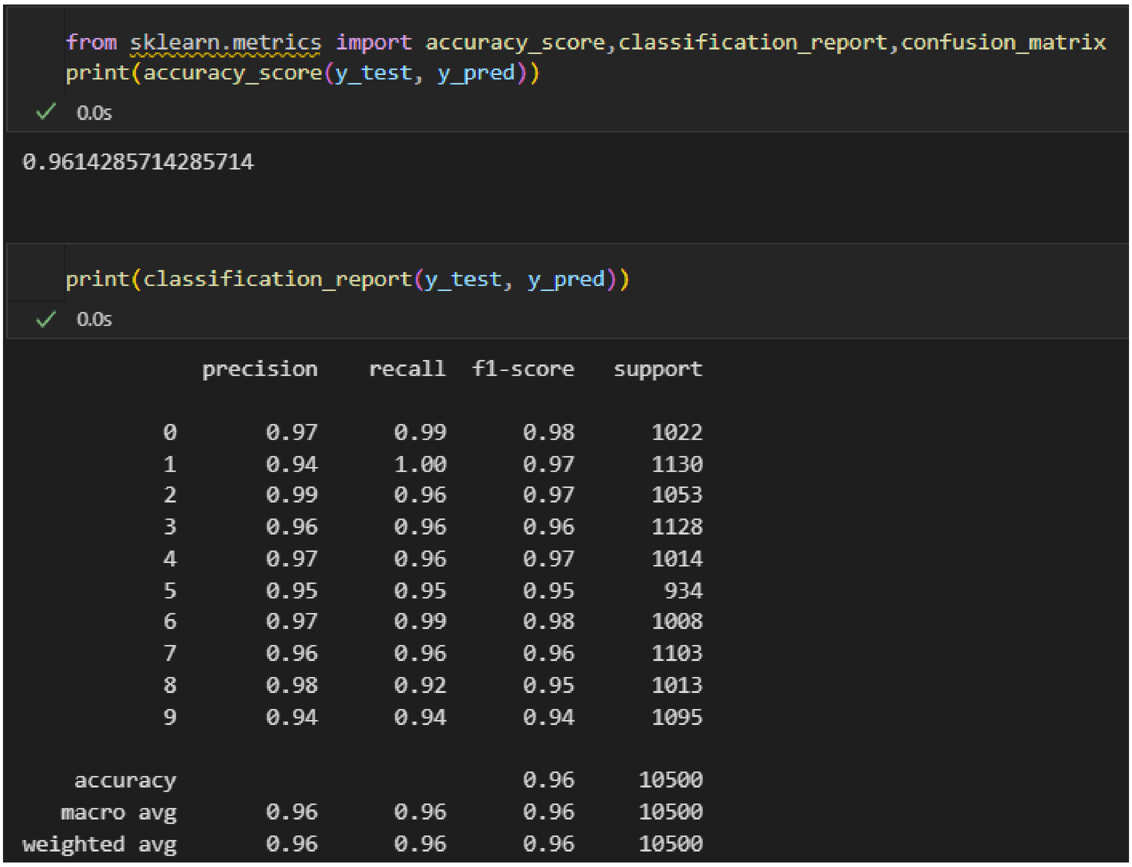
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[814   0   1   0   0   1   3   1   0   1]
 [  0 895   1   0   0   0   0   1   1   1]
 [  4  12 820   2   1   0   1  13   3   2]
 [  1   5   3 873   0  15   1   6   7   2]
 [  0   9   0   0 758   0   3   1   0  20]
 [  2   2   0  14   1 721  14   1   2   5]
 [  5   1   0   0   1   2 798   0   1   0]
 [  0  15   2   0   1   0   0 847   0  15]
 [  6   7   2  11   7  14   6   2 727   7]
 [  7   6   1  11  14   2   0  14   2 822]]
```

4.) RATIO OF TRAIN AND TEST DATA – 75:25

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

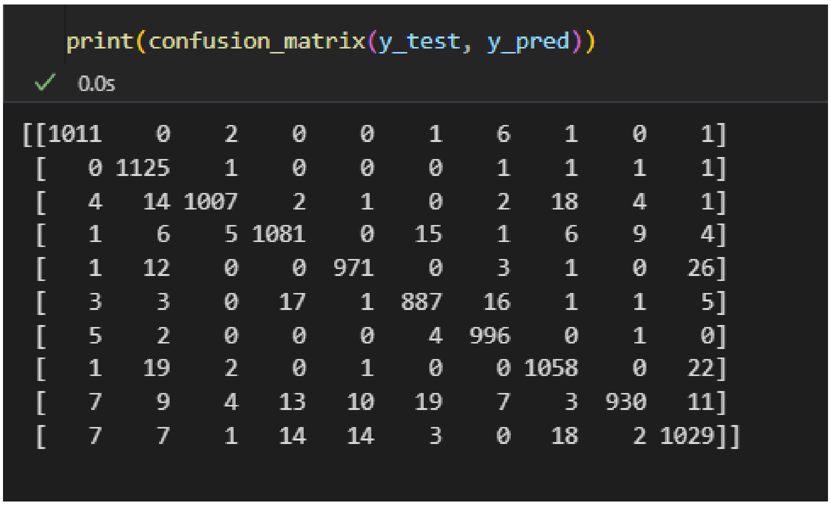
```
0.9614285714285714
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1022
           1       0.94      1.00      0.97      1130
           2       0.99      0.96      0.97      1053
           3       0.96      0.96      0.96      1128
           4       0.97      0.96      0.97      1014
           5       0.95      0.95      0.95       934
           6       0.97      0.99      0.98      1008
           7       0.96      0.96      0.96      1103
           8       0.98      0.92      0.95      1013
           9       0.94      0.94      0.94      1095

    accuracy                           0.96     10500
   macro avg       0.96      0.96      0.96     10500
weighted avg       0.96      0.96      0.96     10500
```

CONFUSION MATRIX



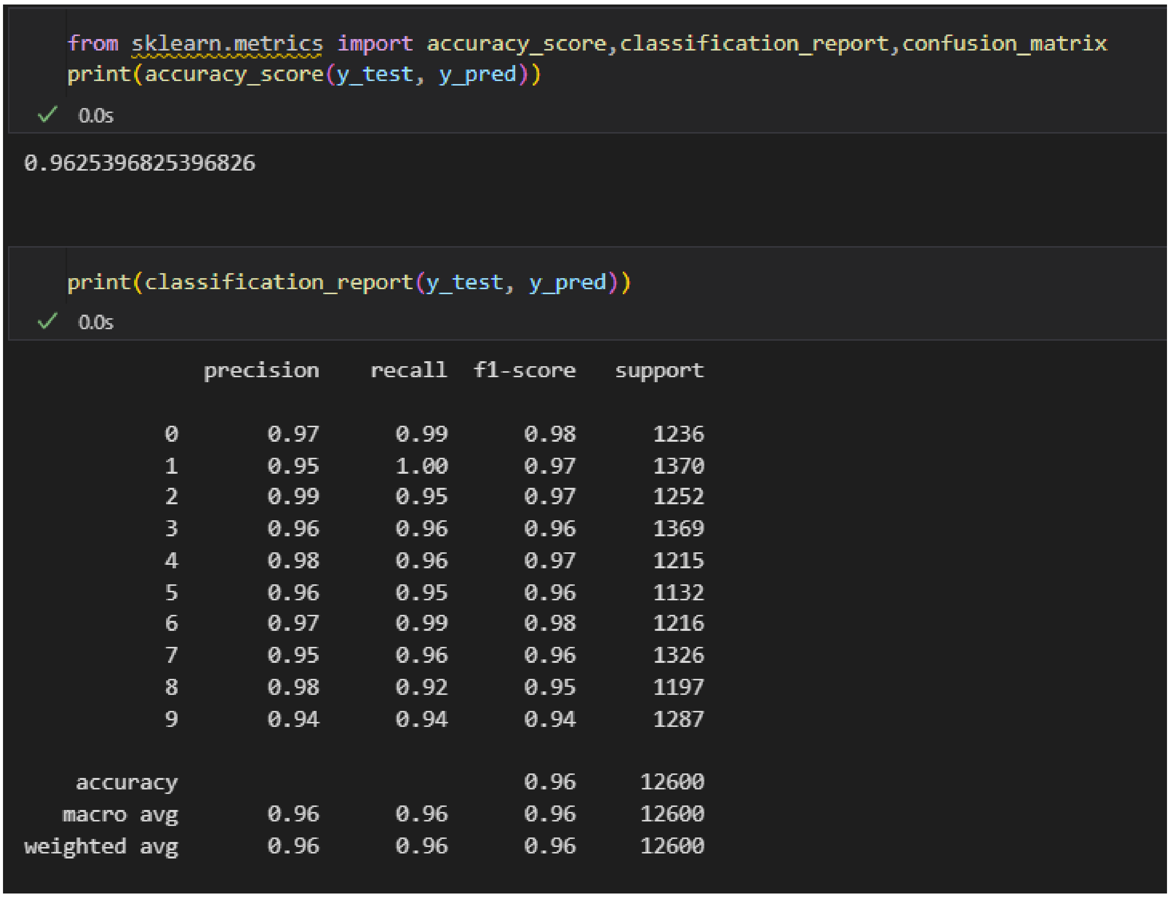
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1011    0    2    0    0    1    6    1    0    1]
 [   0 1125    1    0    0    0    1    1    1    1]
 [   4   14 1007    2    1    0    2   18    4    1]
 [   1    6    5 1081    0   15    1    6    9    4]
 [   1   12    0    0  971    0    3    1    0   26]
 [   3    3    0   17    1  887   16    1    1    5]
 [   5    2    0    0    0    4  996    0    1    0]
 [   1   19    2    0    1    0    0 1058    0   22]
 [   7    9    4   13   10   19    7    3  930   11]
 [   7    7    1   14   14    3    0   18    2 1029]]
```

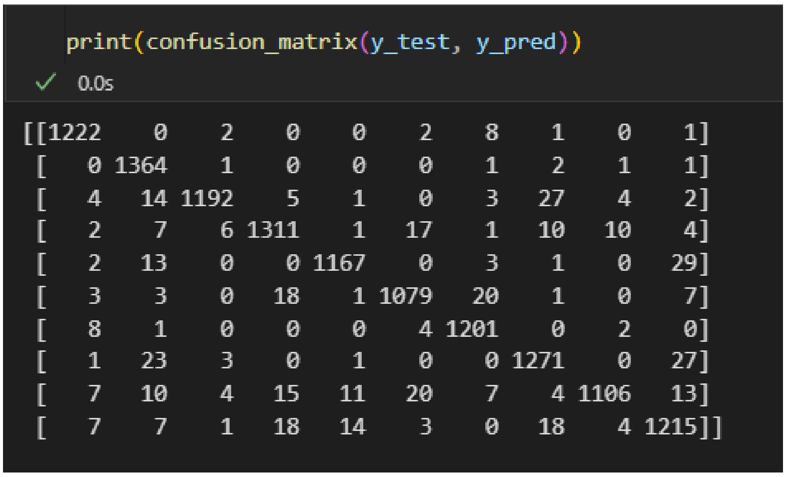
5.) RATIO OF TRAIN AND TEST DATA – 70:30

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
✓ 0.0s

0.9625396825396826

print(classification_report(y_test, y_pred))
✓ 0.0s

              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1236
           1       0.95      1.00      0.97      1370
           2       0.99      0.95      0.97      1252
           3       0.96      0.96      0.96      1369
           4       0.98      0.96      0.97      1215
           5       0.96      0.95      0.96      1132
           6       0.97      0.99      0.98      1216
           7       0.95      0.96      0.96      1326
           8       0.98      0.92      0.95      1197
           9       0.94      0.94      0.94      1287

    accuracy                           0.96     12600
   macro avg       0.96      0.96      0.96     12600
weighted avg       0.96      0.96      0.96     12600
```

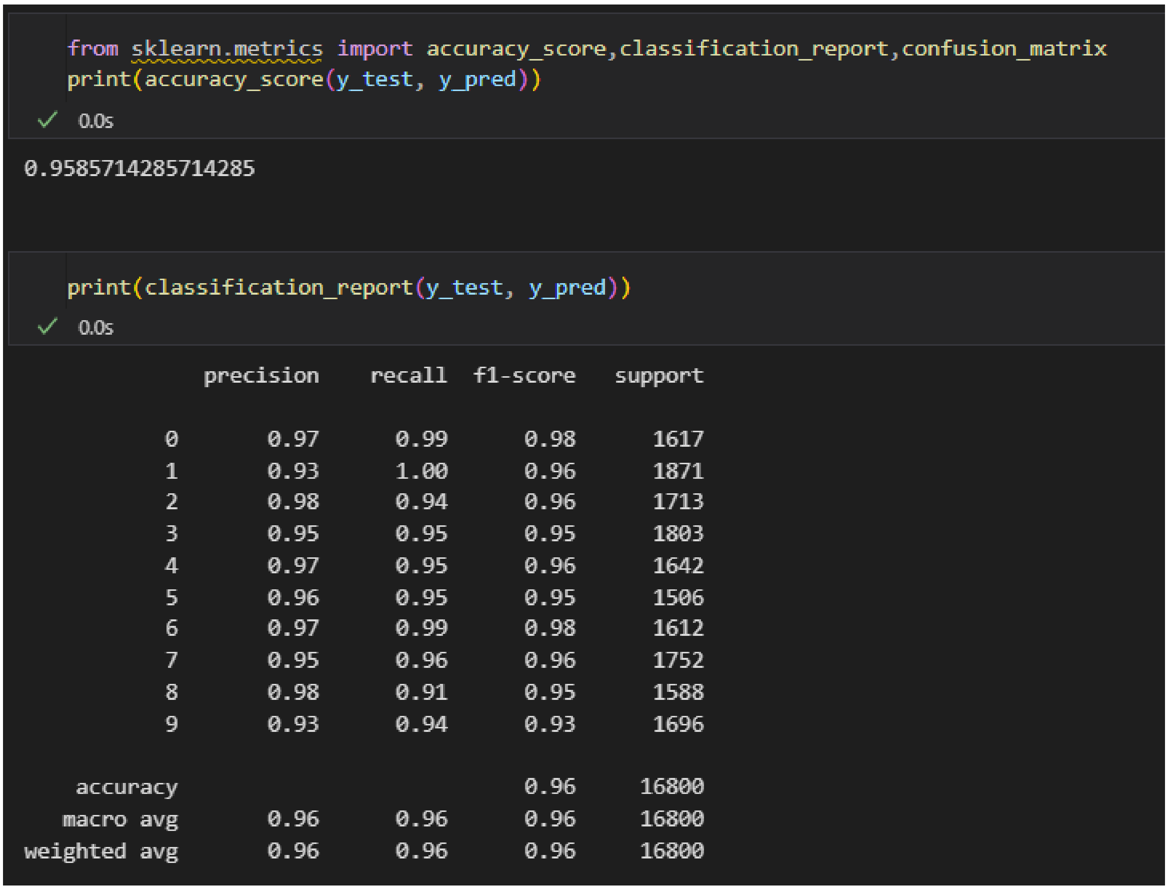
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
✓ 0.0s

[[1222    0    2    0    0    2    8    1    0    1]
 [   0 1364    1    0    0    0    1    2    1    1]
 [   4   14 1192    5    1    0    3   27    4    2]
 [   2    7    6 1311    1   17    1   10   10    4]
 [   2   13    0    0 1167    0    3    1    0   29]
 [   3    3    0   18    1 1079   20    1    0    7]
 [   8    1    0    0    0    4 1201    0    2    0]
 [   1   23    3    0    1    0    0 1271    0   27]
 [   7   10    4   15   11   20    7    4 1106   13]
 [   7    7    1   18   14    3    0   18    4 1215]]
```

6.) RATIO OF TRAIN AND TEST DATA – 60:40

ACCURACY – 0.95



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

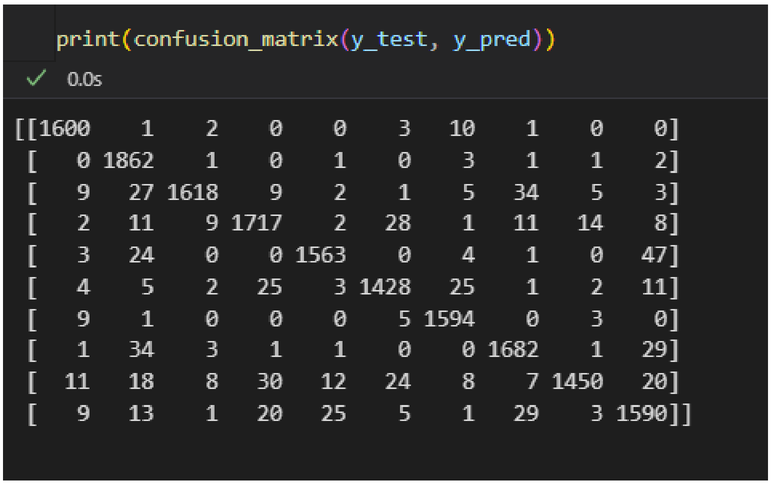
```
0.9585714285714285
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1617
           1       0.93      1.00      0.96      1871
           2       0.98      0.94      0.96      1713
           3       0.95      0.95      0.95      1803
           4       0.97      0.95      0.96      1642
           5       0.96      0.95      0.95      1506
           6       0.97      0.99      0.98      1612
           7       0.95      0.96      0.96      1752
           8       0.98      0.91      0.95      1588
           9       0.93      0.94      0.93      1696

    accuracy                           0.96     16800
   macro avg       0.96      0.96      0.96     16800
weighted avg       0.96      0.96      0.96     16800
```

CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
```

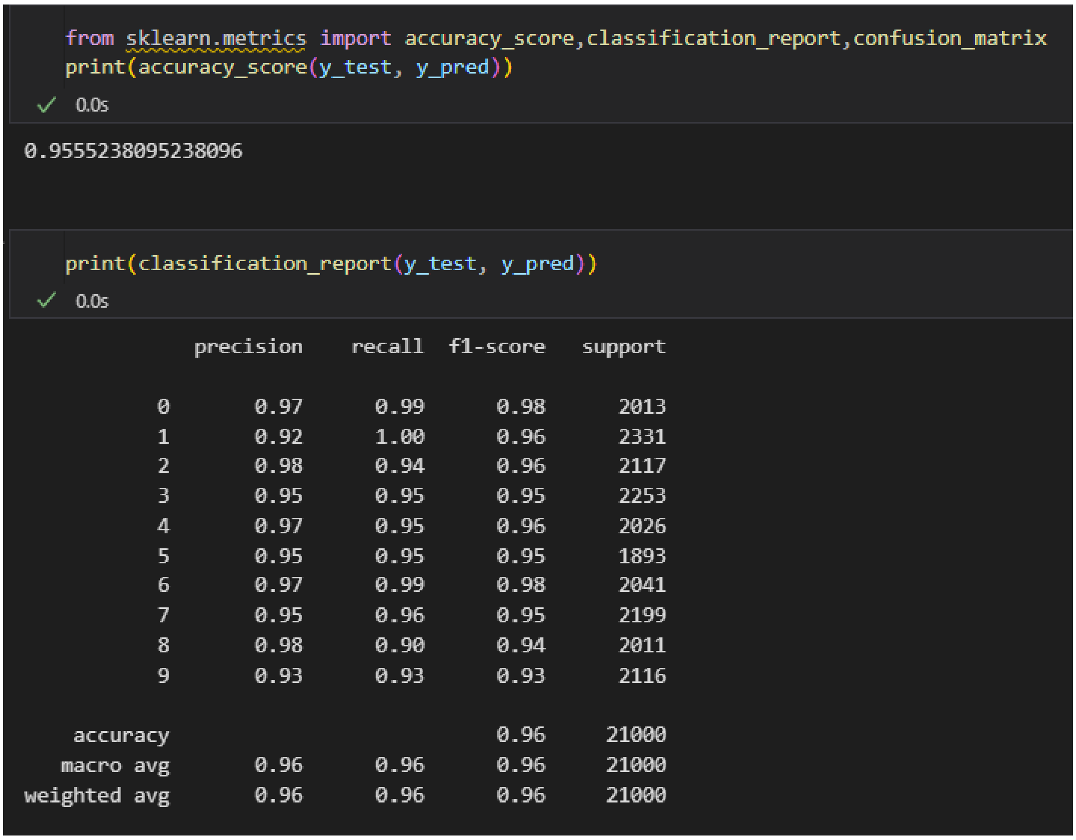
✓ 0.0s

```
[[1600    1    2    0    0    3   10    1    0    0]
 [   0 1862    1    0    1    0    3    1    1    2]
 [   9   27 1618    9    2    1    5   34    5    3]
 [   2   11    9 1717    2   28    1   11   14    8]
 [   3   24    0    0 1563    0    4    1    0   47]
 [   4    5    2   25    3 1428   25    1    2   11]
 [   9    1    0    0    0    5 1594    0    3    0]
 [   1   34    3    1    1    0    0 1682    1   29]
 [  11   18    8   30   12   24    8    7 1450   20]
 [   9   13    1   20   25    5    1   29    3 1590]]
```

FOR K = 10

1.) RATIO OF TRAIN AND TEST DATA - 95:05

ACCURACY – 0.95



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

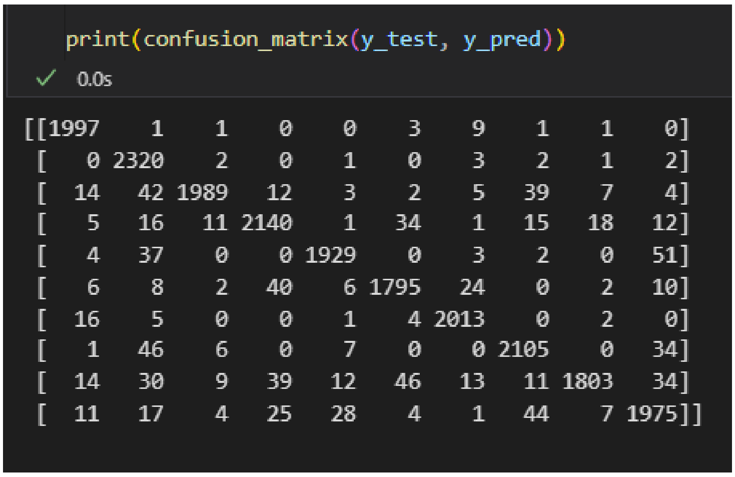
```
0.9555238095238096
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      2013
           1       0.92      1.00      0.96      2331
           2       0.98      0.94      0.96      2117
           3       0.95      0.95      0.95      2253
           4       0.97      0.95      0.96      2026
           5       0.95      0.95      0.95      1893
           6       0.97      0.99      0.98      2041
           7       0.95      0.96      0.95      2199
           8       0.98      0.90      0.94      2011
           9       0.93      0.93      0.93      2116

    accuracy                           0.96     21000
   macro avg       0.96      0.96      0.96     21000
weighted avg       0.96      0.96      0.96     21000
```

CONFUSION MATRIX



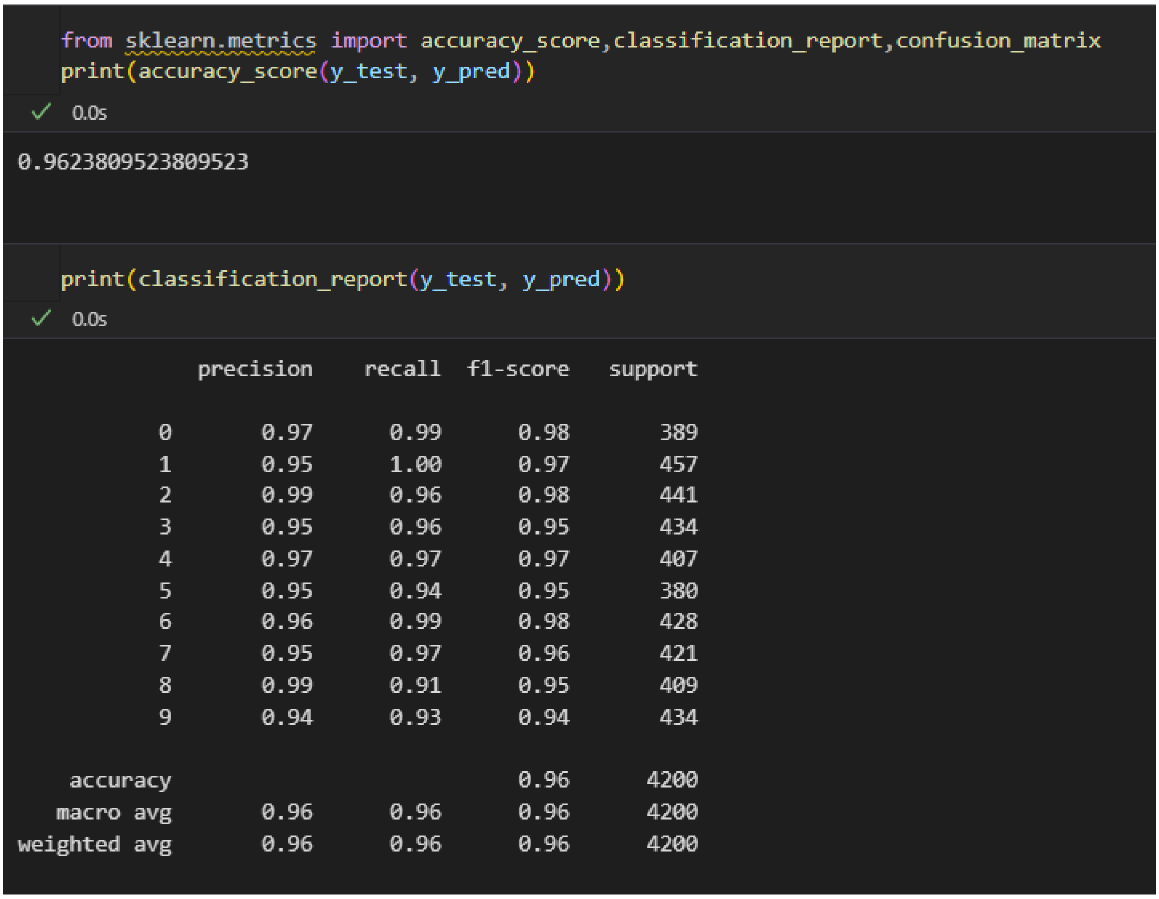
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1997    1    1    0    0    3    9    1    1    0]
 [   0 2320    2    0    1    0    3    2    1    2]
 [  14   42 1989   12    3    2    5   39    7    4]
 [   5   16   11 2140    1   34    1   15   18   12]
 [   4   37    0    0 1929    0    3    2    0   51]
 [   6    8    2   40    6 1795   24    0    2   10]
 [  16    5    0    0    1    4 2013    0    2    0]
 [   1   46    6    0    7    0    0 2105    0   34]
 [  14   30    9   39   12   46   13   11 1803   34]
 [  11   17    4   25   28    4    1   44    7 1975]]
```

2.) RATIO OF TRAIN AND TEST DATA – 90:10

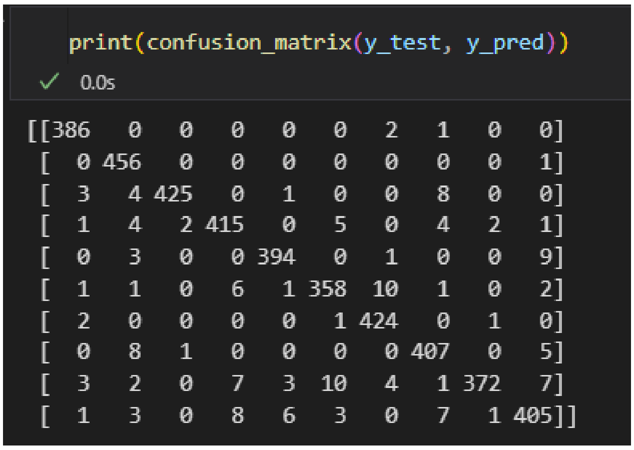
ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
✓ 0.0s

0.9623809523809523
```

```
print(classification_report(y_test, y_pred))
✓ 0.0s

              precision    recall  f1-score   support

           0       0.97      0.99      0.98       389
           1       0.95      1.00      0.97       457
           2       0.99      0.96      0.98       441
           3       0.95      0.96      0.95       434
           4       0.97      0.97      0.97       407
           5       0.95      0.94      0.95       380
           6       0.96      0.99      0.98       428
           7       0.95      0.97      0.96       421
           8       0.99      0.91      0.95       409
           9       0.94      0.93      0.94       434

    accuracy                           0.96      4200
   macro avg       0.96      0.96      0.96      4200
weighted avg       0.96      0.96      0.96      4200
```

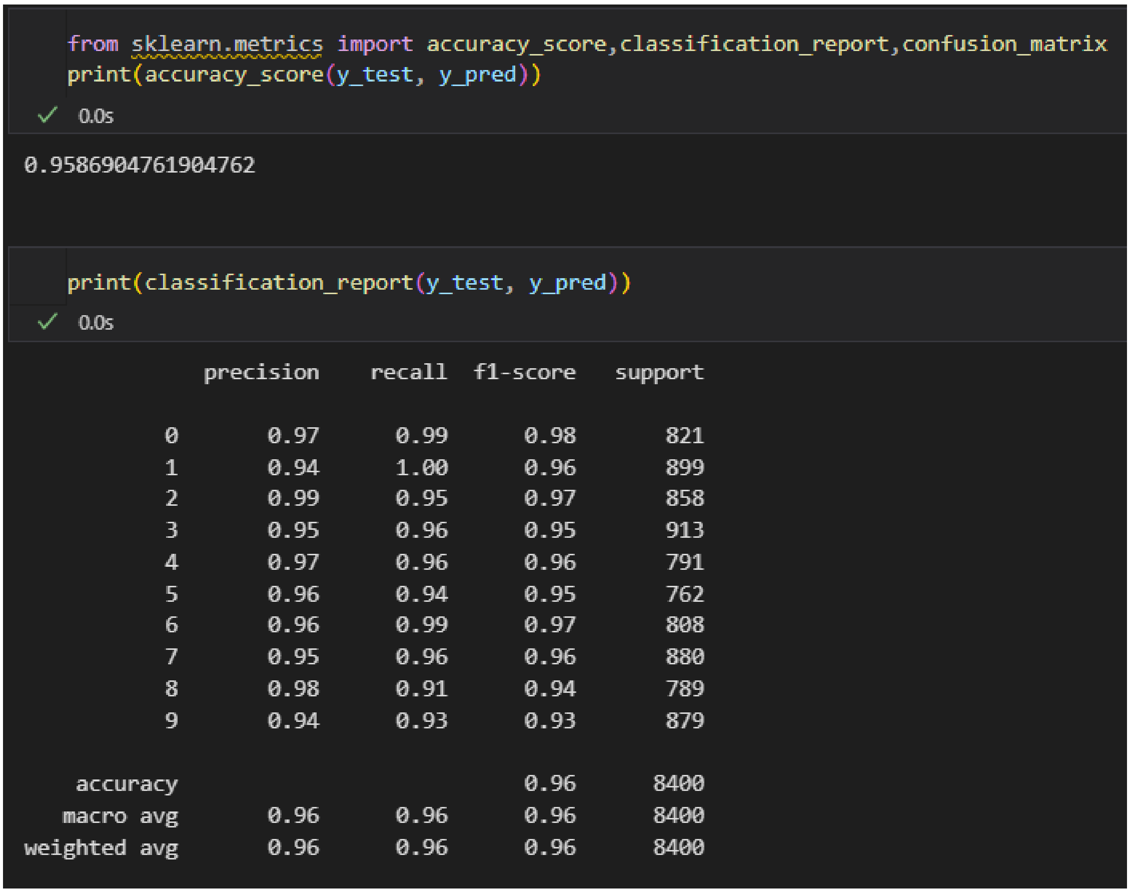
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
✓ 0.0s

[[386   0   0   0   0   0   2   1   0   0]
 [  0 456   0   0   0   0   0   0   0   1]
 [  3   4 425   0   1   0   0   8   0   0]
 [  1   4   2 415   0   5   0   4   2   1]
 [  0   3   0   0 394   0   1   0   0   9]
 [  1   1   0   6   1 358  10   1   0   2]
 [  2   0   0   0   0   1 424   0   1   0]
 [  0   8   1   0   0   0   0 407   0   5]
 [  3   2   0   7   3  10   4   1 372   7]
 [  1   3   0   8   6   3   0   7   1 405]]
```

3.) RATIO OF TRAIN AND TEST DATA – 80:20

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

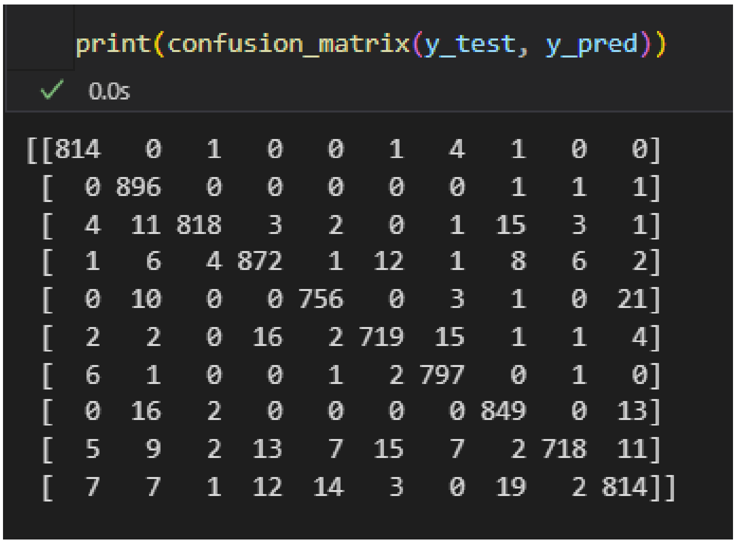
```
0.9586904761904762
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       821
           1       0.94      1.00      0.96       899
           2       0.99      0.95      0.97       858
           3       0.95      0.96      0.95       913
           4       0.97      0.96      0.96       791
           5       0.96      0.94      0.95       762
           6       0.96      0.99      0.97       808
           7       0.95      0.96      0.96       880
           8       0.98      0.91      0.94       789
           9       0.94      0.93      0.93       879

    accuracy                           0.96      8400
   macro avg       0.96      0.96      0.96      8400
weighted avg       0.96      0.96      0.96      8400
```

CONFUSION MATRIX



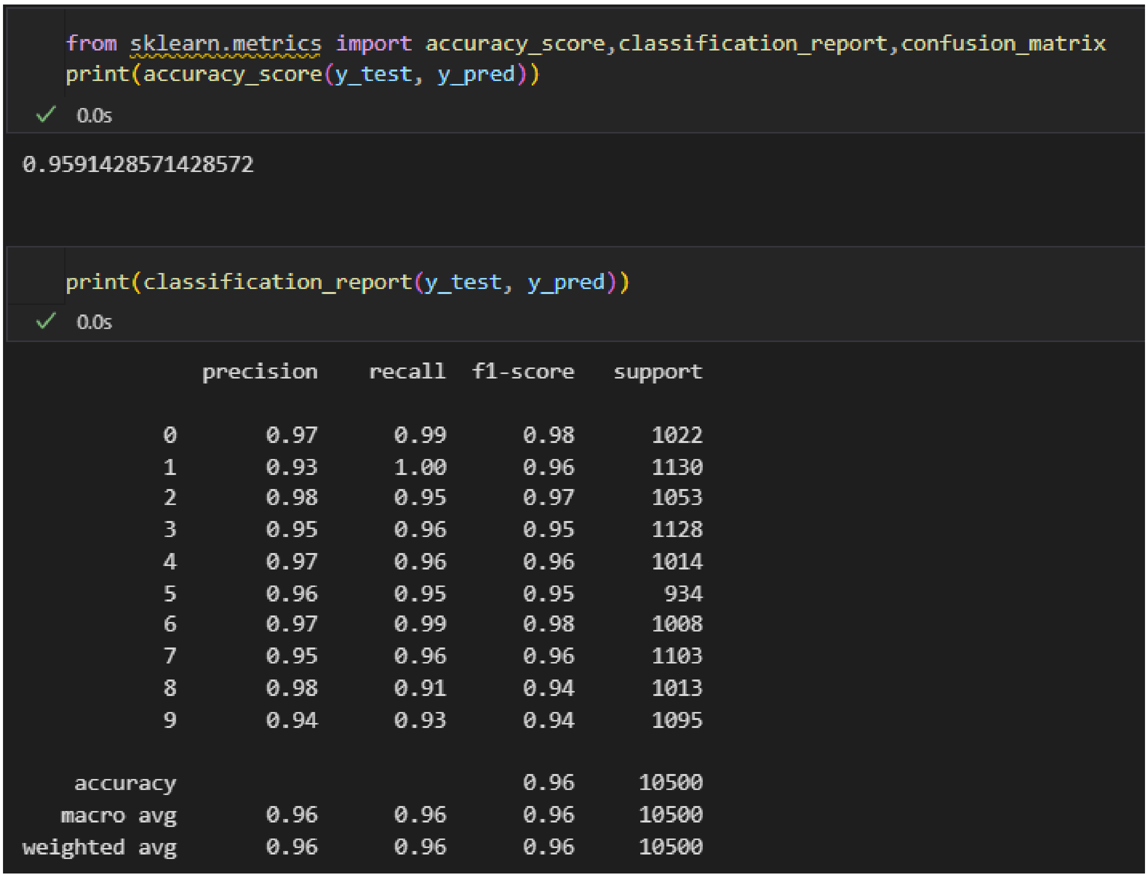
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[814   0   1   0   0   1   4   1   0   0]
 [  0 896   0   0   0   0   0   1   1   1]
 [  4  11 818   3   2   0   1  15   3   1]
 [  1   6   4 872   1  12   1   8   6   2]
 [  0  10   0   0 756   0   3   1   0  21]
 [  2   2   0  16   2 719  15   1   1   4]
 [  6   1   0   0   1   2 797   0   1   0]
 [  0  16   2   0   0   0   0 849   0  13]
 [  5   9   2  13   7  15   7   2 718  11]
 [  7   7   1  12  14   3   0  19   2 814]]
```

4.) RATIO OF TRAIN AND TEST DATA – 75:25

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

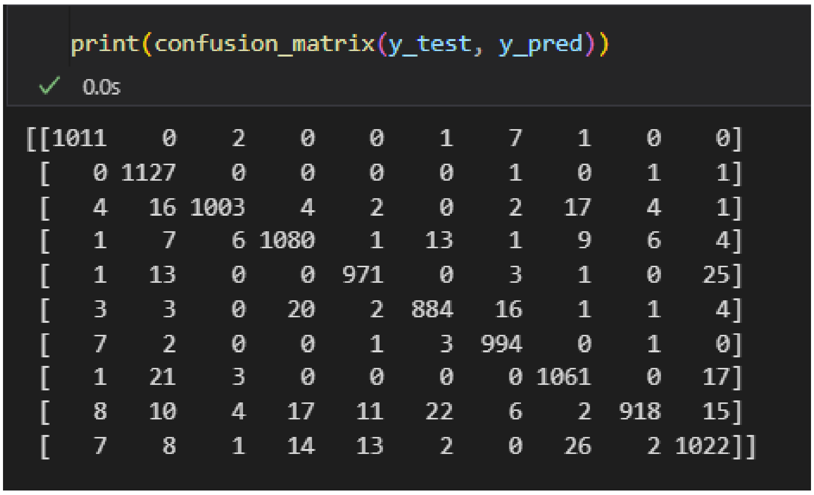
```
0.9591428571428572
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1022
           1       0.93      1.00      0.96      1130
           2       0.98      0.95      0.97      1053
           3       0.95      0.96      0.95      1128
           4       0.97      0.96      0.96      1014
           5       0.96      0.95      0.95       934
           6       0.97      0.99      0.98      1008
           7       0.95      0.96      0.96      1103
           8       0.98      0.91      0.94      1013
           9       0.94      0.93      0.94      1095

    accuracy                           0.96     10500
   macro avg       0.96      0.96      0.96     10500
weighted avg       0.96      0.96      0.96     10500
```

CONFUSION MATRIX



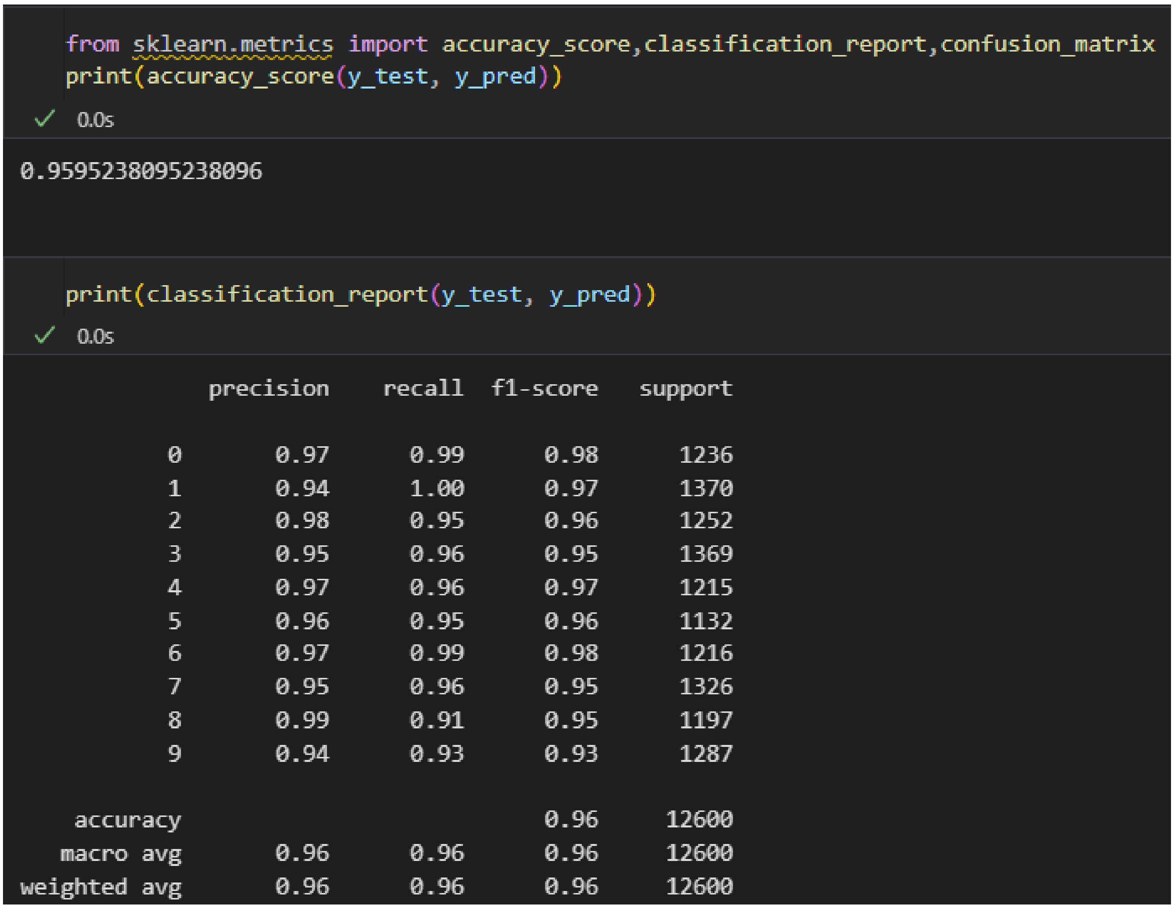
```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1011    0    2    0    0    1    7    1    0    0]
 [   0 1127    0    0    0    0    1    0    1    1]
 [   4   16 1003    4    2    0    2   17    4    1]
 [   1    7    6 1080    1   13    1    9    6    4]
 [   1   13    0    0  971    0    3    1    0   25]
 [   3    3    0   20    2  884   16    1    1    4]
 [   7    2    0    0    1    3  994    0    1    0]
 [   1   21    3    0    0    0    0 1061    0   17]
 [   8   10    4   17   11   22    6    2  918   15]
 [   7    8    1   14   13    2    0   26    2 1022]]
```

5.) RATIO OF TRAIN AND TEST DATA – 70:30

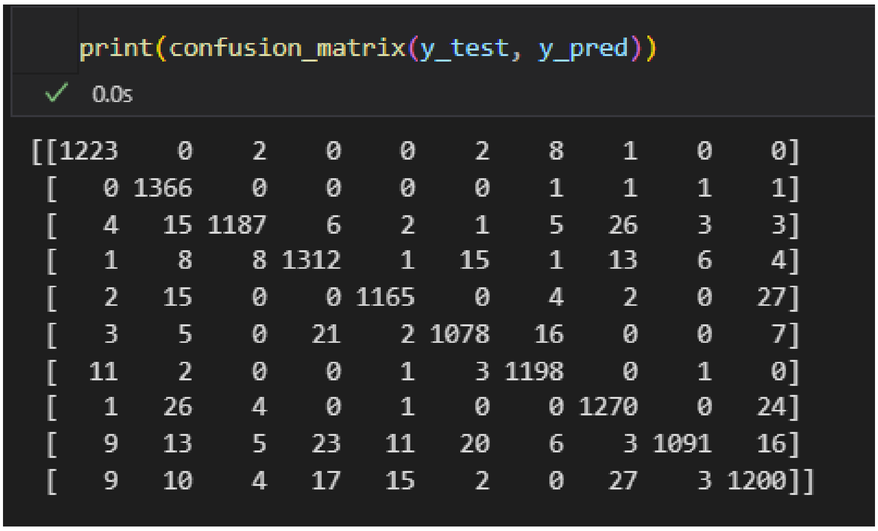
ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
✓ 0.0s

0.9595238095238096
```

```
print(classification_report(y_test, y_pred))
✓ 0.0s

              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1236
           1       0.94      1.00      0.97      1370
           2       0.98      0.95      0.96      1252
           3       0.95      0.96      0.95      1369
           4       0.97      0.96      0.97      1215
           5       0.96      0.95      0.96      1132
           6       0.97      0.99      0.98      1216
           7       0.95      0.96      0.95      1326
           8       0.99      0.91      0.95      1197
           9       0.94      0.93      0.93      1287

    accuracy                           0.96     12600
   macro avg       0.96      0.96      0.96     12600
weighted avg       0.96      0.96      0.96     12600
```

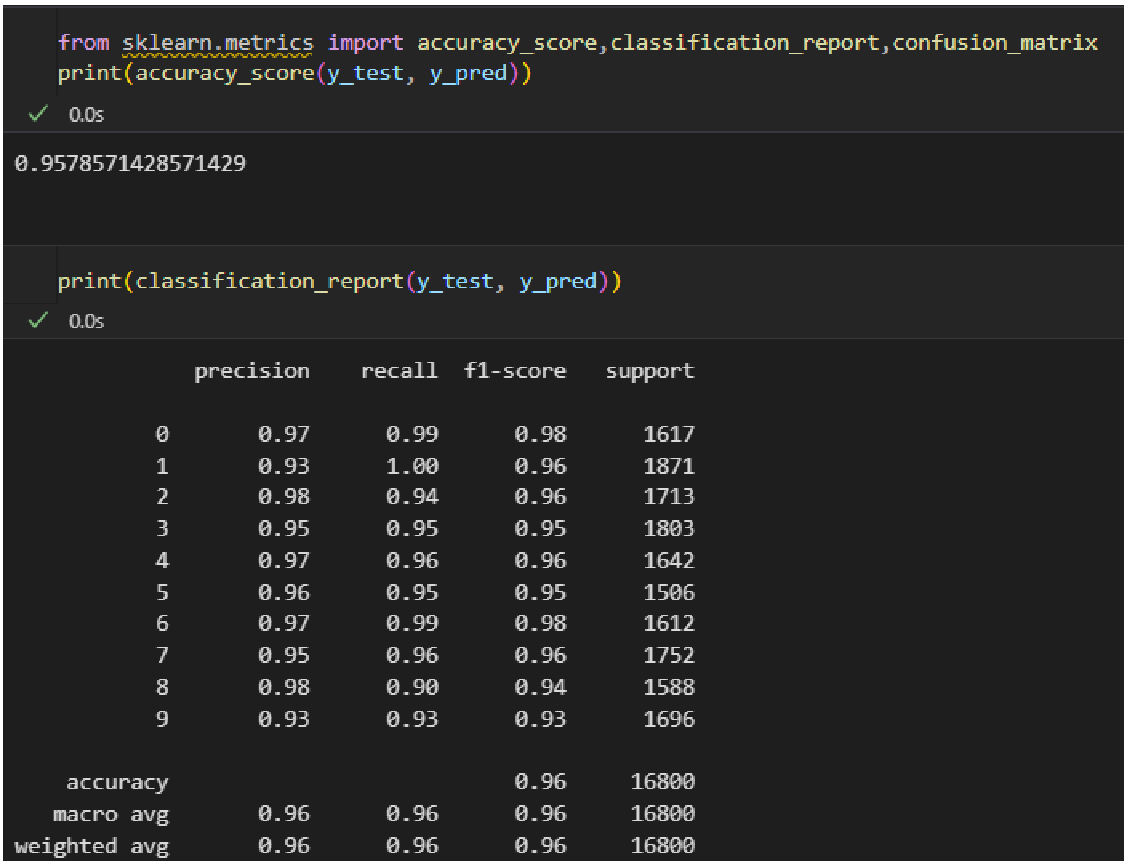
CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
✓ 0.0s

[[1223    0    2    0    0    2    8    1    0    0]
 [   0 1366    0    0    0    0    1    1    1    1]
 [   4   15 1187    6    2    1    5   26    3    3]
 [   1    8    8 1312    1   15    1   13    6    4]
 [   2   15    0    0 1165    0    4    2    0   27]
 [   3    5    0   21    2 1078   16    0    0    7]
 [  11    2    0    0    1    3 1198    0    1    0]
 [   1   26    4    0    1    0    0 1270    0   24]
 [   9   13    5   23   11   20    6    3 1091   16]
 [   9   10    4   17   15    2    0   27    3 1200]]
```

6.) RATIO OF TRAIN AND TEST DATA – 60:40

ACCURACY – 0.96



```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
print(accuracy_score(y_test, y_pred))
```

✓ 0.0s

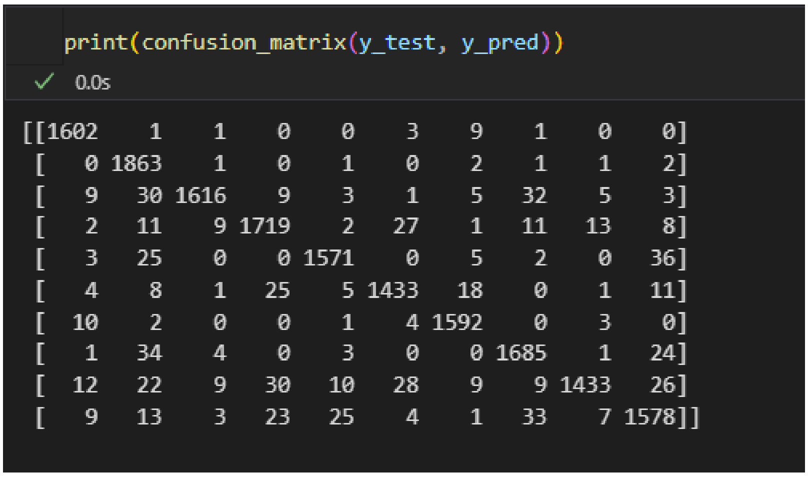
```
0.9578571428571429
```

```
print(classification_report(y_test, y_pred))
```

✓ 0.0s

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      1617
           1       0.93      1.00      0.96      1871
           2       0.98      0.94      0.96      1713
           3       0.95      0.95      0.95      1803
           4       0.97      0.96      0.96      1642
           5       0.96      0.95      0.95      1506
           6       0.97      0.99      0.98      1612
           7       0.95      0.96      0.96      1752
           8       0.98      0.90      0.94      1588
           9       0.93      0.93      0.93      1696

    accuracy                           0.96     16800
   macro avg       0.96      0.96      0.96     16800
weighted avg       0.96      0.96      0.96     16800
```

CONFUSION MATRIX



```
print(confusion_matrix(y_test, y_pred))
```

✓ 0.0s

```
[[1602    1    1    0    0    3    9    1    0    0]
 [   0 1863    1    0    1    0    2    1    1    2]
 [   9   30 1616    9    3    1    5   32    5    3]
 [   2   11    9 1719    2   27    1   11   13    8]
 [   3   25    0    0 1571    0    5    2    0   36]
 [   4    8    1   25    5 1433   18    0    1   11]
 [  10    2    0    0    1    4 1592    0    3    0]
 [   1   34    4    0    3    0    0 1685    1   24]
 [  12   22    9   30   10   28    9    9 1433   26]
 [   9   13    3   23   25    4    1   33    7 1578]]
```

Analysis of Model Performance Dependency on Training-Testing Split and K Value

1. Training-Testing Split Ratio:

The training-testing split ratio plays a crucial role in determining the performance of the KNN model. Here's how it impacts the model:

- **Effect on Model Bias and Variance**:
  - **Higher Training Ratio**: When more data is allocated to training (e.g., 80% or more), the model tends to have lower bias but higher variance. This is because the model learns more from the training data, potentially capturing more complex patterns but becoming more sensitive to noise.
  - **Higher Testing Ratio**: Conversely, with a higher testing ratio (e.g., 30% or more for testing), the model may have higher bias but lower variance. It sees less data during training, potentially leading to oversimplification and underfitting, but it generalizes better to unseen data.
- **Impact on Model Performance**:
  - **Accuracy**: In general, as the training ratio increases (more data for training), the model tends to perform better on the training data itself (higher training accuracy). However, the performance on the test data might decrease if the model overfits to the training data.
  - **Generalization**: A balanced split (e.g., 70:30 or 80:20) often strikes a good balance between training the model adequately and evaluating its generalization ability on unseen data.

2. Choice of K Value:

The choice of K in KNN significantly influences model performance. Key observations include:

- **Bias-Variance Trade-off**:
  - **Small K (e.g., K = 2)**: Leads to low bias but high variance. The model might capture intricate patterns in the training data but can be sensitive to outliers and noise.
  - **Large K (e.g., K = 10)**: Results in higher bias but lower variance. The model tends to generalize better but might miss finer details present in the data.
- **Impact on Decision Boundary**:
  - **Smaller K**: Results in a more complex decision boundary that closely fits the training data, potentially leading to overfitting.
  - **Larger K**: Produces a smoother decision boundary that may underfit the training data but generalizes better to unseen data.
- **Model Performance**:
  - **Accuracy**: The optimal K value for accuracy depends on the dataset and the underlying distribution of data points. Typically, a cross-validation approach or grid search is used to determine the best K.
  - **Confusion Matrix**: Different K values can lead to varying confusion matrix patterns, affecting precision, recall, and F1-score metrics.

Conclusion:

- **Finding the Balance**: Achieving optimal performance involves finding the right balance between training-testing split ratio and K value. It requires consideration of trade-offs between bias and variance, model complexity, and generalization ability.
- **Experimentation and Validation**: It's essential to experiment with different split ratios and K values, evaluate their impact on model performance (via accuracy metrics and confusion matrices), and validate results through cross-validation or hold-out validation to ensure robustness.

Understanding these dependencies allows for informed decisions in model selection and tuning, ensuring the KNN model performs optimally for specific datasets and tasks.