CST8502 - Lab 6
Neural Networks

Student Name: Alvin Litani Liauw
Student Number: 041118874

For every step, include screenshot of the code and the results in this document (screenshot from colab/jupyter notebook). Also, in your words, explain your code and results. If there is no explanation, no marks will be given. No need to write long paragraphs, but one or 2 lines per step.

1

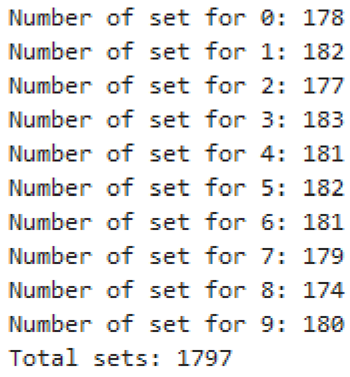
```
# load the dataset
digits = load_digits()
```

2

```
total = 0

# show how many sets in the target label
for name, count in zip (digits.target_names, np.bincount(digits.target)):
   total += count
   print (f'Number of set for {name}: {count}')

print (f'Total sets: {total}')
```



```
Number of set for 0: 178
Number of set for 1: 182
Number of set for 2: 177
Number of set for 3: 183
Number of set for 4: 181
Number of set for 5: 182
Number of set for 6: 181
Number of set for 7: 179
Number of set for 8: 174
Number of set for 9: 180
Total sets: 1797
```

3

```
# load feature matrix into x and target label into y
x = digits.data
y = digits.target

# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=74)
```

4

```
# Create MLP classifier
mlp = MLPClassifier(hidden_layer_sizes=(15,), max_iter=550, random_state=74)
```

5

```
# Train the classifier
mlp.fit(x_train, y_train)
```

6

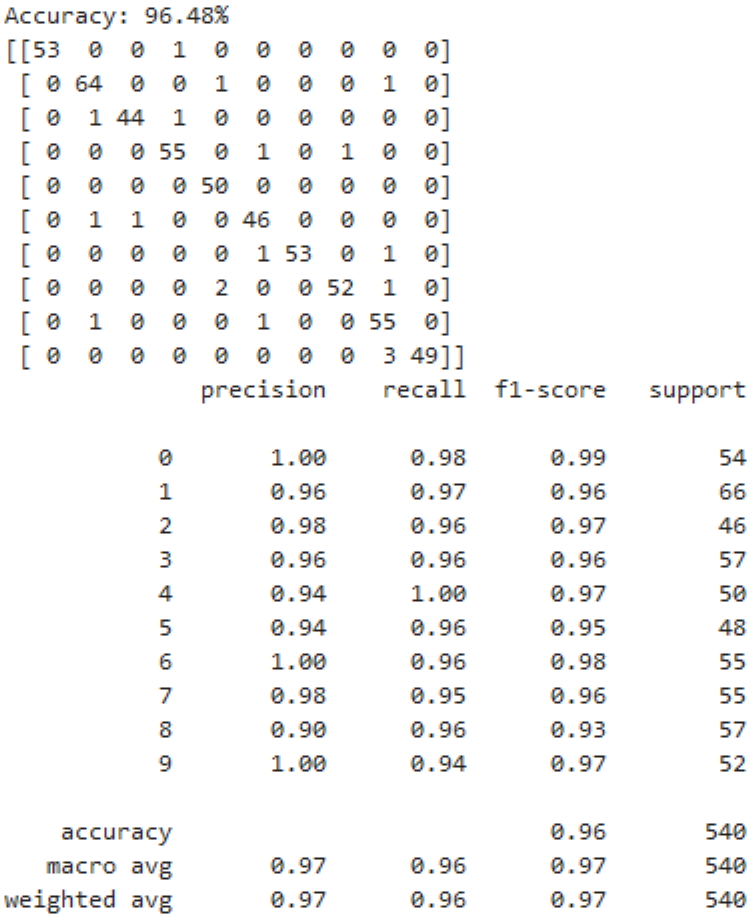
```
# Make predictions on the test set
y_pred = mlp.predict(x_test)
```

7

```
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Show confusion matrix
ConfusionMatrix = confusion_matrix(y_test,y_pred)
print(ConfusionMatrix)

# Show classification report
ClassificationReport = classification_report(y_test, y_pred)
print(ClassificationReport)
```



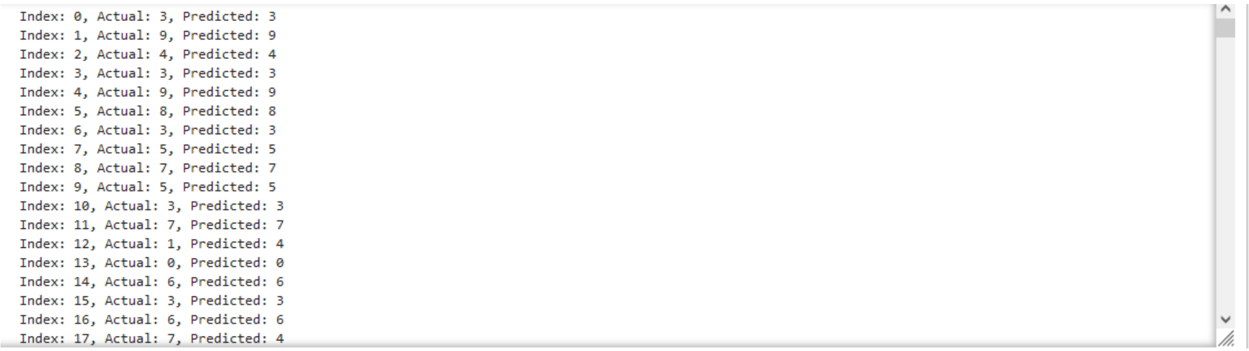
```
Accuracy: 96.48%
[[53  0  0  1  0  0  0  0  0  0]
 [ 0 64  0  0  1  0  0  0  1  0]
 [ 0  1 44  1  0  0  0  0  0  0]
 [ 0  0  0 55  0  1  0  1  0  0]
 [ 0  0  0  0 50  0  0  0  0  0]
 [ 0  1  1  0  0 46  0  0  0  0]
 [ 0  0  0  0  0  1 53  0  1  0]
 [ 0  0  0  0  2  0  0 52  1  0]
 [ 0  1  0  0  0  1  0  0 55  0]
 [ 0  0  0  0  0  0  0  0  3 49]]
              precision    recall  f1-score   support

           0       1.00      0.98      0.99        54
           1       0.96      0.97      0.96        66
           2       0.98      0.96      0.97        46
           3       0.96      0.96      0.96        57
           4       0.94      1.00      0.97        50
           5       0.94      0.96      0.95        48
           6       1.00      0.96      0.98        55
           7       0.98      0.95      0.96        55
           8       0.90      0.96      0.93        57
           9       1.00      0.94      0.97        52

    accuracy                           0.96       540
   macro avg       0.97      0.96      0.97       540
weighted avg       0.97      0.96      0.97       540
```

8

```
# Print the actual and predicted result together with the index number
for index, (actual, predicted) in enumerate(zip(y_test, y_pred)):
    print(f"Index: {index}, Actual: {actual}, Predicted: {predicted}")
```



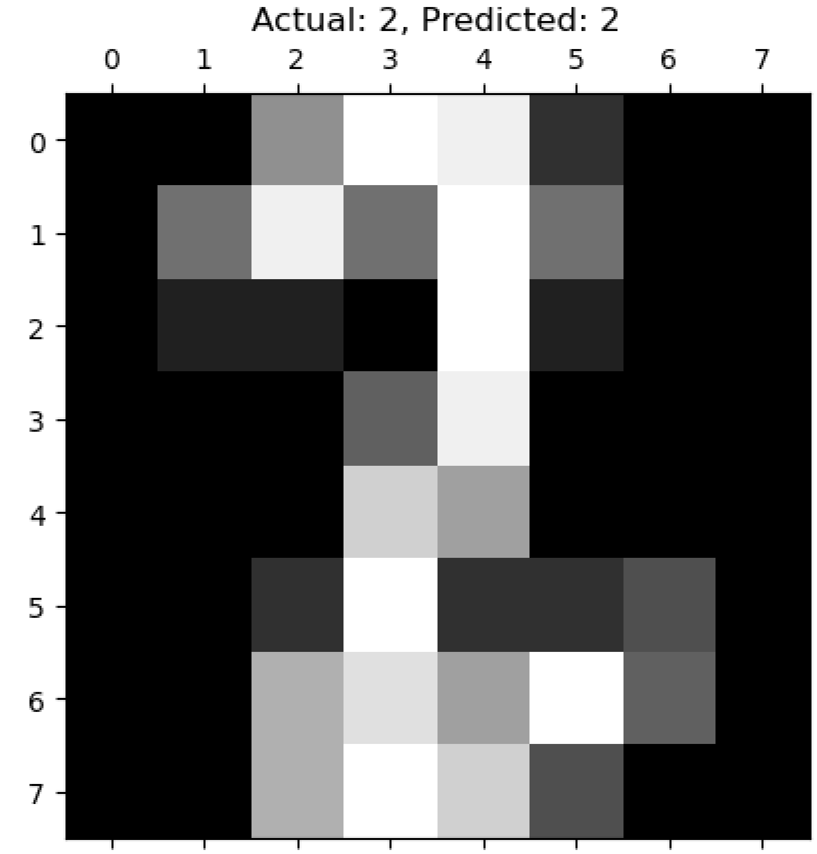
```
Index: 0, Actual: 3, Predicted: 3
Index: 1, Actual: 9, Predicted: 9
Index: 2, Actual: 4, Predicted: 4
Index: 3, Actual: 3, Predicted: 3
Index: 4, Actual: 9, Predicted: 9
Index: 5, Actual: 8, Predicted: 8
Index: 6, Actual: 3, Predicted: 3
Index: 7, Actual: 5, Predicted: 5
Index: 8, Actual: 7, Predicted: 7
Index: 9, Actual: 5, Predicted: 5
Index: 10, Actual: 3, Predicted: 3
Index: 11, Actual: 7, Predicted: 7
Index: 12, Actual: 1, Predicted: 4
Index: 13, Actual: 0, Predicted: 0
Index: 14, Actual: 6, Predicted: 6
Index: 15, Actual: 3, Predicted: 3
Index: 16, Actual: 6, Predicted: 6
Index: 17, Actual: 7, Predicted: 4
```

9

```
# image index based on student number
image_index = 74

# Reshape the image data (8x8 matrix)
image_data = x_test[image_index].reshape(8, 8)

# Plot the image for the 74th image
plt.matshow(image_data)
plt.title(f"Actual: {y_test[image_index]}, Predicted: {y_pred[image_index]}")
plt.show()
```


Actual: 2, Predicted: 2