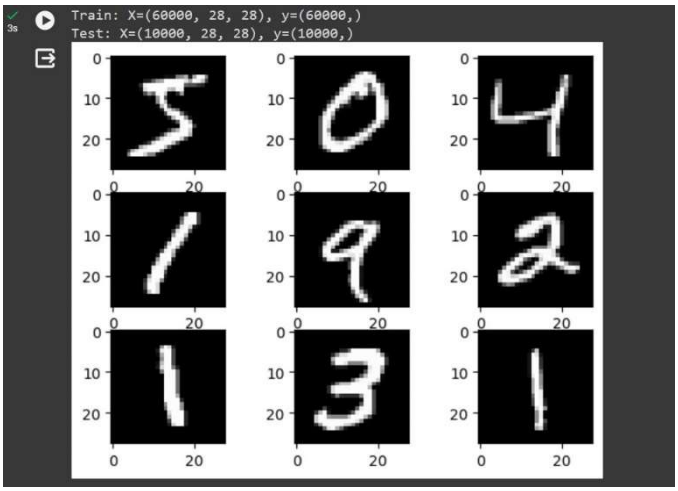
- The assignment involves the task of identifying/recognizing handwritten digits as given in MNIST Dataset. The dataset is located at in inbuilt-tensorflow.keras.
- We implement various machine learning techniques eventually and see how they perform in contrast with each other.
- For this assignment, we select a csv dataset. The csv was taken from here: [Digit Recognizer | Kaggle]



- We perform a train-test split in the following options: 80:20, 50:50, 99:1,, 20:80. To achieve this, we use random.choice() from random module with test_size values incorporated in a list. In other words, we pick test_size values randomly and see how our model performs.
- We then define a function **fit_predict** wherein, we fit the model and evaluate its performance against various algorithms.
- The results are then compared.

  For Decision Tree classifier, accuracy score is **83.57**

  For Random Forest, it is **95.71**

  For Logistic Regression, it is **90.57**

  Likewise it is compared.

```
fit_predict(lr, X_train, y_train, X_test, y_test)
```

```
Accuracy Score is: 0.9057619047619048

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.97      0.95      2052
           1       0.96      0.96      0.96      2330
           2       0.89      0.89      0.89      2096
           3       0.88      0.87      0.87      2222
           4       0.92      0.91      0.91      2053
           5       0.86      0.84      0.85      1833
           6       0.93      0.94      0.93      2079
           7       0.93      0.92      0.92      2191
           8       0.87      0.86      0.86      2062
           9       0.87      0.89      0.88      2082

    accuracy                           0.91     21000
   macro avg       0.90      0.90      0.90     21000
weighted avg       0.91      0.91      0.91     21000


Confusion Matrix:
[[1982    0    9    2    4   21   15    5    8    6]
 [   0 2245   17   11    1    5    3    8   34    6]
 [   8   19 1863   41   26   17   31   29   53    9]
 [  12    6   68 1928    2   93   12   20   55   26]
 [  10    8   11    6 1872    3   25    4   19   95]
 [  30    6   17   90   22 1535   42    6   63   22]
 [  29    9   30    6   21   26 1949    2    6    1]
 [   6    5   27   15   15    2    0 2018   12   91]
 [  17   40   35   66   13   66   18    6 1774   27]
 [  13    4    7   27   63   13    0   77   23 1855]]
```