Objective

To develop an image recognition algorithm using the logistic regression based on the gradient descent, trained on the MNIST dataset to correctly classify handwritten images as ID or non-ID, where ID corresponds to the last digit of my Student ID, which is 0.

My Solution

1. Preprocessing:

Before training the model on the MNIST dataset, it was required to preprocess the dataset. This included the following steps:

- Rescaling the train and test datasets such that each column in the dataset corresponds to an image. Thus, each image in the dataset was flattened to a column vector. The dimensions of the train set thus obtained = 784 x 60,000 and the test set dimensions = 784 x 10,000. Additionally, the train and test datasets were normalized by dividing each pixel value in both the sets by 255.

- The labels for the train and test datasets were rescaled into a row vector. The dimensions of the train label = 1 x 60,000 and test set label = 1 x 10,000. Additionally, the labels were modified such that the label of the image = 1, if the image shows the ID (i.e. 0) and label of the image = 0 if the image does not show the ID (i.e. not 0). This was achieved by making masks train labels and test labels corresponding to image of ID and using these masks to modify the labels.

2. Learning:

To learn the model for classification of ID and non-ID, I have written the class logisticRegression. The constructor method of the class initializes the training data, test data and validation data (splitting the original training data into training + validation dataset), the learning rate (alpha), the weights and the bias. The weights and the bias are initialized randomly values between 0 and 1. The learning rate alpha, can contain a list of multiple alpha values, and the best alpha value is decided by testing the trained model (with a particular alpha) on the validation set and selecting the alpha value which results in minimum cost on the validation set.

The method train_model() implements the Gradient Descent algorithm for performing the logistic regression task. If multiple values of alpha are provided, then it determines the best alpha value by evaluating the cost of the trained models on the validation set and selects the value for alpha which results in minimum cost. This alpha value is used for training the model.

For each instance of training the model (whether for validation or final training), the weights and bias are initialized randomly. The number of iterations (epochs) during the validation and final training is set to 6000. This number was determined by experimenting with different number of iterations including: 1000, 2000, 3000, 5000, 6000, 7000.

The cost obtained after training the model for different values of alpha is displayed in the output and are shown in fig. 1.

```
Training Mode
Cross Validation for alpha =  5
C:\Users\shrey\AppData\Roaming\Python\Python36\site-packages\ipykernel_launcher.py:33: RuntimeWarning: divide by zero encounter
ed in log
C:\Users\shrey\AppData\Roaming\Python\Python36\site-packages\ipykernel_launcher.py:34: RuntimeWarning: invalid value encountere
d in multiply
Cost for alpha =  5 =  0.025280193789209208
Cross Validation for alpha =  2
Cost for alpha =  2 =  0.024563777874491228
Cross Validation for alpha =  1
Cost for alpha =  1 =  0.025258516704746853
Cross Validation for alpha =  0.5
Cost for alpha =  0.5 =  0.026332561386352385
The best learning rate =  2
Training the model using alpha =  2
C:\Users\shrey\AppData\Roaming\Python\Python36\site-packages\ipykernel_launcher.py:75: RuntimeWarning: divide by zero encounter
ed in log
C:\Users\shrey\AppData\Roaming\Python\Python36\site-packages\ipykernel_launcher.py:76: RuntimeWarning: invalid value encountere
d in multiply
The training error =  0.022221658814956618
The Training accuracy =  0.9934375
```

Figure 1. Cost for various values of alpha after performing the gradient descent for 6000 iterations

3. Evaluation:

During the final training of the model, the cost is evaluated during every iteration of the gradient descent algorithm and the learning curve was plotted, shown in fig. 2.

During my last execution of the algorithm, the model achieved a cost = 0.02222169 on the training dataset, and an accuracy of 99.34375% (by rounding the prediction values).

Additionally, the model achieved a cost = 0.02504383 on the Test set and an accuracy = 99.18% (by rounding the prediction values). The output corresponding to training and test cost and accuracy is shown in fig. 3.

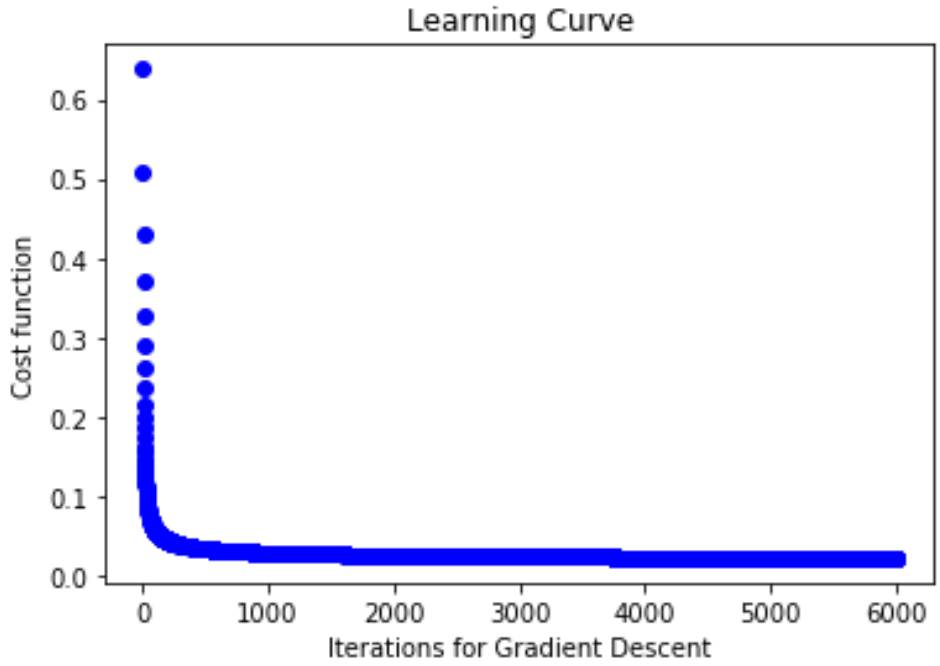

Figure 2. Plot of Cost function vs the number of iterations

```
The training error =  0.022221658814956618
The Training accuracy =  0.9934375

Testing Mode
The testing error  =  0.025043834704278557
Testing accuracy =  0.9918
```

Figure 3. Cost and accuracy obtained on training and test dataset

Additionally, I tested the model on a random image from the test set containing the ID, the ground truth value = 1 and the predicted value = 0.999047. And on the image from the test set not containing the ID, the ground truth = 0 and the predicted value = 5.248 e-08.

Thus, we can see that the model has been trained well and the performance on the test set is very close to the performance on the training set.