# Analysis Report

Compared why Linear regression is not suitable for Classification against Logistic Regression using both Mnist and Kaggle Dataset:

**MNIST Analysis**

The MNIST database of handwritten digits has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

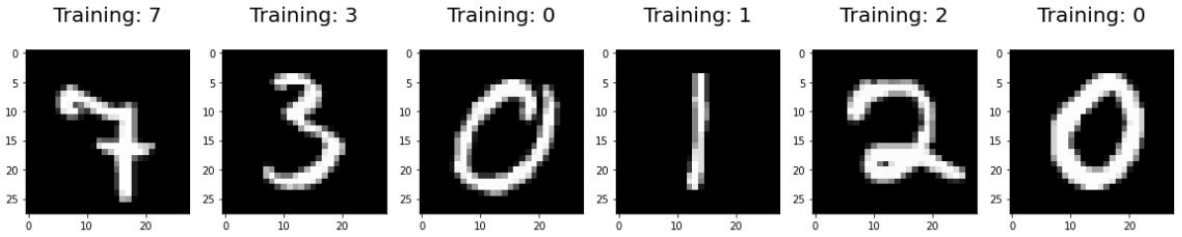
## Logistic Regression Results :

Executed Logistic Regression on MNIST dataset using different classes.

Using different label or numbers  classes like  ( 0,4) , (1,5),(2,6), (3,7),(4,8)  and plotted the image and its corresponding labels.

**Showing Training Digits and Labels**

```
[142…
plt.figure(figsize=(20,4))
for index, (image, label) in enumerate(zip(train_img[0:6], train_lbl[0:6])):
    plt.subplot(1, 6, index + 1)
    plt.imshow(np.reshape(image, (28,28)), cmap=plt.cm.gray)
    plt.title('Training: %s\n' % label, fontsize = 20)
```




Run Logistic Regression on the dataset and captured the weights/accuracy and Std deviation as given below:

**intercept ----->: [0.]**

**Weight ------>: [ 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.**

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.

0. 0. -0. -0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. -0. -0. -0. 0. -0. 0. -0. -0.
-0. 0. 0. 0. 0. -0. -0.01 0. 0.01 0. 0. 0.
0. -0. 0. 0. 0. 0. 0. 0. 0. 0. -0. -0.
-0. 0. 0. 0. 0. 0. 0. 0. -0. -0. 0. 0.
0. 0.01 0. -0. 0. 0. 0. 0. 0. 0. 0. -0.
0. -0. -0. -0. -0. -0. -0. 0. 0. -0. 0.01 -0.01
-0.01 -0. -0. 0. -0.01 0. 0. 0. 0.01 0. 0. 0.
0. 0. 0. 0. -0. -0. -0.01 -0. 0. 0. 0. -0.01
-0. -0.01 -0.01 0. -0. -0. 0. 0.01 0.01 -0. -0. 0.
0.01 0. -0. -0. 0. 0. 0. 0. 0. 0. 0. 0.
-0. 0. 0. 0. -0. -0.01 -0. 0. 0. -0.01 -0.01 0.
0. -0. 0.01 0.01 0.01 0. 0. -0. 0. 0. 0. 0.
0. 0. 0. 0. -0. -0. -0.01 0.01 -0. -0. -0. -0.
-0. -0. -0. -0. 0. -0. -0. 0.01 0.01 0. 0. 0.
0. 0. -0. 0. -0. 0. 0. -0. -0. -0. -0. 0.01
-0. 0. -0.01 -0.01 -0. 0.01 0. -0.01 -0.01 0. -0. 0.
0. 0. 0. 0. 0. 0. -0. -0. -0. 0. 0. -0.
-0. 0. 0. 0. -0.01 -0.01 -0.01 -0.01 -0.01 -0. -0.01 -0.
-0.01 -0. -0.01 -0.01 0. 0. 0. 0. 0. 0. 0. 0.
-0. 0. 0. -0. -0.01 0.01 0. -0.01 -0. -0. -0.01 -0.01
-0.01 -0.01 0. -0.01 0. 0.01 -0.01 -0.01 -0. 0. 0. 0.
0. 0. 0. 0. 0. 0. -0. -0. 0. -0. 0. 0.
-0. -0. -0. 0. 0. 0. -0. -0. -0.01 0.01 -0.01 -0.01
-0. 0. 0. 0. 0. 0. -0. 0. 0. 0. -0. 0.
0. -0.01 0.01 0.01 0. 0. 0.01 -0. 0.01 0.01 0. -0.
0. -0. -0.01 -0.01 0. -0. 0. 0. 0. 0. 0. 0.
0.01 -0. -0. 0.01 0.01 -0. 0.01 0.01 0. -0. 0.02 0.01
0.02 0.01 -0. 0.01 0. -0.01 -0.01 -0. -0. -0. 0. 0.
0. 0. 0. 0. -0. -0. -0. 0.01 0.01 0.01 0. 0.01
-0. -0. 0.02 0.01 0.01 0. 0. -0. -0. -0.01 -0. -0.
-0. -0. 0. 0. 0. -0. 0. 0. -0. -0. -0. 0.
0. 0.01 -0.01 -0. 0. 0.01 0.01 0.02 0.01 -0. 0.01 -0.
0. 0. 0. -0.01 -0. 0. 0. -0. 0. 0. 0. 0.
-0. 0. 0. 0.01 -0.01 -0.01 -0.01 -0.01 0.01 0.01 0.01 0.01
0. 0. 0.01 -0.01 0.01 0.01 0. -0. 0. 0. 0. -0.01
0. 0. 0. 0. -0. -0. 0. 0.01 -0.01 0. -0. -0.
0.01 0.01 0.01 0.01 0. 0. 0.01 0. 0.01 0.01 -0. -0.
0. -0. 0. 0. 0. -0. -0. 0.01 -0. -0.01 -0. -0.
-0. -0.01 -0. -0. -0.01 -0.01 0. 0.01 0.01 0.01 0. 0.
0.01 0. -0. -0. -0. -0. 0. 0. 0. -0. -0. 0.
-0. -0.01 -0. -0. -0.01 -0.01 -0.01 -0.01 -0.01 -0. 0.01 0.01
0. 0. -0.01 -0. 0.01 0.01 0.01 0. -0. -0. 0. 0.
0. 0. -0. 0. 0. -0. -0.01 -0. -0. -0. -0. -0.
-0.02 -0.01 -0.01 -0. -0. 0. -0. 0. 0. 0.01 0.01 -0.

-0. -0. 0. 0. 0. 0. 0. -0. -0. -0. -0.01 -0.01
-0.01 -0.01 -0. -0.01 -0.01 -0.01 -0.01 -0. -0. -0. -0. 0.
0. 0.01 0.01 -0. -0. 0. 0. 0. 0. 0. 0. 0.
-0. -0. -0. -0. -0.01 -0. -0. 0. -0. -0. -0.01 -0.
0. -0. 0. 0. -0. -0. -0. -0. 0. 0. 0. 0.
0. 0. 0. 0. -0. 0. -0. -0. -0. 0. -0. 0.01
0. 0. -0.01 -0.01 0. -0. 0. 0. 0. 0. 0. 0.
-0. 0. 0. 0. 0. 0. 0. 0. 0. 0. -0. -0.
0. 0.01 0.01 0. 0. 0.01 0. 0. 0. 0.01 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. ]

**Accuracy -----> 0.9933605720122574**

**Shape ['1' '5' '5' ... '1' '1' '1']**

**intercept ----->: [0.]**

**Weight ------>: [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.**

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. -0. -0. -0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. -0. -0. -0. -0. -0.
-0. -0.01 -0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. -0. -0. -0. 0.
-0. -0. -0. -0. 0. -0. 0.01 0.02 -0. -0. -0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
-0. -0.01 -0. 0. -0.01 -0. -0. -0. 0. -0. -0. 0.
0.01 0.01 0.01 0. 0. 0. 0. 0. 0. 0. 0. -0.
-0. -0. -0. 0. 0. -0. -0.01 -0. -0.01 -0. -0.01 -0.
0. -0.01 -0. 0.02 0.02 -0.01 -0.01 -0.01 -0. 0. 0. 0.
0. 0. 0. -0. -0. -0. -0. -0. 0. 0. -0. 0.
-0. 0. -0. -0. 0. -0. -0.01 0.01 0.01 0.01 0. 0.01
0.01 0.01 0.01 0. 0. 0. -0. -0. -0. -0. -0. -0.
-0. 0. 0. 0.01 -0. 0.01 -0.01 0. -0. 0.01 0.01 0.01
0.01 0.01 0.01 0.01 0.01 0. 0. 0. 0. 0. -0. -0.
0. 0. -0. -0. 0. 0.01 0. -0. -0. -0. -0.01 -0.
0. 0. 0. 0. 0. 0.01 0. 0.02 0.02 0.01 0. 0.
0. 0. 0. 0. 0. 0. -0. -0. 0.01 0.01 -0. -0.01
-0. -0.01 -0. -0. -0. -0.01 0. 0. 0.01 0.01 0.01 0.02
0.02 0.01 0. 0. 0. 0. 0. 0. 0. 0. -0. 0.
-0. 0. -0. 0. 0. 0.01 -0.01 -0. -0. -0.01 -0.01 -0.
0. 0. 0.01 0.02 0.01 0.01 0.01 0. 0. 0. 0. 0.

```
 0.    0.    0.   -0.   -0.01 -0.   -0.    0.01 -0.    0.   -0.   -0.01
-0.   -0.01 -0.01  0.    0.    0.    0.01  0.01  0.01  0.01  0.    0.
 0.    0.    0.    0.    0.   -0.   -0.   -0.    0.    0.01  0.01 -0.
 0.    0.01 -0.01 -0.   -0.01 -0.01 -0.01  0.01 -0.   -0.01  0.    0.
 0.    0.    0.   -0.   -0.    0.    0.    0.    0.   -0.   -0.01  0.
 0.01  0.01  0.01  0.01  0.   -0.   -0.01 -0.02 -0.01 -0.02  0.    0.
 0.    0.    0.01  0.01  0.    0.    0.    0.    0.    0.   -0.    0.
 0.   -0.01  0.    0.01  0.01  0.01  0.01  0.   -0.    0.   -0.01 -0.02
-0.01 -0.01  0.01  0.02  0.01  0.    0.    0.    0.    0.    0.    0.
 0.    0.   -0.    0.    0.    0.    0.01  0.01 -0.    0.    0.01  0.01
-0.   -0.01 -0.02 -0.01 -0.01  0.01  0.02  0.02  0.01  0.    0.    0.
 0.    0.    0.    0.    0.    0.   -0.    0.    0.    0.    0.01  0.01
-0.    0.01 -0.   -0.01 -0.01 -0.01 -0.01  0.    0.01  0.02  0.02  0.01
 0.01  0.    0.    0.    0.    0.   -0.    0.    0.    0.    0.    0.
 0.01  0.01  0.01 -0.   -0.   -0.01 -0.   -0.01 -0.01 -0.01  0.01 -0.
 0.01  0.02  0.01  0.    0.    0.01 -0.    0.    0.    0.   -0.    0.
 0.    0.    0.    0.    0.01  0.02  0.02  0.01  0.01  0.   -0.   -0.01
-0.    0.    0.01 -0.    0.01  0.01  0.   -0.    0.    0.01  0.01  0.01
 0.    0.   -0.    0.    0.    0.    0.    0.    0.    0.01  0.01  0.
 0.    0.    0.    0.    0.01  0.   -0.   -0.01 -0.    0.01  0.   -0.
 0.01  0.01  0.    0.    0.    0.   -0.    0.    0.    0.    0.   -0.
-0.    0.01 -0.    0.01 -0.   -0.01 -0.   -0.   -0.01 -0.01 -0.    0.01
-0.   -0.   -0.01 -0.   -0.    0.   -0.   -0.   -0.    0.   -0.    0.
 0.    0.    0.   -0.01 -0.    0.   -0.   -0.   -0.    0.01 -0.    0.01
 0.01  0.   -0.    0.   -0.01  0.01 -0.   -0.   -0.   -0.   -0.   -0.
-0.   -0.   -0.    0.    0.    0.    0.   -0.   -0.   -0.01  0.    0.01
 0.01  0.01  0.01  0.01  0.01  0.01  0.    0.01  0.01  0.   -0.   -0.01
-0.01 -0.01  0.   -0.   -0.   -0.    0.    0.    0.    0.    0.    0.
 0.   -0.    0.    0.   -0.   -0.01  0.01  0.01  0.    0.01 -0.    0.
-0.    0.    0.01  0.   -0.01 -0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.   -0.    0.   -0.01 -0.01 -0.    0.01  0.01
 0.01  0.01  0.    0.    0.01  0.01  0.01  0.   -0.   -0.   -0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.01  0.01  0.    0.   -0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.  ]
```

**Accuracy -----> 0.995**

**Shape ['2' '6' '2' ... '6' '2' '6']**

**intercept ----->: [-0.]**

**Weight ------>: [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.**

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. -0.
-0. -0. 0. 0. 0. 0.02 -0. -0.02 -0.01 0.01 0.01 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. -0. 0. 0. 0.02 0.03 -0.01 -0. 0.01 0.01 0.01 0.01
0. 0.01 0.02 0. -0. 0.01 0.02 0. -0.01 -0. 0. 0.
0. 0. -0. -0. 0.01 0.01 0.01 0.01 0.02 0. 0.02 0.01
-0.03 -0.01 0.02 -0.02 -0.02 0. 0.01 -0.01 0.03 0.02 0.03 0.02
0.01 -0. 0. 0. 0. 0. -0. -0. 0.01 0.02 -0.02 -0.01
-0. -0.02 -0.01 -0. -0.02 0. -0.01 -0.01 0.01 0.01 0.01 0.
0.03 0.02 0.02 0.03 0.02 0.01 0.01 0.01 0. 0. 0. -0.
0.02 0. -0.01 0.01 -0.02 -0.01 -0. 0.02 0.01 -0.02 0.01 0.01
-0.04 -0. -0. 0.01 0.01 0. 0.01 0.03 0.04 0.02 0.02 0.01
0. 0. 0. -0. 0.01 0.01 -0.01 0.02 0.01 0. -0.01 -0.01
-0.02 -0.01 -0.04 0.01 -0. 0.01 0. 0.01 0.01 0.01 -0.01 0.02
0.01 -0. 0. 0. 0. 0. 0. -0. -0.02 -0.02 -0.03 0.01
-0.01 0.01 -0.01 -0.01 0.02 -0. 0.01 0.01 0.01 -0. -0.01 -0.01
-0.01 0. -0.02 -0.01 -0.02 -0.02 0. 0. 0. 0. -0. -0.
-0.01 -0. -0.01 -0.03 0.01 -0.01 0.01 -0.02 0.02 0.01 0. -0.02
-0.02 -0.01 -0.03 -0.03 -0.01 0.01 -0.03 -0.04 -0.02 -0. 0. 0.
0. 0. -0. -0. 0.01 -0.02 -0. 0.01 -0.01 -0.02 -0.01 -0.
0.01 -0.02 0. -0.04 -0. 0. 0.01 -0.02 -0. 0.02 -0.04 -0.01
0.01 -0. 0. 0. 0. 0. -0. -0. 0.01 -0.01 0. 0.01
0. -0.01 0. 0. -0.03 -0. -0.04 -0. -0.01 -0.02 0.02 -0.01
-0. -0.01 -0.01 0.03 0.02 -0. -0. 0. 0. 0. 0. -0.
0.02 0.02 -0. -0.02 0.01 0.01 0.02 0.01 0.03 0.03 0.01 -0.02
0.01 -0.05 -0.02 -0.01 -0.02 -0.02 0.02 0.04 -0.01 -0.01 -0.01 0.
0. 0. 0. -0. 0.01 -0.03 0.02 0.03 0.04 0. 0.01 -0.02
-0.01 0.03 0.02 -0.01 0.02 0.01 0.03 -0.02 0.01 -0. 0.02 0.03
0. -0.02 -0.02 0. 0. 0. 0. -0. -0.05 -0. 0.01 0.04
0.02 0.02 0.03 -0. 0.03 -0.03 0.01 0.01 -0. -0.01 -0.01 0.01
0.01 0.02 0.02 0.02 -0. -0.02 -0.01 0. 0. 0. 0. -0.
-0.02 0.03 0.03 -0. -0. 0.01 -0.03 0. 0.01 -0.01 -0.01 0.
-0. -0.01 -0.01 -0. -0.02 0.01 0. 0.02 -0.01 0.01 -0. 0.
0. 0. -0. -0. -0.03 -0.01 0.01 -0.02 0. 0. 0.01 0.01
-0.01 0.01 0.02 -0.02 -0. 0.01 0. 0.04 0.03 0. -0.01 -0.01
-0.02 0.01 -0.03 -0. 0. 0. -0. 0. -0.03 -0.02 0.03 0.02
0.02 0. 0. 0. -0.01 -0.01 -0.02 0.01 -0.01 -0.01 0. -0.02
-0. -0.03 0.02 0. 0.01 -0.04 -0.02 -0. 0. 0. -0. 0.
-0.04 0.01 -0.02 -0.03 0. -0. 0. 0.02 0.01 -0.01 0. -0.01
-0.02 0.01 -0. 0.01 -0. 0.02 0.01 -0.02 0. -0.04 -0.01 -0.
0. 0. -0. 0.01 -0.02 -0.01 0.01 -0.02 0. -0. 0.01 -0.03
-0.01 0.04 0.01 0.04 -0.01 0.01 0.01 -0.03 0.01 -0. -0.02 -0.03

-0.01 -0.02 -0.01 -0.01  0.    0.   -0.    0.   -0.   -0.04 -0.04 -0.01
-0.01  0.   -0.    0.02  0.01 -0.   -0.03  0.02 -0.02  0.01  0.01 -0.01
-0.    0.01 -0.04  0.   -0.01 -0.02 -0.03 -0.01  0.    0.   -0.   -0.
-0.01 -0.01 -0.02 -0.01 -0.01 -0.03  0.01  0.    0.    0.02  0.01  0.
 0.01 -0.    0.01 -0.01  0.   -0.01 -0.    0.02 -0.   -0.   -0.    0.
 0.    0.   -0.   -0.   -0.   -0.02 -0.01 -0.02  0.03  0.01  0.01 -0.
 0.01  0.01  0.01  0.02 -0.01 -0.01  0.01 -0.02 -0.01  0.02 -0.01 -0.
 0.02  0.    0.    0.    0.    0.   -0.   -0.   -0.   -0.01 -0.02 -0.01
-0.   -0.02 -0.01 -0.01  0.    0.01  0.01  0.01  0.01  0.01  0.03 -0.02
 0.   -0.   -0.03 -0.02 -0.   -0.    0.    0.    0.    0.    0.   -0.
-0.   -0.   -0.01 -0.01 -0.02 -0.02 -0.01 -0.01  0.   -0.01  0.   -0.02
-0.02 -0.   -0.    0.01 -0.01 -0.02 -0.01 -0.   -0.   -0.   -0.    0.
 0.    0.    0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.01 -0.01 -0.01
-0.01 -0.   -0.   -0.   -0.01 -0.02 -0.01 -0.02 -0.02 -0.01 -0.   -0.
-0.   -0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
-0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.
-0.   -0.   -0.   -0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.  ]

**Accuracy -----> 0.973**

**Shape ['7' '3' '7' ... '3' '7' '7']**

**intercept ----->: [16.35]**

**Weight ------>: [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.**

 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.   -0.   -0.   -0.   -0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.   -0.   -0.   -0.   -0.   -0.   -0.01 -0.01
-0.01 -0.01 -0.02 -0.01 -0.01 -0.01 -0.   -0.   -0.   -0.   -0.   -0.
-0.    0.    0.    0.    0.   -0.   -0.   -0.   -0.   -0.   -0.   -0.
-0.   -0.   -0.02 -0.02 -0.02 -0.01 -0.02 -0.02 -0.02 -0.02 -0.01 -0.01
-0.   -0.   -0.   -0.   -0.    0.    0.    0.    0.    0.   -0.   -0.
-0.    0.01  0.    0.01  0.01  0.01 -0.01 -0.   -0.01 -0.   -0.01 -0.01
-0.01 -0.02 -0.   -0.01 -0.02 -0.02 -0.01 -0.   -0.    0.    0.    0.
 0.   -0.    0.    0.   -0.02  0.    0.01 -0.01  0.01 -0.   -0.01  0.01
-0.01  0.01  0.01 -0.02 -0.   -0.01 -0.01 -0.   -0.   -0.    0.   -0.01
 0.    0.    0.    0.    0.    0.    0.   -0.   -0.01  0.    0.02 -0.

-0.  0.01  0.01  0.01 -0.  -0.02 -0.01 -0.  0.  0.01  0.01  0.01
 0.01  0.01  0.01  0.02  0.  0.  0.  0.  0.  0.  -0.  -0.01
 0.  0.  0.01  0.02  0.  0.01  0.01  0.01  0.01 -0.01  0.  0.01
-0.01 -0.01  0.01  0.03 -0.  -0.  -0.01  0.01  0.  0.  0.  0.
 0.  0.01  0.  0.01  0.01  0.01 -0.  0.02 -0.01 -0.02 -0.01  0.01
 0.02  0.  -0.01 -0.01 -0.02  0.03 -0.01  0.  -0.01 -0.01  0.  -0.
 0.01  0.  0.  0.  0.  0.  0.  -0.01 -0.  0.01  0.02 -0.01
 0.02  0.02  0.01 -0.02 -0.  -0.01  0.01  0.03  0.03 -0.  -0.  -0.01
-0.01 -0.  -0.  -0.01  0.01  0.  0.  0.  0.  0.  0.01  0.01
 0.  -0.  0.02  0.  -0.01 -0.02  0.01  0.02 -0.  -0.01  0.02  0.
 0.01  0.02  0.02  0.01  0.01  0.01 -0.  -0.02  0.  0.  0.  0.
 0.  0.01  0.01  0.01  0.  0.01 -0.  0.01 -0.  -0.  -0.01  0.01
-0.02 -0.01 -0.02  0.01 -0.01 -0.  0.01 -0.01 -0.02  0.  -0.01 -0.02
 0.  0.  0.  0.  0.  0.  0.01  0.01 -0.01  0.  0.01  0.
 0.02  0.02 -0.  -0.  -0.03 -0.01 -0.02 -0.01 -0.01 -0.01 -0.01  0.
 0.01  0.01 -0.01  0.01  0.  0.  0.  0.  0.  0.  0.  0.01
 0.  0.02  0.01  0.01  0.02 -0.01 -0.  0.02 -0.01 -0.02  0.01  0.
 0.01  0.01  0.  -0.  0.01 -0.  0.04  0.01 -0.  0.  0.  0.
 0.  0.  0.  0.01  0.01 -0.01 -0.01  0.  0.  0.01  0.  -0.01
-0.03 -0.01 -0.01  0.  0.01 -0.01  0.02  0.02 -0.  0.  0.01  0.
 0.  0.  0.  0.  0.  0.  -0.  -0.01  0.01 -0.  0.01 -0.
 0.01  0.01 -0.01  0.01  0.01 -0.03  0.01 -0.  -0.01  0.01 -0.02  0.
-0.  0.  -0.01 -0.01 -0.01  0.  0.  0.  0.  -0.  -0.  -0.02
-0.01 -0.  0.  -0.  -0.03  0.  0.  0.01 -0.01  0.01  0.01  0.02
 0.01  0.01 -0.  -0.  0.  -0.01  0.  0.  0.01  0.  0.  0.
 0.  -0.  -0.  -0.02 -0.01 -0.02 -0.01  0.01 -0.03 -0.01 -0.02  0.02
-0.01 -0.  0.01  0.  -0.01  0.02  0.01 -0.01 -0.  -0.02  0.01  0.01
 0.02  0.  0.  0.  0.  0.  -0.  -0.01 -0.01 -0.01 -0.01 -0.01
 0.  -0.  -0.01  0.02  0.02  0.  0.01 -0.01 -0.02 -0.01 -0.  0.02
-0.02 -0.02 -0.01  0.  0.01 -0.  0.  0.  0.  0.  -0.  -0.01
-0.01 -0.03 -0.03 -0.  0.02  0.  0.01  0.02  0.01 -0.02 -0.  0.01
-0.02 -0.02 -0.03 -0.03 -0.03 -0.01 -0.01 -0.02  0.  -0.  0.  0.
 0.  -0.  -0.  -0.01 -0.03 -0.02 -0.01 -0.  -0.01 -0.01 -0.  0.01
 0.01  0.  -0.03  0.02 -0.04  0.  -0.02 -0.02 -0.03 -0.01 -0.02 -0.01
-0.  0.  0.  0.  0.  0.  -0.  -0.  -0.02  0.  -0.  -0.02
-0.02 -0.01 -0.01 -0.  -0.01  0.01 -0.  0.01 -0.01 -0.  -0.02 -0.01
-0.02 -0.01 -0.01 -0.01  0.  0.  0.  0.  0.  0.  -0.  -0.01
-0.01  0.  -0.01  0.  -0.  0.02 -0.01 -0.  -0.  0.01  0.  -0.01
-0.  -0.02 -0.01 -0.01  0.  -0.01 -0.01 -0.01  0.  0.  0.  0.
 0.  0.  0.  -0.  0.  0.  -0.01  0.01 -0.  -0.04 -0.02 -0.01
-0.01 -0.01 -0.01  0.01  0.01 -0.  -0.  0.01  0.  -0.01 -0.  -0.
 0.  0.  0.  0.  0.  0.  0.  0.  0.01  0.01 -0.  -0.01
-0.02 -0.02  0.01 -0.  -0.02 -0.02 -0.  0.01 -0.  0.01  0.02  0.01
 0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.
 0.  -0.01 -0.02  0.  0.02  0.01  0.  -0.  -0.03  0.  -0.01 -0.03
-0.  0.03  0.01  0.  0.  0.  0.  0.  0.  0.  0.  0.

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. -0. -0. -0. 0.
0. 0. 0. 0. ]

**Accuracy -----> 0.981**

**Shape ['8' '8' '8' ... '8' '4' '8']**

**intercept ----->: [-0.]**

**Weight ------>: [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.**

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. -0. -0. -0. -0.
-0. 0. 0.01 0. -0. -0. -0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. -0. -0. -0. -0. -0.
-0. -0. -0.01 -0.01 -0.01 -0.01 0. 0.01 0. -0. -0. -0.
-0. 0. 0. 0. 0. 0. 0. -0. -0. -0. -0. -0.
-0. -0. -0.01 -0.01 0.01 -0. -0.01 -0. -0. 0. 0. 0.01
0.01 -0. -0. -0.01 -0. -0. 0. 0. 0. 0. -0. -0.
-0. -0. 0. 0. 0.01 -0. -0. 0. 0. 0. 0.01 -0.
0.01 0.02 0.01 0. -0.01 0.01 0.01 -0. -0.01 0. 0. 0.
0. 0. -0. -0. -0. -0. 0. 0. 0. 0. 0. 0.01
0.01 0.01 0. 0.01 -0.01 0. 0.01 -0.01 -0. -0. -0. 0.01
0. 0.01 0. 0. 0. 0. -0. -0. -0. -0.01 -0.01 -0.01
-0. -0. 0.01 0.01 0. 0.01 0. 0.01 -0. -0.01 0.01 0.01
0.01 0.01 0. -0. -0. 0.01 -0. 0. 0. 0. -0. -0.
-0.01 -0.01 -0.01 -0.01 0.01 0.01 0. -0. -0. 0. 0.01 0.01
0.01 0.01 0.01 0.02 -0. -0. 0. 0.01 0.01 0.01 0. 0.
0. 0. -0. -0. 0. 0. 0.01 0.01 0.01 -0. 0.01 0.
-0.01 -0. -0.01 -0. -0. -0.01 -0. -0. 0.01 -0. 0.01 0.01
0.01 -0. 0. 0. 0. 0. -0. 0. 0.01 0.01 0.01 0.
-0. -0. -0. 0. -0.01 0. 0.02 -0. -0.01 -0.01 -0. 0.01
0.01 0.01 0. 0.02 0.01 -0. 0. 0. 0. -0. -0. 0.01
0.01 0. 0.01 -0. 0.01 0. -0. 0.01 -0.01 0. 0.01 -0.01
-0.01 -0.02 -0. 0. 0. 0. 0. 0.01 0.03 0.01 0.02 0.01
0. -0. -0. 0.01 0. 0. -0.01 0. -0. -0.01 -0.02 -0.
-0.02 -0. 0.02 -0.01 0. 0.01 -0.01 -0.01 0. -0. -0. 0.01
0.01 0. 0. 0. 0. -0. 0. 0.01 -0. 0. -0.01 -0.01
-0.01 -0. -0.02 -0. -0.02 0.01 -0.01 -0.01 -0.01 -0.02 0.01 -0.01
-0.02 -0. 0. 0.01 -0. -0. 0. 0. 0. -0. -0. 0.
-0.01 -0.01 0. -0.01 -0.01 -0. 0. 0. -0. 0.01 0. -0.01
-0.02 -0.01 -0.01 -0. -0.01 0.01 -0. 0. -0. -0. 0. 0.
0. -0. -0. -0. -0.02 -0.02 0. -0.02 -0.01 -0.01 -0.01 -0.01
0.01 0. 0.01 -0.01 -0.01 -0.02 -0.01 -0.01 -0.01 0. -0. -0.
-0. -0. -0. 0. 0. 0. -0. -0. -0.02 -0.01 -0. -0.01

```
-0.01 -0.    0.01 -0.    0.   -0.   -0.01 -0.02  0.01 -0.01 -0.01  0.01
-0.   -0.01 -0.   -0.   -0.   -0.   -0.    0.    0.    0.   -0.   -0.
-0.02 -0.01  0.01 -0.01  0.    0.01  0.    0.   -0.01  0.01 -0.   -0.02
-0.02 -0.    0.01 -0.   -0.01 -0.01  0.   -0.   -0.   -0.   -0.    0.
 0.    0.   -0.   -0.   -0.01 -0.01  0.   -0.   -0.   -0.01  0.01 -0.
 0.   -0.01 -0.01 -0.01 -0.01  0.   -0.01  0.   -0.   -0.01  0.01 -0.
-0.   -0.   -0.    0.    0.    0.    0.   -0.   -0.   -0.01  0.    0.
 0.01  0.02  0.01 -0.01 -0.    0.01 -0.01 -0.   -0.01 -0.   -0.01 -0.01
-0.01 -0.01  0.01 -0.    0.   -0.   -0.    0.    0.    0.    0.   -0.
-0.    0.   -0.    0.    0.    0.02  0.03  0.01  0.    0.01 -0.    0.
-0.    0.02  0.    0.01  0.01  0.01  0.01 -0.01  0.    0.    0.    0.
 0.    0.    0.   -0.   -0.    0.01  0.02  0.01  0.01  0.01  0.01  0.03
 0.02  0.02  0.02  0.02  0.01  0.01  0.02  0.02  0.02  0.01  0.    0.
 0.   -0.    0.    0.    0.    0.    0.    0.    0.    0.01  0.01  0.01
 0.02  0.01 -0.    0.02  0.01  0.01  0.02  0.01  0.01  0.01  0.01  0.01
 0.01  0.    0.    0.   -0.   -0.   -0.    0.    0.    0.    0.    0.
 0.    0.    0.   -0.    0.01  0.   -0.    0.   -0.    0.02  0.01  0.01
 0.01  0.01  0.    0.    0.01  0.01  0.    0.   -0.   -0.   -0.    0.
 0.    0.    0.    0.    0.   -0.   -0.01 -0.01 -0.    0.   -0.   -0.
 0.    0.   -0.   -0.01  0.    0.01  0.01  0.   -0.   -0.   -0.    0.
-0.   -0.    0.    0.    0.    0.    0.    0.    0.    0.   -0.   -0.
-0.01 -0.02 -0.01 -0.    0.    0.    0.    0.    0.01 -0.   -0.   -0.
-0.   -0.   -0.   -0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.   -0.   -0.    0.    0.    0.    0.    0.    0.    0.
 0.   -0.   -0.   -0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.    0.
 0.    0.    0.    0.  ]
```

## Accuracy -----> 0.9880519480519481

# Standard Error ----> 0.0082

Also using logistic regression plotted for [3,4] Numbers.

logisticRegr = LogisticRegression(solver = 'lbfgs',max_iter=1200) and tested for cl

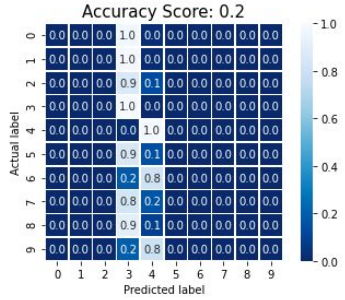
```
logisticRegr.predict(test_img[0:10])
array(['3', '4', '3', '3', '4', '4', '3', '3', '3', '3'], dtype='<U1')
```

Accuracy  = logisticRegr.score(test_img, test_lbl)

# Accuracy → 0.183

**Also plotted the accuracy for one of the instance ( predicted value VS actual value)**

```
#plthttp://localhost:8890/notebooks/Documents/github/Python_Tutorials/Sklearn/Logistic_Regression/LogisticRegression_MNIST-Easier_Loading.ipynb
sns.heatmap(cm_normalized, annot=True, fmt=".1f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {:.1f}'.format(score)
plt.title(all_sample_title, size = 15);
```




# Linear Regression Output -

Experimented Mnist Dataset and captured the output  ( 0,4) (1,5)(2,6)2,7)(3,8) wi

**intercept ----->: 1.871914749230491**
**Weight ------>:  -225609.32392792773**

Prediction ----> [ 0  4  4 ... -1  4  0]
Actual l----> [0 4 4 ... 0 4 0]
**Accuracy -----> 0.6634320735444331**
---
Shape ['1' '5' '5' ... '1' '1' '1']
**intercept ----->: 4.102126213801093**
**Weight ------>:  425407.34558768675**
prediction----> [1.   1.24 1.2  ... 3.72 4.03 1.95]
Prediction ----> [1 1 1 ... 4 4 2]
Actua ----> [1 1 1 ... 5 5 1]

**Accuracy -----> 0.7665**

**----------**

<class 'list'>
Shape ['2' '6' '2' ... '6' '2' '6']
**intercept -----> : 3.797575014773756**
**Weight ------> : -1402430.064593433**
prediction----> [2.48 2.33 4.38 ... 1.76 4.95 1.89]
Prediction ----> [2 2 4 ... 2 5 2]
Actual ----> [2 2 6 ... 2 6 2]

**Accuracy -----> 0.5335**

**----------**

<class 'list'>
Shape ['7' '3' '7' ... '3' '7' '7']
**intercept -----> : 5.612925636491273**
**Weight ------> : 378931.6271637447**
prediction----> [6.96 7.48 5.62 ... 7.5 4.63 6.96]
Prediction ----> [7 7 6 ... 7 5 7]
Actual l----> [7 7 7 ... 7 3 7]

**Accuracy -----> 0.563**

<class 'list'>
Shape ['8' '8' '8' ... '8' '4' '8']
**intercept -----> : 5.678687956075184**
**Weight ------> : -313486.29262392083**
prediction----> [4.07 4.36 4.17 ... 3.69 7.78 6.79]
Prediction ----> [4 4 4 ... 4 8 7]
Actual----> [4 4 4 ... 4 8 8]

**Accuracy -----> 0.5641558441558442**

**Standard Error ----> 0.0862**

==========

Similar for Mnist dataset the Standard Deviation with Logistic :

**Standard Deviation Error ----> 0.0082**

**Vs**

**Linear regression Std Deviation Err - 0.08**

**Clearly we can see Logistic regression performs better for classification compated to Linear regression:**

**Analysed why Linear regression is not suitable for classification using different example AGE vs Purchasing Power for Linear and Logistic and plotted the graph-**

Plotted Age vs Purchasing Power.
From the Mean Square Errors and Linear/Logistic Graph Plots.

**Label/Class '0' indicates, Person is not able to purchase.**

**Label/Class '1' indicates, Person is able to purchase**

.

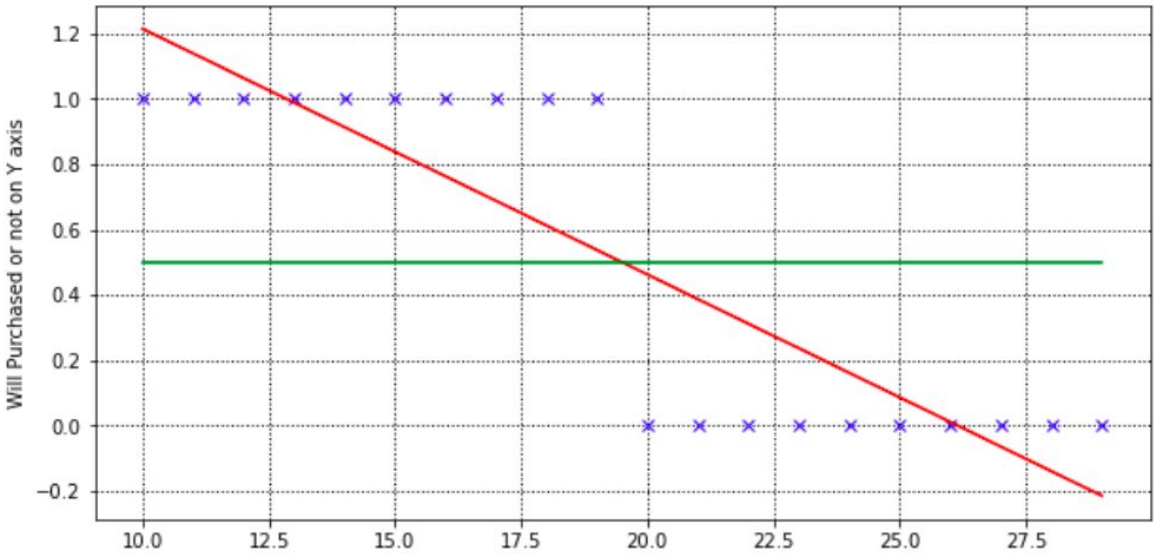
Linear regression mean square error is very high compared to Logistic regression.

**Linear regression R2: 0.42112651342340734**
**Logistic regression R2: 0.9553066567250714**

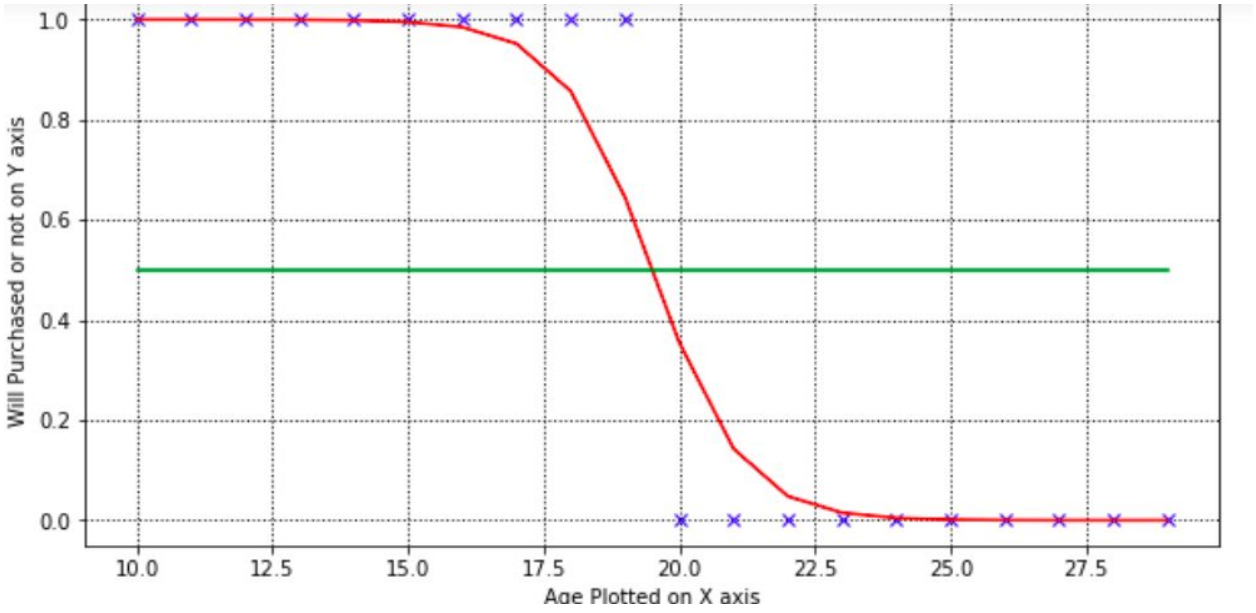
Linear regression RMSE: 0.1286385525725761 1

Logistic regression RMSE: 0.009931854061095247

**Linear Regression PLot**




array([ 1.21428571,  0.83834586,  0.46240602,  0.08646617, -0.28947368])



**RMSE  is less  for Logistic compared to Linear**

# Kaggle Analysis:

# For  IBM Dataset to determine the Attrition in the company using Logistic and Linear Regression.

**The Features consists of both Numerical and Categorical Data as shown below:**
**Also attached the HR.CSV sheet for reference.**

['Age', 'DailyRate', 'DistanceFromHome', 'Education', 'EmployeeCount',
'EmployeeNumber', 'EnvironmentSatisfaction', 'HourlyRate',
'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
'YearsSinceLastPromotion', 'YearsWithCurrManager', 'n_Gender',
'n_JobRole', 'n_BusinessTravel', 'n_Department', 'n_EducationField'],
dtype='object')

In [11]:

**LogisticRegression(max_iter=9000, warm_start=True)**

Weights [-0.03 -0.  0.04  0.04  0.  -0.  -0.39 -0.  -0.35 -0.05 -0.35 -0.
0.  0.14 -0.02  0.06 -0.12  0.05 -0.35 -0.05 -0.19 -0.18  0.08 -0.14
0.14 -0.12  0.08  0.03 -0.05  0.13  0.08]

Predict [0 0 0 0 0 0 0 0 0 0]

mean squared error 0.1463

intercept: 0.1463
Weights [-0.03 -0. 0.03 -0.05 0. -0. -0.32 -0. -0.25 -0.01 -0.35 -0.
0. 0.17 -0.03 0.01 -0.26 0.05 -0.39 -0.04 -0.18 -0.13 0.08 -0.14
0.16 -0.16 0.07 0.04 0.04 0.14 0.03]

Predict [0 0 0 0 0 0 0 0 1 0]
mean squared error 0.1701

intercept: 0.1701
Weights [-0.03 -0. 0.04 0.09 0. -0. -0.32 -0. -0.43 -0.08 -0.35 -0.
0. 0.13 -0.02 0.03 -0.17 0.05 -0.46 -0.05 -0.27 -0.28 0.08 -0.13
0.11 -0.11 0.12 0.06 0.01 0.14 0.09]

Predict [0 0 0 0 0 0 0 0 0 1]
mean squared error 0.1531

**intercept: 0.1531**
**Weights [-0.04 -0. 0.03 0.03 0. -0. -0.35 -0. -0.4 -0.03 -0.32 -0.**
**0. 0.11 -0.02 0.01 -0.23 0.05 -0.49 -0.03 -0.15 -0.11 0.07 -0.12**
**0.12 -0.11 0.08 0.05 0.04 0.15 0.02]**

Predict [0 0 0 0 0 0 0 0 0 0]
mean squared error 0.1429

intercept: 0.1429
Weights [-0.03 -0. 0.03 -0.02 0. -0. -0.42 -0. -0.46 -0.06 -0.4 -0.
0. 0.15 -0.02 0.07 -0.09 0.06 -0.52 -0.04 -0.14 -0.27 0.1 -0.15
0.14 -0.14 0.02 0.02 0.05 0.12 0.06]

Predict [0 0 1 0 0 0 0 0 0 0]
**mean squared error 0.1395**

In [9]:

```
print(f'Mean Square Error ---> {s_list}')
std_dev = s_list
```

```
p = round(np.std(std_dev, dtype=np.float64),4)
print(f' Standard Error ----> {p}')
```

## Mean Square Error ---> [0.1463, 0.1701, 0.1531, 0.1429, 0.1395]

## Standard Error ----> 0.0108

## Linear Regression for Kaggle Dataset

Similarly executed for Kaggle Dataset and captured the results as given below:

```
intercept: 0       1
Name: n_Attrition, Length: 1470, dtype: int64
 Weights -0.003922572756349346

 Predict [0.15 0.37 0.16 0.21 0.01 0.32 0.34 0.13 0.07 0.43]
mean squared error 0.85
intercept: 0.85
 Weights -0.0037047644726774655

 Predict [0.29 0.12 0.06 0.18 0.1  0.2  0.34 0.17 0.18 0.06]
 Actual     n_Attrition
mean squared error 0.83
intercept: 0.83
 Weights -0.004673571955363548

 Predict [ 0.04  0.01  0.13  0.16  0.06  0.1  -0.19  0.09  0.13  0.25]
mean squared error 0.84
intercept: 0.84
 Weights -0.003037182499893588

 Predict [ 0.4   0.14  0.42  0.19 -0.03  0.25 -0.03  0.28  0.33  0.33]
mean squared error 0.85
intercept: 0.85
 Weights -0.0033438458684491566

 Predict [ 0.27  0.48  0.27  0.2   0.17  0.13  0.36 -0.07  0.19  0.1 ]
```

mean squared error 0.85

In [16]:

```
print(f'Mean Square Error ---> {s_list}')

std_dev = s_list




p = round(np.std(std_dev, dtype=np.float64),4)

print(f' Standard Error ----> {p}')
```

# Mean Square Error ---> [0.85, 0.83, 0.84, 0.85, 0.85]

# Standard Error ----> 0.008

**Conclusion : Logistic Regression is better for classification problems compared linear regression.**

Thank you