# Report
# ML Assignment 4

Sankalp Rajendran IIT2019173

## IPYNB LINK

- **Batch Gradient Descent (BGC)**

**Approach:** Used scaled features list to iterate over all samples the no. of times taken as argument and updated coefficients each time.

- **Stochastic Gradient Descent (SGD)**

**Approach:** It takes the no. of iterations as argument and in each iteration a random index is selected from the training set and coefficients are updated after each iteration. The error calculated using a separate function is returned.

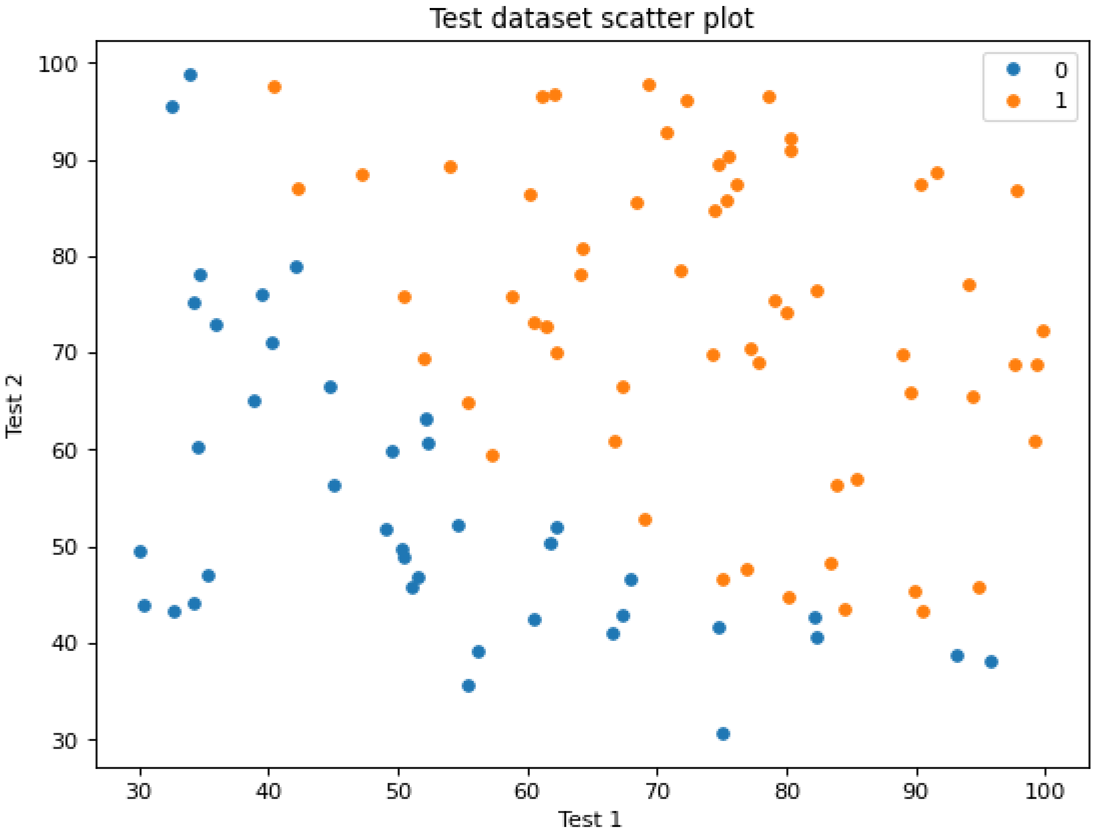
- **Mini-Batch Gradient Descent (MBGD)**

**Approach:** It takes the learning rate(**α**), no. of iterations(k) and batch size(b) as argument and in each iteration a batch index(i) is selected from the range of number of batches. Then in the nested loop the next step of gradient descent is calculated using the (b) no of samples starting from i*b index and coefficients are updated after each iteration. The error calculated using a separate function is returned.

# Q1

**Classification model that estimates an applicant's probability of getting admission to an institution based on the scores from those two examinations.**

## Dataset scatter plot:



-

## a-*No feature scaling, No higher order terms, No regularisation*

| Algorithm | α(Learning Rate) | k(No. of Iterations) | % Error |
|---|---|---|---|
| BGD | 0.01 | 100000 | 10.0 |
| SGD | 0.00001 | 10000 | 20.0 |
| MBGD | 0.000001 | 100 (Batch Size=15) | 20.0 |

## *With feature scaling, No higher order terms, No regularisation*

| Algorithm | **α**(Learning Rate) | k(No. of Iterations) | % Error |
|---|---|---|---|
| BGD | 0.00000000000000001 | 400 | 6.66 |
| SGD | 0.000000000000001 | 300 | 13.33 |
| MBGD | 0.0001 | 1000 (Batch Size=15) | 10.0 |

## **b-***With feature scaling, with higher order terms, No regularisation*

| Algorithm | **α**(Learning Rate) | k(No. of Iterations) | % Error |
|---|---|---|---|
| BGD | 0.00000000000000001 | 60 | 10.0 |
| SGD | 0.00000000000000001 | 3000 | 10.0 |
| MBGD | 0.000000000000000001 | 6000 (Batch Size=5) | 6.66 |

## **c-***With feature scaling, with higher order terms, with regularisation*

| Algorithm | **α**(Learning Rate) | k(No. of Iterations) | % Error |
|---|---|---|---|
| BGD | 0.0000000000000001 | 100 | 33.33 |
| SGD | 0.00000000000000001 | 500 | 10.0 |
| MBGD | 0.00000000000000001 | 50 (Batch Size=10) | 10.0 |

(Note: In regularisation, lambda was taken as 1.)

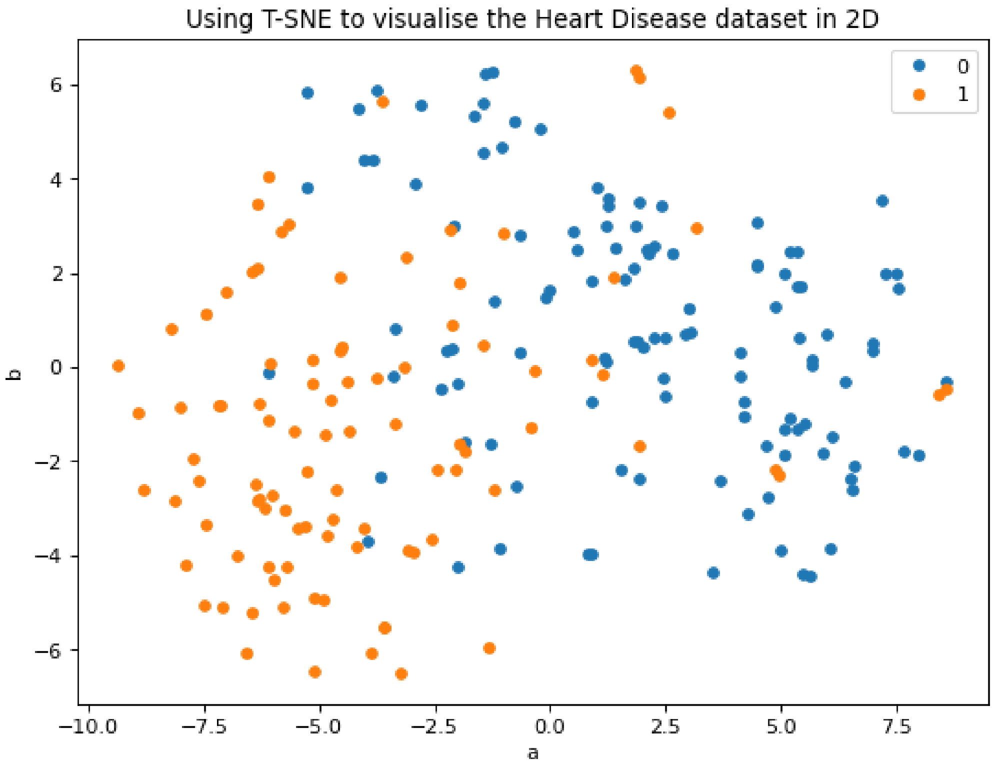
**Analysis:** There were  70 points in the training dataset and 30 in the testing dataset. Since the dataset was very small, it is observed that the results are very inconsistent. In stochastic and mini-batch gradient descent we have an element of randomisation which combined with the small sample size caused the results to vary by a large amount. The % error values in the tables above are taken to be the best possible after lots of trial and errors over a range of **α**(Learning Rate)  and k(No. of Iterations).

The Predict() function takes the two test scores as an argument and returns 1 and 0 to indicate admission or not.

# Q2

**Classifier using logistic regression on Cleveland Medical data set for heart disease diagnosis**

## Dataset scatter plot:



Since, the dataset has 14 features, here t-sne was used to visualize it in 2D plot.

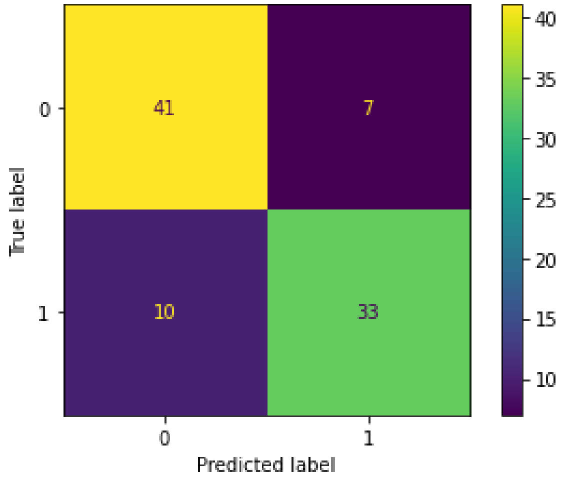
| No. of features taken | α(Learning Rate) | k(No. of Iterations) | % Error |
|---|---|---|---|
| 4(reduced with PCA) | 0.001 | 10000 | 17.58 |
| 13 | 0.001 | 10000 | 17.58 |

Note: PCA was used to reduce the dataset to 4 features. LDA was not used because in it the no. of components are always less than no. of classes of labels. Error was observed to remain the same when all 13 features were used in logistic regression.

# Analysis:

- Batch Gradient Descent was used to calculate the error %.
- The classifier() function uses the same algorithm to return the prediction labels on the test set.
- The predicted list and the correct labels were used to create the confusion matrix which shows that 74 out of 91 were classified correctly while there were 7 false positives and 10 false negatives.
- The single_classifier() function takes as argument the features of a single patient and predicts heart disease.

Confusion matrix for 4 features case:



- Specificity= 41/(41+7) = 0.85
- Sensitivity= 33/(33+10) = 0.77
- Precision= 33/(33+7) = 0.82
- Negative Predictive value= 41/(41+10) = 0.80