A Project Report

On

Logistic Regression

BY

AMAN BADJATE	2017B3A70559H
GARVIT SONI	2017B3A70458H
KSHITIJ VERMA	2017B1A71145H

UNDER THE SUPERVISION OF

DR. BHANU MURTHY

SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS OF

BITS F464: MACHINE LEARNING

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJASTHAN)

HYDERABAD CAMPUS

APRIL, 2021

Contents

List of Figures

1 Introduction

Logistic Regression is an algorithm mostly used for classification problems. We have a binary classification problem here. This algorithm uses the sigmoid function to calculate the probability of a datapoint to belong to a particular class. Sigmoid function is basically an S-shaped curve which can take any real value lying between 0 and 1. In this assignment, the aim is to detect forged banknotes by classifying a datapoint as 0 or 1.

2 Design Decisions

2.1 Dataset

The dataset consists of five columns, the first four columns represent features and the last column contains binary value representing the two classes for our classification problem.

2.2 Feature Scaling

Expressing an attribute in smaller units will lead to a larger range for that attribute, and thus tend to give such an attribute greater effect. To avoid dependence on the choice of measurement units, the data should be normalized or standardized. Standardization is used to normalize the features and bring them to the same scale. In standardization , the values for an attribute, A, are normalized based on the mean (i.e., average) and standard deviation of A.

$$z = (x - \mu) / \sigma$$

2.3 Splitting the dataset

The dataset is firstly shuffled to make sure that any ordering of the dataset is removed. This ensures that our training model does not get biased. The dataset is then divided using 70:30 train:test split.

2.4 Logistic Regression

The dataset is trained and tested using Logistic Regression. Two models are developed: (i) **Using Gradient Descent Algorithm** (ii) **Using Stochastic Gradient Descent Algorithm**. In both models, we have experimented with different learning rates. The results obtained using each of them is shown in the results section.

2.5 Determining feature importance

The 4-dimensional weight vector is used to get important features. Large positive values of w[i] indicates higher importance of the i-th feature in the prediction of positive class. Large negative values of w[i] indicates higher importance of the i-th feature in the prediction of negative class. This is seen from the expression of logistic loss function. Stochastic Gradient Descent minimises the loss by setting large positive weights for features that are more important for predicting a data point to belong to a positive class and the same goes with negative class . So, overall we conclude that the magnitude of the weight matters in determining the importance of features.

The magnitude of the weight obtained for each of the four features, respectively for Gradient Descent are **3.88, 3.47, 3.26, 0.29**.

The magnitude of the weight obtained for each of the four features, respectively for Stochastic Gradient Descent are **2.84, 2.17, 1.93, 0.17**.

So we can clearly see that the first three features are important and play a major role in determining the class a data point belongs to.

3 Comparison of Different Learning rates

As discussed in the design section, we have experimented with different learning rates. We have plotted loss and accuracy versus epochs for 3 different learning rates for both GD and SGD

3.1 Small Learning Rate

We have plotted the graphs for error and accuracy vs epochs for learning rate = 0.000005.

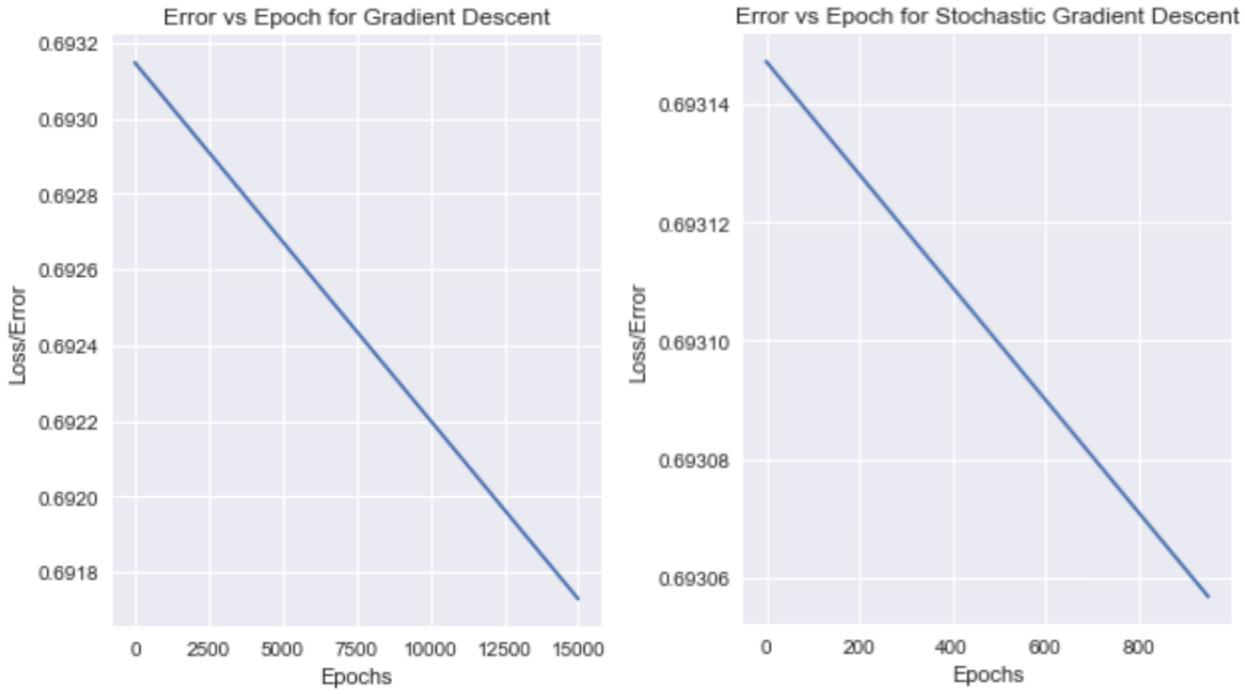

Figure 1: Error vs Epoch for both GD and SGD for small eta

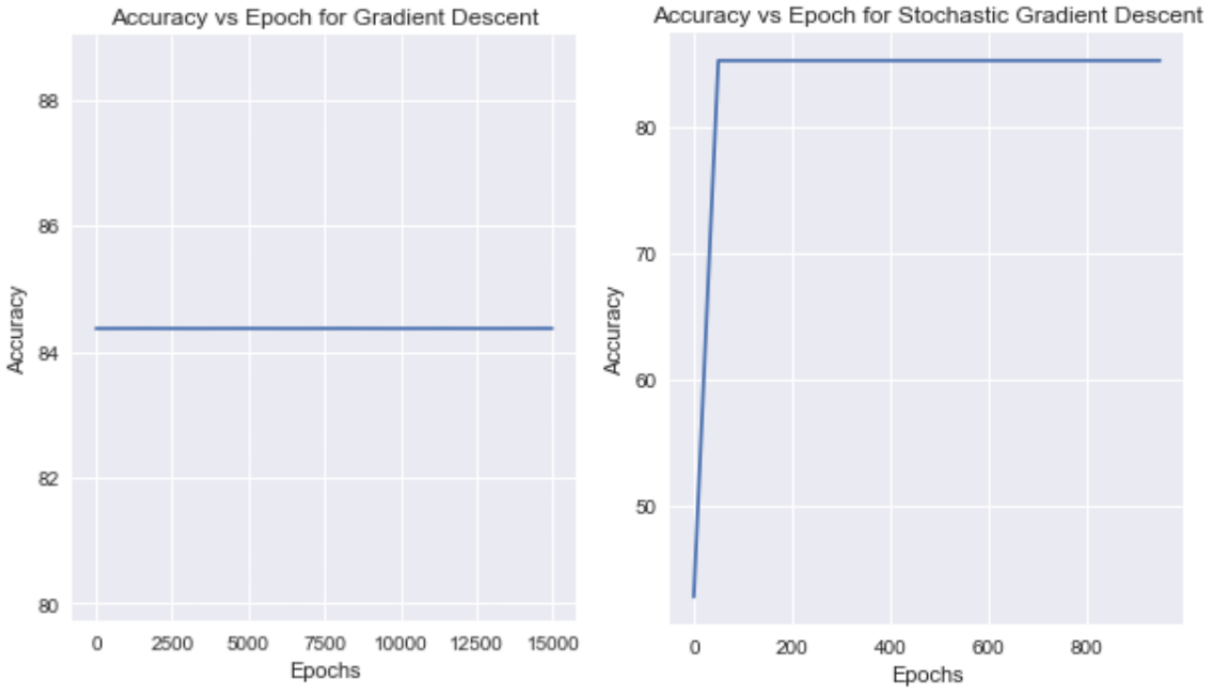

Figure 2: Accuracy vs Epoch for both GD and SGD for small eta

3.2 Large Learning Rate

We have plotted the graphs for error and accuracy vs epochs for learning rate = 50.

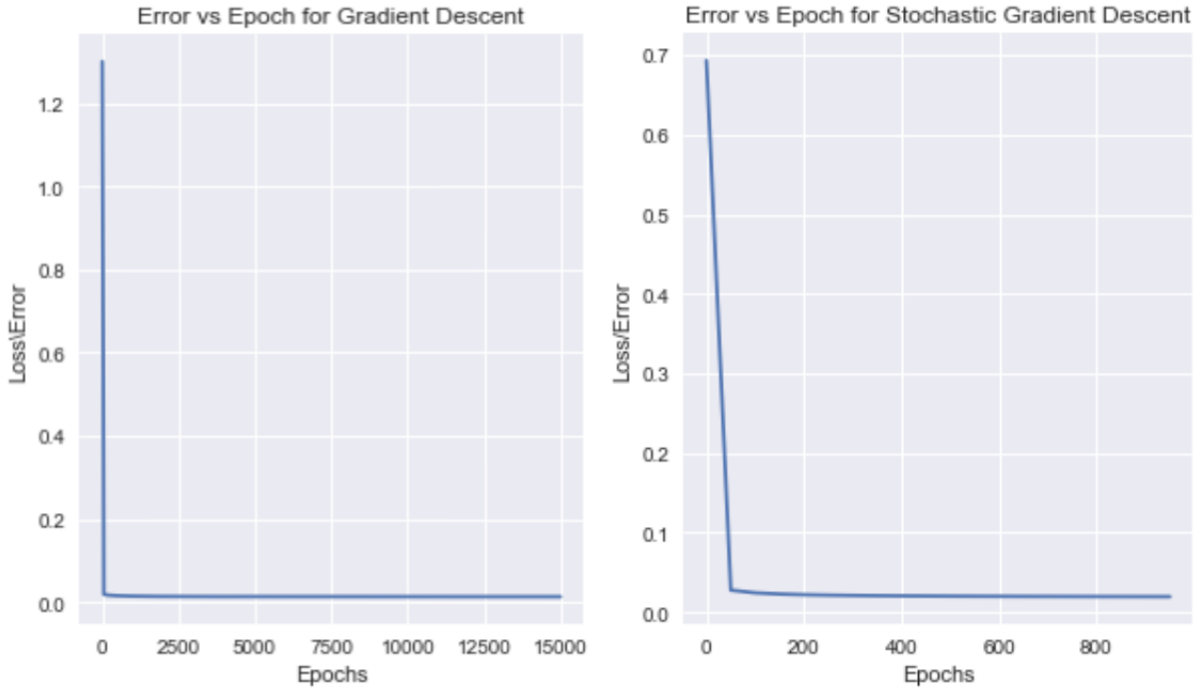

Figure 3: Error vs Epoch for both GD and SGD for large eta

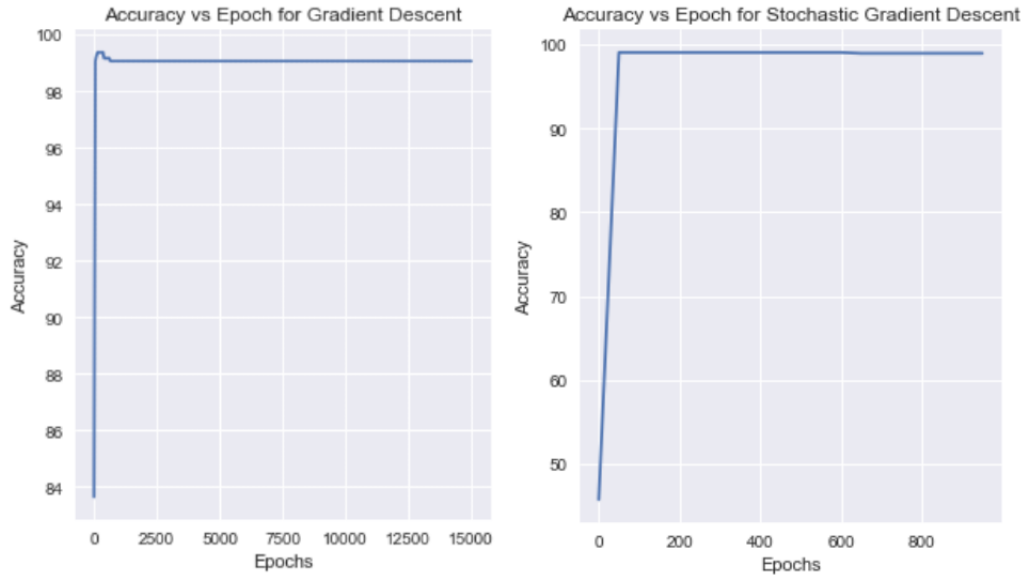

Figure 4: Accuracy vs Epoch for both GD and SGD for large eta

3.3 Optimal Learning Rate

We have plotted the graphs for error and accuracy vs epochs for learning rate = 0.01.

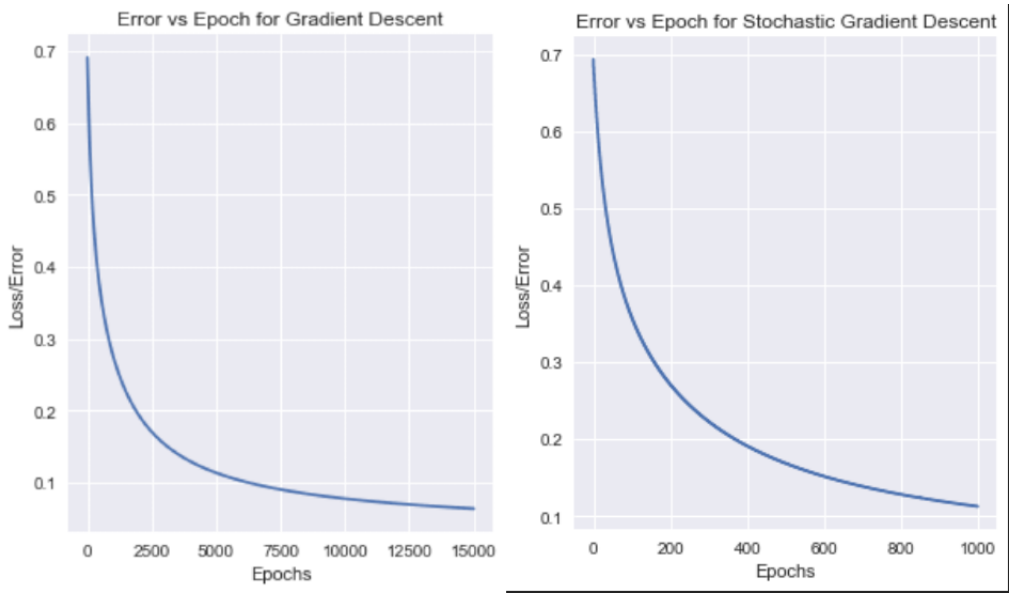

Figure 5: Error vs Epoch for both GD and SGD for optimal eta

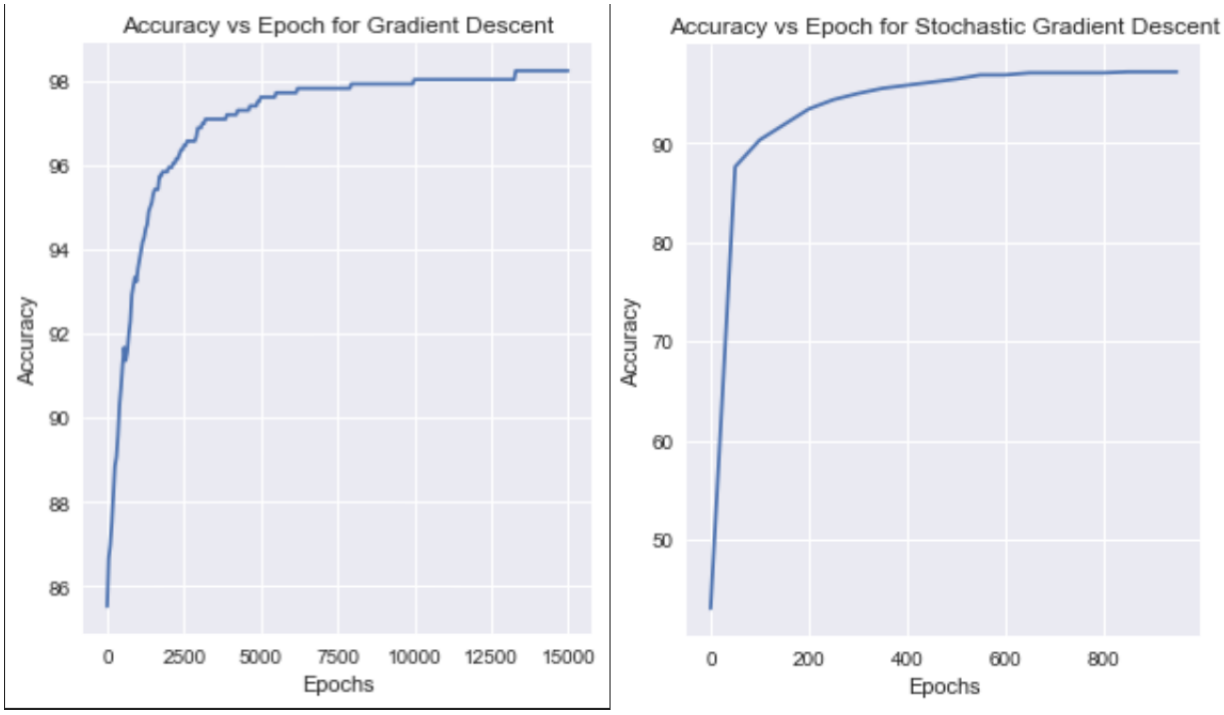

Figure 6: Accuracy vs Epoch for both GD and SGD for optimal eta

4 Results

4.1 For Gradient Descent

The result, including accuracy, F-score, Recall and Precision obtained are shown below.

Average Train Statistics using GD

Average Accuracy: 98.06

Average Fscore: 0.98

Average Recall: 0.96

Average Precision: 0.99

Average Test Statistics using GD

Average Accuracy: 97.74

Average Fscore: 0.97

Average Recall: 0.95

Average Precision: 1.00

4.2 For Stochastic Gradient Descent

The result, including accuracy, F-score, Recall and Precision obtained are shown below.

Average Train Statistics using SGD

Average Accuracy: 97.47

Average Fscore: 0.97

Average Recall: 0.96

Average Precision: 0.99

Average Train Statistics using SGD

Average Accuracy: 97.16

Average Fscore: 0.97

Average Recall: 0.95

Average Precision: 0.99

5 Conclusion

Both models i,e, Gradient Descent and Stochastic Gradient Descent gave similar results, all giving accuracy around 98%. Lower values of weights in terms of magnitude were observed for lower learning rate. A very low learning rate like 0.000005 gave poor results and a very high learning rate like 50 also gave poor results but learning rate = 0.05 gave good results.

From the plot of Error vs Epochs, we can see that the error function reaches its minimum in 2000 to 2500 iterations for GD and 200-250 iterations for SGD, which does not require very heavy computation and too much training time. Hence, this classifier serves its purpose very well. We can conclude that logistic regression was one of the best algorithms suited for this classification problem as we are getting a very high accuracy of around 99