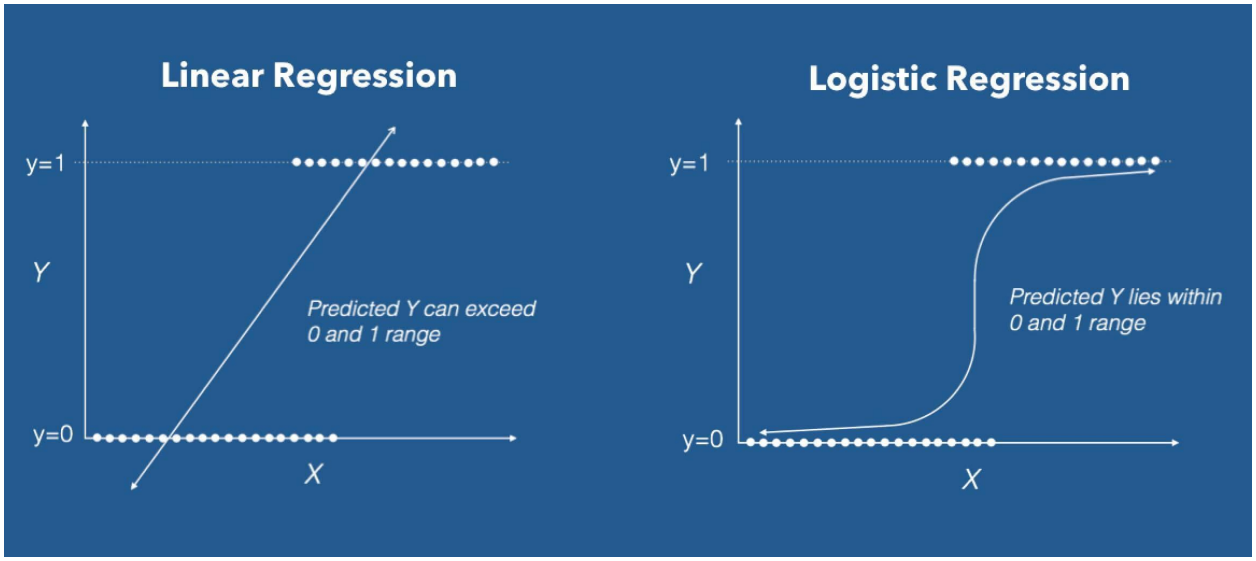
2A - Logistic Regression

1. Brief description of the model and its implementation

Logistic Regression is a Machine Learning algorithm that is used for classification problems, it is a predictive analysis algorithm and based on the concept of probability.

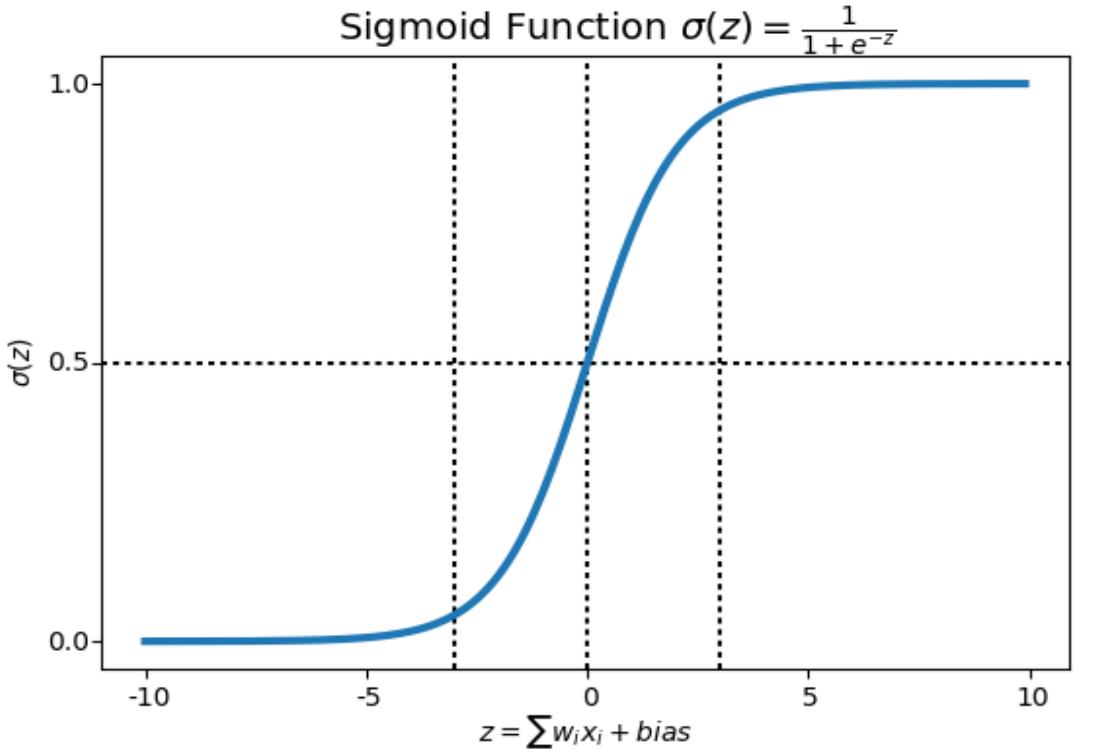


We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the '**Sigmoid function**' or also known as the 'logistic function' instead of a linear function.

The hypothesis of logistic regression tends to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

$$0 \leq h_\theta(x) \leq 1$$

In order to map predicted values to probabilities, we use the Sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.



$$f(x) = \frac{1}{1+e^{-(x)}}$$

Implementation:

1. We read the data set.
2. We created a 70:30 split 10 times.
3. Using Gradient Descent we found out weights, loss, accuracy and bias for each split.
4. Now using Stochastic Gradient Descent we found out weights, loss, accuracy and bias for each split.
5. After this, we found precision, recall and f-score for both SGD and GD.
6. At last, we plotted the loss and accuracy for the model every 50 iterations.

2. The most important feature in the dataset

The most important feature in our dataset is '**attr1**' since it has the highest absolute weight both in GD as well as SGD.

GD weights: [-7.1283898 -3.8592469 -4.83226118 -0.53333067]

SGD weights: [-1.17823966 -0.63425724 -0.65358423 -0.26335889]

3. The final train and test metrics (loss, accuracy, recall, precision and f score) achieved by the model with GD and SGD:

.....Gradient Descent.....

GD weights: [-7.1283898 -3.8592469 -4.83226118 -0.53333067]
GD bias: 6.732066125824214
Loss: 16.189998909778826

_____Training GD_____
Accuracy : 0.9917708333333334
Precision : 0.9897803387525836
Recall : 0.9918353536930902
F score : 0.990804202659895

_____Testing GD_____
Accuracy : 0.9881067961165049
Precision : 0.9856978403028613
Recall : 0.9871012716081433
F score : 0.9863146302442994

.....Stochastic Gradient Descent.....

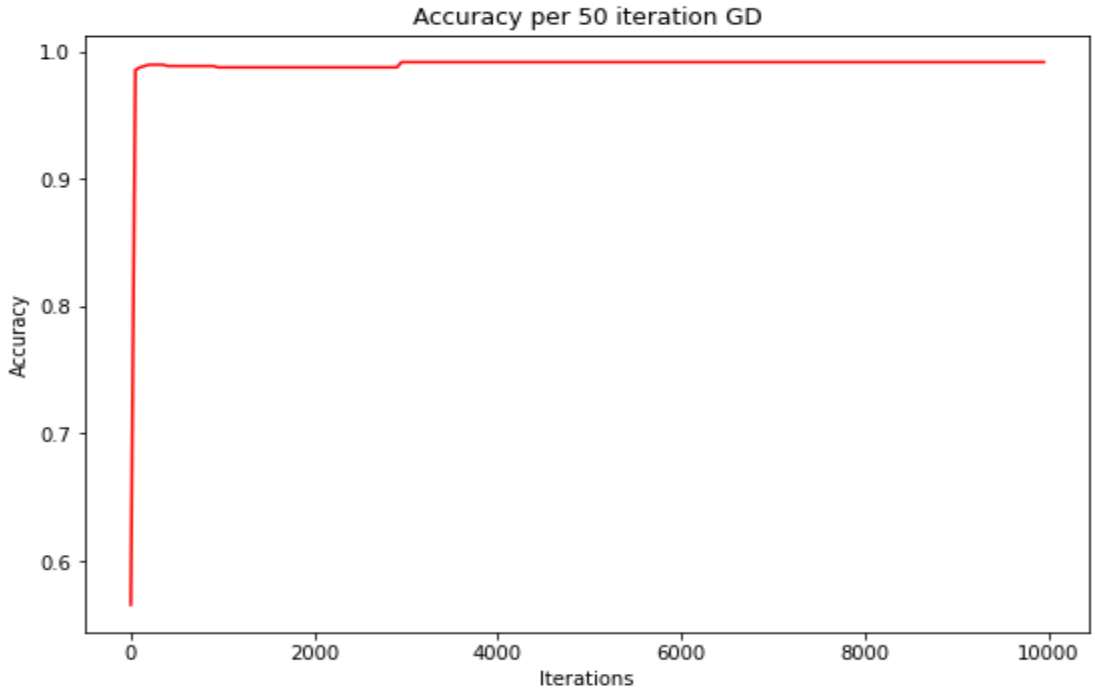
SGD weights: [-1.17823966 -0.63425724 -0.65358423 -0.26335889]
SGD bias: 0.5548669149810307
Loss : 0.056107717315380966

_____Training SGD_____
Accuracy : 0.9722916666666667
Precision : 0.9829818690137808
Recall : 0.9546232678381671
F score : 0.9685836515510057

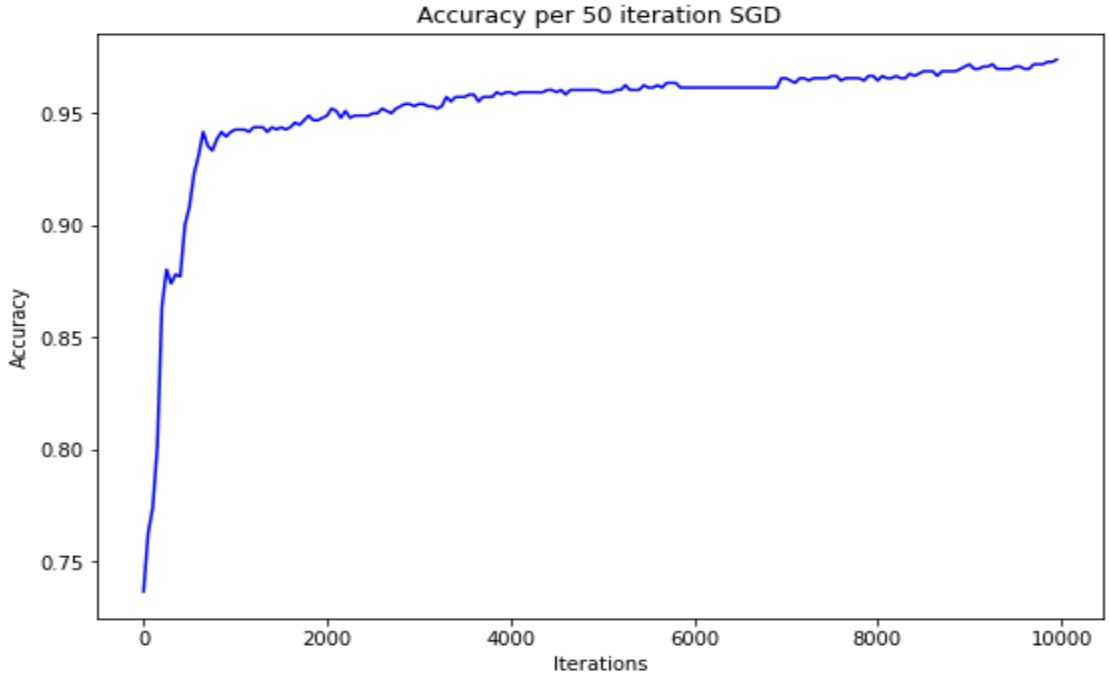
_____Testing SGD_____
Accuracy : 0.9754854368932039
Precision : 0.9779780396249536
Recall : 0.9654897698386384
F score : 0.971627306737441

4. Plots of accuracy for three different learning rates using GD and SGD:

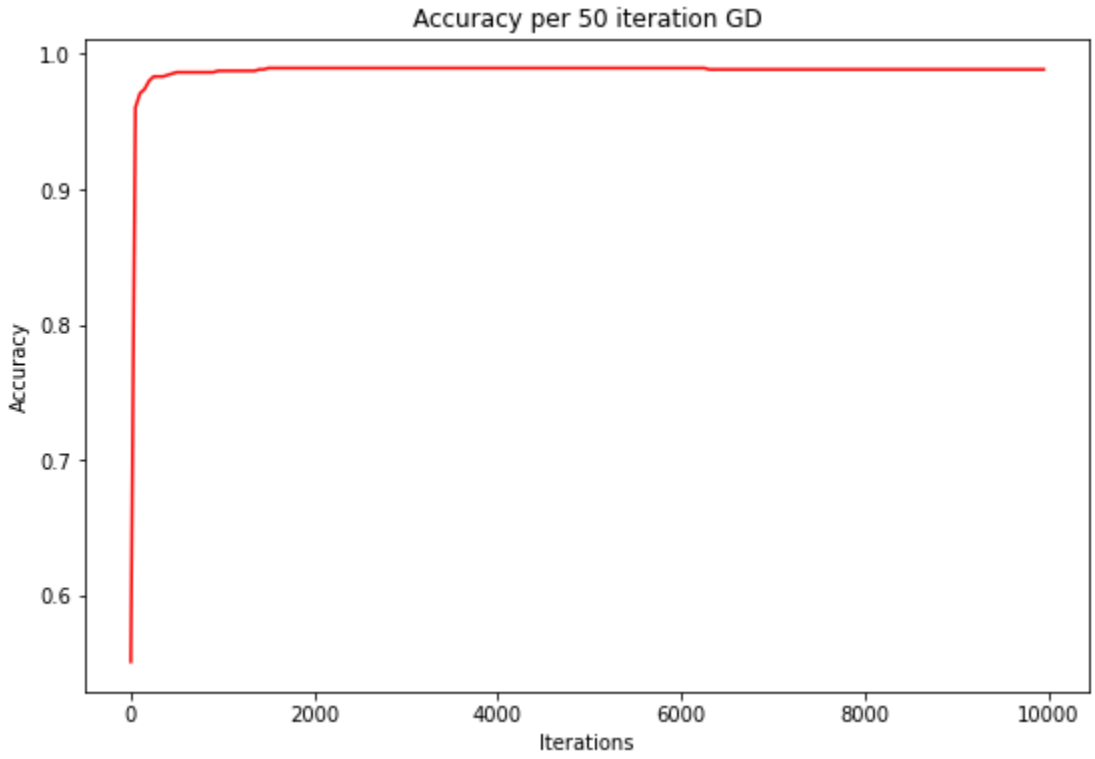
GD with 0.001 learning rate-



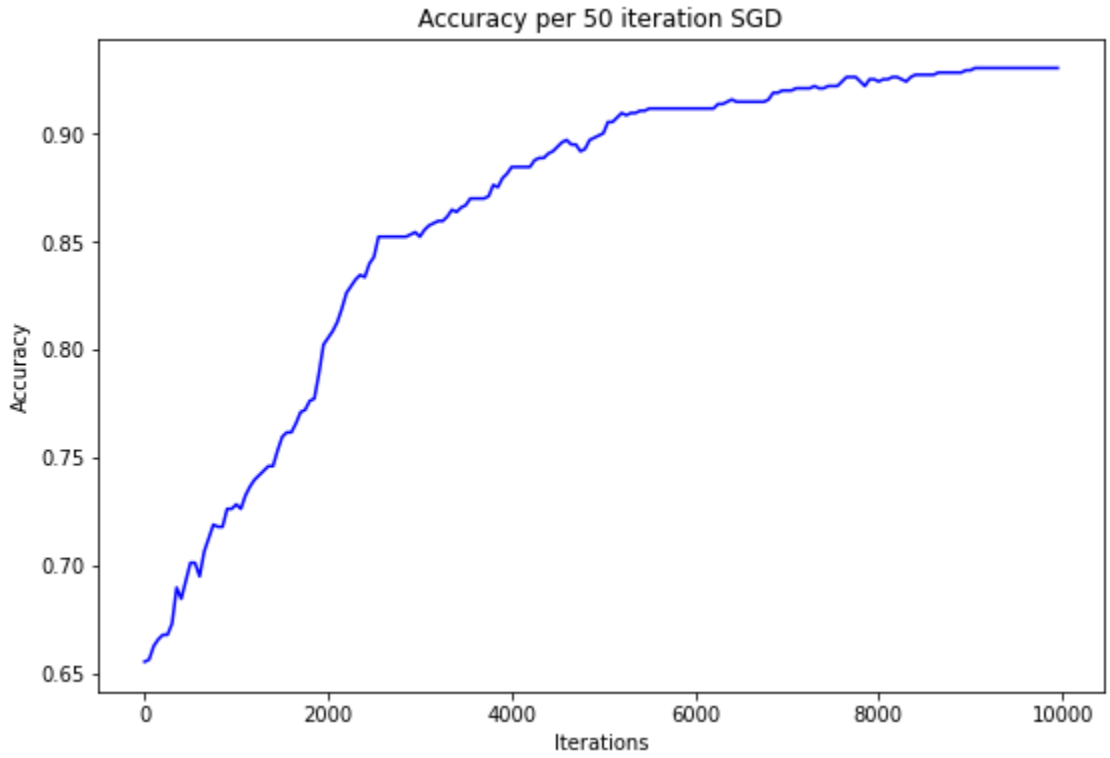
SGD with 0.001 learning rate-



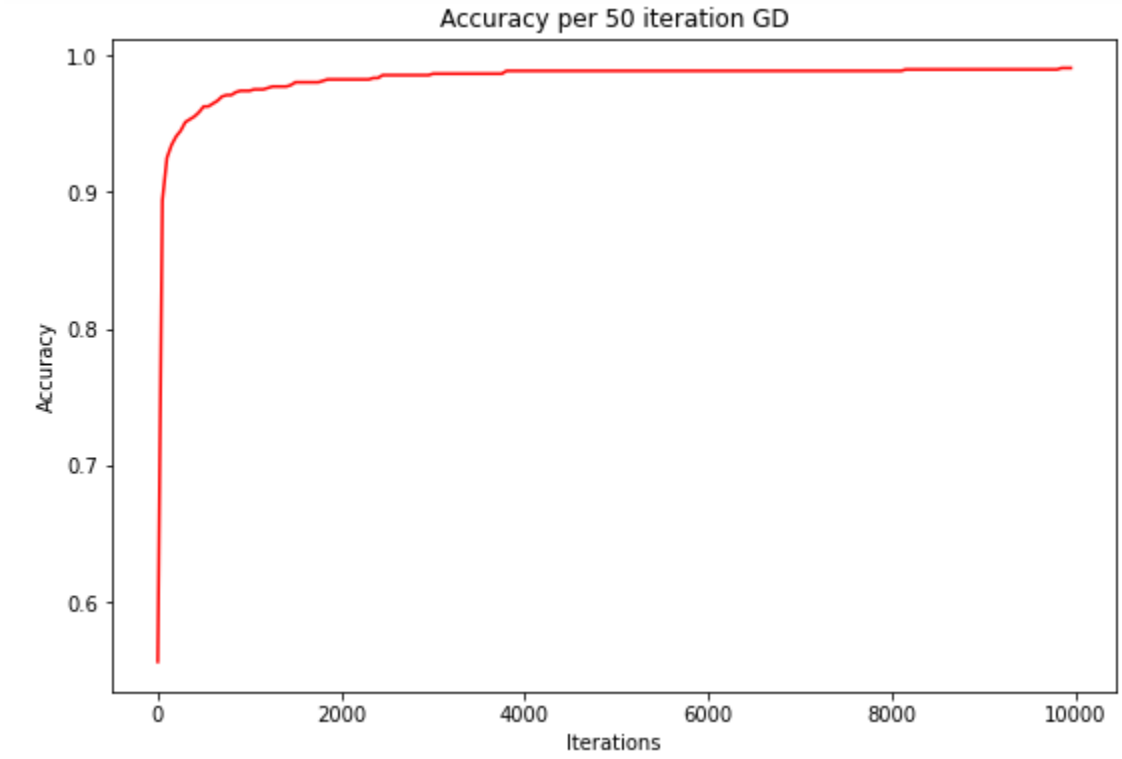
GD with 0.0001 learning rate-



SGD with 0.0001 learning rate-



GD with 0.00001 learning rate-



SGD with 0.00001 learning rate-