CS 725: Foundations of Machine Learning Homework 1

Soumen Kumar Mondal
23m2157@iitb.ac.in

August 21, 2023

Abstract

In this assignment, we will implement logistic regression model on a toy datasets for binary classification and linear classification model on a toy datasets for multi class classification. Our key goal in this assignment is to correctly implement these models and analyze the results we obtain.

1 Logistic Regression

1.1 Learning Rate

The learning rate is a crucial hyper-parameter in machine learning algorithms, especially in gradient-based optimization methods like gradient descent. It determines the step size at which the algorithm updates the model's parameters during training. A higher learning rate can lead to faster convergence, but it might also cause overshooting and divergence. Conversely, a lower learning rate can lead to slower convergence, potentially getting stuck in local minima. Finding the right balance is essential for achieving optimal training performance and avoiding convergence issues. Experimenting with different learning rates helps identify the optimal rate that maximizes the model's accuracy and minimizes its loss during training and validation.

The impact of varying learning rates on accuracy and loss was explored for the Logistic Regression (LR) model. The outcomes of these experiments can be summarized as follows:

Training accuracy: Analyzing Figure 1 reveals that the Logistic Regression (LR) model achieves its highest accuracy when the learning rate is set to 10^{-1}. As anticipated, a higher learning rate contributes to swift convergence, requiring fewer than 100 epochs. However, the experiment's scope extends to a higher number of epochs to gauge the impact of a slower learning rate, which demands more epochs for convergence. The figure also underscores that a slower learning rate like 10^{-6} necessitates very large number of epochs for convergence whereas 1000 epochs as shown in the figure are not sufficient. The relationship between learning rate and training performance is not always linear (as evident from the figure), and it's possible to encounter situations where a smaller learning rate (e.g., $10^{-}6$) yields better training accuracy than a larger learning rate (e.g., $10^{-}4$). This phenomenon can be attributed to the concept of the learning rate's impact on optimization.

Training loss: From Figure 2, it is evident that the loss is minimum and achieves convergence in lowest number of epochs when the learning rate is set to 10^{-1}. As expected, a lower learning rate will require more number of epochs to achieve minimum loss.

Validation accuracy: It can be seen from Figure 3 that the validation accuracy is highest when the learning rate is set to 10^{-1}. For lower learning rates, the number of epoch 1000 is insufficient to achieve convergence.

Validation loss: It can be seen from Figure 4 that the validation loss is minimum when the learning rate is set to 10^{-1}. It is also evident that the number of epoch 100 is optimum in terms of loss — increasing from 100 will lead to increase in the loss.

In summary, the learning rate of 10^{-1} is optimal for this LR model as it reaches to convergence in less number of epochs without having significant impact on the accuracy and loss of the LR model.

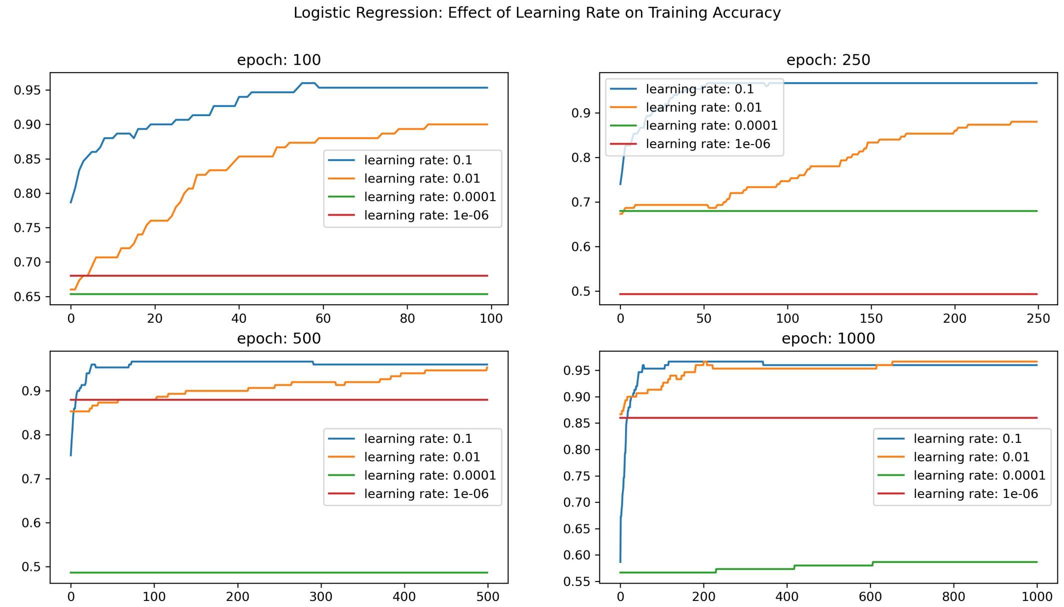

Figure 1: Effect of learning rate on training accuracy of LR model

1.2 Number of Epoch

The number of epochs in machine learning signifies the count of times the entire data set is iterated through during training. The choice of the number of epochs plays a pivotal role in model performance. Too few epochs may lead to under-fitting, where the model hasn't learned enough from the data, while too many epochs might cause over-fitting, where the model captures noise in the training data, failing to generalize well to new data. Striking the right balance by monitoring validation performance helps achieve optimal model training and prevent over-fitting or premature convergence.

The number of epoch are varied while keeping the learning rate same as 10^{-1} which is the best value of learning rate. It can be observed on Figure 5 and Figure 6 that the number of epoch 250 is optimal as it reaches convergence early and it has slightly better accuracy than the number of epoch 100. Nevertheless, the difference in accuracy and loss in both training and validation data set between epoch 100 and epoch 250 is insignificant. However, we select number of epoch 250 as best because of the uncertainty in unseen data where number of epoch 100 might not be sufficient to reach the convergence.

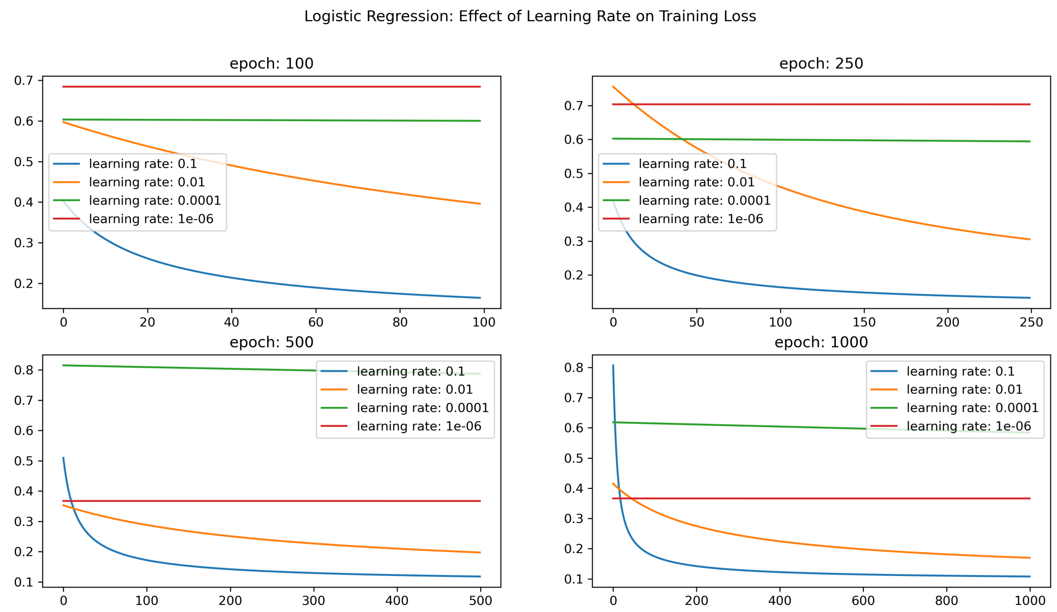

Figure 2: Effect of learning rate on training loss of LR model

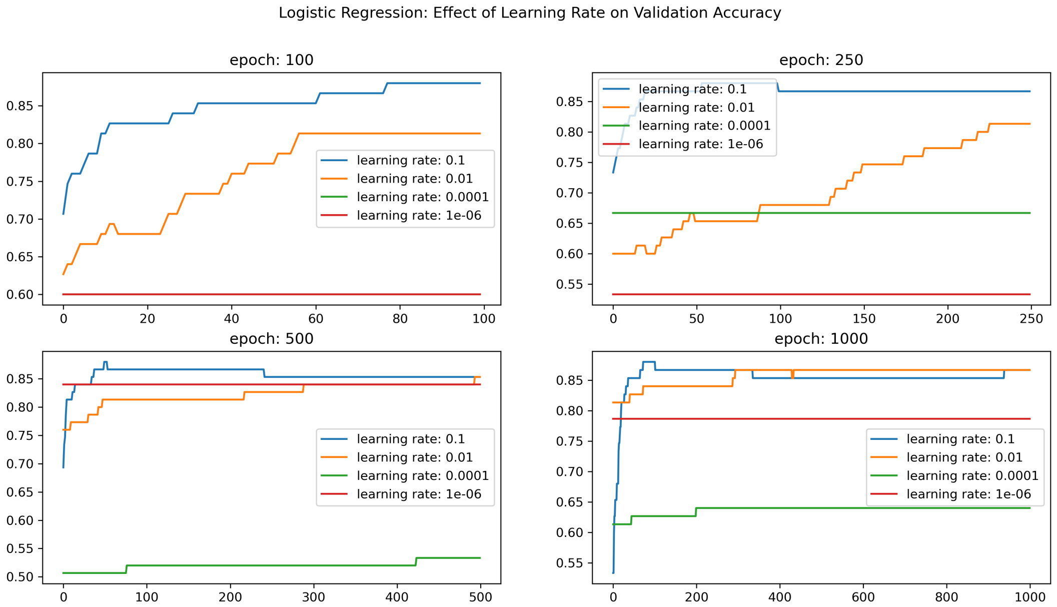

Figure 3: Effect of learning rate on validation accuracy of LR model

1.3 Momentum

Momentum is a critical concept in optimization algorithms that enhances the convergence of machine learning models. By introducing a notion of momentum, the optimization process gains the ability to overcome local minima and accelerate convergence towards optimal solutions. In the context of a Logistic Regression (LR) model, momentum affects the parameter updates during training. The equation $v = \mu v - \lambda \nabla L(w)$ showcases how the gradient descent direction is influenced by the accumulated momentum term (v) and the learning rate (λ). The updated

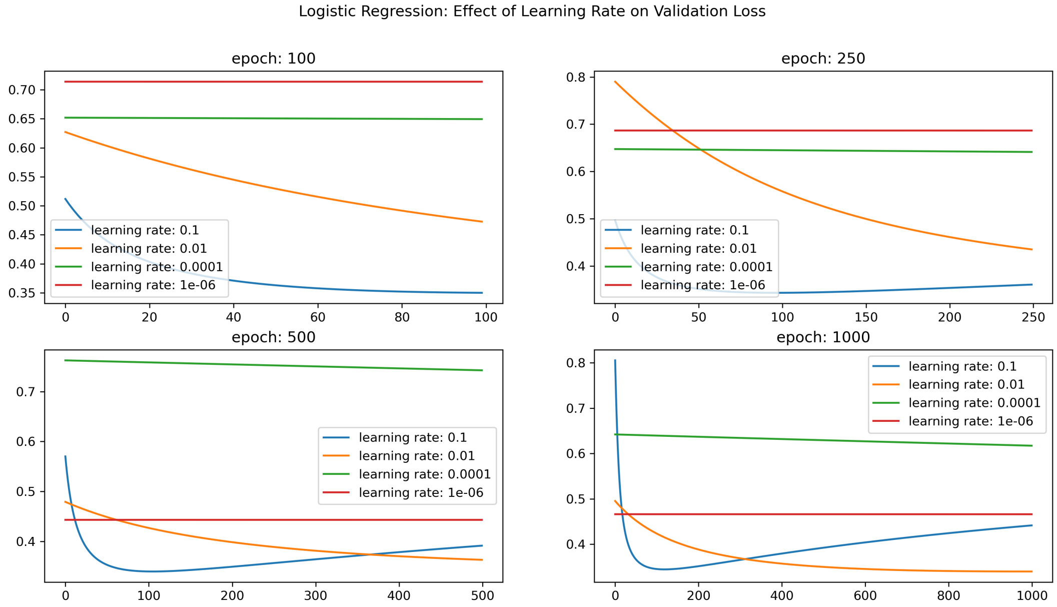

Figure 4: Effect of learning rate on validation loss of LR model

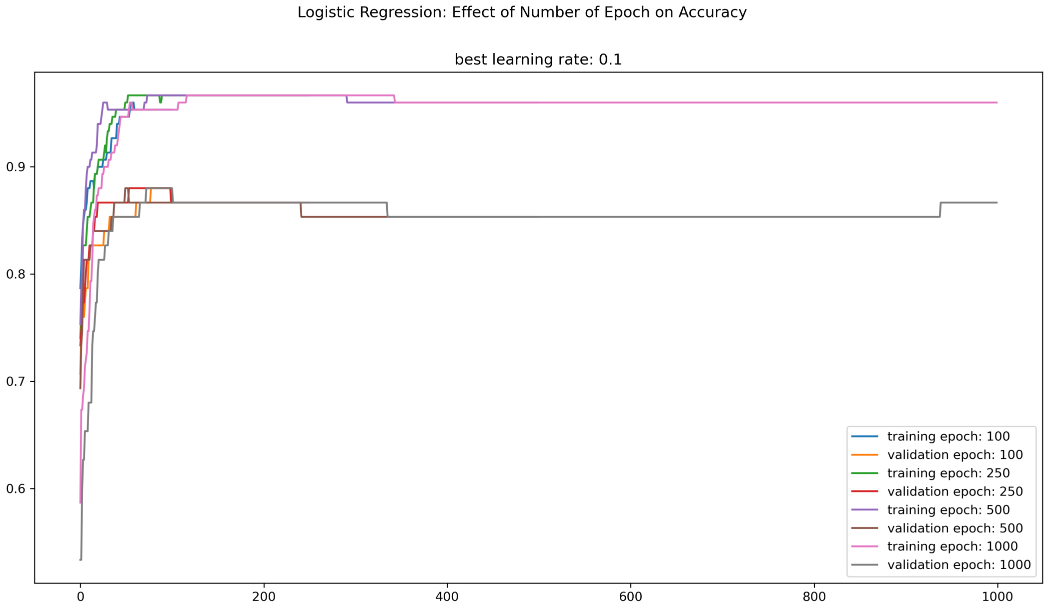

Figure 5: Effect of number of epoch on accuracy of LR model

weight parameters ($w = w + v$) then reflect the combined effect of both momentum and learning rate, effectively steering the model towards better convergence paths.

The momentum values are varied while keeping the learning rate same as 10^{-1} which is the best value of learning rate and keeping the number of epoch same as 250 which is the best value of epoch. It can be observed on Figure 7 and Figure 8 that the momentum value of 0.9 helped to achieve the convergence on accuracy earlier than the momentum value of 0. However, the higher momentum value has a slightly lower accuracy than the lower momentum value of 0. Similarly, the loss corresponding to higher momentum value of 0.9 is slightly higher than the

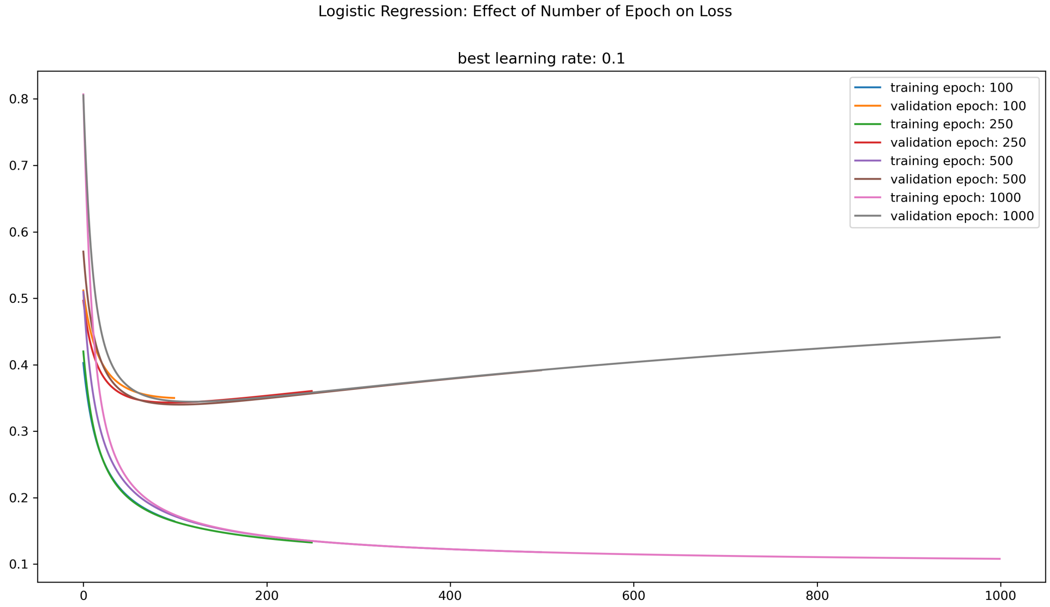

Figure 6: Effect of number of epoch on loss of LR model

loss corresponding to lower momentum value of 0. In summary, it is evident that keeping a high momentum value increases the speed of the training process but there will be a trade off with the accuracy. For this example, since the number of epoch is kept as 250 and there is no problem of slow convergence, we give more weight to the accuracy over speed. Therefore, we choose momentum value of 0 as the best value of momentum.

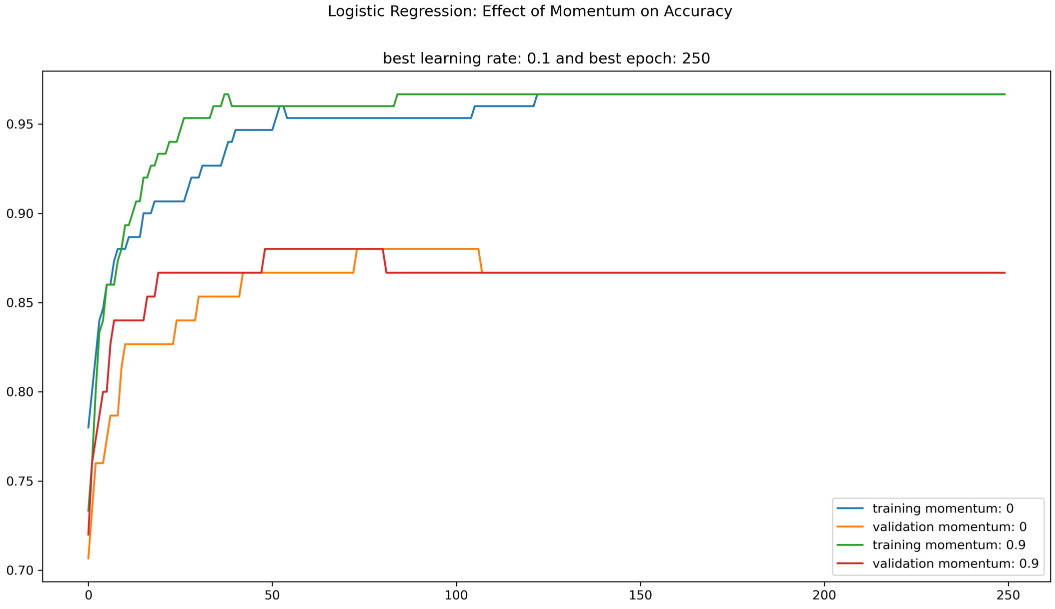

Figure 7: Effect of momentum on accuracy of LR model

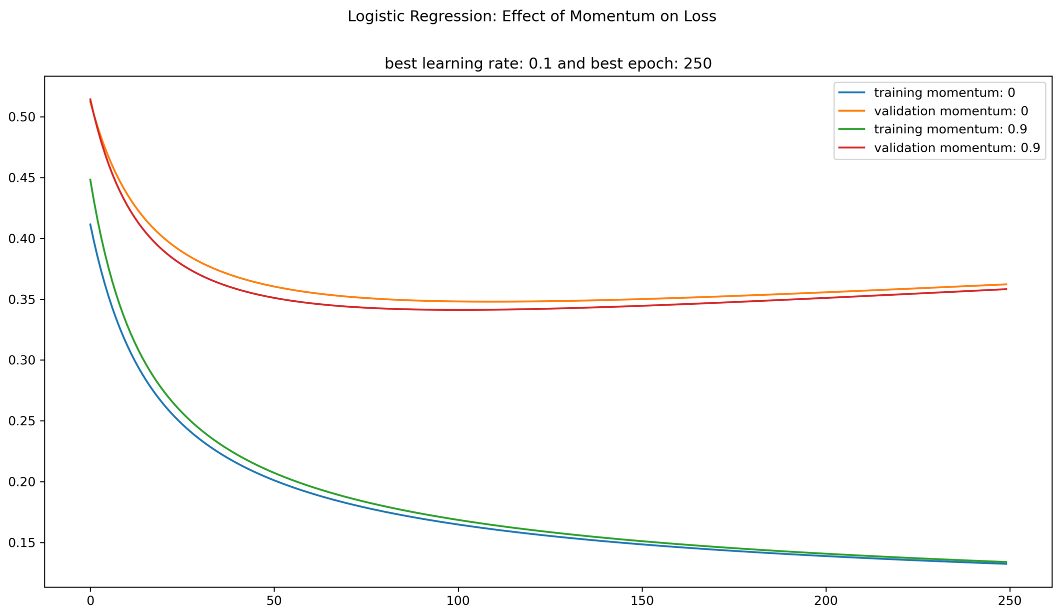

Figure 8: Effect of momentum on loss of LR model

1.4 Initial Weights

Choosing initial weights properly is crucial as it significantly affects the optimization process and the final performance of a machine learning model. Poor initial weights can lead to slow convergence, getting stuck in local minima, or even divergence. Well-chosen initial weights provide the model with a favorable starting point, enabling faster convergence and better chances of finding globally optimal solutions. Appropriate initialization strategies help enhance training stability, prevent vanishing or exploding gradients, and improve the overall efficiency of the learning process.

Random initialization assigns different starting points to each weight, which breaks symmetry and introduces diversity in the learning process. This diversification aids in capturing distinct features and patterns in the data. Moreover, random initialization prevents the model from converging to the same values, which enhances the model's capacity to learn complex relationships and generalize better to new data. Overall, random initialization promotes effective training by introducing variety and reducing the risk of symmetrical weight updates.

In this example, since there were only 2 input features, we could have initialized the weights manually by plotting the data in 2D plane. One of such manually selected weight could be $[w_0 = 0, w_1 = 1, w_2 = 0]$ since the decision boundary equation would be $x_1 = 0$ for the given training example. However, to keep the model generalized that works on any type of training example (in case we decide to train the model on different set of training data), we select the initial weights by random initialization as described above in this section.

1.5 Gradient of Loss Function

The scoring function for logistic regression is typically represented as the linear combination of the input features weighted by the model's learned parameters. Given a set of input features $\bar{x} = (x_1, x_2, \ldots, x_d)$ and the corresponding weights $\bar{w} = (w_1, w_2, \ldots, w_d)$, the scoring function z

can be expressed as:

$$z = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_d \cdot x_d \tag{1}$$

Where w_0 is the bias term.

This scoring function computes a linear combination of the input features, incorporating the model's learned weights and a bias term. The output of this scoring function is then used to compute the probability of the positive class in binary logistic regression using the sigmoid function. The sigmoid function is a key component in logistic regression and is used to transform the output of the scoring function into a probability value between 0 and 1. The sigmoid function is defined as:

$$f(\bar{x}; \bar{w}, w_0) = \frac{1}{1 + e^{-(\bar{w}\cdot\bar{x} + w_0)}} \tag{2}$$

In logistic regression, the common loss function used is the log loss or cross-entropy loss. The loss function for a single training example is defined as:

$$L(f(\bar{x}^i; \bar{w}, w_0), y^i) = -y^i \cdot \log(f(\bar{x}^i; \bar{w}, w_0)) - (1 - y^i) \cdot \log(1 - f(\bar{x}^i; \bar{w}, w_0)) \tag{3}$$

Where y^i is the true label of the i^{th} input data point.

Then the average of the total loss over the entire training dataset is given as:

$$J(\bar{w}, w_0) = -\frac{1}{N} \sum_{i=1}^{N} \left[L(f(\bar{x}^i; \bar{w}, w_0), y^i) \right] \tag{4}$$

Where $L(f(\bar{x}^i; \bar{w}, w_0), y^i)$ is substitute from Equation 3. N is the total number of training data points. $J(\bar{w}, w_0)$ is sometimes called the average loss function or cost function.

Three useful results that would be required to derive the gradient of the loss function are given as follows:

$$f'(z) = f(z) \cdot (1 - f(z)) \tag{5}$$

$$\frac{\partial J(\bar{w}, w_0)}{\partial \bar{w}} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial \bar{w}} \tag{6}$$

$$\frac{\partial J(\bar{w}, w_0)}{\partial w_0} = \frac{\partial J}{\partial z} \cdot \frac{\partial z}{\partial w_0} \tag{7}$$

Where $f(z)$ is the standard sigmoid function defined as per Equation 2 and J is the loss function which is differentiated by following the chain rule.

Using the above three equations, the gradient of the loss function can be derived as follows:

$$J(\bar{w}, w_0) = -\frac{1}{N} \sum_{i=1}^{N} \left[y^i \cdot \log(f(z)) + (1 - y^i) \cdot \log(1 - f(z)) \right] \tag{8}$$

$$\frac{\partial J}{\partial z} = -\frac{1}{N} \sum_{i=1}^{N} \left[y^i \cdot \frac{f'(z)}{f(z)} + (1 - y^i) \cdot \frac{-f'(z)}{1 - f(z)} \right] \tag{9}$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \left[y^i \cdot (1 - f(z)) - (1 - y^i) \cdot f(z) \right] \qquad \text{(From Equation 5)} \tag{10}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \left[f(z) - y^i \right] \tag{11}$$

Applying the chain rule, the differentiation can be written as:

$$\frac{\partial J(\bar{w}, w_0)}{\partial \bar{w}} = -\frac{1}{N}\sum_{i=1}^{N}\left[f(\bar{x}^i; \bar{w}, w_0) - y^i\right] \times \left[\bar{x}^i\right] \tag{12}$$

$$\frac{\partial J(\bar{w}, w_0)}{\partial w_0} = -\frac{1}{N}\sum_{i=1}^{N}\left[f(\bar{x}^i; \bar{w}, w_0) - y^i\right] \times [1] \tag{13}$$

Where $z = \bar{w} \cdot \bar{x} + w_0$

2 Linear Classifier

2.1 Learning Rate

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

2.2 Number of Epoch

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

2.3 Momentum

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.