FML : Assignment 01 Report

Department of Computer Science

Indian Institute of Technology Bombay

Guided by

Prof. Sunita Sarawagi

Foundation of Machine Learning

CS 725

Prepared By

Bharat Patidar

23M0761

Rohit Singh Yadav

23M0773

Report :

In this Exercise/Assignment we have learnt the training of Logistic Regression Model and Linear Classification Model.

The optimal value of accuracy depends on the following three parameters.

1. Learning Rate

2. Epochs

3. Momentum

The below table states the different parameter value for optimal accuracy :

	Logistic Regression	Linear Classification
Learning Rate	0.1	0.1
Epochs	1000	250
Momentum	0.9	0
Accuracy(%)	85.33%	96%
Loss	4.35	7.65

Report of Logistic Regression

In the Logistic Regression Model the Dataset is taken from https://github.com/ashutoshbsathe/cs725-hw . In this model the binary dataset is used where the number of classes were 2 also the number of features of dataset is also 2.

We have implemented functions : calculate_loss(), calculate_gradient(), update_weights() and get_prediction() functions. Also implemeted preprocess() and sigmoid() functions for model training purpose.

In this report we have to analyse the effect of the three parameters on the model's accuracy and loss. As the parameters are :

1. Learning Rate
2. Epochs
3. Momentum

Effect of Learning Rate on Loss and Accuracy :

As the Learning rate of the model increases the step size or the gradient loss increases hence , the rate of change of learning rate directly affects the accuracy . Also the epochs is also a important parameter while finding the effeicient value of learning rate.

As in the calculation we have taken the epochs as 100 initially for calculating learning rate.

# Epochs = 100

Momentum = 0

Learning Rate	Accuracy (%)	Loss
1e-1	82.67%	2.25
1e-2	77.33%	0.91
1e-4	77.33%	0.69
1e-6	77.33%	0.69

Conclusion :

As the Accuracy is most optimal when we use Learning Rate = 1e-1 = 0.1 . As we decrease the size of the Learning Rate alpha the accuracy decreases . But with the decrease in learning rate the loss also decreases.

Hence , the most efficient Learning Rate is 0.1 as it gives the accuracy 82.67%.

Effect of # of Epochs on Loss and Accuracy :

As the Epochs of the model increases the number of iterations increases . hence , the rate of change of Epochs directly affects the accuracy . # of Epochs states that how much time the model will search for the minima or in search of valley the model will how much effort. Also the epochs is also a important parameter while finding the effeicient value of learning rate.

As in the calculation we have taken the Learning Rate as .0.1 for calculating # of Epochs.

Learning Rate = .0.1

Momentum = 0

# of Epochs	Accuracy (%)	Loss
100	82.67%	2.25
250	84%	3.03
500	84%	3.66
1000	85.33%	4.35

Conclusion :

As the Accuracy is most optimal when we use Learning Rate = 1e-1 = 0.1 and use the # of Epochs = 1000 . As we increase the number of the Epochs the accuracy increases . But the increase in the Epochs causes the increase in loss.

Hence , the most efficient Learning Rate is 0.1 and Number of Epochs is 1000.

Effect of Momentum on Loss and Accuracy :

As the Momentum states that gradients accumulated from past iterations will push the cost further to move around a saddle point even when the current gradient is negligible or zero. It is better when we use gradient descent .

As in the calculation we have taken the Learning Rate as 0.1 And # of Epochs 1000 for calculating # of Epochs.

Learning Rate = 0.1

# of Epochs = 1000

Momentum	Accuracy(%)	Loss
0	85.33%	4.35
0.9	85.33%	4.35

Conclusion :

As the Accuracy is most optimal when we use Learning Rate = 1e-1 = 0.1 and use the # of Epochs = 1000 . As we increase the momentum the accuracy increases . But the increase in the Momentum causes the increase in loss.

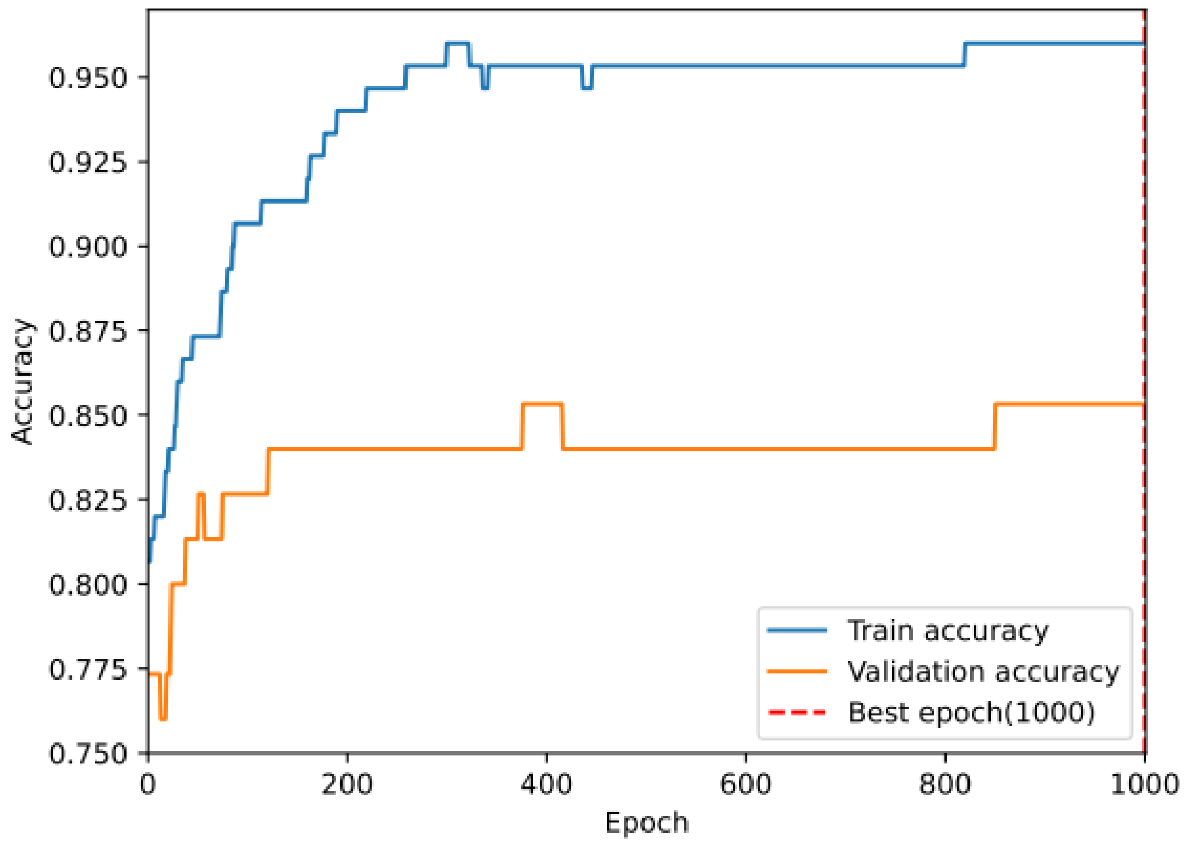
Hence , the most efficient Learning Rate is 0.1 and Number of Epochs is 1000 and Momentum is 0.9 .

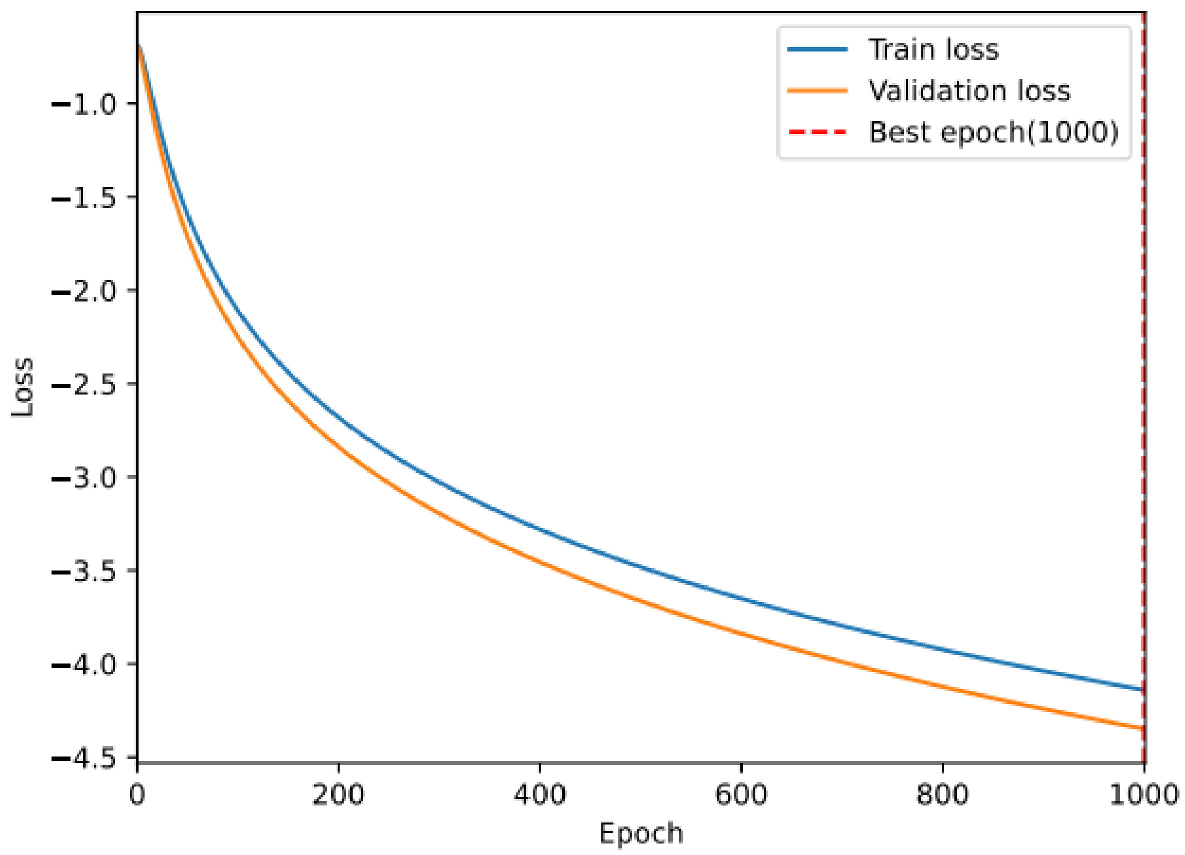
Graph :

Learning Rate = 0.1

# of Epochs = 250

Momentum = 0





Train loss
Validation loss
Best epoch(1000)
Loss
Epoch
−1.0
−1.5
−2.0
−2.5
−3.0
−3.5
−4.0
−4.5
0
200
400
600
800
1000

Report of Linear Classifier

In the Linear Classifier Model the Dataset is taken from https://github.com/ashutoshbsathe/cs725-hw . In this model the Iris dataset is used where the number of classes were 3 also the number of features of dataset is also 4.

We have implemented functions : calculate_loss(), calculate_gradient(), update_weights() and get_prediction() functions. Also implemeted preprocess() and sigmoid() functions for model training purpose.

In this report we have to analyse the effect of the three parameters on the model's accuracy and loss. As the parameters are :

1. Learning Rate
2. Epochs
3. Momentum

Effect of Learning Rate on Loss and Accuracy :

As the Learning rate of the model increases the step size or the gradient loss increases hence , the rate of change of learning rate directly affects the accuracy . Also the epochs is also a important parameter while finding the effeicient value of learning rate.

As in the calculation we have taken the epochs as 100 initially for calculating learning rate.

# Epochs = 100

Momentum = 0

Learning Rate	Accuracy (%)	Loss
1e-1	96%	6.23
1e-2	80%	4.10
1e-4	40%	2.56
1e-6	40%	2.41

Conclusion :

As the Accuracy is most optimal when we use Learning Rate = 1e-1 = 0.1 . As we decrease the size of the Learning Rate alpha the accuracy decreases . But with the decrease in learning rate the loss also decreases.

Hence , the most efficient Learning Rate is **0.1**

Effect of # of Epochs on Loss and Accuracy :

As the Epochs of the model increases the number of iterations increases . hence , the rate of change of Epochs directly affects the accuracy . # of Epochs states that how much time the model will search for the minima or in search of valley the model will how much effort. Also the epochs is also a important parameter while finding the effeicient value of learning rate.

As in the calculation we have taken the Learning Rate as 0.1 for calculating # of Epochs.

Learning Rate = 0.1

Momentum = 0

# of Epochs	Accuracy (%)	Loss
100	96%	6.23
250	100%	7.65
500	100%	11.23
1000	100%	13.26

Conclusion :

As the Accuracy is most optimal when we use Learning Rate = 1e-1 = 0.1 and use the # of Epochs = 250 . As we increase the number of the Epochs the accuracy increases . But after a thresold value the accuracy remains constant. But the increase in the Epochs causes the increase in loss.

Hence , the most efficient Learning Rate is 0.1 and Number of Epochs is 250.

Effect of Momentum on Loss and Accuracy :

As the Momentum states that gradients accumulated from past iterations will push the cost further to move around a saddle point even when the current gradient is negligible or zero. It is better when we use gradient descent .

As in the calculation we have taken the Learning Rate as 0.1 And # of Epochs 250 for calculating Momentum.

Learning Rate = 0.1

# of Epochs = 250

Momentum	Accuracy(%)	Loss
0	100%	7.65
0.9	100%	13.53

Conclusion :

As the Accuracy is most optimal when we use Learning Rate = 1e-1 = 0.1 and use the # of Epochs = 250 . As we increase the momentum the accuracy increases . But the increase in the Momentum causes the increase in loss.

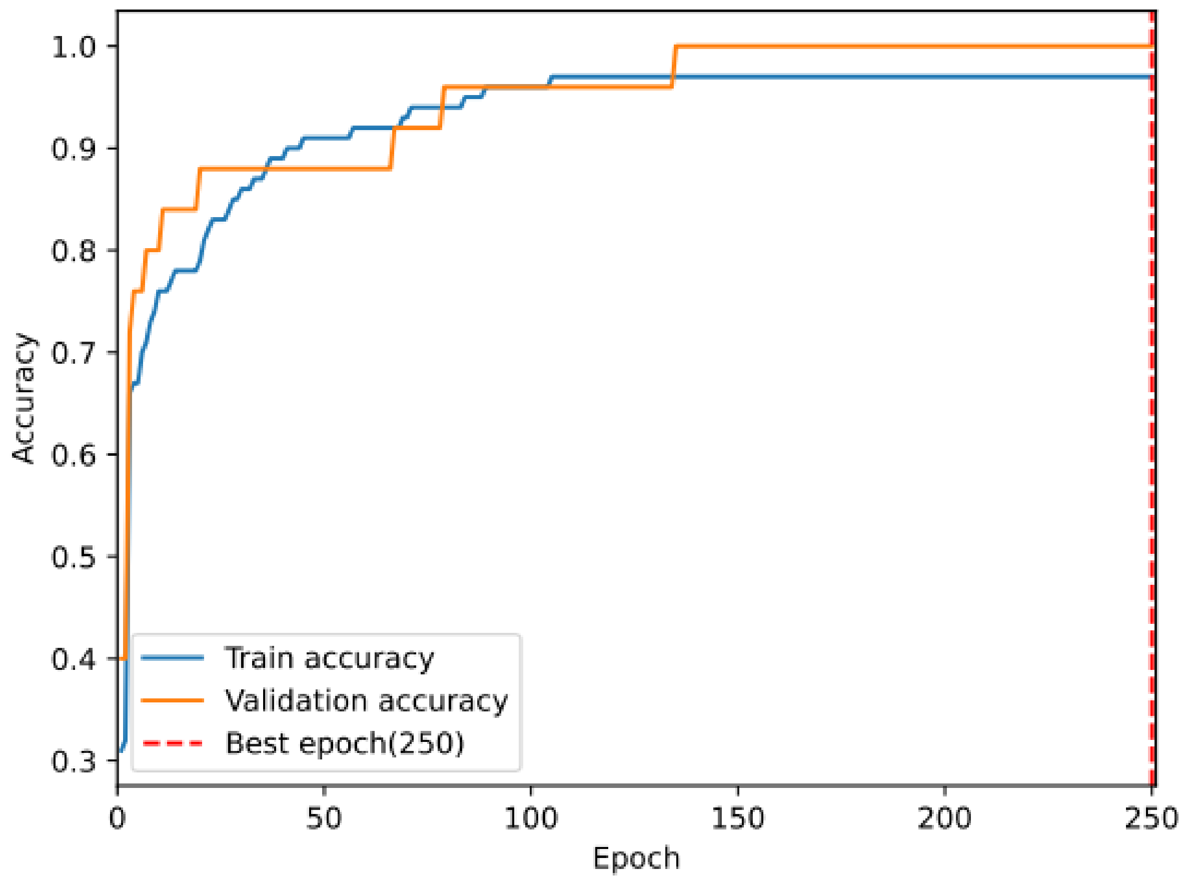
Hence , the most efficient Learning Rate is 0.1 and Number of Epochs is 250 and Momentum is 0.

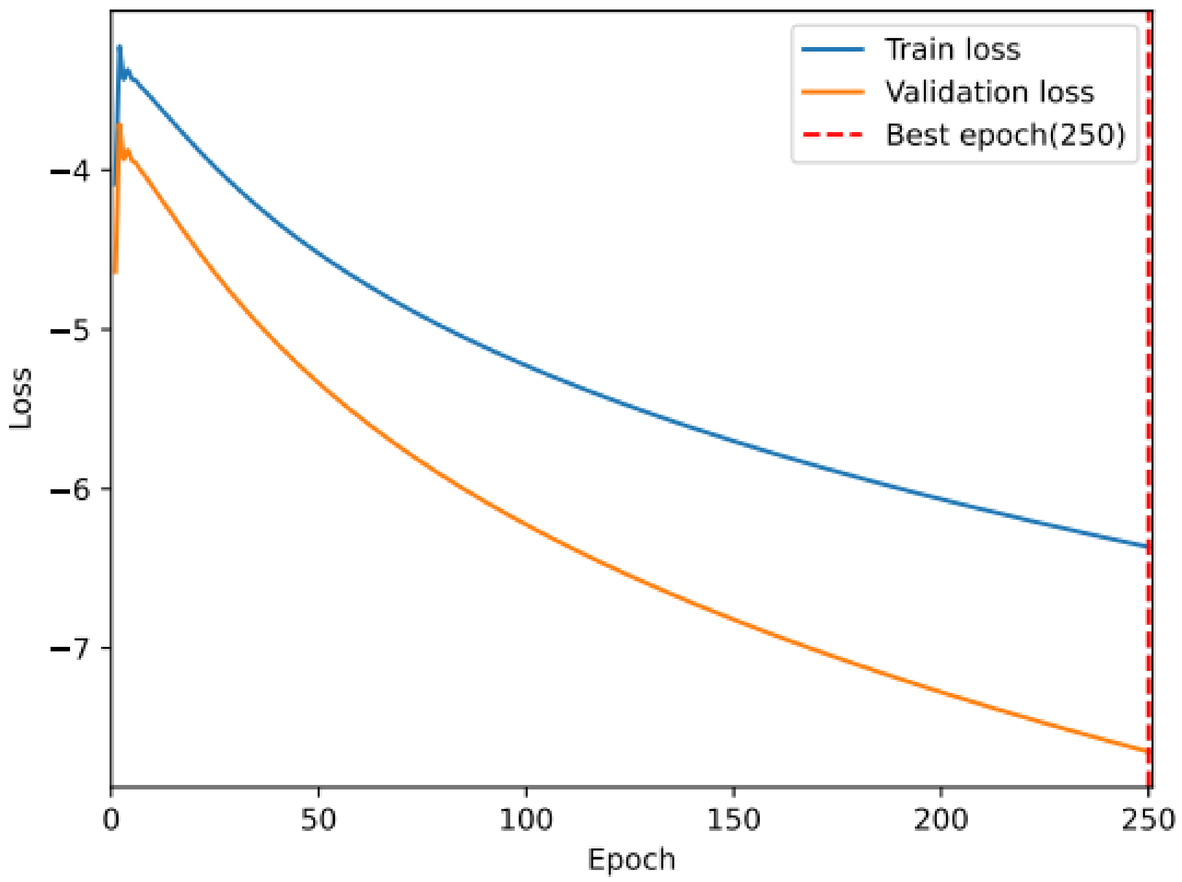
Graph :

Learning Rate = 0.1

# of Epochs = 250

Momentum = 0




Train loss
Validation loss
Best epoch(250)
Loss
Epoch
−4
−5
−6
−7
0
50
100
150
200
250

1) **Binary Cross-Entropy Loss for Logistic Regression:**

The binary cross-entropy loss for logistic regression is defined as:

$$\text{loss} = - (1/n) * \Sigma [y * \log(g(z)) + (1 - y) * \log(1 - g(z))]$$

w represents the parameters (coefficients) of the model.

n :number of data points.

y :true binary label (0 or 1).

z :combination of features and parameters: $z = w^T * x$

sigmoid function: $g(z) = 1 / (1 + \exp(-z))$

Derivation of the Gradient:

derive the gradient of sigmoid function g(z) with respect to z:

$$d(g(z))/dz = \sigma(z) * (1 - g(z))$$

---->The derivative of the loss function with respect to w for a single data point (x, y):

$$d(\text{loss}(\theta))/d\theta = - [y * (1/g(z)) * d(g(z))/dz * x + (1 - y) * (-1/(1 - g(z))) * d(g(z))/dz * x]$$

$$= - [y * (1 - g(z)) * x + (1 - y) * g(z) * (-x)]$$

$$= - [y * x - y * g(z) * x - g(z) * x + y * g(z) * x]$$

$$= - [y - g(z)] * x$$

gradient for data set is average of individual

$$\nabla L(w) = - (1/n) * \Sigma [(y - g(z)) * x]$$

2) Study on iris. How to adopt logistic regression to multi-class settings. Ans) The Iris flower data set or Fisher Iris data set is a multivariate data set used. It is sometimes called Anderson's Iris data set because Edgar Anderson collected the data. Let's go through an explanation of how the Iris dataset can be used for machine learning:

1. Dataset Overview: The Iris dataset consists of measurements from three species of iris flowers: Setosa, Versicolor, and Virginica. For each species of, four attributes are recorded.

• Sepal length in cm

• Sepal width in cm

• Petal length in cm

• Petal width in cm

• Class / Species

- Iris Setosa

- Iris Versicolor

- Iris Virginica

The goal is to build a machine learning model that can classify iris flowers based on these four features. Iris Virginica Iris Versicolor Iris Setosa

2. **Loading and Exploring Data:**

Put the dataset into a scripting environment of your choice (like Python). Examine the dataset to discover its elements, such as the quantity of features, classifications, and samples. Use scatter plots or other approaches to visualize the data in order to see the relationships between the features and classes.

3. dividing the data: Take the dataset and divide it into a training set and a testing set. The most common split ratios are 70-30 or 80-20, where the larger portion is used for testing and the smaller piece for training across classes & features.

4. Selecting an Algorithm for Machine Learning: Several methods and algorithms, such as logistic regression, discriminant analysis with linearity, classification trees, the use of support vector machines, and many others, can be used in supervised learning to predict an outcome from a data set. In some nearest neighbor classification methods, a class can be given to an observation based on how closely it resembles another observation.

5. Training the Model: With all 4 features and our techniques, we will train the model after dividing the dataset into testing data and training data. Using the training data, test the selected algorithm to see how well it performs.

6. Using the trained model, make predictions about the testing set. Compare the predicted class labels with the true class labels to assess the model's performance. The confusion matrix, F1-score, recall, accuracy, and precision are frequently used evaluation measures for categorization.

7. Conclusion: We learned how to develop our own supervised machine learning model by using the Iris Flower Classification Project using Machine Learning. This research taught us new things about model building, data analysis, data visualization, and other subjects. Using "One-vs-Rest" or "Softmax" allows logistic regression to be applied to a multi-class context.

● One-vs-Rest, occasionally referred to as One-vs-All, is a heuristic technique for multi-class classification utilizing binary classification methods.It is possible to divide a multiclass data set into many binary classification issues. The best accurate model is then utilized to generate predictions after a binary classifier is trained for each binary classification task. For instance: 'Iris Setosa', 'Iris Versicolor', & 'Iris Virginica' each have a multi-class grouping issue example. The following three binary classification datasets might be created from this: Classification in Binary