CSE 574 Project 1: Two class Classification using Logistic Regression

Bhagyashri Thorat (Person ID: 50290581)
Email: bthorat@buffalo.edu

25th September 2019

Abstract

A patient can be classified as having malignant or benign cancer. The goal of this project was to use 2 class classification using machine learning and predict the cancer type based on 10 features as either malignant or benign. Logistic regression was used to train the data. The dataset used was of Wisconsin Diagnostic Breast cancer.

1 Introduction

Machine learning is a subset of Artificial Intelligence. It's used to model the computer systems based on specific algorithms to perform a specific task based on the previous results and errors.

The dataset was further divided into training (80%), validation (10%), testing (10%). The model was trained on this training data and then accuracy, recall, and precision were calculated for validation data and training data using logistic regression. The loss function used was gradient descent.

1.1 Linear Regression

The Linear Regression model is trained using gradient descent function. And the prediction is done using the sigmoid function. The linear regression formula used for this problem is of the form:

$$y = f(w^T(x) + bias)$$

where (x) is a Gaussian radial basis function.
And bias is intercept added as shown in the figure below.

The hyperparameters are:

- Number of basis function or epoch
- Learning rate

Deciding the number of basis functions to use is one of the hyperparameters for both the dataset. The weights are initialized to a random number.

1.2 Methodology

The steps used to predict the output are as follows:

1. Read data file
2. Preprocess the data
3. Normalize the data
4. Partition the data into training, validation, and testing
5. Initialization of the weights, bias and learning rate
6. Train the model

7. Predict the output for validation and testing data

2 Dataset

In this project, the dataset used was of Wisconsin Diagnostic Breast cancer and it had 569 instances and 32 attributes including the ID and the target values. The 30 features were generated by calculating the mean, standard error and worst of the 10 features computed for a digitized image of a fine needle aspirate (FNA) of breast mass

3 Preprocessing

In the preprocessing step, the data is converted from raw data into clean data to achieve better results. The following steps were performed in the preprocessing step:

1. **Dropping unnecessary columns:** As the data contained 32 columns, containing ID, target and then 30 features, the ID column was dropped as it wasn't necessary to either train or test the output.
2. **Mapping of target values:** Then, as the linear regression predicts output as either 1 or 0 depending on the sigmoid function, the second column containing the target variables Malignant(M) and benign(B) were mapped to
   - M as 0 and
   - B as 1

   I used the .replace function to map the values as follows:

```
data = data.replace({'M':0,'B':1})
```

3. **Normalization:** In this step, the feature values were normalized. That is, all the values were rescaled to a common scale of 0 and 1. The formula for normalization is:

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

   Or else, there is sklearn.preprocessing.normalize(X, norm, axis =1, copy, return_norm) function available which gives us normalized data: I have used this function as follows:

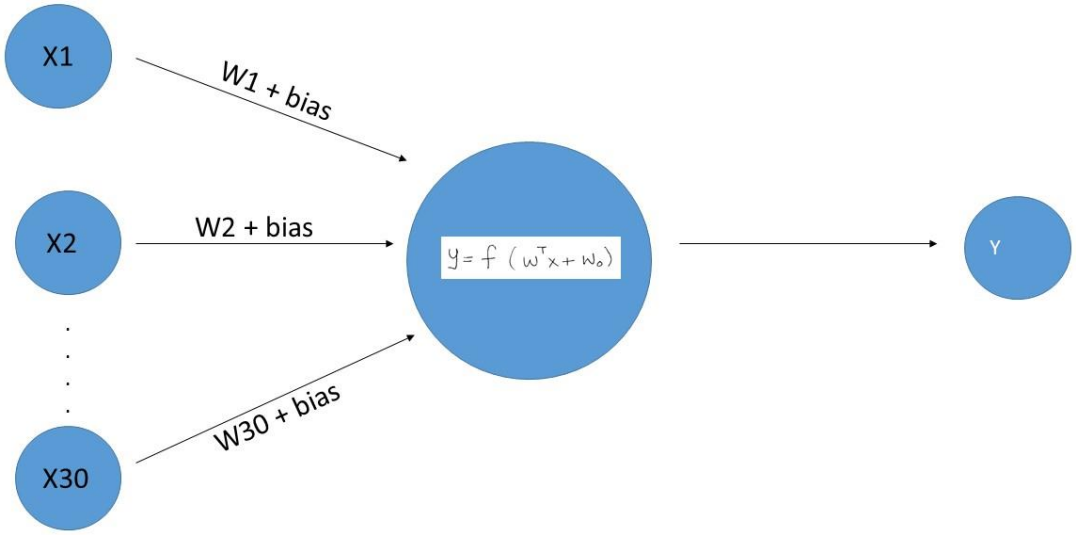
```
features = preprocessing.normalize(features)
```

4. **Other:** In addition, as data did not contain any null values, no further procession was done

4 Architecture

In this project,

Computational graph:



Where bias is W_0

Formulae:

$$z = \theta^T x$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

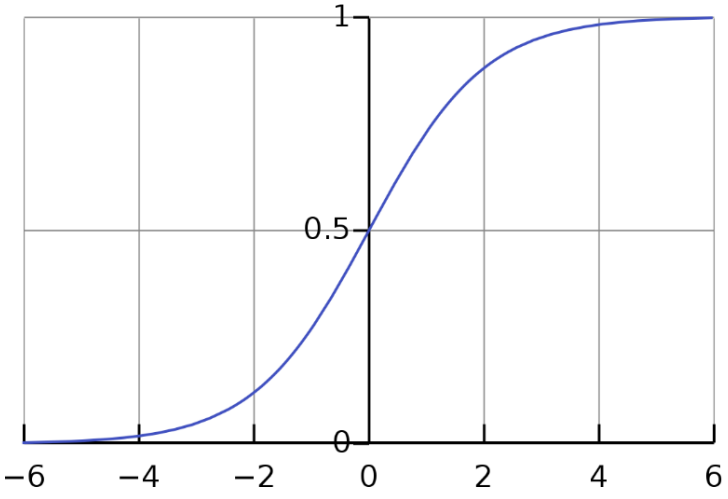


Fig. 2 — Logistic function

$$h = g(X\theta)$$

$$J(\theta) = \frac{1}{m} \cdot \left(-y^T \log(h) - (1 - y)^T \log(1 - h)\right)$$

Fig. 4— Loss function

$$\frac{\delta J(\theta)}{\delta \theta_j} = \frac{1}{m} X^T (g(X\theta) - y)$$

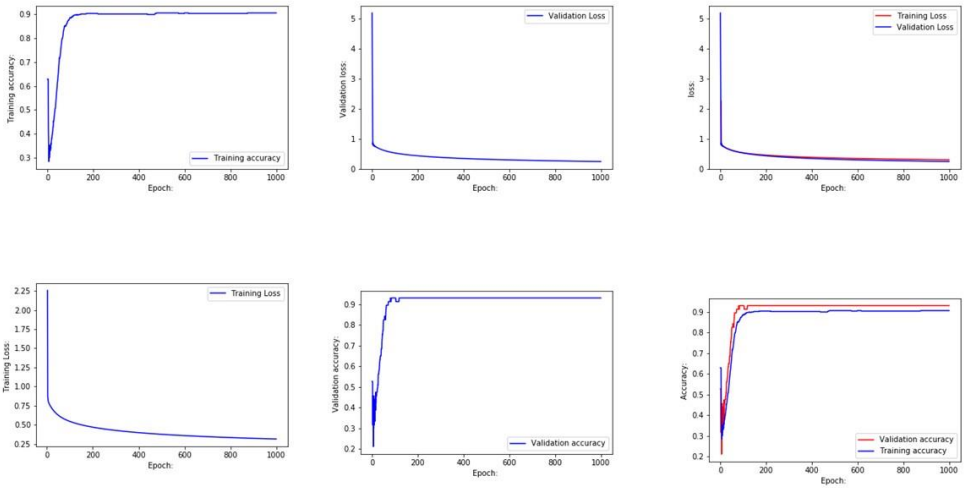
Fig. 4 — Partial derivative

5 Results

Based on the model created on the 80% part of data or training data,
the accuracy obtained for validation is:
the accuracy obtained for testing is:

1. epoch 1000
LR: 0.000005
Final accuracy for validation: 92.98245614035088
Final accuracy for Testing: 92.98245614035088
Final accuracy for Training: 90.54945054945055

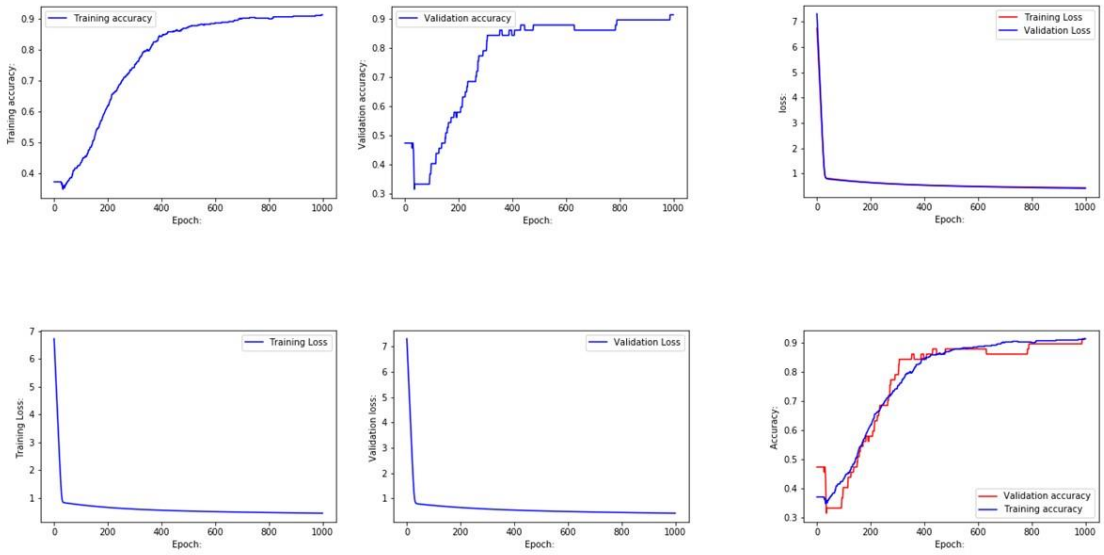
Conclusion: overfit



2.
epoch: 1000
LR: 0.000001

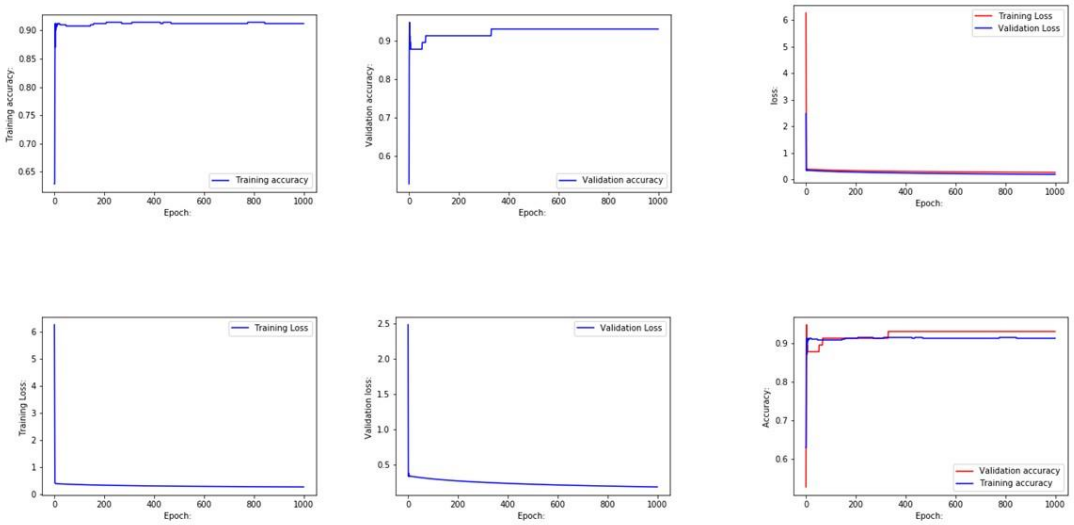
Final accuracy for validation: 92.98245614035088
Final accuracy for Testing: 91.22807017543859
Final accuracy for Training: 91.20879120879121

Conclusion: Underfit



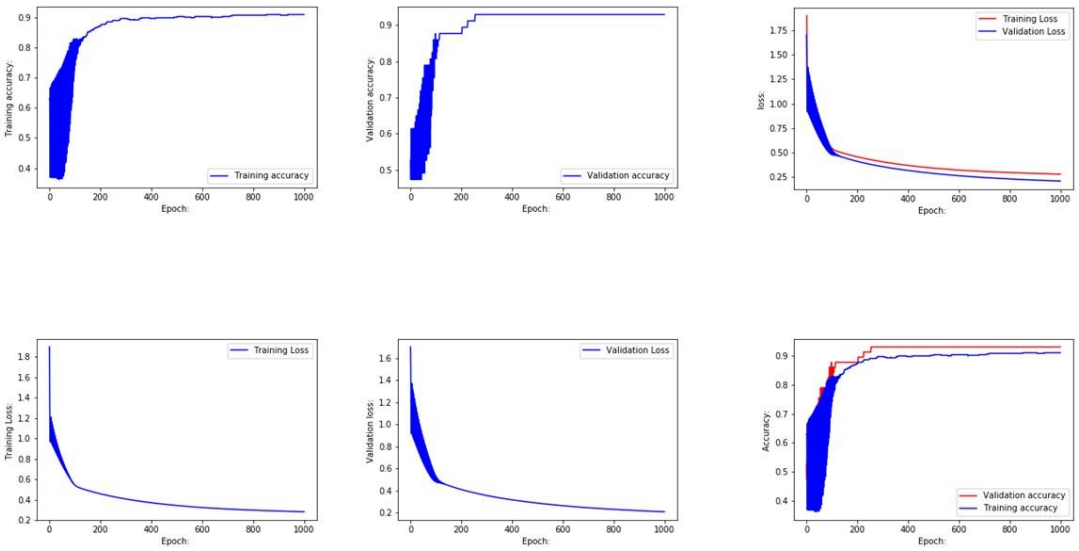
3.
epoch: 1000
LR: 0.000008
Final accuracy for validation: 98.24561403508771
Final accuracy for Testing: 92.98245614035088
Final accuracy for Training: 91.20879120879121

Conclusion: Underfit



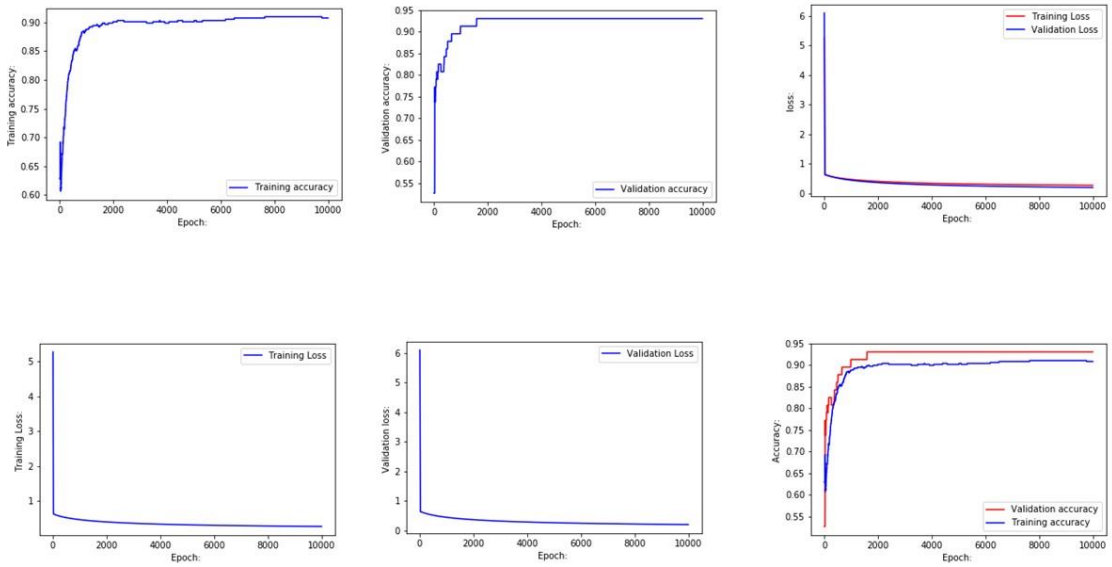
4.
epoch: 1000
LR: 0.00001
Final accuracy for validation: 96.49122807017544
Final accuracy for Testing: 92.98245614035088
Final accuracy for Training: 90.98901098901099

Conclusion: Underfit.



5.
epoch: 10000
LR: 0.01
Final accuracy for validation: 92.98245614035088
Final accuracy for Testing: 96.49122807017544
Final accuracy for Training: 90.98901098901099

Conclusion: Overfit



As we can see from the accuracy calculated that the max accuracy reached by testing data is 92%. And for training is 91%, and for validation is 98%. Also, as we can see that most of the accuracy results are underfit, we can conclude that the model is doing a good job in training the data.

6 Conclusion

Based on the results, on the features that are precomputed from images of needle aspirate (FNA) of breast mass for the Wisconsin Diagnostic breast cancer dataset, for classification using logistic regression, maximum accuracy is 92% for testing data. And 98% For the validation data. So, we can say using the results that the model is able to classify the data according to 92% accuracy. The best accuracy was observed for epoch: 1000, Learning Rate: 0.000005 with 92.98% result.

7 References:

1. https://medium.com/@martinpella/logistic-regression-from-scratch-in-python-124c5636b8ac
2. Lecture notes.