Project 1 : Logistic Regression in Python

Mohd Ehtesham Shareef
Department of Computer Science
University at Buffalo
Buffalo, NY 14214
mohdehte@buffalo.edu

Abstract

This report presents a methodology to diagnosing breast cancer based on a set of input variables. The methodology involves building a logistic regression model which classifies between malignant and benign cases. We compare logistic regression with linear regression and define the activation function for logistic regression. The model hyperparameters are optimized by a hit and trial process which analyses the loss v/s epochs graph to decide the optimum values of the hyperparameters. After finding the optimum hyperparameters, the performance metrics are evaluated.

1 Introduction

Logistic Regression models the probabilities for classification problems with two possible outcomes. It is an extension of the linear regression model.

While the linear regression model works well for predicting continuous target variables, it fails for classification problems. This is because it can have any one of an infinite number of possible values which can be below 0 and above 1. A linear model does not output probabilities. It treats the classes as numbers instead of categorical data and tries to fit the best hyperplane which minimizes the distance between the points and the hyperplane (mean squared error). To overcome this problem, we use a logistic activation function which converts the outcome of the linear model to a probability (a value in the range of 0 to 1).

$$\mathbf{y = \sigma(w^T x + b)}$$

where y = target variable

w = weight vector

x = input variable

b = bias

σ = logistic activation function

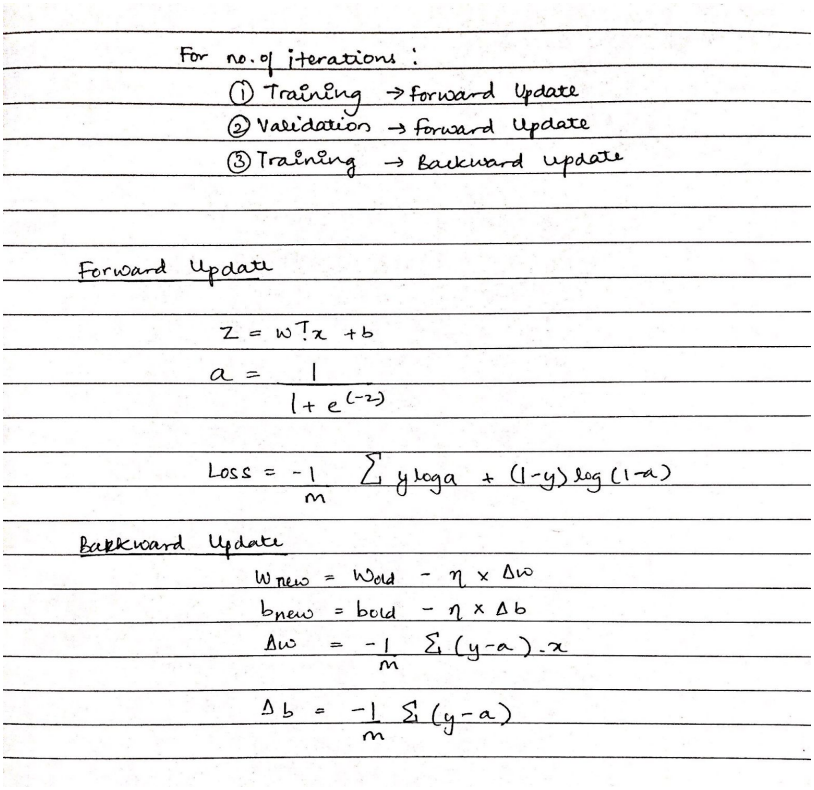

Figure 1. Equations for logistic regression

Figure 1 defines the training process for our logistic regression model. For a fixed learning rate, at every epoch (iteration) we calculate the cost function for the training and validation set and update the weights and bias so that the loss is lower than the last epoch. We stop once we reach the minima. Once the training is complete, we get the optimal weights and bias for the model.

2 Dataset

We perform our model evaluations on the Wisconsin Diagnostic Breast Cancer Database. The dataset contains 569 samples with 32 attributes. The first column of the dataset represents the PatientID and the second column is the target variable which represents the final outcome of the diagnosis. The other 30 features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. We use Pandas, which is a data analysis python library, to load the dataset and further preprocess it. We refer to the loaded dataset as a dataframe from here on.

	0	1	2	3	4	5	6	7	8	9	...	22	23	24	25	26	27	28	29	30	
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	25.38	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	24.99	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	23.57	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	14.91	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	22.54	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07

Figure 2. Dataframe with the first 5 values of the dataset

The dataset is then divided into a 90-10 ratio for the training and test data. The training set is then further divided into a 90-10 ratio to get the validation set.

3 Preprocessing

The PatientID is not a relevant feature for our model and hence, it is dropped from the dataframe. The problem at hand is a binary classification problem, and hence there can be only two output classes. The target variable in our dataset classifies the output of the diagnosis as either "Malignant" or "Benign". Logistic Regression cannot work on the labels directly and requires all the input and output variables to be numeric. For this purpose, we convert the target variable column to a categorical data type which outputs "1" when the tumour is Malignant and "0" when the tumour is benign.

Furthermore, we use a MinMaxScaler from the scikit-learn python library to normalize our data so that the data has values within 0 to 1. Upon analysing columns 5 and 6 from Figure 1, we notice that the values in column 5 are on average about 1000 times the values from column 6. These features are in very different ranges and the larger values of column 5 might influence the outcome of the model even though it might not necessarily be very important as a predictor. This is the intuition behind normalizing the dataset.

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Figure 3. Formula for normalizing the data used by MinMaxScaler

5 Architecture

Figure 4 depicts the computation graph for logistic regression. $x_1, x_2 \ldots\ldots x_{30}$ represent the 30 input variables of the data. The edges represent the weights by which the input variables are multiplied to get the hypothesis function. The hypothesis is then passed to a sigmoid function so that the output is a value between the range 0 to 1. The sigmoid function is an activation function for logistic regression which maps predictions to probabilities.

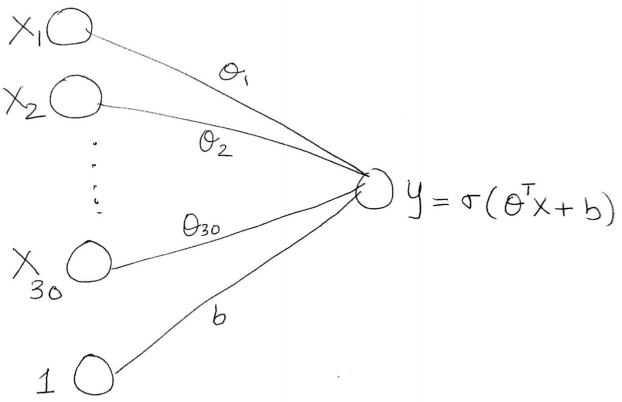

Figure 4. Computation graph for logistic regression

Figure 5 depicts the graph of a sigmoid function against a variable. In order to map the output of the sigmoid function to a discrete class, we select a threshold known as the decision boundary. For our model, the decision boundary is set to 0.5 . For any value of the sigmoid function equal to and above 0.5, the output of the sigmoid function is interpreted as "1" (Malignant) and "0" (Benign) otherwise.

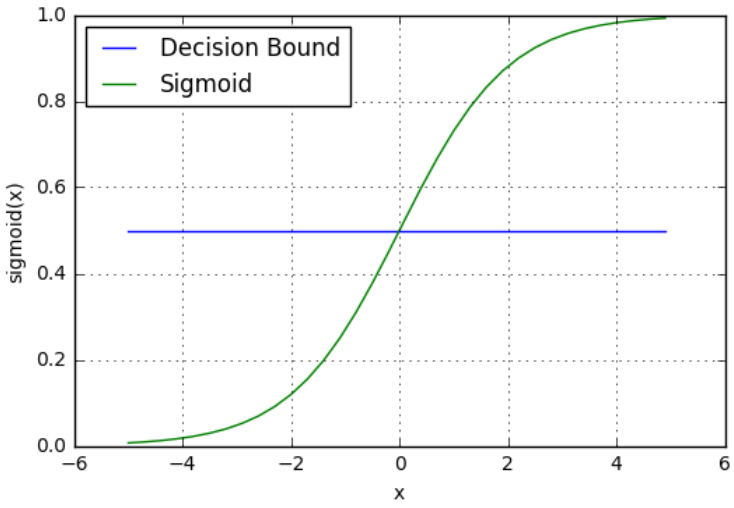

Figure 5. Decision boundary and output of sigmoid function for logistic regression [1]

6 Results

We start off the training process by setting our learning rate (η) to a value of 0.001 and number of epochs to 10000. From Figure 6, we observe that the learning rate is too low for both the training and validation loss to converge to the minima. Therefore, we must either increase the value of η or the number of epochs for the curves to converge.

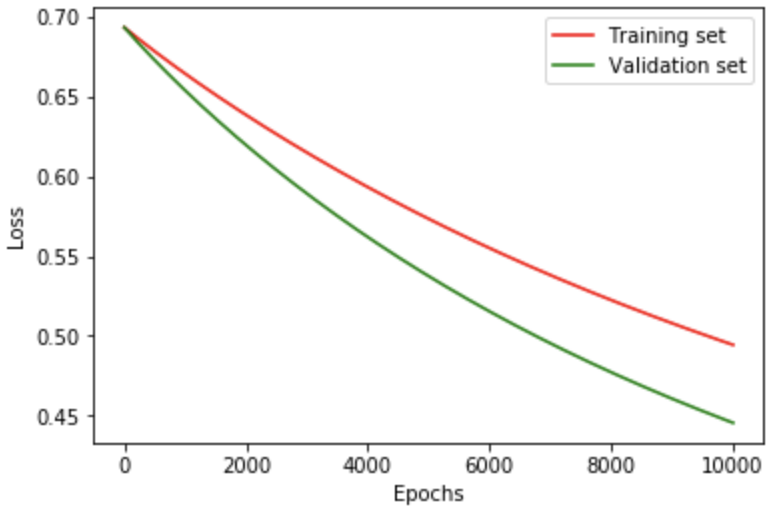

Figure 6. Loss v/s Epochs graph for η = 0.001 and epochs = 10000

To speed up the process of making the curves converge faster, we set η = 2 and number of epochs to 10000. Figure 7 depicts the graph for this hyperparameter combination. We notice that the curve is very steep and the validation loss curve overshoots the minima. This is an indication that η is very high gradient descent is accelerating at a very high rate. Hence, it is necessary to reduce η to get an optimal solution.

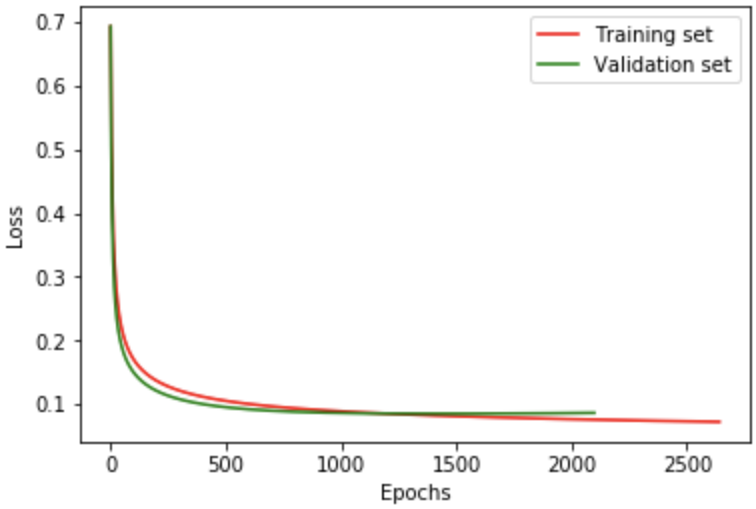

Figure 7. Loss v/s Epochs graph for η = 2 and epochs = 10000

Figure 8 depicts the most optimal solution for our model. We set η = 0.14 and number of epochs to 10000. We observe that both the training and validation curve converge at the same point. This means that the weights and bias calculated by the training set perform well for the unseen validation set as well. Thus, we can infer that for the given weights and bias, any unseen data should give a fairly generalised output.

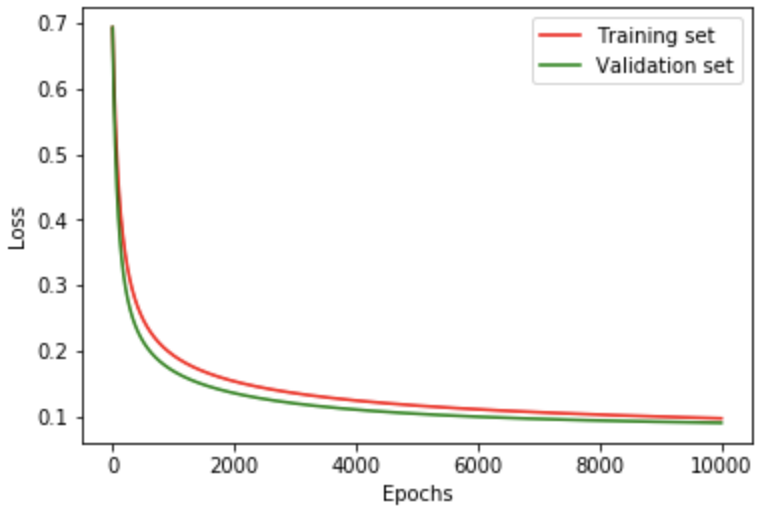

Figure 8. Loss v/s Epochs graph for $\eta = 0.14$ and epochs = 10000

To verify the results of our model, we use the following evaluation metrics :

- **Accuracy** - Percentage of correctly predicted samples to the total number of samples.
- **Precision** - Ratio of correctly predicted positive samples to total positive samples.
- **Recall** - Ratio of correctly predicted positive samples to total samples of that class.

Evaluation Metric	**Validation Set**	**Test Set**
Accuracy	98.07 %	96.49 %
Precision	1	1
Recall	0.944	0.92

Table 1. Evaluation metrics for the logistic regression model

7 Conclusion

In this report, we introduce the basic concept of logistic regression and implement it in Python from scratch. We analyse how the loss function varies with changes in the learning rate and find the optimum values of the hyperparameters. The model evaluation metrics are analysed it is concluded that logistic regression gives satisfactory results for the current binary classification problem with an accuracy of 96.49%.

References

[1] https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html

[2] sci-kit learn documentation https://scikit-learn.org/