Classification of two class problem using Logistic Regression in python from scratch

Sankeerth Tella
50317364
stella3@buffalo.edu

Abstract

In this project, we perform classification of two class problem using logistic regression technique. The dataset provided is Wisconsin diagnostic breast cancer which is basically classified into two classes namely Benign and Malignant. So by using logistic regression technique I developed a model of two classes so that for given input it maps into either of the classes.

1 INTRODUCTION

In the given task we have cancer dataset and it is basically a two class problem which comprises of two classes namely benign and malignant. We need to classify and develop a model using logistic regression. Before coming to logistic regression let's see under what category of learning strategy this logistic regression falls under. Based on the way how they learn or train a model they are classified into four categories

i) supervised learning
ii) unsupervised learning
iii) semi-supervised learning
iv) reinforcement learning

In our given scenario logistic regression falls under supervised learning. Supervised learning is nothing but model is trained on a labelled dataset. Labelled dataset is the dataset which have both input and output parameters.

Logistic regression is the classification algorithm used to develop a model and assign the observations to any of the discrete classes. Unlike linear regression which is used when dependent variable is continuous, logistic regression produces output using sigmoid function to return a probability value and that probability value is mapped to any one of the discrete classes.

1.1 LOGISTIC REGRESSION

Logistic regression is one of the popular supervised machine learning algorithm especially for binary classification. In simple words logistic regression is nothing but estimating parameters using logistic model and mapping them to various classes. Going in deep based on the output classifications logistic regression is further divided into three types

i) Binary logistic regression:

Binary logistic regression is nothing but it has only two possible outcomes and there will be only two classes to classify. Consider simple example of student data set where we have the student marks and their respective grades. Based on the given data we need to classify it into two classes namely pass and fail. So for any given new input our designed model must map to appropriate class.

ii) Multinomial logistic regression

In this type of regression we generalize our model to multiclass problem i.e with more than two possible outcomes. For example consider animal data set where we have characteristics of more then 2 animals and we need to develop model using this given features. For any given input we need to map to appropriate class.

iii) Ordinal logistic regression

Ordinal logistic regression is a statistical technique that is used to predict behavior of ordinal level dependent variables with a set of independent variables.

1.2 CORE OF LOGISTIC REGRESSION

1.2.1 SIGMOID FUNCTION

Logistic regression is named for the function used at the core of the method, the logistic function. In order to map predicted values to probabilities, we use this sigmoid function. This function simply maps any given value to scale between 0 and 1.

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

2 DATASET

Wisconsin Diagnostic Breast Cancer (WDBC) dataset is used for training, validation and testing. The dataset contains 569 instances with 32 attributes (ID, diagnosis (B/M), 30 real-valued input features). Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. Computed features describes the following characteristics of the nuclei present in the image.

1	radius (mean of distances from center to points on the perimeter)
2	texture (standard deviation of gray-scale values)
3	perimeter
4	area
5	smoothness (local variation in radius lengths)
6	compactness ($perimeter^2/area - 1.0$)
7	concavity (severity of concave portions of the contour)
8	concave points (number of concave portions of the contour)
9	symmetry
10	fractal dimension ("coastline approximation" - 1)

The mean, standard error, and \worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.

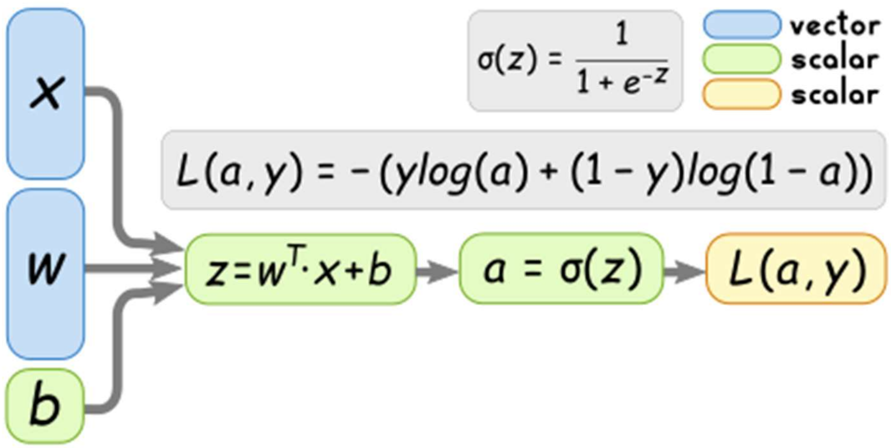
3 PREPROCESSING

Before processing the given data there needs to be some preprocessing done to the data. Initially I loaded the entire data using readcsv function imported from pandas library. Then from whole data I selected 2nd column which is of output column and stored in y. Next I stored all the remaining columns other than y and ids i.e first column into x. Later I splitted the whole data into three parts namely training data, testing data, validation data. Training data comprises of 80% of given data and remaining data is divided between testing and validation 10% each. And next after splitting the data scaling of data is done using standardscalar function which is imported from sklearn kit.

4 ARCHITECTURE

4.1 COMPUTATIONAL GRAPH

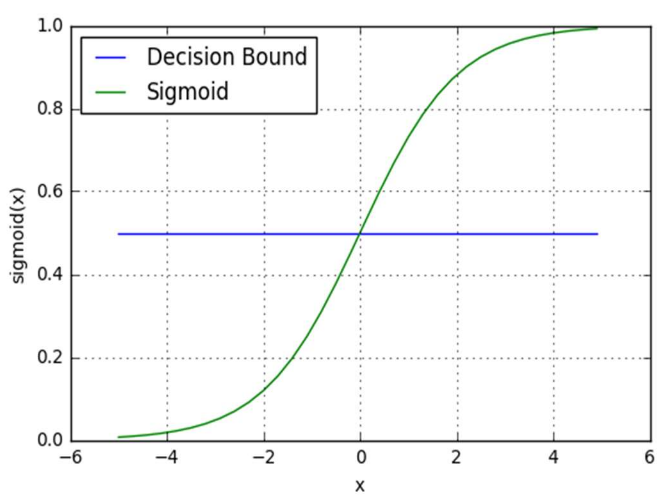
The computational graph of the logistic regression can be visualized as follows:



w, x are input vectors and their size depends on input variables.

4.2 DECISION BOUNDARY

After calculating the probabilistic value we need to map it to either of the class. So we need set threshold and based upon the threshold value we will divide the classes.



4.3 COST FUNCTION

Instead of Mean Squared Error, we use a cost function called cross-entropy, also known as Log Loss. Cross-entropy loss can be divided into two separate cost functions: one for y=1 and one for y=0.

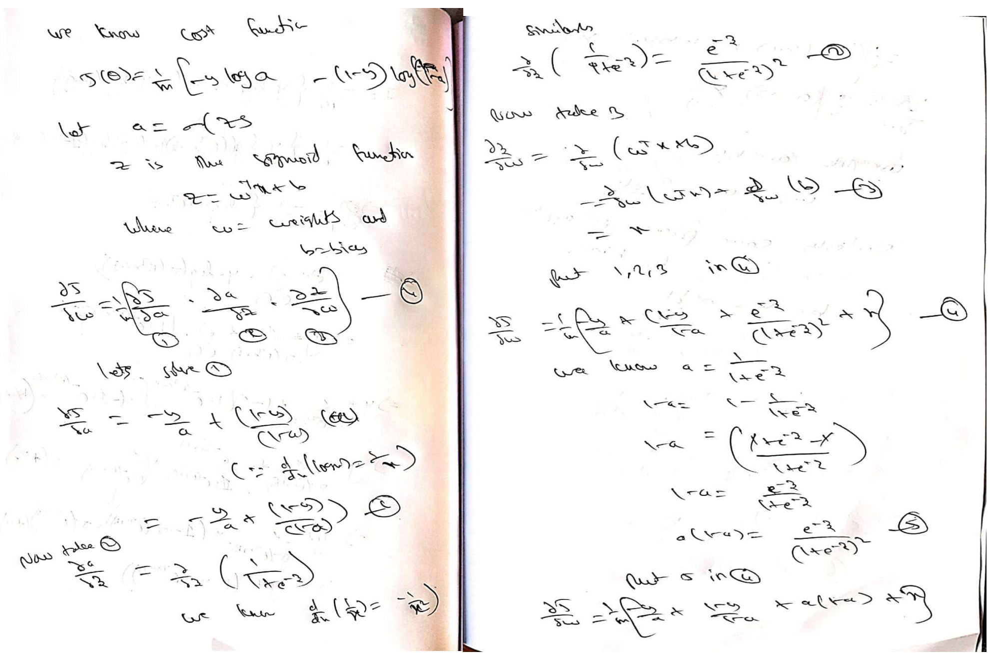
$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \qquad \text{if y = 1}$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) \qquad \text{if y = 0}$$

4.4 GRADIENT DESCENT

The objective of gradient descent is to find out optimal parameters that result in optimising a given cost function. In the Logistic Regression algorithm, the optimal parameters are found by minimizing the loss function and updating the weights and bias.



we know cost function

$$J(\theta) = \frac{1}{m}\left[-y \log a - (1-y)\log(1-a)\right]$$

let $a = \sigma(z)$

z is the sigmoid function

$z = w^T x + b$

where w = weights and b = bias

$$\frac{\partial J}{\partial w} = \frac{1}{m}\left(\frac{\partial J}{\partial a} \cdot \frac{\partial a}{\partial z} \cdot \frac{\partial z}{\partial w}\right) \quad — (4)$$

lets solve (1)

$$\frac{\partial J}{\partial a} = \frac{-y}{a} + \frac{(1-y)}{(1-a)}$$

$$\left(\because \frac{d}{dx}(\log x) = \frac{1}{x}\right)$$

$$= -\frac{y}{a} + \frac{(1-y)}{(1-a)} \quad (1)$$

Now take (2)

$$\frac{\partial a}{\partial z} = \frac{\partial}{\partial z}\left(\frac{1}{1+e^{-z}}\right)$$

we know $\frac{d}{dx}\left(\frac{1}{x}\right) = \frac{-1}{x^2}$

similarly

$$\frac{\partial}{\partial z}\left(\frac{1}{1+e^{-z}}\right) = \frac{e^{-z}}{(1+e^{-z})^2} \quad — (2)$$

Now take 3

$$\frac{\partial z}{\partial w} = \frac{\partial}{\partial w}(w^T x + b)$$

$$= \frac{\partial}{\partial w}(w^T x) + \frac{\partial}{\partial w}(b) \quad — (3)$$

$$= x$$

put 1,2,3 in (4)

$$\frac{\partial J}{\partial w} = \frac{1}{m}\left\{\frac{-y}{a} + \frac{(1-y)}{1-a} + \frac{e^{-z}}{(1+e^{-z})^2} + x\right\} \quad — (4)$$

we know $a = \frac{1}{1+e^{-z}}$

$$1-a = 1 - \frac{1}{1+e^{-z}}$$

$$1-a = \left(\frac{1+e^{-z}-1}{1+e^{-z}}\right)$$

$$1-a = \frac{e^{-z}}{1+e^{-z}}$$

$$a(1-a) = \frac{e^{-z}}{(1+e^{-z})^2} \quad — (5)$$

put 5 in (4)

$$\frac{\partial J}{\partial w} = \frac{1}{m}\left\{\frac{-y}{a} + \frac{1-y}{1-a} + a(1-a) + x\right\}$$

$$= \frac{1}{m}\left\{ -y(1-a) + (1-y)a \right\} x$$

$$\frac{\partial J}{\partial w} = \frac{1}{m}\left\{ (a-y)\, x \right\}$$

weights are now

$$w_i = w_i - \alpha \frac{\partial L}{\partial w_i}$$

Similarly same process is applied for bias

$$b = b - \alpha \frac{\partial L}{\partial b}$$

After minimizing the cost function through various epochs, we get the updated weights and bias. And using these weights and bias we predict the output values again using sigmoid function with updated final weights and bias. After predicting the final values, we need to calculate true positive(TP), true negative(TN), false positive(FP), false negative(FN) by comparing with actual output values. After this using the values of TP, TN, FP, FN we need to calculate accuracy, precision, recall, f-measure.

5 RESULTS

After predicting the values using the model, the performance of model can be evaluated using four metrics namely accuracy, precision, recall, f-measure.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

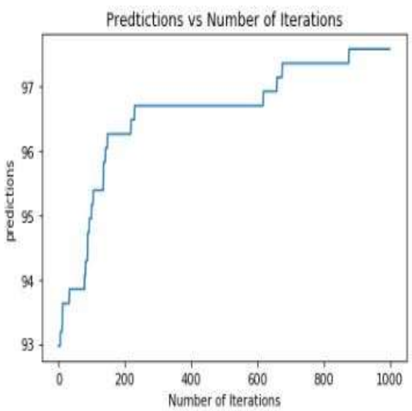
$$Precision = \frac{TP}{TP + FP} \tag{2}$$

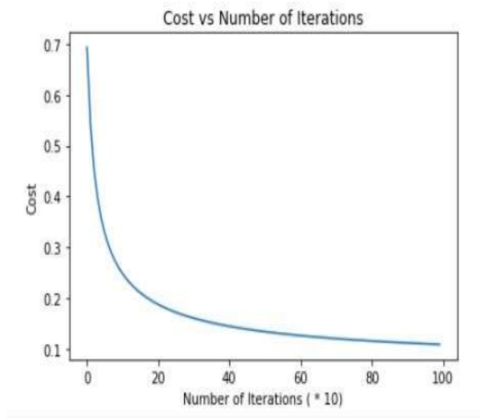
$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

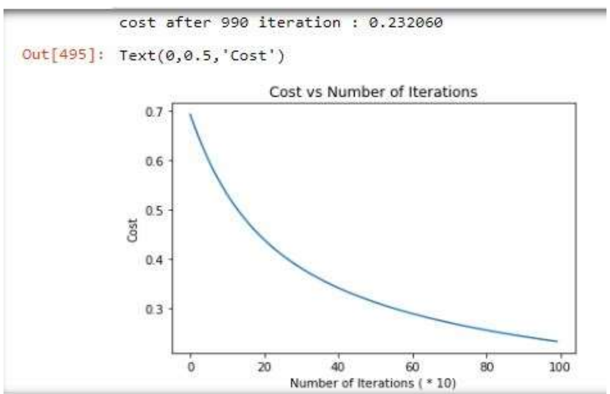
```
In [339]: accuracy=num/den*100
          print(accuracy)

          98.24561403508771

In [340]: n=cm[0][0]
          m=cm[0][1]
          recall=n/(n+m)*100
          print(recall)

          100.0

In [341]: p=cm[1][0]
          precision=n/(n+p)*100
          print(precision)

          97.14285714285714

In [342]: fmeasure=(2*recall*precision)/(recall+precision)
          print(fmeasure)

          98.55072463768116
```

I trained my model with learning rate=0.01 and number of epochs=1000, then I plotted graph between predictions vs number of iterations and number of iterations and cost.

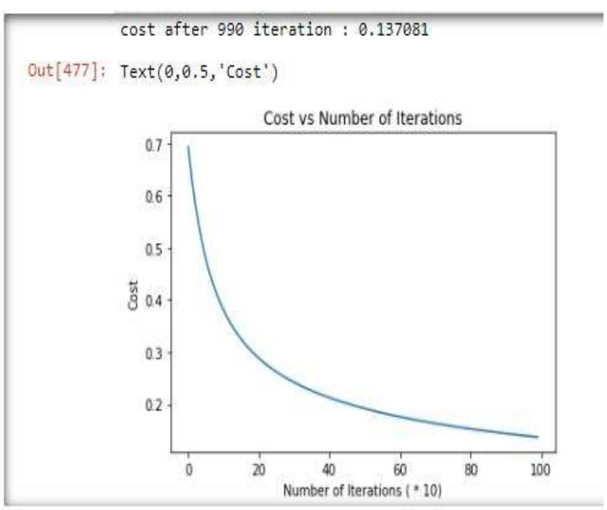




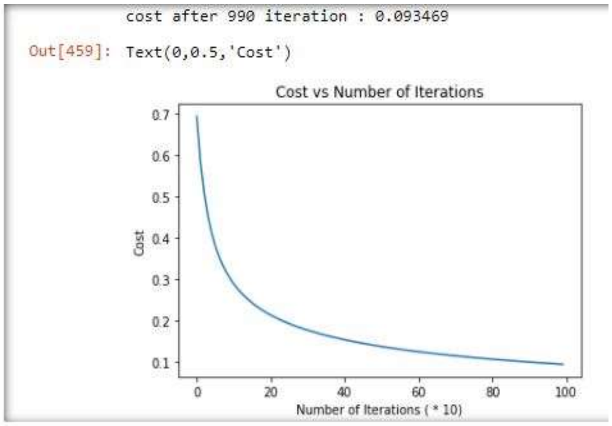
I validated my model using validation data with learning rate=0.001 and number of epochs=1000 and I got cost=0.232060



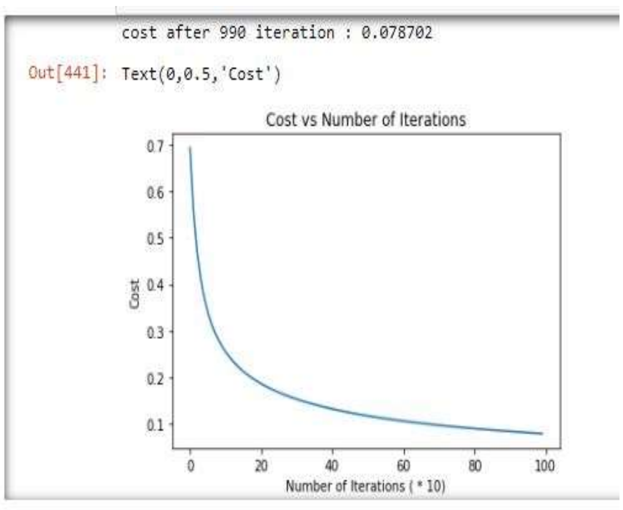
I validated my model using validation data with learning rate=0.003 and number of epochs=1000 and I got cost=0.137081



I validated my model using validation data with learning rate=0.006 and number of epochs=1000 and I got cost=0.093469



I validated my model using validation data with learning rate=0.008 and number of epochs=1000 and I got cost=0.078702



6 CONCLUSION

I successfully trained my model using given cancer dataset and able to validate my dataset as per the graphs shown above. And the following results are drawn.

Accuracy = 98.24561403508771

Precision = 97.14285714285714

Recall = 100.0

F-measure = 98.55072463768116

References

1) https://machinelearningmastery.com/logistic-regression-tutorial-for-machine-learning/

2) https://en.wikipedia.org/wiki (htt)/Logistic_regression

3) https://stats.idre.ucla.edu/r/dae/ordinal-logistic-regression/

4) https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html

5) https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc

6) https://www.statisticssolutions.com/regression-analysis-ordinal-regression/

7) http://ronny.rest/blog/post_2017_08_12_logistic_regression_derivative/