CLASSIFICATION USING LOGISTICS REGRESSION

Prakhathi Murugesan
University of Buffalo
prakhath@buffalo.edu

Abstract

Arthur Samuel, who coined the term 'Machine Learning' defines Machine Learning as, the field of study which provides Computer the ability to learn without being programmed. The Computer can be able to perform a specific task by relying on patterns and interference instead of instructions. Some of the application of machine learning are spam detection, search engines, photo tagging, and medical diagnosis, etc. The dataset of Wisconsin Breast Cancer diagnosis is given for this project and the task is to classify the patients into class 0 (Benign) and class 1 (Malignant). The goal of this project is to perform classification using logistic regression. The classification algorithm/ classifier maps the dataset into specific categories and the logistic regression is used to build the predictive model which predicts the output as either 0 or 1. This report records the process of building a model and evaluates its performance with different parameters.

1. INTRODUCTION

Classification is a learning approach in which it takes training data as input and learns how given input variables are related to the class or category. Some of the examples of Binary classification are spam identification, and medical diagnosis. Logistic regression is a supervised classification algorithm which takes features (input) X and predicts the target variable (output) Y. Target variable Y is the dependent variable and depends on the independent variable X. Y takes two discrete values, i.e. 0 or 1. Consider we plot a graph for a dataset with features of two classes and develop a model which is a linear line; it should pass through the graph splitting class 1 on one side and class 0 on the other side. This implies that a well-fitting model should be able to classify the class. In this project, Wisconsin Diagnostic Breast Cancer (WDBC) dataset is used and it consists of 569 records of patients and 32 attributes (including ID, diagnosis) which will help in predicting the output.

2. PROCESSING OF DATASETS

The Initial step of the project is the processing of the dataset. As a part of this step, panda library is used to read the content of the dataset which is in CSV file and convert it into Pandas Data frame. The first attribute of the dataset is Patient's ID which is not a useful feature in predicting the output. So, the first attribute should be removed from the Data frame. The second attribute has either Benign (B) or Malignant (M) value. This attribute indicates whether the patient is identified with cancer or not. Thus, it gives out the target value (Y). As the target variable takes either 0 or 1, B and M values should be mapped to 0 and 1 correspondingly. This value should be dropped from Data frame. Thus, X contains 569 instance and 30 features. The next step of processing is the normalization of the dataset. As the values in the dataset range differently, it is must to convert the values to a common scale without distorting the differences in the range of values. To normalize the data, values from the first column are taken and each value is subtracted from a minimum value of that column and then divided by the range of the values of that column. The same has to be done for each column. Thus, at the end of this step, all the values in the dataset are normalized.

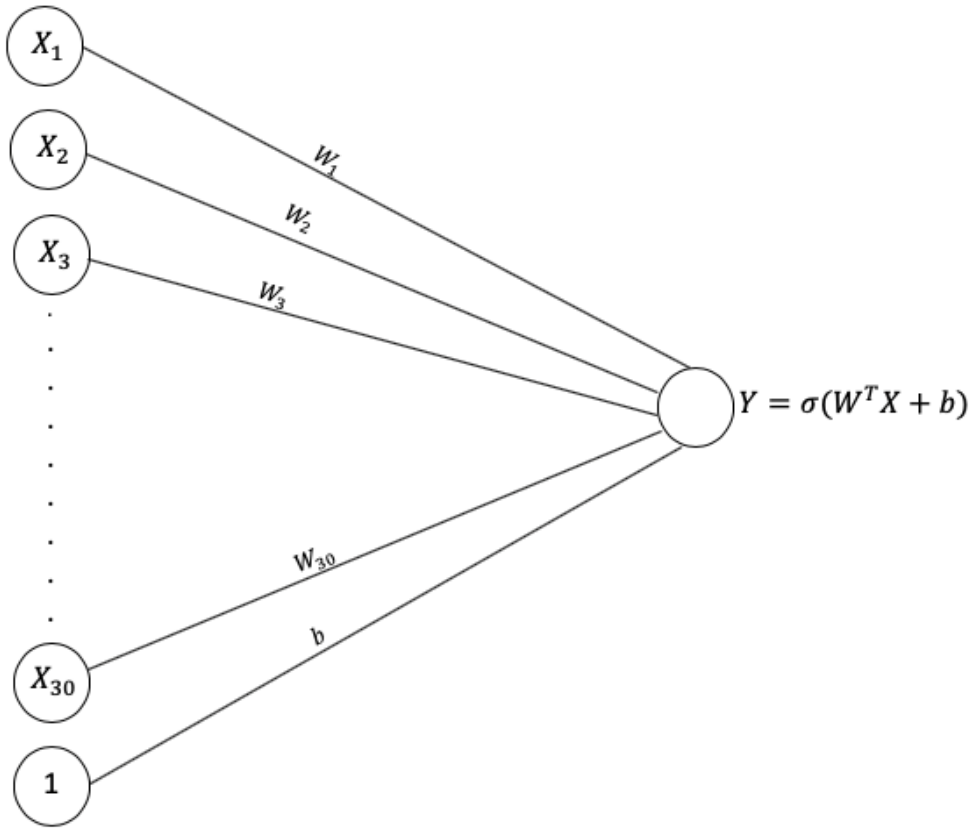
$$\tilde{x}_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

The final step of the preprocessing is partitioning datasets into Training, Validation, and Testing datasets. The training, validation and testing dataset consists of 80%, 10% and 10% of the overall dataset respectively. The training dataset

is for training the model and then the model is evaluated with validation dataset to tune the hyperparameter and then the performance is verified using testing dataset.

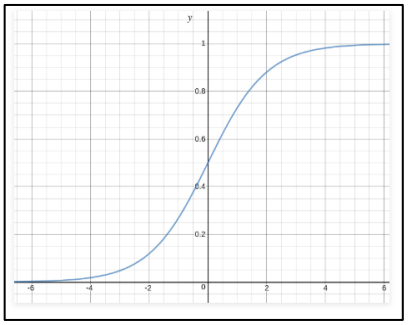
3. ARCHITECTURE:

The model is developed with passing training dataset X of 30 features along with the bias and weight to get the predicted output Y. This output Y is verified for the accuracy and performance of the model. The Architectural diagram of the predictive model using logistic regression is given as:



3.1 TRAINING USING LOGISTIC REGRESSION:

Logistic regression builds a model that predicts the target variable Y, which can take only two distinct values for a given set of features X. Logistic regression models use the sigmoid function. The value of the sigmoid function lies between 0 and 1.



Sigmoid function is represented by,

$$\sigma(t) = \frac{1}{(1+e^{-t})}$$

A Linear regression model can be represented as:

$$\mathbf{Z = WX + B \text{ or } Z = \Theta x}$$

Where W is the weight and B is the Bias. These two are learnable parameter. Weight refers to the strength of the connection and on increasing its value, it denotes how much influence does it have on output. Bias refers how far off the predicted value from the real one. Then, on applying sigmoid function to the output of linear regression, we get

$$\mathbf{P = \sigma (Z)}$$

The hypothesis of logistic regression is given by,

$$\mathbf{P} = \frac{1}{(1+e^{-\theta x})}$$

By Sigmoid graph, the Predicted value will become 1 when Z goes to infinity and Predicted value will become 0 when Z goes to negative infinity. Initialize the value of Bias 'B' to zero and Weight 'W' to random number with dimension equal to X. By passing the training dataset to the model, the model is trained. The cost function is used to verify the accuracy of the model. And the validation dataset is passed to the model which helps in updating the hyperparameter. This process is continued for n iterations.

3.2 COST FUNCTION:

The Cost function is expressed as a difference between the actual value and the predicted value. Cost function helps the model to correct its behavior to minimize the mistakes. It is a measure of how wrong the model is in terms of its ability to estimate the relationship between X and Y. The Cost function is also referred to as error or loss function. The Cost function is given by:

$$\mathbf{Cost = -\log(p), \text{ if } y=1}$$

$$\mathbf{Cost = -\log(1-p), \text{ if } y=0}$$

For the graph -log(p), as p approaches 1, the cost is 0, and as p approaches 0, the cost is infinity. Similarly, for graph -log(1-p), the cost is 0 when the actual value is 0, and the model predicts 0, and the cost is infinity as p approaches 1. So, the combined equation of cost function for single data is:

$$\mathbf{Cost = -y \log(p) - (1-y) \log(1-p)}$$

Cost function for 'm' training data can be calculated by:

$$\mathbf{Cost = J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [y^i \log p^i + (1 - y^i) \log(1 - p^i)]}$$

Gradient descent is used to minimize the cost function.

3.3 GRADIENT DESCENT:

Gradient descent is used to minimize the loss obtained from the model. Gradient descent is an efficient algorithm that attempts to find a global or local minima of a function. It helps a model to learn the direction that the model should take to reduce errors. For each iteration, parameters W, and B are updated to minimize the cost function and the model gradually converges towards the minimum where further twerk to the parameters produce little or zero changes in the loss. The model has optimized the weights and bias at this point such that they minimize the cost function.

By using the partial derivative of the cost function, we derive gradient descent rule.

$$dw = \partial \frac{J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} (p^i - y^i)\, x_j^i$$

$$db = \frac{1}{m} \sum_{i=1}^{m} (p^i - y^i)$$

$$w = w - lr * dw$$

$$b = b - lr * db$$

where lr is learning rate.

3.4 WORKING WITH HYPER-PARAMETERS:

Hyper-parameters are factors that eventually affect the performance of the system. The hyper-parameters that are of significant importance are:

- Number of iteration or epoch
- Learning Rate

The number of iterations gives us the time to correct the mistakes, and reduce the cost, and find the optimum value of weight and bias.

Similarly, the learning rate also plays a vital role in the performance of the model. Learning rate is the amount that the weight is updated. It refers to how big the steps of learning are. Choosing the ideal learning rate is challenging because a smaller learning rate indicates that the step size is small, which leads to longer training process, and convergence is prolonged. Larger learning rate means larger step size, which leads to rapid change in weight and higher convergence which tends to miss out the minimal point i.e., it overshoots the minimum and starts moving upward.
By tuning the number of iteration and learning rate, the weight and bias get updated. And the best fitting model is obtained for the optimal values.

For learning rate 0.0009 and epoch 10000, the predictive model becomes underfit and found accuracy is of 84% as shown in diagram 1. As we increase the learning rate to 0.5, the predictive model becomes overfit and accuracy is obtained as 100% using validation dataset. For learning rate 0.01, the model becomes stable and gives accuracy of about 96%. Thus, the updated weight and bias for the best model is calculated as:

W_1 = 0.72042607
W_2 = 0.26645591
W_3 = 0.82736609
W_4 = 0.90985861
W_5 = -0.39566899
W_6 = 0.70469882
W_7 = 1.45455875
W_8 = 1.74948382
W_9 = -0.47568372
W_{10} = -0.87963962

W_{11} = 0.63323129
W_{12} = -0.54010588
W_{13} = 0.56404496
W_{14} = 0.54355843
W_{15} = -0.5494033
W_{16} = -0.09286978
W_{17} = -0.0926078
W_{18} = -0.05099244
W_{19} = -0.5856061
W_{20} = -0.32033052

W_{21} = 1.20971251
W_{22} = 0.49907151
W_{23} = 1.20971251
W_{24} = 0.49907154
W_{25} = 1.20811519
W_{26} = 1.10288488
W_{27} = 0.1225111
W_{28} = 0.86689296
W_{29} = 1.16702777
W_{30} = 0.16286217
b = -3.8270153

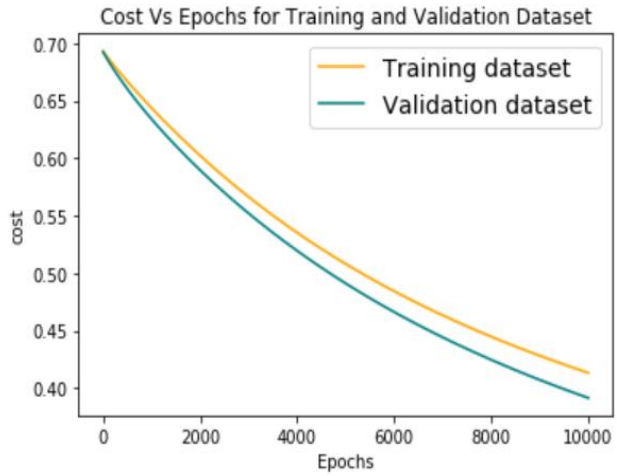


Figure 1. Graph shows Cost Vs Epoch for learning rate 0.0009 (Underfit model)

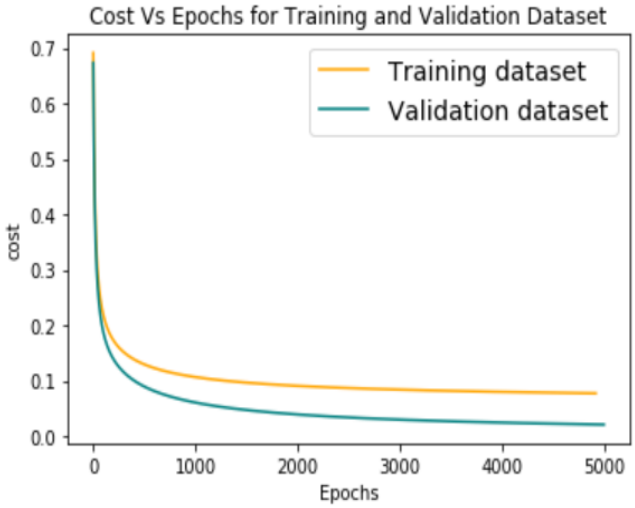


Figure 2. Graph shows Cost Vs Epoch for learning rate 0.5 (Overfit model)

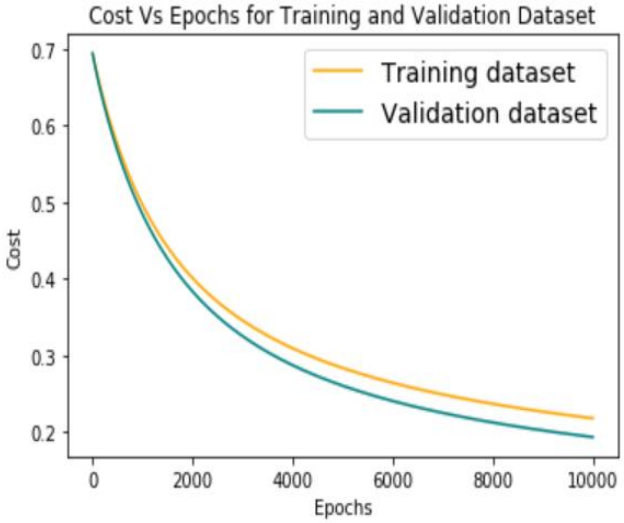
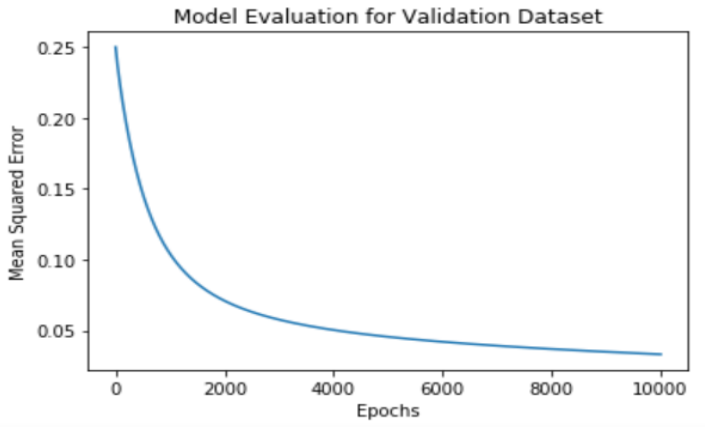


Figure 3. Graph shows Cost Vs Epoch for learning rate 0.01

3.5 MODEL EVALUATION METRICS:

It is crucial to evaluate the predicted value against the actual value so that the performance of the model is determined. One of the popular methods of evaluation is the Mean Squared Error Method. It is an estimator that measures the average of error squares i.e., the average squared difference between the predicted values and actual value. It is always non – negative, and values close to zero are better.

$$\mathbf{MSE} = \frac{\mathbf{1}}{\mathbf{N}}\sum_{\mathbf{i=1}}^{\mathbf{N}}(\mathbf{Y_i} - \widehat{\mathbf{Y}}_\mathbf{i})^2$$



4. RESULT:

A confusion matrix is computed to describe the performance of a model. The number of correct and incorrect predictions are summarized in the confusion matrix. The Confusion matrix gives the count of total TP, TN, FP, and FN occurred in the model.

The Confusion Matrix for the predictive model is obtained as:
Confusion Matrix = [34 0]
[2 21]
Where True Positive is 34, False Positive is 0, False Negative is 2, and True Negative is 21.

Accuracy or Classification Rate is given by:
$\mathbf{Accuracy} = \frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FP+FN}}$ which is obtained as **0.96491**

Recall refers to the ratio of the total number of correctly classified positive to the total number of positive data
$\mathbf{Recall} = \frac{\mathrm{TP}}{\mathrm{TP+FN}}$ which is calculated as **0.94444**

Precision refers to the ratio of the total number of correctly classified positive to the total number of predicted positive value.
$\mathbf{Precision} = \frac{\mathrm{TP}}{\mathrm{TP+FP}}$ which is calculated as **1.0**

High recall, low precision means that most of the positive data is correctly recognized (low FN), but the number of false positives is high. **Low recall, high precision** shows that a lot of positive data is missed (high FN), but whatever predicted as positive are positive (low FP).

5. CONCLUSION:

This project helped in getting a clear idea of the working of logistic regression. It has also helped in understanding the sigmoid function and the gradient descent. We calculated the updated weights by computing the error function and minimizing it by gradient descent. Finally, the target values are computed for the testing dataset. We also calculated the mean square error value and the accuracy of the model. This project helped in getting a deeper considering of the impact of the hyper-parameters on the performance of the system. Thus, we performed a logistic regression to classify the dataset.

REFERENCES

[1] Logistic Regression – Introduction to Logistic Regression- https://www.geeksforgeeks.org/understanding-logistic-regression/

[2] Logistic Regression – Overview of Logistic Regression - https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc

[3] Fundamentals of Machine Learning - https://towardsdatascience.com/machine-learning-fundamentals-via-linear-regression-41a5d11f5220

[4] Mean Squared Error - https://www.geeksforgeeks.org/python-mean-squared-error/