Classification Using Logistic Regression

Lan H. Le
Department of Mechanical and Aerospace Engineering
University at Buffalo
Buffalo, NY 14226
hoanglan@buffalo.edu

Abstract

Based on data from Wisconsin Diagnostic Breast Cancer (WDBC), we use logistic regression to determine whether fine needle aspirate (FNA) cells are Benign or Malignant.

1 Introduction

For a long time, we have been relying solely on doctors to make diagnoses and determine whether patients' tumors are benign or malignant. As humans are bound to make mistakes, so are doctors. Now with the advance of machine learning, diagnoses can be more accurate when we combine technology and human experience.

The task of this project is classifying FNA cells into two classes, Benign or Malignant, using logistic regression. The features used classification are pre-computed from images of a fine needle aspirate (FNA) of a breast mass. The dataset in use is from WDBC. The code is written in Python from scratch.

2 Dataset

WDBC dataset will be 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 cell 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 (perimeter2/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

Preprocessing the dataset consists of several different steps. First, the column ID from the dataset is dropped since it does not contribute to the classification process. Next, the diagnosis labels need to be mapped to 0s and 1s for more convenient processing.

To mitigate bias and the risk of overfitting, the dataset is then split into training (80%), validation (10%), and test (10%) sets.

Lastly, before training the model, training data needs to be normalized to make sure they are of a similar scale.

Data preprocessing is performed by utilizing Pandas Dataframe and scikit-learn tools.

4 Architecture

So as to generate the predictions, Sigmoid function is used to produce values between 0 and 1 (and from there we can classify values based on a threshold, e.g. 0.5).

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

SIGMOID FUNCTION

We also have:

$z = w^T . x + b$

In order to find the "best" w and b, we update them for a number of iterations (epochs) using gradient descent.

$b := b - adb$

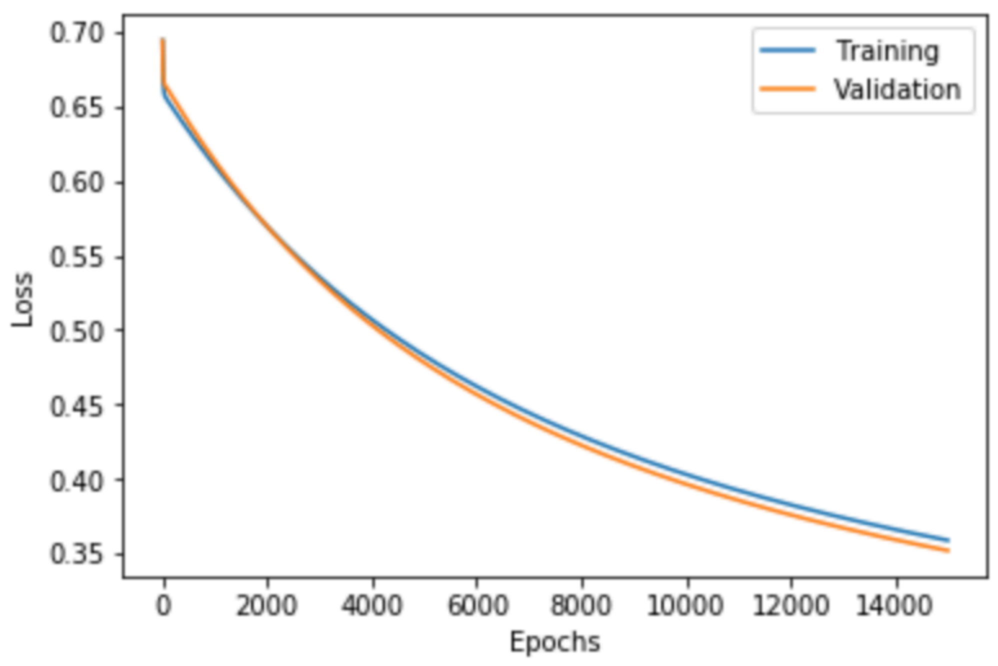
$w := w - adw$

(a is the learning rate)

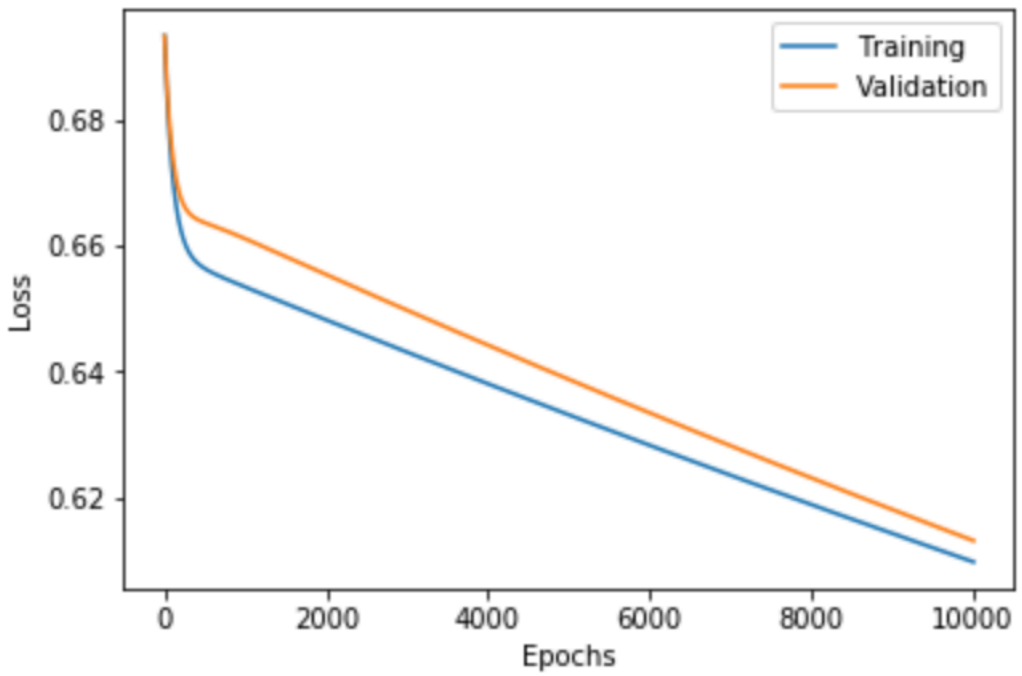
So as to keep track of the progress of the training, loss function is used. If loss decreases after each epoch then the training is going in the right direction. Loss is recorded for both training and validation data.

Loss function:

$L(p, y) = -(ylogp + (1 - y)log(1 - p))$



LEARNING RATE = 0.1 & EPOCHS = 15000



LEARNING RATE = 0.01 & EPOCHS = 10000

It is also important to tune the hyper-parameters (i.e. learning rate and number of epochs for this project). We start with a learning rate of 0.01 and a number of epochs of 10000. Using trial and error, the optimal values are 0.1 for learning rate and 15000 for number of epochs. Changing the values above or below this point does not improve the performance significantly.

5 Results

After training with logistic regression, we are able to calculate the coefficients (w and b) that produce the optimal results. We can now use this value to evaluate the performance of our model on the test set. To be more specific, predictions are made based on the features in the test set and then the predictions are compared to the actual (test) targets.

For a learning rate of 0.1 and 15000 epochs, the scores are:

- Accuracy: 82.45%
- Precision: 92.30%
- Recall: 57.14%

For reference, here are the scores when the learning rate is 0.01 with 10000 epoch:

- Accuracy: 66.66%
- Precision: 100%
- Recall: 9.52%

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

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

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

PERFORMANCE EVALUATION

6 Conclusion

We use logistic regression to divide our data into two classes, Benign and Malignant. In order to do this, we have to preprocess the data, train the model, and tune the hyper-parameters. As we go along, we also have to track the loss to make sure that our end result is optimal. In the end, the accuracy and precision scores are high but the recall score is just above average. In the future, we would like to work on perfecting the current model or applying another model on the dataset to see if performance can be further improved.

Acknowledgments

The description of the dataset is from the original project description. The equations for logistic regression and performance evaluation are also derived from class materials.

References