Two-Class Logistic Regression Classification of Breast Cancer Data

Project 1 CSE 574 Intro to Machine Learning

Krishna Naga Karthik BODAPATI
Department of Computer Science
(SUNY) University at Buffalo
Buffalo NY, 14221
kbodapat@buffalo.edu

Abstract

The Goal of this project to predict any given tumor whether it is benign or malignant. Logistic Regression is used to achieve that and "Wisconsin Diagnostic Breast Cancer" dataset is used to train the model which has 30 features, which are precomputed Finite Needle Aspiration images and 569 training examples. The model has good accuracy in predicting the tumor type, even though it is a simple linear model

1 Introduction

1.1 What is FNA Diagnosis:

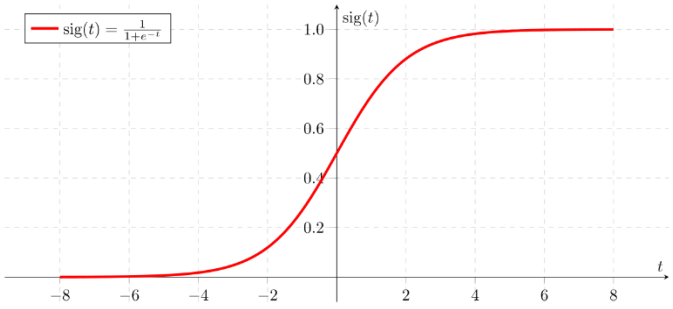
Fine-needle aspiration (FNA) is a diagnostic procedure used to investigate lumps or masses. In this technique, a thin (23–25 gauge), hollow needle is inserted into the mass for sampling of cells that, after being stained, will be examined under a microscope (biopsy). The sampling and biopsy considered together are called fine-needle aspiration biopsy (FNAB) or fine-needle aspiration cytology (FNAC) (the latter to emphasize that any aspiration biopsy involves cytopathology, not histopathology). Fine-needle aspiration biopsies are very safe minor surgical procedures. Often, a major surgical (excisional or open) biopsy can be avoided by performing a needle aspiration biopsy instead, eliminating the need for hospitalization. In 1981, the first fine-needle aspiration biopsy in the United States was done at Maimonides Medical Center. Today, this procedure is widely used in the diagnosis of cancer and inflammatory conditions.

1.2 Logistic Regression:

As Logistic regression is a supervised machine learning Algorithm, We need to train this model with labelled data, which we can do using "Wisconsin Diagnostic Breast Cancer" Dataset. Once the model is trained we can preform predictions to unlabeled data

In this problem we need to predict to which class the tumor belongs i.e. either to Benign class or Malignant class. So, Binary Logistic Regression classifier is the ideal

model for this problem. In this model we used Sigmoid as Basis function because it takes any input and outputs floating-point number between 0 and 1, We take this a probability of tumor to be malignant



2 DATASET

The Wisconsin Diagnostic Breast Cancer (WDBC) dataset used for training, validation and testing of the regression model contains 569 instances with 32 attributes (ID, diagnosis (B/M), 30 real-valued input features). Features are computed from digitized images of FNA of breast tissues. Computed features describe the following characteristics of the cell nuclei:

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 the output features are labelled as B for Benign and M for Malignant Tumors.

3 Pre-Processing

3.1 Normalization

It is not a great idea to use un normalized data for training, Normalizing data gives equal importance all features and stops the domination of features with larger values.

In this problem Min-Max Normalization is used. In Min-Max Normalization we subtract each feature vector by its minimum and then divide it by (maximum – minimum)

Before Normalization

	1	2	3	4	5	6	7	8	9	10	...	22	23	24	25	26	27	28	29	30	31
0	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	...	25.38	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890
1	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	...	24.99	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902
2	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	...	23.57	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758
3	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	...	14.91	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300
4	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	...	22.54	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678

Normalized Data

	0	1	2	3	4	5	6	7	8	9	...	21	22	23	24	25	26
0	1.0	0.521037	0.022658	0.545989	0.363733	0.593753	0.792037	0.703140	0.731113	0.686364	...	0.620776	0.141525	0.668310	0.450698	0.601136	0.619292
1	1.0	0.643144	0.272574	0.615783	0.501591	0.289880	0.181768	0.203608	0.348757	0.379798	...	0.606901	0.303571	0.539818	0.435214	0.347553	0.154563
2	1.0	0.601496	0.390260	0.595743	0.449417	0.514309	0.431017	0.462512	0.635686	0.509596	...	0.556386	0.360075	0.508442	0.374508	0.483590	0.385375
3	1.0	0.210090	0.360839	0.233501	0.102906	0.811321	0.811361	0.565604	0.522863	0.776263	...	0.248310	0.385928	0.241347	0.094008	0.915472	0.814012
4	1.0	0.629893	0.156578	0.630986	0.489290	0.430351	0.347893	0.463918	0.518390	0.378283	...	0.519744	0.123934	0.506948	0.341575	0.437364	0.172415

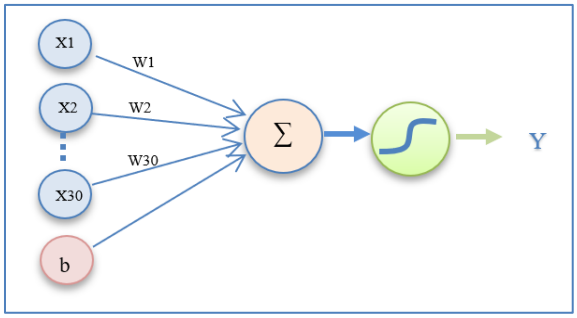
The result is, all the data is between 0 and 1

3.2 Partitioning of Data:

It is bad idea to use all examples to train the model, because it would be impossible to test accuracy of the model as It remembers all training examples and also, we cannot distinguish whether the fit is good fit or overfit. So, it is a good idea to partition dataset before Training

In this problem we split the data set into 3 parts: Training set (80%) which is used to train the model, Validation set (10%) to tune the Hyper Parameters, and Testing Set (10%) to test and determine the accuracy of the model

4 Model Architecture

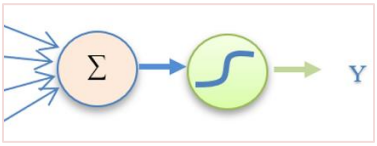


There are a total of 30 features which are represented by (x1,x2,…….x30), each feature has its appropriate weight which is represented by (w1,w2,……..w30) and bias of B is added. The output is the linear combination of the Weights and features

$$z = w_1x_1 + w_2x_2 + \text{---------} + w_{30}x_{30} + b$$

$$z = \sum w_i x_i + b$$

Since the output can be any real valued, we apply sigmoid to this output which returns all the values in range (0,1)



$$A = \sigma(Z)$$

4.1 Loss Function:

Typically, the Loss function is output Vector – Prediction Vector i.e. in our case it is (Y-A). The problem with this loss function is it not convex function (it has more than 1 local minimum points) so we cannot to use Gradient descent. To solve this problem, we use Binary Cross Entropy Function which is:

If Y = 0 $\quad (-1/m) \sum (y*\log(a))$
If Y =1 $\quad (-1/m) \sum (y*\log(1-a))$

Which can be Written as

$$L = (-1/m) \sum (y*\log(a)) + ((1-y)*\log(1-a))$$

4.2 Gradient Descent:

We need to determine the weights for our model. First we initialize all weights to 0s and update these weights for each iteration. Here our goal is to minimize cost function which, Cost function is minimum when slope of cost function = 0. So we compute the gradient for cost function for each weight and subtract that from weight. For each iteration the cost function is reduced.

$$dL/dw_i = (-1/m)\, d/dw\{ y*\log(\sigma(z)) + (1-y)*\log(1-\sigma(z))\}$$

$$dL/dwi = (-1/m) \{ (y - \sigma(z)) \} xi$$

$$\Delta wi = (-1/m) \{ (y - \sigma(z)) \} xi$$
$$\Delta b = (-1/m) \{ (y - \sigma(z)) \}$$

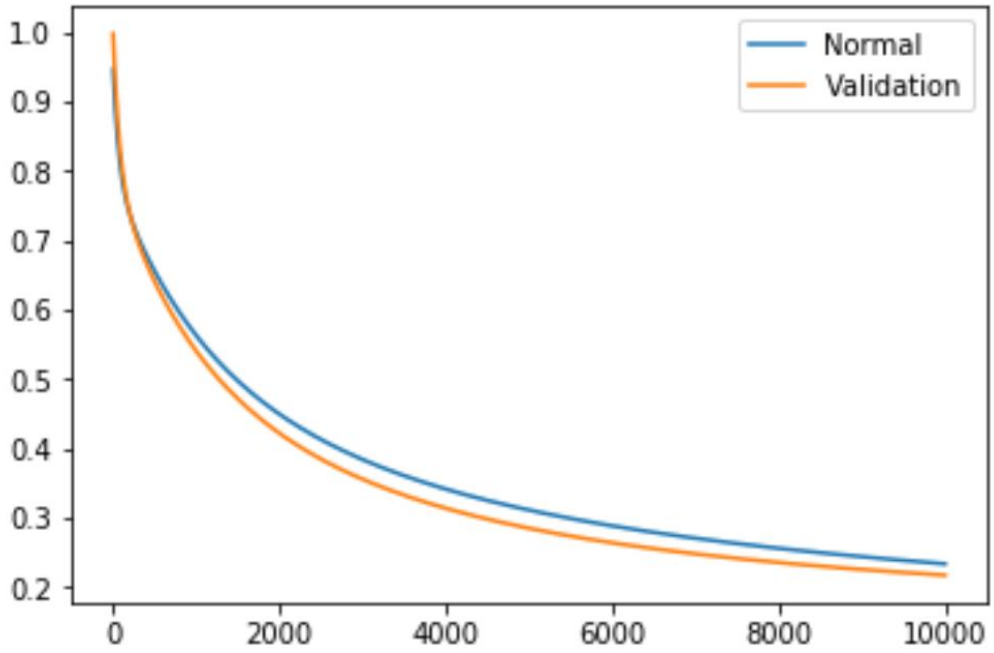
So new Weights and bais will be

$$W = w - \Delta wi$$
$$B = b - \Delta b$$

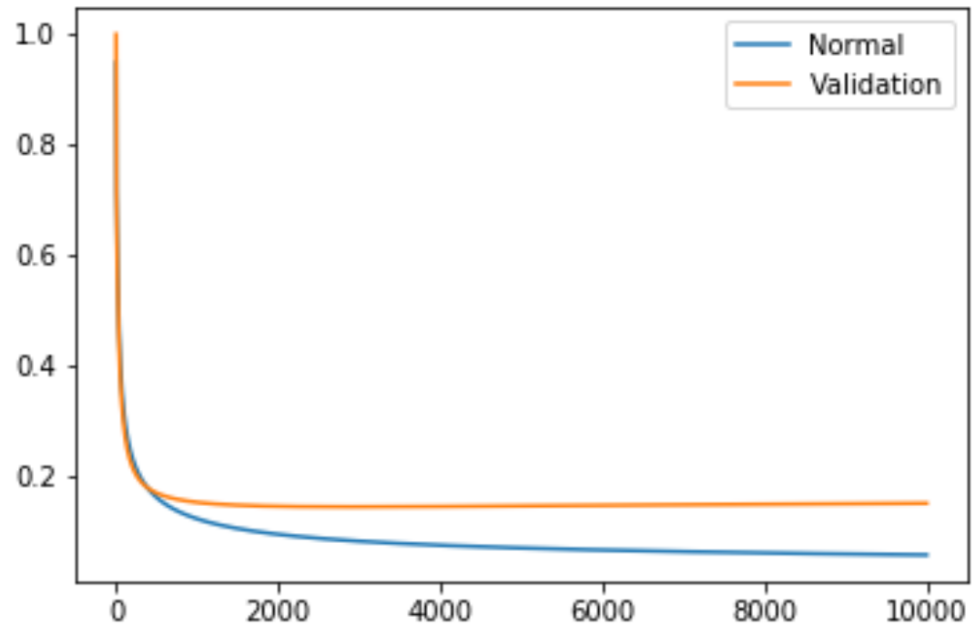
4.3 Tuning Hyperparameter

Here we need to decide two hyper parameters which are: 1. No. of iterations 2. Learning rate, tuning theses parameters is crucial because we don't want to overfit or under fit the data and also, we need our model to converge fast. If the learning rate is too high the model converges too fast and cost function may increase and If the learning rate is too small, it takes longer to converge.

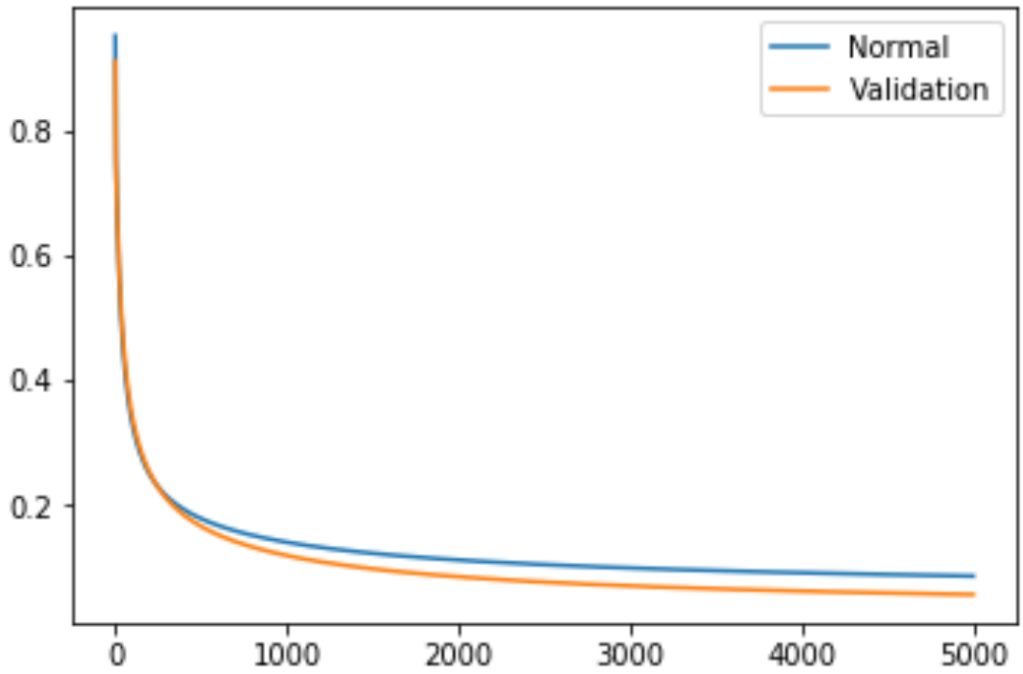
Plots



Alpha = 0.01 and epochs = 10000
Less Learning rate and underfit



Alpha = 1 and epochs = 10000
Over Fit



Alpha = 0.4 and Epochs = 5000
Good fit

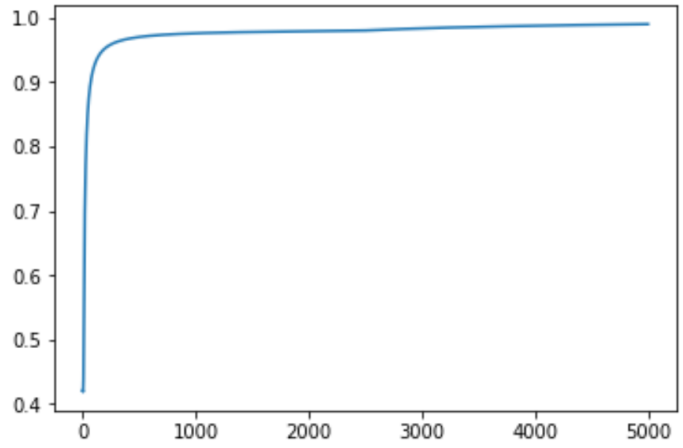
5 Results

Metrics of the model are:

Accuracy: `0.9649122807017544`

Precision: `0.9583333333333334`

Recall: `0.9583333333333334`



Accuracy vs Epochs

6 Conclusion

We used Logistic Regression to predict the probability of tumor to be benign and malignant based on FNA images. Even though this model is fairly simple linear model, has a good accuracy