Breast Cancer Prediction using Logistic Regression

Manisha Biswas
mbiswas2@buffalo.edu

Abstract

This paper presents an approach to classify two-class problem using Logistic Regression. The dataset used to create the classifier is the WDBC (Wisconsin Diagnostic Breast Cancer) dataset which was created by Dr. William H. Wolberg, where the features used for classification are pre-computed from images of a fine needle aspirate (FNA) of a breast mass. The aim is to classify the suspected FNA cells whether it is benign (class 0) or malignant (class 1). The first part of the project is dedicated to pre-process the data. To achieve this we examine the data, what it contains, when and how it was created, if it is noisy, if it has missing values. This section is important to understand what are the issues that will need to be processed while preparing the data to create the classifier. Then we partition the dataset as: 80% for training, 20% for testing and 10% of testing data to validation dataset. The next step is to apply Logistic Regression algorithm to optimize the training set. We also select a set of hyperparameters for the learning process and estimate the best model parameters which gives us 94.74% of accuracy.

1. Introduction

Since the beginning of human existence, we have been able to cure many diseases, from a simple bruise to complex neurological disorders. Now, humanity is on the cusp of conceiving of something new: a cure to cancer.

Breast cancer is the most common cancer among women and one of the major causes of death among women worldwide. Breast Cancer occurs as a result of abnormal growth of cells in the breast tissue, commonly referred to as a Tumor. A tumor does not mean cancer - tumors can be benign (not cancerous), pre-malignant (pre-cancerous), or malignant (cancerous). Every year approximately 124 out of 100,000 women are diagnosed with breast cancer, and the estimation is that 23 out of the 124 women will die of this disease, yet we haven't been able to find a cure for it. But, by merging the power of artificial intelligence and human intelligence, we may be able to step-by-step optimize the cancer treatment process, from screening to effectively diagnosing and eradicating cancer cells.

Thus, the correct diagnosis of BC and classification of patients into malignant or benign groups can improve the prognosis and chances of survival for patients significantly, as it can promote timely clinical treatment to patients.

2. Dataset Definition

The dataset used for analysis is WDBC (Wisconsin Diagnostic Breast Cancer) dataset. The dataset contains 569 instances with 32 attributes (ID, diagnosis (B/M), 30 real-valued input

features). This dataset consists of 10 real-valued features that are computed for each cell nucleus:

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. Data Preprocessing

We are using a Jupyter Notebook to work on the dataset. First we import all the necessary dependencies and load the dataset in a .csv file format to the dataframe. Then we perform **Exploratory Data Analysis** to understand the basic properties and pattern of the data. Creating visuals helps people understand the data set without manually reading the entire range of rows and allows for digestible pieces of information that sometimes code and predictive analytics can't accurately portray.

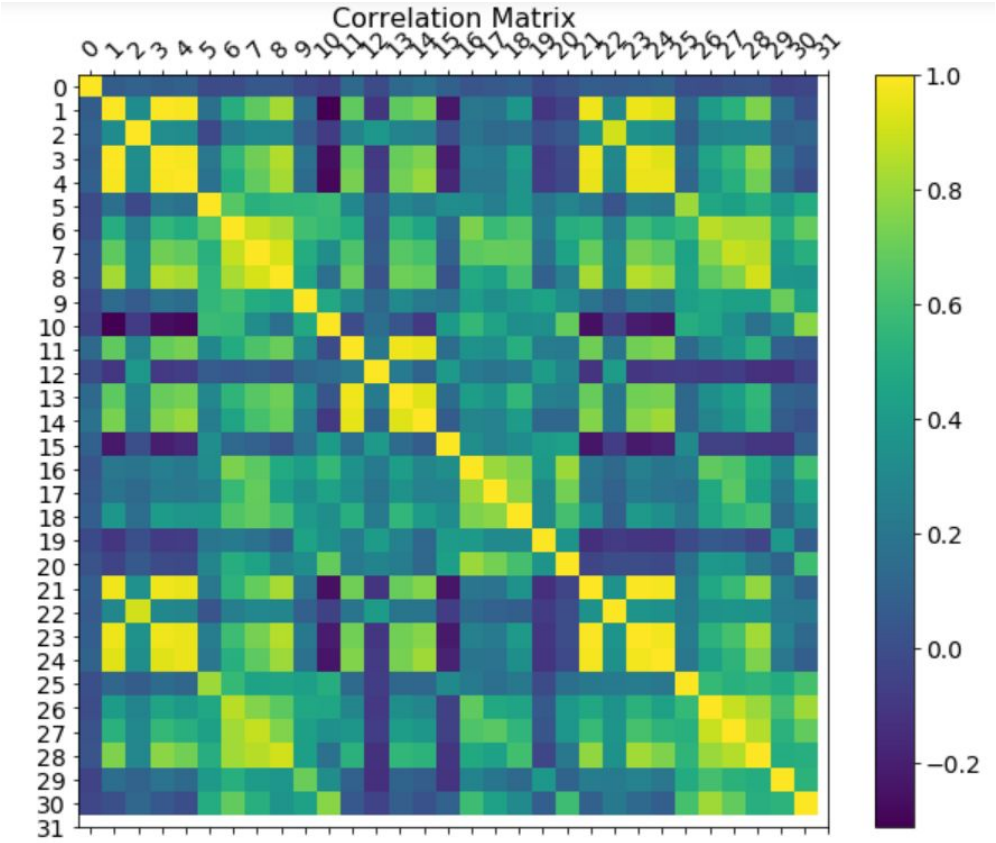
											...								
566	926954	M	16.600	28.08	108.30	858.1	0.08455	0.10230	0.092510	0.053020	...	18.980	34.12	126.70	1124.0	0.11390	0.30940	0.34030	0.14180
567	927241	M	20.600	29.33	140.10	1265.0	0.11780	0.27700	0.351400	0.152000	...	25.740	39.42	184.60	1821.0	0.16500	0.86810	0.93870	0.26500
568	92751	B	7.760	24.54	47.92	181.0	0.05263	0.04362	0.000000	0.000000	...	9.456	30.37	59.16	268.6	0.08996	0.06444	0.00000	0.00000

569 rows × 32 columns

We find the dimensions of the data set using the panda dataset 'shape' attribute and observe that the data set contains 569 rows and 32 columns.Then we find any missing or null data points of the data set.

```
0    0.0
1    0.0
2    0.0
3    0.0
4    0.0
dtype: float64
```

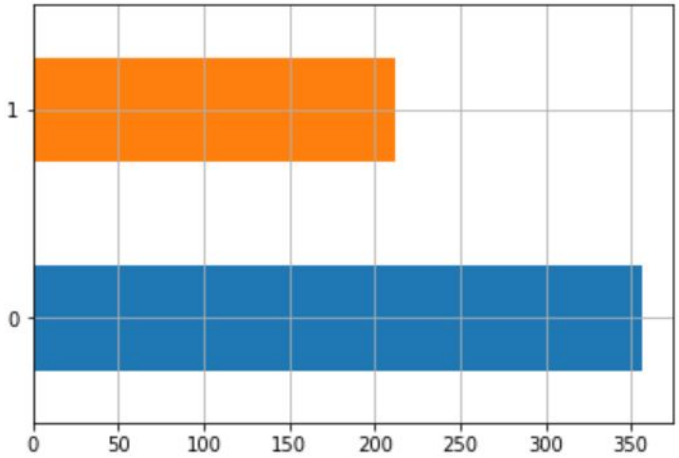
We can also perform Visual Exploratory Analysis with Correlation Matrix and Pairplot:



Now we find that the Id column is redundant and not useful, So we drop it.

'Column 2' is the column which we are going to predict, which says if the cancer is M = malignant or B = benign and it contains categorical data, meaning that it consists of labeled values instead of numerical values so we convert it to binary representation where 1 = **Malignant** and 0 = **Benign**.

We can identify that out of the 569 persons, 357 are labeled as B (benign) and 212 as M (malignant).



Now we split our dataset into train (80% of the data), test (20% of the data) and validation data(10% of the test data).The training set contains a known output and the model learns on this data to be generalized to other data later on. The test dataset (or subset) to test our model's prediction on this subset. The validation dataset is used to give an estimate of model skill while tuning the model's hyperparameters.

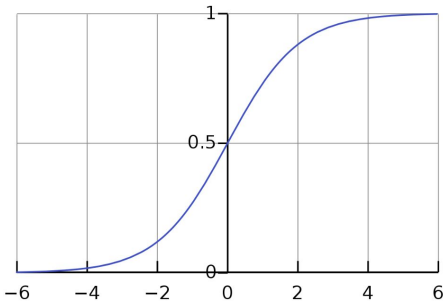
Then we normalize the data to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values.

4. Model Architecture

Logistic Regression

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable. It is a predictive analysis algorithm and uses the concept of probability.

The name logistic regression comes from something known as the *logistic function*, also known as the *sigmoid function,* rising quickly and maxing out at the carrying capacity of the environment. Logistic regression algorithm also uses a linear equation with independent predictors to predict a value. The predicted value can be anywhere between negative infinity to positive infinity. We need the output of the algorithm to be a class variable, i.e. 0- No, 1- Yes. Therefore, we squash the output of the linear equation into a range of [0,1] we use the sigmoid function to squash the predicted value.



The linear equation for the above curve can be represented as:

$$z = \theta_0 + \theta_1 . x_1 + \theta_2 . x_2 + ...$$

To ensure the output falls between 0 and 1, we can squash the linear function into a sigmoid function. The linear function becomes:

$$z = (\theta^T)x + b$$

$$a = \sigma(z)$$

where Sigmoid Function(σ) = $\frac{1}{1+e^{-z}}$

The function (σ), maps any real number to the (0, 1) interval, making it useful for transforming an arbitrary-valued function into a function better suited for classification.

a will give the **probability** and if the predicted probability is greater or equal to 0.5, then we can label it as '**Malignant**'; if the predicted probability is smaller to 0.5, then we can label it as '**Benign**'.

The cost function for logistic regression looks like

$$cost\,function = -\frac{1}{m}\sum_{i=1}^{m} y\ log\ \sigma(z) + (1-y)\ log(1-\sigma(z))$$

Now To reduce the cost function we introduce the idea of Gradient Descent which works towards minimizing the cost function.

The general idea of gradient descent is to tweak parameters *w* and *b* iteratively to minimize a cost function. on each parameter:

$$^{repeat\ until\ convergence}\ \{w := w - \alpha\Delta w\}$$

$$^{repeat\ until\ convergence}\ \{b := b - \alpha\Delta b\}$$

Where (α) is the learning rate

An initial value is assigned to *w;* then iteratively we update *w, b* by Learning Rate.

5. Results

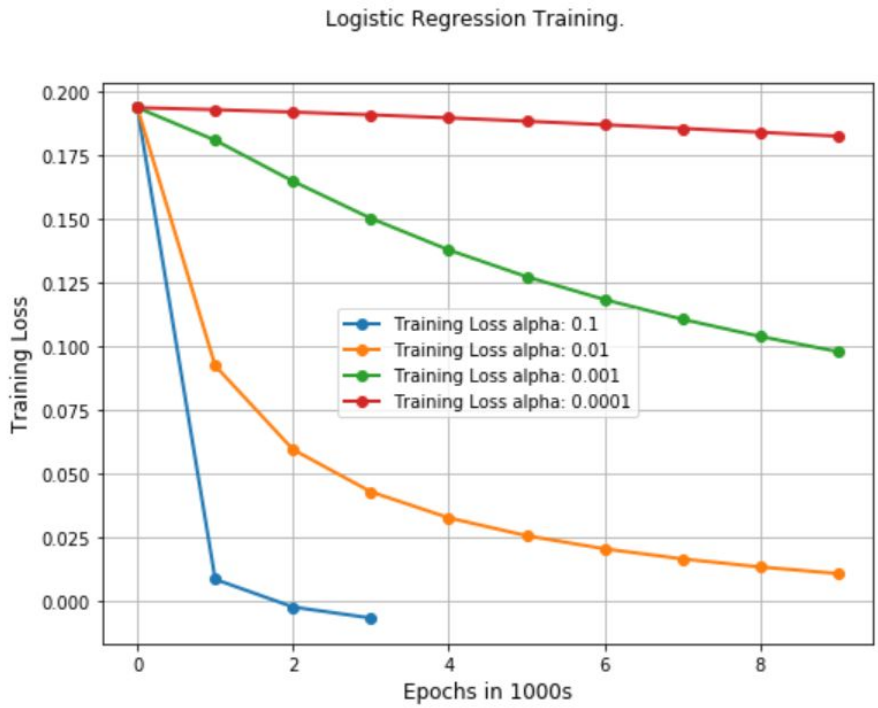
The results obtained in this paper were very accurate with possibility of being better by usage of different techniques, methods of data pre-processing and including more data into the training. We discuss a few such methods below along with the results we obtained by applying them.

Hyperparameter Tuning

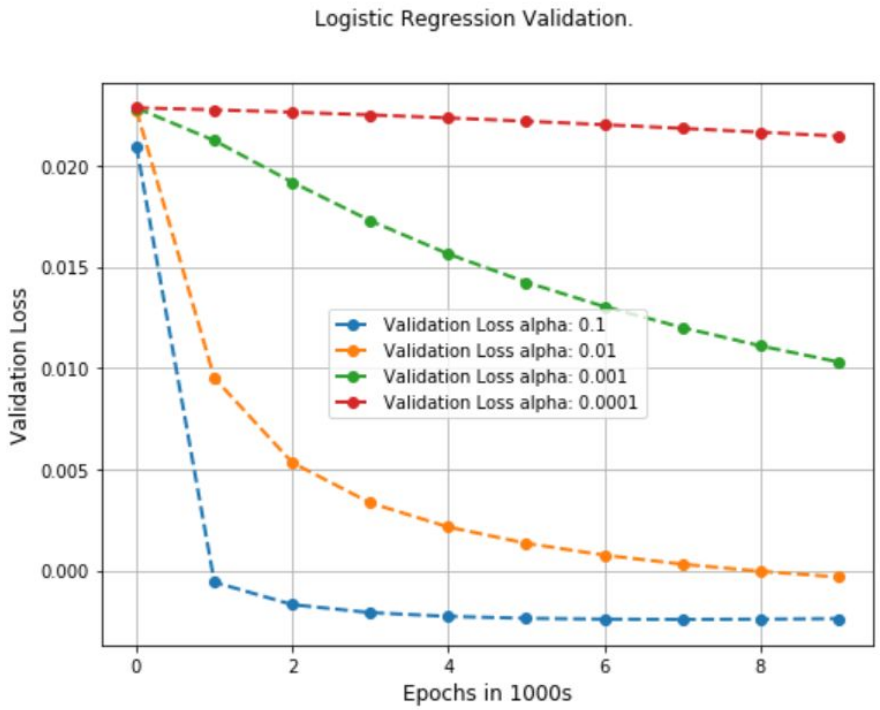
We consider one hyperparameter as learning rate with different values as 0.1, 0.01, 0.001, 0.0001

Then we iteratively update the value of w and b with respect to the Training and Validation data to estimate the best parameter for accuracy.

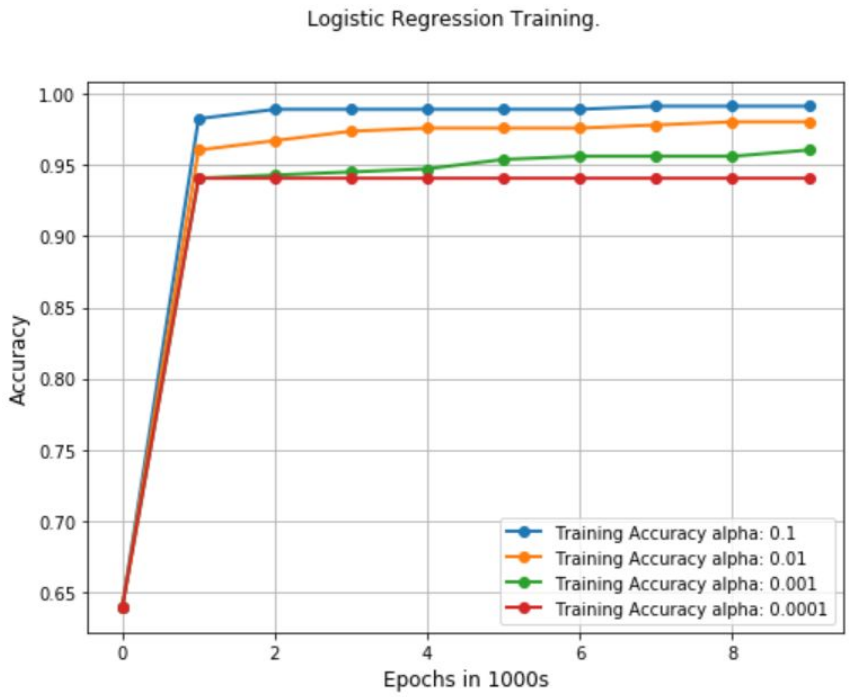
We record the Training loss for different learning rate as:



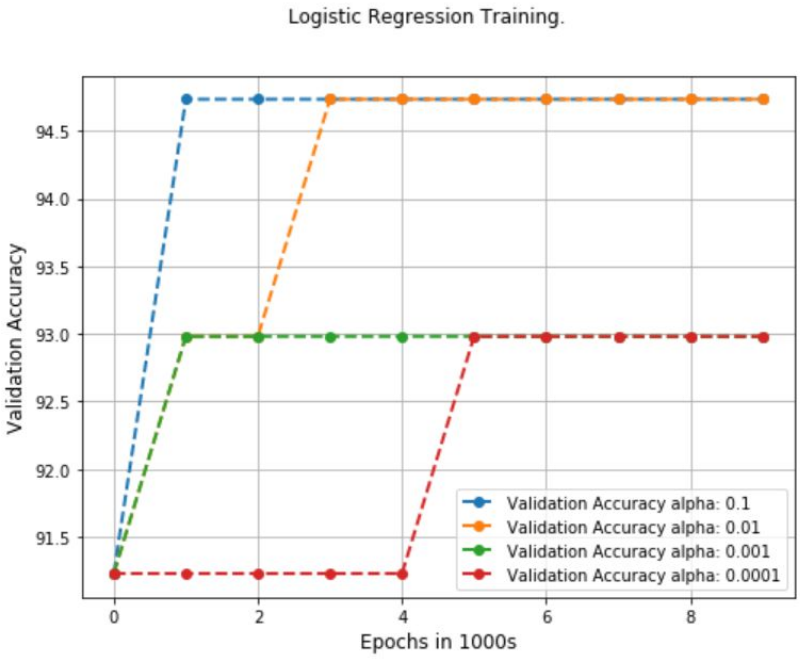
We record the Validation loss for different learning rate as:



We record the Training Accuracy for different learning rate as:

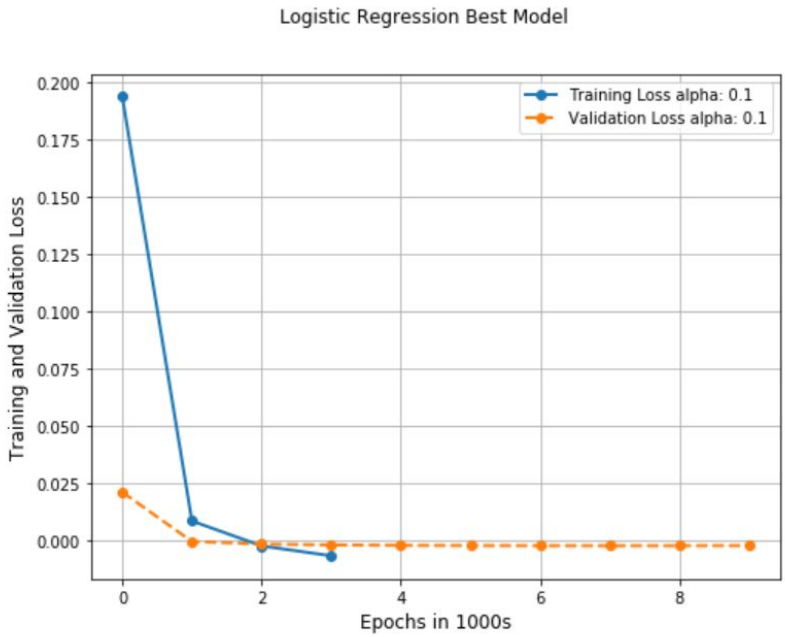


We record the Validation Accuracy for different learning rate as:

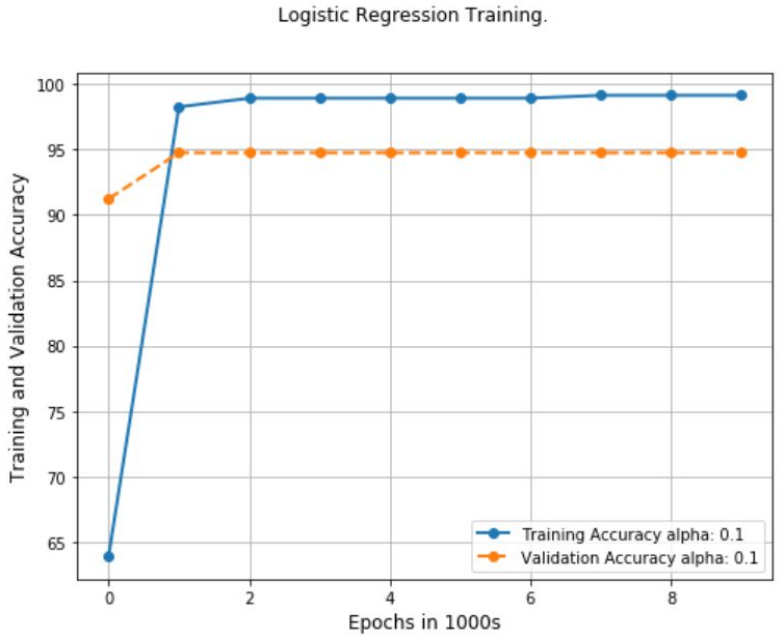


We estimated that the best learning rate parameters for the model is 0.1 which gives us a training accuracy of 99.12%. Then we retrain the model with best hyperparameters

The Training and Validation Loss plot for best parameter(i.e $\alpha = 0.1$):



The Training and Validation Accuracy plot for best parameter(i.e $\alpha = 0.1$):



We now fix the learning rate α and model parameter and test our models performance on the testing set which gives us an Accuracy of 94.74%

	Training Data	Test Data
Accuracy	99.12%	94.74%

Confusion Matrix

To check the accuracy of test dataset we introduce a concept of confusion matrix which we import using the confusion_matrix method of metrics class. The confusion matrix is a way of tabulating the number of misclassifications, i.e., the number of predicted classes which ended up in a wrong classification bin based on the true classes.It is a table with four different combinations of predicted and actual values.

The confusion matrix in our case is as follows:

	TP	FP
FN	30	0
TN	3	24

We will use the Confusion Matrix to calculate the value of Accuracy, Precision and Recall.

Accuracy : We calculate the accuracy as the ratio of the number of correct predictions to the total number of input samples.

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

Precision: We will calculate the precision as the number of correct positive results divided by the number of positive results predicted by the classifier.

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

Recall: We will calculate recall as the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

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

The final results are as follows:

Accuracy	94.74%
Precision	100%
Recall	90.91%

6. Conclusion

The method used in this paper is a simple Logistic Regression model which lacks to take benefit of regularization and performs a limited hyperparameter tuning. Yet, the accuracy achieved is commendable. The scope of improvement of the current model is vast and worth the effort because in consideration of the importance of detecting early stage cancer, even a fraction of improvement in the accuracy would help several patients. We can further improve the accuracy by collecting more data and training the model upon it. A future prospect would be to use neural networks with deep hidden layers.

References:

[1] Analysis of the Wisconsin Breast Cancer Dataset and Machine Learning for Breast Cancer Detection: https://www.researchgate.net/publication/311950799_Analysis_of_the_Wisconsin_Breast_Cancer_Dataset_and_Machine_Learning_for_Breast_Cancer_Detection

[2] The cost function in logistic regression:

https://www.internalpointers.com/post/cost-function-logistic-regression