**Department of Electrical Engineering**

**Faculty Member:** LE Munadi Sial **Date:** 1-Nov-2023

**Semester:** 7th **Group:**

# CS471 Machine Learning

**Lab 8: Logistic Regression**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **PLO4 - CLO4** | **PLO4 -CLO4** | **PLO5 -CLO5** | **PLO8 -CLO6** | **PLO9 -CLO7** |
| **Name** | **Reg. No** | **Viva /Quiz / Lab Performance** | **Analysis of data in Lab Report** | **Modern Tool Usage** | **Ethics** | **Individual and Team Work** |
|  |  | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

## Introduction

This laboratory exercise will focus on the python implementation of logistic regression. Logistic regression is a supervised learning technique that incorporates a sigmoid function activation with the linear regression algorithm to implement classification. Unlike regression, classification involves discrete labels such 0/1, true/false, cat/not a cat, benign/malignant etc. The sigmoid function causes the hypothesis values to take place between 0 or 1. Similar to regression, weight parameters are trained on a dataset so as to fit a model that can make accurate predictions from that dataset. To prevent overfitting, regularization can be used in logistic regression.

## Objectives

The following are the main objectives of this lab:

* Extract and prepare the training and test datasets
* Use feature scaling to ensure uniformity among the feature columns
* Implement cost function to get the overall loss
* Implement gradient descent algorithm to update weight parameters
* Plot the training and test losses
* Make scatter plots of the classification

## Lab Conduct

* Respect faculty and peers through speech and actions
* The lab faculty will be available to assist the students. In case some aspect of the lab experiment is not understood, the students are advised to seek help from the faculty.
* In the tasks, there are commented lines such as #YOUR CODE STARTS HERE# where you have to provide the code. You must put the code/screenshot/plot between the #START and #END parts of these commented lines. Do NOT remove the commented lines.
* Use the tab key to provide the indentation in python.
* When you provide the code in the report, keep the font size at 12

**Theory**

Linear Regression is a very basic supervised learning technique. To calculate the loss in each training example, the difference between a hypothesis and the label (y) is calculated. The hypothesis is a linear equation of the features (x) in the dataset with the coefficients acting as the weight parameters. These weight parameters are initialized to random values at the start but are then trained over time to learn the model. The cost function is used to calculated the error between the predicted y^ and the actual y.

A brief summary of the relevant keywords and functions in python is provided below:

**print()** output text on console

**input()** get input from user on console

**range()**  create a sequence of numbers

**len()** gives the number of characters in a string

**if** contains code that executes depending on a logical condition

**else** connects with **if** and **elif**, executes when conditions are not met

**elif** equivalent to **else if**

**while** loops code as long as a condition is true

**for** loops code through a sequence of items in an iterable object

**break** exit loop immediately

**continue** jump to the next iteration of the loop

**def** used to define a function

**Lab Task 1 - Dataset Preparation, Feature Scaling \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

You have been provided with a dataset containing several feature columns. You will need to select any 3 of the feature columns to make your own dataset. The “y” is the label column that your model will predict. The dataset examples are to be divided into 2 separate portions: training and test datasets (choose from 80-20 to 70-30 ratios). Save the prepared datasets as CSV files. Next, load the datasets into your python program and store them as NumPy arrays (Xtrain , ytrain, Xtest, ytest,). Next, use feature scaling to rescale the feature columns of both datasets so that their values range from 0 to 1. Finally, print both of the datasets (you need to show any 5 rows of the datasets).

***### TASK 1 CODE STARTS HERE ###***

*### TASK 1 CODE ENDS HERE ###*

***### TASK 1 SCREENSHOT STARTS HERE ###***

*### TASK 1 SCREENSHOT ENDS HERE ###*

**Lab Task 2 – Sigmoid Activation \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

For logistic regression, you will implement the following hypothesis:

h(x) = g(w0 + w1x1 + w2x2 + w3x3 + …)

g(z) = 1 / (1 + e-z)

The w represents the weights and the x represent the features. h(x) is to be calculated for each training example and its difference with the label y of that training example will represent the loss. The g(z) function represents the sigmoid activation function. In this task, you will write a function that takes in a value z as argument and outputs the result of the sigmoid activation g(z). Provide the code for this task:

***### TASK 2 CODE STARTS HERE ###***

*### TASK 2 CODE ENDS HERE ###*

**Lab Task 3 - Cost Function \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

In this task, you will write a cost function that calculates the overall loss across a set of training examples. This cost function will be useful to calculate the losses in both the training and test phases of the program.

cost\_function(X, y)

The X and y are the features and labels of the training/test dataset. The function will return the cost value. The cost function is given by:

The m is the number of the training/test examples in the dataset. Remember that the hypothesis requires the sigmoid activation. Write the code for the cost function and implement it to print out the cost. You will need to initialize the weights to some random values in order to calculate the hypothesis. Provide the code and all relevant screenshots showcasing the use of your cost function.

***### TASK 3 CODE STARTS HERE ###***

*### TASK 3 CODE ENDS HERE ###*

***### TASK 3 SCREENSHOT STARTS HERE ###***

*### TASK 3 SCREENSHOT ENDS HERE ###*

**Lab Task 4 – Gradient Descent \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

In this task, you will write a function that uses gradient descent to update the weight parameters:

gradient\_descent(X, y, alpha)

The X and y are the features and labels of the training dataset, *alpha* is the learning rate which is a tuning hyperparameter. The gradient descent algorithm is given as follows:

The gradient descent for logistic regression may seem identical to that in linear regression, however, it should be noted that they are not the same formulas as the hypothesis for the logistic regression is different from that of linear regression. Provide the code and all relevant screenshots showcasing the use of your gradient descent function.

***### TASK 4 CODE STARTS HERE ###***

*### TASK 4 CODE ENDS HERE ###*

***### TASK 4 SCREENSHOT STARTS HERE ###***

*### TASK 4 SCREENSHOT ENDS HERE ###*

**Lab Task 5 – Training and Test Implementation \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

In this task, you will use the functions from the previous two tasks to write a “main” function that performs the actual training and testing. Use the cost function and gradient descent function on the training examples to determine the training loss and update the weights. Then, use the cost function on the test examples to determine the test loss. This single iteration over the entire dataset (both training and test) marks completion of one epoch. You will need to perform the training and testing over several epochs (the epoch number is another hyperparameter that must be chosen). Ensure that at the end of each epoch, the training loss, test loss, precision value and recall value are stored for plotting purposes. To get the precision and recall, use the following:

Start the training at some value of alpha. Try multiple training attempts with varying alpha values and find the best value of the alpha. Once you have found the best alpha value, showcase the output by making three plots:

1. training loss and test loss vs. the epoch number
2. precision and recall vs. the epoch number
3. precision vs. recall

Ensure all axes are labeled appropriately. Provide the code (excluding function definitions) and all plots of the final output.

***### TASK 5 CODE STARTS HERE ###***

*### TASK 5 CODE ENDS HERE ###*

***### TASK 5 PLOT 1 STARTS HERE ###***

*### TASK 5 PLOT 1 ENDS HERE ###*

***### TASK 5 PLOT 2 STARTS HERE ###***

*### TASK 5 PLOT 2 ENDS HERE ###*

***### TASK 5 PLOT 3 STARTS HERE ###***

*### TASK 5 PLOT 3 ENDS HERE ###*

**Lab Task 6 – Prediction and Scatter Plot \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Save the weights that fit the best model and use them to create a prediction function. The prediction function will take the features as input and output the predicted class of the label. Additionally, the prediction function must make a scatter plot showing the training and test examples. The coordinates in the scatter plot correspond to the inputs (x). The class is denoted by (red) O and (blue) X for 0 and 1 respectively. The predicted value must be shown as a black O or X. Provide the code and screenshot of is being used.

***### TASK 6 CODE STARTS HERE ###***

*### TASK 6 CODE ENDS HERE ###*

***### TASK 6 SCREENSHOT STARTS HERE ###***

*### TASK 6 SCREENSHOT ENDS HERE ###*