**Department of Electrical Engineering**

**Faculty Member:** LE Munadi Sial **Date:** 18-Oct-2023

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

# CS471 Machine Learning

**Lab 6: Linear Regression I - Feature Scaling, Cost Function and Gradient Descent**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **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 implementation of linear regression in python. Linear regression is a basic supervised learning technique which serves as a good starting point for learning supervised learning and also sets the fundamental basis for learning other machine learning techniques. In linear regression, a dataset with various features and a label is trained. It consists of weighted parameters that are trained to fit a model that best approximates the dataset.

## Objectives

The following are the main objectives of this lab:

* Extract and prepare the training dataset
* Use feature scaling to ensure uniformity among the feature columns
* Implement cost function to get the overall loss
* Implement gradient descent algorithm to train the weight parameters
* Plot the training loss
* Use a prediction function to use the trained model

## 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. This cost is used to determine how the weights are to be adjusted in what is called the gradient descent algorithm. The gradient descent uses a step size (alpha) as a hyperparameter which can be tuned. This hyperparameter is varied to determine the model that best fits the dataset.

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 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

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 “Sale Price” is the label column. Load the dataset into your python program as NumPy arrays (Xtrain ,ytrain). Print 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 - Feature Scaling \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

In the input matrix (Xtrain), use feature scaling to rescale the feature columns so that their values range from 0 to 1:

You will use these rescaled values in the upcoming tasks. Print the rescaled dataset (you need to show any 5 rows of the datasets).

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

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

***### TASK 2 SCREENSHOT STARTS HERE ###***

*### TASK 2 SCREENSHOT ENDS HERE ###*

In the tasks 3 and 4, you will write the cost function and gradient descent function respectively. In task 5, you will use both of these functions in order to perform linear regression.

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

For linear regression, you will implement the following hypothesis:

h(x) = b + w1x1 + w2x2 + w3x3

The wj and b represent the weights while the xj represents the features. The feature number is denoted by j. The linear hypothesis 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. Initialize the weights and bias to random values between 0 and 1.

In this task, you will write a cost function that calculates the overall loss across a set of training examples:

cost\_function(X, y)

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

The m is the number of the training examples in the dataset. Write the code for the cost function and implement it to print out the cost. Provide the code and all relevant screenshots of the final output.

***### 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 Algorithm \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

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:

Provide the code and any relevant screenshots of the final output.

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

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

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

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

**Lab Task 5 – Training across Epochs \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

In this task, you will use the functions from the previous two tasks to write a “main” function that performs the actual training. Use the cost function on the entire training dataset to determine the training loss. You will need to store this training loss for plater plotting. Next, use the gradient descent function to update the weights and bias. This iteration over the entire dataset is called an “epoch”. You will need to perform the training over several epochs (the epoch number is a hyperparameter you must select at the start of the training). Thus, you will compute training loss and weight update at each epoch. At the last epoch, note down the final weight values and plot the training loss (y-axis) over the epochs (x-axis). Provide the code (excluding function definitions of tasks 2 and 3) as well as all releveant screenshots of the final output.

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

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

***### TASK 5 SCREENSHOT STARTS HERE ###***

*### TASK 5 SCREENSHOT ENDS HERE ###*

**Lab Task 6 – Hyperparameter Tuning \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

In this task, you will use your code from the previous task. Tune the alpha hyperparameter at different values to get various plots. You will need to provide at least 3 plots. Mention the alpha value in the plot titles. Ensure all the axes are labeled appropriately. Note down the weights at the final epochs.

***### TASK 6 PLOTS START HERE ###***

*### TASK 6 PLOTS 1 END HERE ###*