# Department of Computing

# CS370: Artificial Intelligence

# Class: BSCS-8C

# Lab 02: Optimization

# Date: 05-03-2021

# Time: 10:00-1:00

# Instructor: Dr. Hashir Kiani

# Lab 02: Optimization in Machine Learning

**Introduction**

Since machine learning algorithms are implemented on a computer, the mathematical formulations are expressed as numerical optimization methods. Training a machine learning model often boils down to finding a good set of parameters. The notion of “good” is determined by the objective function (loss function). Given an objective function, finding the best value is done using optimization algorithms.

If our objective function is differentiable, we have access to a gradient at each location in the space to help us find the optimum value. By convention, most objective functions in machine learning are intended to be minimized, that is, the best value is the minimum value. Intuitively finding the best value is like finding the valleys of the objective function, and the gradients point us uphill. The idea is to move downhill (opposite to the gradient) and hope to find the deepest point.

**Objective**

The objective of this lab is to implement optimization algorithms including Gradient Descent and Stochastic Gradient Descent.

**Tools/Software Requirement**

Python, & its libraries

**Description**

Please go through the lecture slides and recordings for detailed description of gradient descent and stochastic gradient descent algorithm.

## 

**LAB TASKS**

**Task 1:**

Write code in Python to fit a linear regression model to the data given below. Use squared loss as the loss function. Optimize the parameters of your model using gradient descent algorithm.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | 1 |
|  | 3 |

**Task 2:**

Change your code in Task 1 to introduce artificially generated random training examples instead of using the data given in Task 1. For this task, you need to generate 100,000 training examples. The input vector for each training example should be 8-Dimensional. As you are doing linear regression so output would be a real scalar value. You can generate artificial data by reverse engineering. Start with a final value of weight vector and use it to find the score for each random input. Then add some random noise to that score to get appropriate output. Again use squared loss as the loss function and optimize the parameters of your model using gradient descent algorithm.

**Task 3:**

Change your code in Task 2 to implement stochastic gradient descent algorithm instead of normal gradient descent. Use the same training data as in Task 2 and use squared loss as the loss function to optimize the parameters of your model.

**Deliverables:** submit a report containing code and screenshots of output.

**Submission Time**: **check lms.**