Laboratory Assignment 2
Optimising Parameters for a Linear Regression Model using Manual Search, Grid Search and Random Search

CSC372-M72: Optimisation

2022-23

You are supposed to do this lab on the 19th and the 26th October. For the first session, please concentrate on completing sections 2.1 and 2.2. We will cover the materials for section 2.3 in the lecture on the 20th of October.

1 Objectives.

- To explain how the parameters in a linear regression model affect the prediction.
- To apply a manual search and gain an appreciation for how changes in parameters may affect an objective function.
- To implement a simple function encapsulating the objective function for fitting a linear model for a given dataset.
- To implement Grid Search and Random Search to identify a good approximation of the optimal solution.

2 Tasks

2.1 Exploring How Linear Regression Modelling Work in Spreadsheet

A linear model is one of the simplest regression techniques, but it captures all the key aspects of data-driven modelling and machine learning. You are usually given a set of data: usually, you have a (or more) predictor x and a predictand y. We may believe that x and y have a linear relationship between them, i.e. if I increase x by a certain amount y increases linearly, and *vice-versa*. This relationship is often described with the following equation:

$$y = mx + c,$$

where m is the slope of the linear relationship and c is the intercept. In machine learning literature, you may find that they denote the slope as weight and the intercept as bias.

Now, of course, if someone is simply giving us the data – a bunch of measurements of ys for respective xs – we do not quite know the original m and c of the model, so we cannot exactly predict the outcome y for an unknown x. To make matters further complicated, the data we are given often

has noise, as in we cannot exactly measure the predictand and thus the values we got from our measurements are slightly off. So, we want to locate an appropriate set of values for m and c such that we fit the data best. To fit the data best, we often try out various different ms and cs, and return the values for which we received the *lowest mean squared error*. The whole idea is that if we can correctly (or with a small error) predict the ys for the given xs, then we should be able to make good predictions of ys for unseen xs. The predictions from the model are often denoted as $\hat{y}$s to distinctly identify them from given ys, which are the measured responses for various xs.

Clearly this is an optimisation problem. In this context, our independent variables are $\boldsymbol{\theta} = (m, c)^\top$ and the optimisation problem is:

$$\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) = \sum_{i=1}^{M} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{M} (y_i - mx_i - c)^2,$$

when you are given M data points.

Of course, there are known solutions for this problem, but we are going to go back to ancient times and try out the manual methods to gain an appreciation for what we are actually doing when we optimise a problem, and appreciate the pain our ancestors had to endure without modern optimisation methods.

Your next step is to download the spreadsheet entitled "linearRegression.xlsx" from the Canvas page of this lab assignment. In the spreadsheet, there are a couple of cells that you can alter: they represent the slope m (cell A3) and the intercept c (cell B3). If you change these variables you will see that the predicted line in orange will move. Your task is to move the line in a position such that it fits the data (plotted with blue squares) best.

You are given that the optimal $f(\boldsymbol{\theta}^*) = 2.4$ (in cell C3). **Can you find the $\boldsymbol{\theta}^*$ for this problem by manually changing m and c?**

2.2 Implementing the Squared Error Function in Python

Implement the squared error function defined in Section 2.1 using Python (in Jupyter Notebook). The function should take an array $\boldsymbol{\theta} = (m, c)^\top$ and two matrices X and Y which contains the training dataset. Use the dataset from the spreadsheet and your values for m and c to test your function.

For your convenience, the training data is given below.

$$X = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix} \text{ and } Y = \begin{bmatrix} 2 \\ 4 \\ 5 \\ 4 \\ 5 \end{bmatrix}$$

2.3 Implementing Automated Optimisation Algorithms

In this section, your task is to implement the following algorithms to minimise the function that you implemented in Section 2.2:

1. Grid Search.
2. Random Search.

You should implement these algorithms as functions. You are free to think and figure out the required input and output parameters for correct functionality.

When it comes to selecting hyper-parameters for these algorithms, you should select them as you see fit.

Finally, you should repeatedly call these algorithms and see if they produce deterministic or stochastic performance. Plot these final performances in a boxplot.

HINT: You may find `boxplot` function from the `matplotlib` library useful for the last part.