Homework Project 1

Announced on 8/18. Due on Tuesday, September 15th, by noon. Submit via Blackboard. Total points: 15.

Required Background Knowledge

- Derivative (Undergraduate Calculus)
- Chain rule of differentiation (Undergraduate Calculus)
- Partial derivative (Undergraduate Multivariate Calculus)
- Gradient (Undergraduate Multivariate Calculus)
- Python (Undergraduate Programming Languages)
- Numpy Python Library (study on your own)

Objective

Get up to speed with: python, numpy, mean squared error fitting, iterative solution improvement method, and partial derivatives/gradients.

Project Description

Let $x \in \mathbb{R}$ (a single real number), $y \in \mathbb{R}$; a pair (x, y) is a training sample. A training set of size m is a set of m such pairs, (x_i, y_i) for $i = 1, ..., m$. In numpy, you can have a single 1D array for all x_i, and separately a 1D array for all y_i.

For a given $(n+1)$-dimensional vector $w \in \mathbb{R}^{n+1}$, let $h(x, w) = \sum_{j=0}^{n} w_j x^j$ be a polynomial of n-th degree of x with coefficients w_j. For example, for $n = 2$, we will have a 2nd degree polynomial $h(x, w) = w_0 + w_1 x + w_2 x^2$ (if you prefer $ax^2 + bx + c$, substitute $a = w_2$, $b = w_1$, $c = w_0$).

Let $L(h(x), y) = (h(x) - y)^2$ be the squared error objective function $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ showing how good the polynomial h specified by w is at predicting the y from x in a given training sample (x, y). The lower the value of L, the higher the accuracy; ideally, the prediction is perfect, $h(x) = y$, and $L = 0$.

Given a sequence of m pairs (x_i, y_i) – the training set – and the value for n (n=1,2,3,4,5), your task is to write a python/numpy code to find a good set of values w_j for that n, for the given training set. A set of values w_j is good if the objective function averaged over the m training pairs is low - the values w lead to mostly accurate predictions for all samples in the training set.

That is, the task is to write python/numpy code to solve

$$w_{good} \approx \arg\min_{w} \sum_{i=1}^{m} L(h(x_i, w), y)/m.$$

How to Solve It

You are required to follow the following procedure, with only minor changes if it improves your results.

For a given n:

(1) Using pencil and paper, derive the formula for $g(x_i, y_i) = \nabla_w L$, the gradient of L with respect to w, as a function of training sample values x_i, y_i. That is, find the gradient - the vector of partial derivatives $\frac{\partial L}{\partial w_j}(x_i, y_i)$ for $j = 0, ..., n$.

(2) Start with small (e.g. in [-0.001,0.001] range), random values for w_j.
(3) Use your formula to calculate $g(x_i, y_i)$ for all training points, then average them: $g = \sum_i g(x_i, y_i)/m$
(4) modify w slightly: $w_{new} = w_{old} - \gamma g$, where γ is some (very) small positive number, experimentally chosen to lead to good results in not-too-many iterations
(5) repeat the two lines above until the quality of predictions, $\sum_{i=1}^{m} L(h(x_i, w), y)/m$, no longer changes significantly (this can be thousands of iterations)

Once you get the good values of w, plot the function $h(x, w) = \sum_{j=0}^{n} w_j x^j$ in blue color, and the training samples in red color, on the same scatter plot.
Repeat for all $n = 1, 2, 3, 4, 5$.

Organizational Details

Use Python 3.6 or 3.7. A Python distro that is tailored for AI/ML work is here:

`https://www.anaconda.com/products/individual`

One convenient IDE to work with python is Spyder (it comes with Anaconda, it allows you to click on a matrix or array and see its contents).

You are only allowed to import *numpy*, and a plotting library to plot scatter plots. All calculations should by done using numpy library. No machine learning/stats/AI/curve-fitting libraries are allowed.

The training set (x, y) will be uploaded on BB.

Solution code (.py, not a jupyter notebook) + a single PDF with the derivation of the gradient and with 5 plots (for $n = 1, 2, 3, 4, 5$) should be submitted via Blackboard.

This python / machine learning jump-start project is the same as last year. The solution should be your own.