PCML project 1 report

Chenfan Sun
Valentin Le Tourneux De La Perraudi$\grave{e}re$
Ireneu Pla

***Abstract*—TBD**

I. Introduction

This report concerns the first project for the EPFL Pattern Classification and Machine Learning course. The goal of the project was to use the concepts we have learned in class using a real dataset to try to identify if the effect of the Higgs boson was observed.

II. Models and Methods

Our first task was implementing six machine learning methods, which all deliver a model based on the data we train them with. The objective was to have a set of tools, and then to try them, and select the best suited for our problem.

The three first functions we implemented aim to minimise a special cost function : the MSE. It is defined as the mean of the squared errors, i.e. the mean of the square of the difference between our prediction and the real data.

$$MSE(w) := \frac{1}{N}\sum_{n=1}^{N}\left[y_n - f(x_n)\right]^2$$

A. ML techniques

Here is a description of the Machine Learning techniques that we implemented.

Gradient descent

: least_squares_GD(y, tx, initial_w, max_iters, gamma) : this method simply implements a gradient descent, i.e. at each step we move each parameter in the direction where it minimises the loss. This is done by computing the gradient of the loss, using this formula for the MSE : insert formula, and then computing the next step parameters, thanks to this formula : insert formula. Our principal way to influence this method is by choosing gamma, the step size. With a large gamma, we risk to exceed the minimum, and then oscillate around it, but with a small gamma, the descent may be very slow, and then very long to compute.

Stochastic gradient descent

: least_squares_SGD(y, tx, initial_w, max_iters, gamma) : This method is a variation of the one above, because it may be really expensive to compute. In order to accelerate the computing, we do a stochastic gradient descent. The difference is that at each step, we dont compute the mean of the gradient on all the samples, but only on one chosen randomly. In expectation, we get the same gradient direction, but obviously, if we are unlucky, this may not be efficient. This is why we did implement an intermediate version, which is the mini-batch SGD : instead of computing the mean of the gradient on all the samples, or on only one, we compute it on a random subset of samples, and by choosing the size of this subset, we can either be close to the normal gradient descent (large subset), or from the SGD (small subset).

Least squares

: least_squares(y, tx) : This method implements the normal equations, which give the w solution of the minimum loss : insert formula.

B. Feature processing

III. Results

IV. Discussion

V. Summary