Finding the Higgs boson

Robin
Department of Computer Science, EPFL Lausanne, Switzerland Adrian Pace
Department of Computer Science, EPFL Lausanne, Switzerland Yannick Schaeffer
Department of Computer Science, EPFL Lausanne, Switzerland

***Abstract*—The Higgs boson is an elementary particle in the Standard Model of physics which explains why other particles have mass. The Higgs boson decays rapidly into other particles and we observe the resulting signature to determine if it was a Higgs boson or some other particle / process. By using binary classification techniques, we can estimate the likelihood that a given signature was the result of a Higgs boson. This will be helpful to narrow the region where the Higgs is expected to be from the overwhelming number of noise in the dataset.**

I. Introduction

TODO: Find intro, will probably need to have finished the other sections before.

II. Models and Methods

A. Background

1) Basic methods:

general_stochastic_gradient_descent: is the same as the same spirit as the function general_gradient_descent but it perform a stochastic gradient descend loop, this mean only compute correction on part of the sample. all argument are the same as above, with same meaning execept grad_function is gone and we have two new ones. *NOTE : C'EST BIEN DEG LES DESCRIPTIONS, JE DOIS MODIFIER LE TEMPLATE POUR AJOUTER UN NOUVEAU SUBPARAGRAPH.*

batch_size the size of the subset on which we'll compute the "partial gradient" (the g correction term in the course notes)

stock_grad_function the function who will compute the stochastic gradient given a subset of y, a corresponding subset of tx and the current weigth vector w.

the selection of the subset is done inside the function with the help of the batch_iter function from the labs exercises

least_squares_GD: perform a gradient descent with cost function MSE and gradient function the gradient of the MSE function. this is the definition of least square GD. argument have same meaning as their homonyms described above

least_squares_SGD: perform a stochastic gradient descent with cost function MSE and gradient function the gradient of the MSE function. this is the definition of least square GD. argument have same meaning as their homonyms described above

least_squares: solve directly the least square problem using linear algebra, we use numpy linear system solve function to solve the system presented in the course this way as a better accuracy and less rounding error than if we directly use the solution formula who need to compute an inverse

ridge regression: same as the one before expect that we introduce a regularization term in the equation. again we solve the linear system derived in course.

logistic regression: perform logistic regression which is a gradient descent with special cost function and associate gradient function. that what we do we call general_gradient_descent with the new cost and gradient function

reg logistic regression: same as the previous one but with a regularisation term in the cost function. (and so also in gradient function). As general_gradient_descent only call the cost and grad function with the argument y,tx and w we need to fix the lambda_ parameter before passing the cost and gradient function to general_gradient_descent. this is done using the partial function of python

2) Parameters:

There are multiple parameters in the project which can be tuned.

Gamma: is the parameter describing the step of the gradient descent and of the stochastic gradient descent. At each iteration, the new weight will be calculated by adding to the current weights the derivative of the weights times gamma. It mustnt be too high or we will go in the direction of the good weights but miss them and go further. And at each iteration, we will further ourselves from the weights. If it is too small, we will eventually reach the best weights, but it will take a lot of time. Thus, we searched iteratively for the smallest gamma for which we got a reasonable computing time. Least square GD and Least square SGD : gamma=0.000003. Logistic regression and regularized logistic regression : 0.00003 Max iterations: this parameter describes how many steps will we take before stopping the algorithm. The higher, the more finely tuned the weights will get, and the smaller, the shorter the algorithme will be. Least square GD and Least square SGD : max iterations=1000. Logistic regression and regularized logistic regression : max iterations=800

Lambda: is the parameter used to avoid overfitting. For this, we did a cross-validation (with 4 folds) and check the test error. We tried it for 30 different lambdas and we saw that there was no lambdas for which the test error was extremely different from the train error. We can then conclude that we dont have overfitting. This is also because we use linear models. So lambda=0 Which function we will use to estimate the weights. We decided to use the least square method. We chose this before the least square GD and SGD since it yielded a better mse (0.3423 against 0.3616). We chose Least square over logistic regression, since when we applied the weights to the input data, classified it, and compared it to the original y, we saw that Least square was more effective than logistic regression (64695 misclassifications against 68921).

B. *Final implementation*

TODO: When finalized describe here the final implementation with the reasoning behind it.

III. Results

TODO: Present our results.

IV. Discussion

TODO: Explain our results strength and weaknesses.

V. Summary

TODO: Conclusion on how tis will be helpfull in this research area.