The Higgs Boson Particle Machine Learning Challenge

Anselmet Marie, Dandjee Sofia, Monnet Héloïse
Tshtsh club

***Abstract*—In order to recreate the process of discovering the Higgs Boson particle, machine learning techniques were applied to CERN particle accelerator data. This paper explains the decision-making process that led to a Higgs Boson signal accuracy prediction of 82,8%, by using specific data pre-processing and a hyper-tuned ridge regression algorithm.**

I. Introduction

This project deals with a machine learning classification problem applied to particle physics: detecting the Higgs Boson. The Higgs Boson is an elementary particle which is part of the Standard Model of physics and that explains why particles have mass. A way of generating this particle is to smash protons into one another at very high speeds [1]. Scientists use several measurements that come from its rapid decay to detect its presence. Using those measures, the goal of the project was to implement a model in order to accurately predict the signal of a Higgs Boson particle. Linear models with no regularization such as the least squares method and linear gradient descent were first used. The benefits of augmenting the features were explored by implementing the regularized logistic regression and the ridge regression methods. Training and test accuracy as well as F1 score were used to assess the performance of the models implemented.

II. Data Pre-Processing

A. The dataset

The dataset, which can be downloaded on the Kaggle website [2], come from experiences done at the Large Hadron Collider at CERN in Geneva. The training data consists of 250,000 events labelled either 1 (Higgs signal) or -1 (other particle). The Higgs Boson decay's signature is represented in the dataset by 30 features. These are either raw variables measured by a detector or quantities computed from the primitive features [3]. The test data contains 568,238 unlabelled samples.

B. Dealing with the categorical feature 'PRI jet num'

In order to pre-process the data, the first step was to divide the samples depending on their values for the feature called "PRI jet num". Indeed, it was observed in the data that this feature can take only three categorical values: either 0, 1 or 2 (and above). By providing a specific training for the data belonging to each of the three categories, we hoped to improve the precision of our model and to consequently gain in global prediction accuracy. Thus it was decided to split the data in three corresponding groups and to work subsequently with each group separately. In the following sections, the group with 0, 1 and 2 (and above) as jet values will be referenced as 'group 0', 'group 1' and 'group 2' respectively.

C. Cleaning the data

The second step was to standardize each group of samples, as the features have different ranges. The mean and standard deviation (calculated ignoring the undefined values) were respectively subtracted and divided for each dimension. This resulted in each feature having a mean of 0 and a standard deviation of 1. Undefined values of -999 were corrected. Indeed, those are outliers, resulting from a mistake during data measurement or collection, and have a strong influence on the final parameters of the model. Features full of undefined values were removed and the remaining outliers were replaced by 0 (the mean after standardization). Also, features that contained constant values were removed, as they would have no impact on the prediction. They were detected thanks to their standard deviation equal to 0 (before standardization). Finally, the validation and training set were standardized with the training mean and standard deviation.

III. Methods and Results

A. Basic linear models

Firstly, a linear regression using the gradient descent algorithm was implemented to find the optimal weights. Choosing the optimal value of the hyper-parameter γ was a crucial step since an accurate step-size enables to quickly join the minimum of the cost function. It was thus decided to perform a cross validation. By randomly partitioning the data into k folds, the cross-validation limits the bias in the estimation of the error and its variance during the determination of the optimal learning rate, since all the data is used both for training and validation. Performing this step all along the project thus helped to approximate more confidently the optimal value for a given parameter. With a random initial weight vector, a number of iterations of 100, 4 folds for the cross-validation, the best value of γ was found to be 0.1521. Training and test accuracies of 74,44 $\%$ and 74,5 $\%$ respectively were obtained by running 1000 iterations.

The stochastic gradient descent algorithm was then implemented. The parameters used were a random initial weight vector, 100 iterations, a mini batch-size of 1 and 2 folds for the cross validation. The optimal value of λ was found to be 0.0155. This method was faster because of the cheaper computational cost of the algorithm, but the accuracy was lower. With a final number of 1000 iterations, the training accuracy was 66.91 $\%$ and the test accuracy was 66.9 $\%$.

Then, a least squares regression using normal equations was used to find the optimal weights. The training and test

accuracy reached respectively 74.46 % and 74.5 %, which is approximately the same as the gradient descent.

Thus, the least squares method and the gradient descent converge to the same results. We can assume that the stochastic gradient descent would have too if it was run on more batches.

B. Regularized models with augmented features

To increase the representational power of the models that would be used later on, features were "augmented". It means that a polynomial basis of fixed degree was added to each feature. This degree was chosen in order to improve the model fitting. The augmented features were then given as input to models that have regularization factors to avoid over-fitting and penalize complex models. The two regularized models implemented were the ridge regression and the regularized logistic regression. For both methods, a 4-fold cross validation was performed in order to find the optimal degree over a range of (1,20) for the 3 groups. The initial weight was initialized to 0 and a mini-batch size of 1 was used for the logistic gradient descent.

For the logistic regression, $\lambda = 1e{-}3$ and $\gamma = 1e{-}2$, found to be optimal for the linear data, were fixed during the search for the best degree. The augmented features proved to improve the model's performance only up until the second degree. A number of 1000 iterations gave an overall training and test accuracy of 72.18 % and 72.1 %.

One way to improve the penalized logistic regression would have been to increase the number of batches and iterations but this would have been very time costly. This is why the focus was made on the ridge regression algorithm. For this method, the maximum training accuracy was attained for a polynomial degree of 12. Fixing the degree to its optimal value, a 4 cross-validation was done to look for the best regularization factor over the range of (1e−9,1e−2). A fine search for the optimal λ was finally performed by narrowing the original range tested for each group.

As an example, the results of the cross-validations for the degree and λ of the ridge regression for 'group 0' are shown on *Figure 1* and *Figure 2*. The results of the cross-validations for the three jet groups are summarized in *Table I*. The obtained optimal values for λ for 'group 0', 'group 1', 'group 2' were 0.000147, 1e−3, and 0.000464 respectively. We obtained an overall training and test accuracy of 82.86 % and 82.8 % respectively for this hyper-tuned model.

Table I
OPTIMAL PARAMETERS FOR THE FINAL RIDGE REGRESSION MODEL (DETERMINED BY 4-FOLD CROSS VALIDATIONS)

Jet group	Degree	λ before fine search	λ after fine search
Group 0	12	1e−4	0.000147
Group 1	12	1e−3	1e−3
Group 2	12	1e−3	0.000464

It can be seen on *Figure 1* that the training accuracy slightly increases until it reaches its maximum at degree 12. There is then a sharp drop in the accuracy, probably because the model becomes too complex to train.

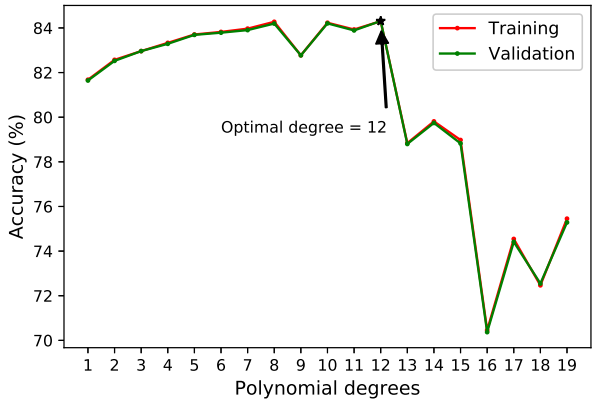

Figure 1. Training and validation accuracies as a function of the polynomial degree for group 0, determined by 4-fold cross validation with fixed $\lambda = 1e{-}3$.

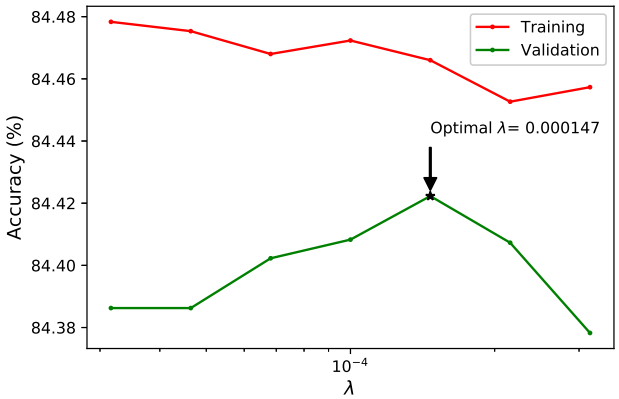

Figure 2. Training and validation accuracies as a function of λ for group 0, determined by 4-fold cross validation with optimal degree value of 12.

Looking now more in detail to *Figure 2*, the training accuracy is better for λ values between 1e−7 and 1e−3. Below 1e−7, this decrease can be explained by the fact that λ will not penalize enough sparse weights. On the contrary, above 1e−3, λ will penalize too much sparse weights.

IV. SUMMARY

The first crucial step of solving the Higgs Boson classification problem was to well understand the data from which we tried to learn and to apply appropriate data pre-processing. The linear models implemented with the gradient descent, stochastic gradient descent and least squares algorithms were too simple to accurately predict the labels of the test data. Thus, it was essential to augment features with a polynomial basis and to use regularized models such as ridge regression and regularized logistic regression to avoid over-fitting. The final hyper-tuned ridge regression model enabled us to reach an accuracy of 82.8 % in the prediction of the Higgs Boson signal and an F1 score of 0.738 on the test data.

REFERENCES

[1] "Learning to discover: the higgs boson machine learning challenge," https://higgsml.lal.in2p3.fr/files/2014/04/documentation_v1.8.pdf, 2014, accessed: 2010-10-28.

[2] "Higgs boson machine learning challenge - data," https://www.kaggle.com/c/higgs-boson/data, accessed: 2010-10-28.

[3] "Epfl machine learning higgs 2019," https://www.aicrowd.com/challenges/epfl-machine-learning-higgs-2019, accessed: 2010-10-28.