The Higgs boson machine learning challenge

Tom MERY Maxime LELIEVRE Matteo PEDUTO

***Abstract*—The Higgs boson challenge is a classification problem based on the ATLAS and CMS experiment. This report is explaining the analysis performed by the group. The assumptions and results obtained are explained and analysed. The specifications of the models are also precisely presented.**

I. Introduction

The Higgs boson is an elementary particle discovered at the Large Hadron Collider at CERN in 2013 that explains why other particles have mass. The goal of this machine learning project is to reproduce the "discovering" process of this particle. By analyzing the decay signature of the collision events, one can predict weather the given event's signature was the result of a Higgs boson or some other particle. The model is based on a vector of features of the collision of two protons smashing into one another at high speeds. Through this report, the reader is able to understand the different assumptions, decisions and results made during the project.

II. Data Analysis

The training set consists of 250'000 data points with 30 features and one column with the labels to predict (s or b). Label 's' stands for signal (predicted as a Higgs boson) and label 'b' stands for background (predicted as something else). The test set has a total of 568'238 data points. Pre-processing the data is critical before starting to implement methods and to fit models. In fact, the data analysis is directly impacting the latter.

A. Data set balancing

First of all, it is important to realize that the data is unbalanced (66%-34%). This makes the accuracy irrelevant. The F1-score is used to check the performance of the models. The test is indeed better in those conditions. [2]

B. Missing values

Even though outliers don't have an important part in the data set, the missing values are primordial. They are replaced by -999 and present with a frequency that is not negligible in the features. There is at least one of those values in 72% of the events. Suppressing those events would cause a loss of information too important that would impact the model. There are different solutions to settle this problem. The first idea is to replace the missing values by the mean or the median of the existent values in the correspondent feature.

Another idea was to replace the missing values by identifying the closest neighbors without missing values of each event with missing values with the cosine distance. The required calculation was too expensive computationally ($\approx 10^{10}$ operations) for a marginal gain.

The chosen solution required some data understanding. In fact, based on this article [3], explaining the Higgs Boson challenge, the missing values can be treated in the right way. The features are divided in two groups. The PRI for primatives variables, the ones actually measured by the detectors, and the the DER for derived variables computed from the first group of features. The article explains that the variable PRI_jet_num, an integer between 0 and 3, determine in which features DER the missing values are found. There is also the quantity DER_mass_MMC that can be an independent source of missing values depending if the topology of the event is too far from the expected topology. The data set can then be separated in 8 groups where each event is sorted depending if a missing value is found or not in DER_mass_MMC and on its PRI_jet_num value. The group are in fact divided in a way that the missing data are in the same features. This means that for group 1, for example, let's say that the features 2-7-12 are always 0, they can then be dropped, same methodology for each group. Separating in 8 groups make things more complicated, in fact, the data trains 8 different models with 8 different sets of weights by splitting the events in the corresponding groups. The test data then also needs to be split in the same way.

C. Data expansion

After cleaning the data set, a polynomial expansion is applied as it often increases the performance of the model. The choice of the best degree is determined and explained in the results section. During the optimization, multiplications between the features have been tried. In fact, the data has been expanded with combining all the columns between them (more than 450 features obtained).

D. Standardization

Once the data set expanded, it was important to standardize it. Indeed, the units of the different variables are different, especially after the features expansion. It was primordial to treat the data before applying the models so each quantity would have the same weight.

E. Correlation and PCA

The last part of the data analysis is to check linear relations between features. A PCA could have been implemented, however, the data set being small, this is not a necessity to reduce the dimension. The calculation time is not too large. The correlation has also been checked to avoid combining correlated variables.

III. Models and Methods

As previously mentioned, the given test set is divided into eight groups according to the values of PRI_jet_num (0,1,2,3) and DER_mass_MMC (missing value or not). The implementations explained hereafter are applied on each of these eight sub-groups to end up with eight different models. Concerning the prediction value of our data set, the labels 's' and 'b' are converted to 1 and -1 respectively while loading the data set from the csv file. Each model has its proper weights and the eight models share the same hyper-parameters.

The six functions that were asked are implemented in the file *implementations.py*, namely linear regression using (stochastic) gradient descent, least squares regression, ridge regression and (regularized) logistic regression using (stochastic) gradient descent. As seen during the course, the mean-squared error has some limits when it comes to perform a binary classification because the result depends crucially on how many points are in each class and where these points lie. Knowing that the data set is unbalanced, the best classification should come from a logistic regression.

Following the processing of the data set, the regularized logistic regression is implemented using gradient descent after having initialized the weights with a uniform distribution between -1 and 1. Though the stochastic gradient descent is faster to process, the reasonable size of the data set makes it possible to stay with a gradient descent. The parameters of the gradient descent γ and the number of iterations *max_iters* are set (γ = 0.1 and *max_iters* = 1000) to ensure a smooth decrease of the loss function until it reaches a stable value. A cross-validation with 4 folds is then implemented while iterating over different values of the regularization term λ (range from 1e-10 to 1) and the degree of the polynomial expansion (range from 1 to 15).

The research protocol that has been set up consists of dividing at the beginning the data set in two sets, a train set and a test set with a ratio of 75-25% respectively, to be able to test locally the performances of the models. The logistic (sigmoid) function is used in a first part to upper-bound to 1 and lower-bound to 0 the predictions values computed with the weights. The classification is achieved by attributing the value 1 to any event with a value of the logistic function strictly above 0,5 and the value 0 for value below or equal to 0,5. To be consistent during the performance computations, the prediction values of -1 from the loaded data set are converted to 0. The prediction values are now either 1 for a boson or 0.

The best parameters are thus sorted out by choosing the combination with the best accuracy.

IV. Results

A. Results of the methods tried

The results obtained with the different methods can be find in the Table I. As it shows, the last method, with splitting the data in groups, is the one giving the best accuracy and F1-Score. The rest of the analysis will then focus on this method.

Method	*Accuracy*	*F1-Score*
Random guess	0.500	0.406
Log reg raw data	0.673	0.624
Log reg mean/med	0.725	0.664
Log reg mean/med poly exp deg 2	0.795	0.678
Log reg mean/med comb	0.802	0.703
Best method	0.818	0.722

TABLE I
THE ACCURACY AND F1-SCORE OF THE DIFFERENT TRIED METHODS

The best method is a regularized logistic regression with polynomial expansion applied on the eight subsets of the data set, each of them trained over 15 000 epochs.

B. Specifications of the chosen model

Hereafter the specifications of each model are presented in the Table II.

PRI_jet_num	*DER_mass_MMC*	*Degree*	*Lambda*	*Proportion*	*Accuracy*
0	Defined	7	1e-05	0.295	0.821
1	Defined	9	1e-04	0.280	0.793
2	Defined	8	1e-05	0.190	0.813
3	Defined	9	1e-04	0.083	0.802
0	Undefined	12	1e-05	0.104	0.935
1	Undefined	10	1e-06	0.030	0.921
2	Undefined	9	1e-10	0.012	0.967
3	Undefined	9	1e-02	0.006	0.934

TABLE II
THE SPECIFICATIONS OF THE 8 TRAINED MODELS

V. Discussion

The results obtained with the regularized logistic regression ran on the eight different groups of data are satisfying giving the F1-Score of 0,722. The analysis can still be discussed. In fact, the optimization has been performed with a cross-validation. This is really time consuming and requires a lot of calculations for the regularized logistic regression. It would have been possible to determine the best parameters with the ridge regression and then change manually some tuning specifications to increase the accuracy. The features could also have been expanded taking into account the correlation of the variables. In fact, combining the quantity didn't give a satisfying enough level of accuracy. However, combining them but avoiding the ones that are correlated with each other can increase a little bit more the accuracy.

VI. Summary

To conclude this project, the Higgs boson challenge showed how important pre-processing of the data is for the results of a model. In fact, even though the algorithm is rather obvious in this project, the way of treating missing values or to expand features induce a big difference in the accuracy. Splitting the data set in groups according to the specifications of the events and training a different model for each of them, was the methodology that gave the best accuracy. There are always some improvements that can be done though.

REFERENCES

[1] S. Ray, "8 proven ways for improving the "accuracy" of a machine learning model," 2015. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/12/improve-machine-learning-results/

[2] M. Olugbenga, "Balanced accuracy: When should you use it?" 2022. [Online]. Available: https://neptune.ai/blog/balanced-accuracy

[3] C. G. I. G. B. K. D. R. Claire Adam-Bourdarios, Glen Cowan, "The higgs boson machine learning challenge," *Respiratory Care*, vol. 1, no. 1, pp. 1–37, 2015.