Discovering Higgs Boson using Machine Learning Algorithms

Fallegger Robin, Christelle Schneuwly and Margot Wendling
Project 1 of CS-433 - Machine Learning, EPFL, Lausanne, Switzerland

Distinguishing Higgs boson from noise in original CERN data could be implemented using machine learning methods. Exploratory analysis, extraction of most relevant data and feature processing were key steps to compare and optimize different algorithms aiming to establish a binary model classification for Higgs Boson measures.

I. Introduction

The Higgs boson is an elementary particle in the Standard Model of particle physics. Its existence was confirmed in 2012 by the ATLAS and CMS collaborations, based on collisions in the LHC at CERN. As scientists are not able to observe the Higgs boson directly, they measure its decay signature instead. The goal of this project [1] is to re-discover the Higgs boson by determining the likelihood that a given event's signature was its result or some noise. The data used here is the real CERN data that has been used in 2012 [2][3]. We built three ridge-regression models, depending on the categorical variable `PRI_jet_num`, with a polynomial expansion of the data set that reached an accuracy of 0.765.

II. Methods

A. Exploratory data analysis

The initial training set of the CERN is composed of the labels `y` of size N, the values of the different features `tX` of size NxF, and the indexes of the samples `ids` of size N. Here, the number of initial samples is `N=250'000`, and features `F=30`.
The features showed very diverse distributions shapes (right-skewed, normal, uniform...). For some of them a clear difference between the boson's and the noise's distributions was visible, whereas for most of them, the distributions were not clearly separable. The test set, without the labels, comprised 568238 samples.

B. Feature processing

First of all, given that some features exhibited right-skewed distributions, we applied a logarithmic transformation on the 11 right-skewed features in order to reshape their distributions: $x_{*j} \longrightarrow \ln(1 + x_{*j})$. The `PRI_jet_num` feature is the only categorical feature with either `0, 1, 2` or `3` integer values. We divided the whole training set into three jets: the first with 99913 samples corresponding to category `0`, the second with 77544 samples corresponding to category `1` and finally the third with 72543 samples corresponding to category `2` and `3`. Jets `2` and `3` were kept as one group, to have similar number of samples in each subset. The next steps were done on each of the jets identically, thus allowing the parameters to fit to the different conditions/categories.
Subsequently, the features with more than 5% of missing values (samples with absent data) were removed from each jet, giving: `jet0` with 18 features, `jet1` wit 22 and `jet2+` with 29.
Knowing that the median is a more robust metric than the mean value, all the remaining missing values were replaced by the feature median. Removing incomplete features and replacing the missing values with the feature mean with or without Gaussian noise decreased the model accuracy.

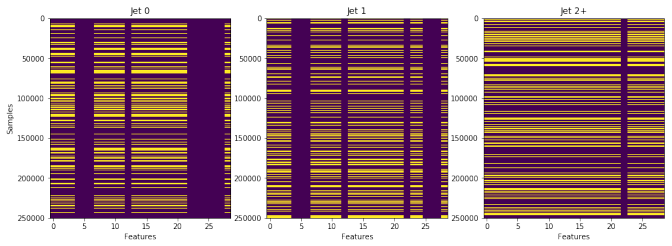

Fig. 1. Data decomposition into jets with dark blue representing the deleted columns (features), and the yellow rows representing the samples of the corresponding jet, used to train the final model.

In addition to data modification and removal, polynomial data augmentation was also necessary to implement due to the very few number of features. Thanks to a high number of samples, the remaining features were augmented using polynomial expansion of degree d and adding a column of ones as the offset:

$$\mathbf{X} = \begin{pmatrix} 1 \ \mathbf{X} \ \mathbf{X}^2 \dots \mathbf{X}^k \end{pmatrix} \text{ for } k \in (1, d) \tag{1}$$

We also built and optimized models with the addition of the first cross term between every two features, i.e. the multiplication of features:

$$\mathbf{X} = \begin{pmatrix} 1 \ (x_i x_j) \ \mathbf{X} \ \mathbf{X}^2 \dots \mathbf{X}^k \end{pmatrix} \ \forall i, j, \text{where} i \neq j \tag{2}$$

where x_i and x_j are the i-th and $j-th$ feature column vector, respectively. This increased the number of features but also the computational cost of cross-validation in a dramatic fashion. Finally, before training any model, each feature, of each jet, was standardized to have zero mean and unit standard deviation. The means and standard deviations were saved in order to standardize the test set with those values later on.
It should be noted that we had to convert the labels when we using logistic regression and regularized logistic regression

algorithms: $y_* = \{-1, 1\}$ for background and signal respectively to $y_* = \{0, 1\}$.

C. Methods used

The following machine learning algorithms were implemented and tested: least squares, least squares with gradient descent, least squares with stochastic gradient descent, ridge regression, logistic regression, regularized logistic regression. For the logistic models we used a decision criterion of 0.5, whereas the other models used a criterion of 0. We could have optimized this decision threshold as well in our cross-validation in order to maximize the model accuracy, however this would have meant to add another layer in the cross-validation process.
Ridge regression and regularized logistic regression were considered as our prime algorithms as they include a penalization coefficient λ to reduce over-fitting.
Learning rates γ, penalty λ and complexity (degree) of our model were tuned using grid searches. We implemented 4-fold cross-validations to optimize polynomial degrees and λ, and did it with and without feature cross-terms. The optimal hyper-parameters were chosen by maximizing test accuracy generated in the cross-validation functions.

III. Results

The method that could better distinguish Higgs boson signal from background was ridge regression with the jet subsets separations. Once we identified which basic regression algorithm was the most suitable, the key step to improve the performance was feature engineering. The optimization of parameters for the ridge regression, with or without interaction terms, and with or without logarithmic transformation yielded very similar results in the cross validation. To tune the hyper-parameters and complexity, we tried to maximize accuracy, precision, recall and F1-score metrics (see Fig. 2). We made a submission on AIcrowd for the models shown in Table I, with optimized hyperparameters.
Without the logistic transformation but with the interaction terms we were able to reach an accuracy of 0.765 and F1-score 0.715 (ID 23453) on AIcrowd. The degrees and lambdas used in our best model are listed in Table II .

Table I : Feature engineering scores on AIcrowd

ID	Log transform	Interaction	Accuracy	F1
23453	No	Yes	0.765	0.715
23328	Yes	No	0.692	0.657
23577	Yes	Yes	0.676	0.648

Table II : Best model (ID: 23453) parameters

Parameter	jet 0	jet 1	jet 2+
Degree n	10	13	14
λ	3.831e-15	1e-12	1e-13

IV. Discussion

Surprisingly feature engineering both including the interaction terms and logistic transform yielded worse results than using these separately. Although the accuracy score is relatively bad for the model that we chose, it remains quite a simple one, in contrast with more advance neural networks for example. However, this really depends on what the cost function of each type of error is, i.e. is it worse to have a false positive or a false negative ? For example, with some of the logistic regression models, we were able to achieve a precision score of 1.0, meaning that we did not have any false positives, albeit at the cost of many false negatives.

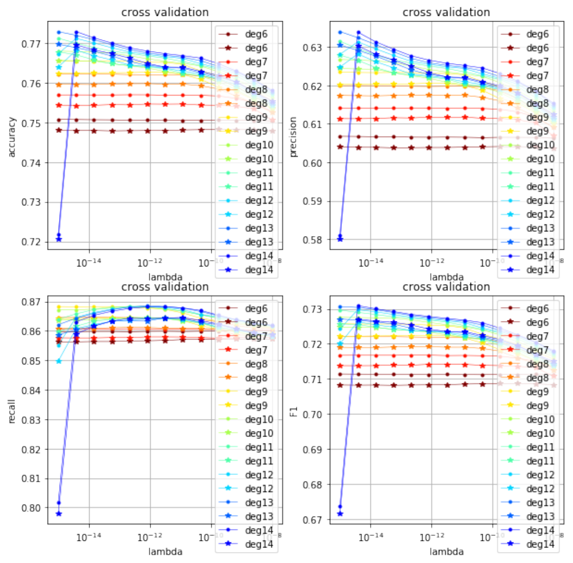

Fig. 2. Cross-validation output of the `jet1` model. Each color corresponds to a degree, curves with round markers for train and stars for test. We decided to use the parameters where the recall is highest and starts to drop off. This is representative for the results from other hyper-parameters (interaction and log-transform)

The main advantage of a linear model is that it can make fast predictions on a big number of samples. In that sense, it could be used as a primary filter in order to discard uninteresting collisions in the LHC. A more complicated and accurate model could then be used to classify the samples that passed this filter.
In order to improve our models, a deeper understanding of the data and particle physics may have been useful in order to make sure to include the most informative features. Furthermore, we could have improved the data cleaning and/or made a better data augmentation. In addition, different feature selection methods, such as forward feature selection or based on class discriminability scores, could have been implemented to select the best performing models. Finally, it would have been useful to have more computational power and more libraries at hand (e.g. for parallelization) in order to test more and a wider range of hyper-parameters (e.g. the decision threshold).

References

[1] C. Adam-Bourdarios, G. Cowan, C. Germain, I. Guyon, B. Kegl, and D. Rousseau, "Learning to discover: the higgs boson machine learning challenge," *URL http://higgsml. lal. in2p3. fr/documentation*, p. 9, 2014.

[2] G. Aad, T. Abajyan, B. Abbott, J. Abdallah, S. A. Khalek, A. A. Abdelalim, O. Abdinov, R. Aben, B. Abi, M. Abolins *et al.*, "Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc," *Physics Letters B*, vol. 716, no. 1, pp. 1–29, 2012.

[3] S. Chatrchyan, V. Khachatryan, A. M. Sirunyan, A. Tumasyan, W. Adam, E. Aguilo, T. Bergauer, M. Dragicevic, J. Erö, C. Fabjan *et al.*, "Observation of a new boson at a mass of 125 gev with the cms experiment at the lhc," *Physics Letters B*, vol. 716, no. 1, pp. 30–61, 2012.