Identifying the Higgs Boson with Generalized Linear Models

Capucine Berger-Sigrist, Mihai David, Tiberiu Mosnoi
Department of Computer Science, EPFL, Switzerland

***Abstract*—In this project, we explore how basic linear models can be used to predict the outcome of simulated events from the ATLAS experiment. Most notably, we are interested in classifying the signal originating from the decay of particles during proton-proton collisions, as either the decay of a Higgs boson or background noise. As our research shows, even fairly simplistic machine learning approaches can yield decent results (83.5% accuracy and 75.3% F1-score) on a complex problem.**

I. Introduction

The ATLAS and CMS experiments were able to confirm the existence of the Higgs boson by analysing the decay of particles produced by the collision of protons in the CERN's LHC. Our aim is to predict whether the observed signals come from the decay of a Higgs boson or from background noise using generalized linear models and clever data analysis and exploitation.

II. Models and Methods

In order to learn this classification problem, we considered two linear predictors: (regularized) linear regression and logistic regression. We implemented three methods for finding the weights with linear regression (gradient descent, stochastic gradient descent and the closed-form solution) and one method with logistic regression (gradient descent). We employed k-fold cross validation to evaluate the performance of our classifiers, using the F1-score and the accuracy as metrics.

A. Data analysis and preprocessing

The dataset comes from 250.000 simulated events of proton-proton collisions. Each event is composed of 30 features that can be divided into two categories: primitive features related to the momenta of particles or derived features obtained from the former.

1) Feature analysis: Firstly, we carefully analyzed the type and distribution of each feature. We concluded that all features but one were continuous. More exactly, the feature *PRI_jet_num* is ordinal, with values in [0, 1, 2, 3]. Undefined values were replaced with -999 because they had no meaning or could not be computed. Secondly, based on our distribution plots and description of features presented in [1], we discovered that several continuous feature values depended on the *PRI_jet_num*. Therefore, a good approach was to split the dataset according to the four ordinal values in order to perform further analysis. Interesting patterns appeared as for events with a *PRI_jet_num* of 0, 10/30 features were completely undefined. For events with *PRI_jet_num* of 1, there were 7/30 features fully undefined. However, for events with *PRI_jet_num* of 2 or 3 almost all other feature values were fully defined.

2) Data splits: As a result of the feature analysis, we decided to split the entire dataset in two ways. Firstly, we divided it into three subsets according to *PRI_jet_num* values, in the following way:

1) *PRI_jet_0* with 99913 samples
2) *PRI_jet_1* with 77544 samples
3) *PRI_jet_2_3* with 72543 samples

Moreover, as all subsets still had missing values for the first feature, *DER_mass_MMC*, we further divided each of the three subsets in two, depending on whether or not this feature was defined, giving a total of six subsets. We conducted experiments on the entire dataset, as well as on both splitting approaches.

3) Feature engineering: Feature standardization (zero mean and unit variance) was performed on all features in order to make our data unitless. In this way we could mitigate the effect of high ranging features on the model behaviour. We also tried only normalizing the data but this approach did not yield better results.

Regarding imputation, we analyzed the impact of replacing undefined values with feature-wise mean, median or mode. As several feature distributions were right skewed, the best approaches turned out to be median or mode, depending on the model.

4) Feature expansion: In order to increase the model complexity and allow the model to identify non-linear patterns, we used three feature expansion techniques.

Firstly, we employed polynomial expansion by carefully choosing a degree d:

$$\phi := [1, X, X^2, ..., X^d]$$

where X represents the initial features matrix and then ϕ the expanded one, used for training.

The second approach was feature multiplication of the initial feature matrix:

$$\forall X_{:i} \neq X_{:j}, \;\; \phi := [\phi \mid X_{:i} * X_{:j}]$$

The last one was adding the square rooted values of the positive feature columns of the initial feature matrix.

		Training		Validation		Testing	
	Methods	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**
Baseline	Regularized logistic regression	44.79%	72.80%	44.80%	72.80%	44.90%	72.90%
	Ridge regression	56.70%	74.43%	56.70%	74.42%	**56.70%**	**74.40%**
Intermediate	Logistic regression	71.20%	80.10%	71.43%	81.66%	71.20%	**80.80%**
	Ridge regression	72.97%	82.22%	72.70%	82.00%	**73.40%**	80.30%
Final	Ridge regression 3 Models	73.45%	83.58%	73.14%	83.36%	75.10%	83.40%
	Ridge regression 6 Models	69.80%	83.65%	69.07%	83.35%	**75.30%**	**83.50%**

Table I
PARAMETERS, F1-SCORE AND ACCURACY RESULTS.

B. Hyperparameter tuning

For hyperparameters tuning we first devised a grid search method to compute *(lambda, degree)* pairs using 5-fold cross-validation. If a logistic regression model was used, we then tuned the regression threshold, keeping the configuration that maximised the accuracy. We also tuned the threshold because of the slightly imbalanced data-set (2:1 ratio) [2].

III. RESULTS

Given that the task is binary classification, employing logistic regression seemed to be the best approach. As Table I shows, starting at a baseline of 72.90% accuracy and a F1-score of 44.90%, we improved our classifier by standardizing the data, replacing missing values with the median, using feature expansion and modifying the decision threshold. These efforts yielded an accuracy of 80.80% and a F1-score of 71.20% on the test set.

Because it seemed that we had hit a glass ceiling with our logistic regression approach, we decided to shift our attention to linear regression since the optimality of the solution and the computational efficiency of its closed-form implementation enabled us to do more experiments in a shorter amount of time. This paradigm shift proved to be worthwhile: we achieved 83.5% accuracy and 75.3% F1-score on the test set. The key contributor to these scores was splitting the dataset into three, as explained in section II-A2, and training one classifier on each subset. Small improvements were also achieved by further splitting the data after the *DER_mass_MMC* feature. The finishing touches consisted in finding the best hyper-parameters for each predictor (see Table II) with the help of a grid search.

IV. DISCUSSION

We tried dropping highly correlated features, as they had a bad impact on the variance of the weights. Also, in order to remove the right skewness of several feature distributions, we applied log transformation on the positive ones [3]. However, the latter two approaches did not bring improvements.

Subsets		Parameters		
PRI_jet	**MMC**	**Lambda**	**Degree**	**Replace with**
0	undefined	1e-8	10	median
	defined	1e-14	10	median
1	undefined	1e-8	6	mode
	defined	1e-16	10	mode
2 or 3	undefined	1e-3	3	mode
	defined	1e-15	11	mode

Table II
PARAMETERS FOR EACH OF THE SIX SUBSETS.

Subsets		Training		Validation	
PRI_jet	**MMC**	**F1-score**	**Accuracy**	**F1-score**	**Accuracy**
0	undefined	35.71 %	95.13 %	34.34 %	95.03 %
	defined	68.84 %	81.28 %	68.68 %	81.17 %
1	undefined	40.77 %	92.70 %	34.77 %	91.96 %
	defined	73.95 %	80.16 %	73.55 %	79.85 %
2 or 3	undefined	67.46 %	93.23 %	60.16 %	91.50 %
	defined	83.23 %	84.09 %	82.80 %	83.66 %

Table III
F1-SCORE AND ACCURACY RESULTS FOR EACH SUBSET.

From Table III, we see that the high imbalance in classes for each subset had a major impact on the metrics, as there were many more true negatives than true positives. Therefore, we should further investigate the effects of weighted logistic regression, by giving a higher weight to rare events [4].

V. SUMMARY

We used two generalized linear models (linear and logistic regression) as the mathematical backbones of our approach. We enriched them with basic machine learning techniques (such as data standardization, feature expansion, hyper-parameter tuning etc.) and further improved them by carefully inspecting the data we were given and exploiting its structure by splitting it into subsets and training a different classifier for each split. This last point was a breakthrough for our approach, propelling us in the top 10% of the teams on AIcrowd.

References

[1] C. Adam-Bourdarios, G. Cowan, C. Germain, I. Guyon, B. Kégl, and D. Rousseau, *Learning to discover: the Higgs boson machine learning challenge*, 2014. [Online]. Available: https://higgsml.lal.in2p3.fr/files/2014/04/documentation_v1.8.pdf

[2] J. Brownlee, "A gentle introduction to threshold-moving for imbalanced classification," 2020. [Online]. Available: https://machinelearningmastery.com/threshold-moving-for-imbalanced-classification/

[3] T. Selvan, "Types of transformations for better normal distribution," 2020. [Online]. Available: https://towardsdatascience.com/types-of-transformations-for-better-normal-distribution-61c22668d3b9

[4] J. Brownlee, "Cost-sensitive logistic regression for imbalanced classification," 2020. [Online]. Available: https://machinelearningmastery.com/cost-sensitive-logistic-regression/