The Higgs Boson Classification Project

Emanuele Nevali, Matteo Francesco Suez, Leonardo Bruno Trentini
Machine Learning (CS-433) Project - Fall 2022
EPFL, Switzerland

***Abstract*—In this project, carried out as part of the Machine Learning course at EPFL, we applied machine learning techniques to actual CERN particle accelerator data to recreate the process of "discovering" the Higgs particle. In this report we present the adopted methodology throughout our journey, from the analysis and processing of the dataset to the choice of the best fitting model.**

I. Introduction

Rarely, collisions between protons at high speed can produce a Higgs Boson, an elementary and unstable particle in the Standard Model of particle physics theory [1]. Since this Boson decays rapidly into other particles, it is not possible to observe it directly, but rather measure its "decay signature", or the products that result from its decay process. The aim of this project is to estimate the likelihood that a given event's signature was the result of a Higgs Boson (signal) or some other process or particle (background).
After the analysis and preprocessing of the dataset, we introduce the models we trained using 6 different algorithms. Thereafter, we explain how we managed to select one among them and the relative hyperparameters, in order to reach the highest accuracy. In conclusion, we present our considerations and final results.

II. Data analysis and preprocessing

A. Exploratory data analysis

Our dataset contains *250000* labeled points (*s* for *signal* and *b* for *background*) for training and *568238* unlabeled ones for testing. These points lie in a space of dimension $D = 30$ (each corresponding to a different feature).

B. Categorization

First of all, we observed that there is only one categorical feature (i.e. *Pri jet num*): it shows the number of jets [2], an integer ranging from 0 to 3.
We noticed that some features are meaningless for some values of jets; therefore, we have split the dataset into 4 subclasses, each characterized by a different *Pri jet num*.

C. Missing Values

In the dataset documentation it is stated that each *-999* value represents a missing value: we decided to replace them all with the median of the corresponding feature. We opted for the median instead of the mean because of its nature of robust estimator to outliers.

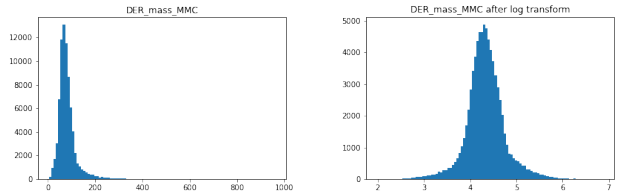

Figure 1: Logarithmic transformation for the 1st feature

Furthermore, for subclasses marked by *Pri jet num* 0 and 1 we decided not to consider columns with too many missing values, in order to avoid singular matrices.

D. Absolute transform of Symmetrical Features

For features whose distribution is symmetric around 0 and for which the proportion of categories is not too even, we substituted them with their absolute value.
This operation increases the gap between the categories, and makes them more detectable by the model.

E. Outliers threshold

After a search for outliers in the data, we set $\alpha = 0.05$ and decided to edge the extreme values of each feature to the α-quantile (for the lower tail) and to the $(1-\alpha)$-quantile (for the upper tail).

F. Angles

We noticed that some of the features represents measurements of angles. For this reason, in order to adapt our model to the periodicity of this type of data, we tried to transform these columns using sine and cosine functions. However, only the latter appeared to be effective, and it was the only one kept in the final model.

G. Heavy-tailed features

We discovered that many of the features were heavy-tailed; thus, we applied to those a logarithmic transformation: $x_k^{'} = log(1 + x_k)$ (where k is the feature column). Thanks to this, we managed to reduce or even remove the skewness of our original data (Fig. 1).

H. Standardization

To ensure our algorithms to be the most effective, after having completed the previously described preprocessing phase, we decided to standardize the data by subtracting from each feature its mean and dividing by its standard deviation:

Method	Subset 1	Subset 2	Subset 3	Subset 4	Total
Gradient Descent	70.7%	65.4%	67.9%	68.3%	68.3%
Stochastic Gradient Descent	69.8%	66.6%	67.5%	68.5%	68.2%
Least squares	84.6%	81.1%	84.4%	84.1%	83.5%
Ridge regression	84.7%	81.3%	84.7%	84.5%	83.7%
Logistic regression	77.1%	68.6%	67.1%	69.8%	71.8%
Regularized logistic	77.6%	69.1%	67.5%	69.8%	72.1%
Variance	$\pm 10^{-3}$				

Table I: Accuracy for each method and subset (the average was weighted on the number of points for each subset)

$Z = \frac{X-\mu}{\sigma}$. The same process was applied both to the train set and to the test set, using on the latter μ and σ of the former.

I. Polynomial Expansion

We managed to improve our predictions by expanding the features. In particular, we augmented them by implementing a polynomial expansion $\phi : [X] \rightarrow [X, X^2, ..., X^D]$ for each column.
The polynomial degree D is one of the hyper-parameters of our model: we found its optimal value by performing a 4-fold cross validation on each subset.
In addition, we added for each couple of features a new feature with their product. By doing so, we exploited the couple of features with a particularly high index of correlation and, thus, we noticed a remarkable increase in the accuracy. Finally, we also added the square root and the cubic root of each feature to our polynomial expansion.

III. Methods

After the preprocessing on the dataset, we managed to find a model for our classification problem by implementing six different methods:

- **Gradient Descent**
- **Stochastic Gradient Descent**
- **Least squares**, using normal equations
- **Ridge regression**, using normal equations
- **Logistic regression**, using gradient descent or stochastic gradient descent ($y \in \{0, 1\}$)
- **Regularized logistic regression**, using gradient descent or stochastic gradient descent ($y \in \{0, 1\}$, with regularization term $\lambda\|\omega\|^2$)

We tested each method by performing a grid search on 3 hyperparameters (some of them are not used by all 6 different algorithms). Such hyperparameters are:

- λ: regularization to reduce overfitting
- γ: step size for gradient and stochastic gradient descent
- D: degree for the polynomial expansion

IV. Results

Due to the fact that we were facing a binary classification problem, we first focused on the Regularized Logistic Regression. However, in the meanwhile we also carried out multiple tests with Least Squares and Ridge Regression, and

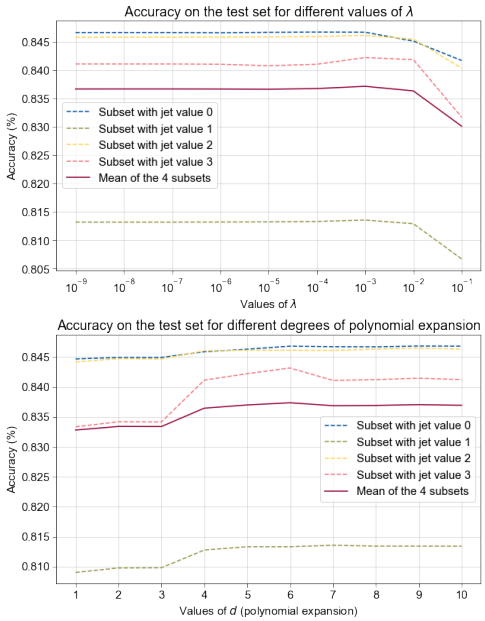

Figure 2: Accuracy for different values of λ and D

remarkably the latter turned out to be the most effective one (Table I). For this reason, we focused on tuning the Ridge Regression hyperparameters, by looking for the best compromise between underfitting and overfitting and adjusting the hyperparameters for each of the four subclasses (Fig. 2). At the end, the hyperparameters through which we obtained our best accuracy are:

- Subset *0*: $\lambda = 10^{-5}$, $D = 7$
- Subset *1*: $\lambda = 10^{-6}$, $D = 6$
- Subset *2*: $\lambda = 10^{-6}$, $D = 6$
- Subset *3*: $\lambda = 10^{-3}$, $D = 6$

Finally, we obtained a model that achieved an accuracy of 0.837 on the test set, with a F1-Score of 0.751 [3].

V. Conclusions

We achieved our best accuracy, 0.838 on the train set and 0.837 on the test set, by using the Ridge Regression with the previously stated hyperparameteres.
It is important to underline that data analysis, data cleaning and feature preprocessing played a key role in this task, enabling us to achieve massive improvements in model's accuracy.

References

[1] M. Goulette, "What should we know about the higgs particle?" *Atlas experiment - CERN*, 2013.

[2] C. Adam-Bourdariosa, G. Cowanb, C. Germain, I. Guyond, B. Kegl, and D. Rousseau, "Learning to discover: the higgs boson machine learning challenge," *Higgs Challenge*, 2014.

[3] "F-Score Definition — DeepAI," https://deepai.org/machine-learning-glossary-and-terms/f-score.