Machine Learning Course: Project 1

Peter Krcmar, Anton Ragot and Robin Zbinden
Machine Learning (CS-433), School of Computer Science and Communications Sciences, EPFL, Switzerland

***Abstract*—Through this report, we aim to tackle a binary classification problem which is linked to the discovery of the Higgs boson using original data from the CERN. We propose several approaches based on regression techniques. We compare these different approaches to find out which one produces the most accurate classifier and the best predictions.**

I. Introduction

The Higgs boson is an elementary particle in the Standard Model of physics which explains why other particles have mass. In order to prove its existence, physicists at CERN perform experiments which reveal its presence. Scientists don't observe it directly, but rather measure its "decay signature". Since many decay signatures look similar, we estimate if a given signature was the result of a Higgs boson (signal) or some other process/particle (background). Therefore, our task consists in predicting if a signature (represented as a vector of features) comes from a signal or the background [1].

First, we pre-process the CERN's Higgs Boson raw data set to clean and filter the data. Then we train a model and explore different regression techniques to figure out which ones suit better for this task.

II. Models and Methods

A. Data pre-processing

By exploring the data we realize that the `PRI_jet_num` feature can only take on four distinct values: 0, 1, 2 and 3. Having a closer look at the provided detailed description of the features [2], we notice that many others features are greatly dependent on this particular number. Most importantly, some features are only defined for certain values of `PRI_jet_num`. Therefore, we separate our data into 4 different sets, based on the `PRI_jet_num` value. The undefined features can then be removed, as they have zero variance and do not give any information. We then need to compute a different prediction model for each of these 4 datasets. The removed features are summarized in table I.

After removing the unwanted features, a couple of data points' `DER_mass_MMC` values are still undefined. We replace them with the median value of the feature, as it is less affected by outliers than the mean.

Table I
Summary of removed features after separating data by `PRI_jet_num`. Crosses indicate the removed features.

Feature \ Jet number	0	1	2	3
`DER_deltaeta_jet_jet`	x	x		
`DER_mass_jet_jet`	x	x		
`DER_prodeta_jet_jet`	x	x		
`DER_lep_eta_centrality`	x	x		
`PRI_jet_leading_pt`	x			
`PRI_jet_leading_eta`	x			
`PRI_jet_leading_phi`	x			
`PRI_jet_subleading_pt`	x	x		
`PRI_jet_subleading_eta`	x	x		
`PRI_jet_subleading_phi`	x	x		
`PRI_jet_all_pt`	x	x		

B. Features expansion

As we deal with the laws of physic, we suppose that the relationship between the features is more complex than just a simple linear relationship. Hence, to take into account this complexity and to avoid underfitting, we expand the number of features by adding new ones, i.e., if $\vec{x}$ is the vector representing the features, we expand this vector by applying the function $\phi : \mathbf{R}^D \mapsto \mathbf{R}^{D'}$ where $D < D'$.

The new added features are combinations and functions of the original features. A starting point is to take every feature to a certain power, by doing polynomial expansion ($x_i \rightarrow x_i^2, x_i^3, ..., x_i^{degree}$). The hyperparameter *degree* has to be tuned to deal with underfitting and overfitting (see section II-D). We also add some cross terms ($x_i, x_j \rightarrow x_i x_j$ $\forall i, j \ 1 \leq i < j \leq D$). By looking at the descriptions of the features, we observe that a lot of them are angles in radians. Thus, we apply the cosine and sine to the features ($x_i \rightarrow cos(x_i), sin(x_i)$).

For each new type of feature added, the accuracy on our train set increases. Empirically, we conclude that the optimal solution is to add all of them.

C. Regression techniques

We implement and test 6 different regression techniques to see which one works best. To make them comparable, we apply the same data processing to each of them and we perform a log-search to find an estimation of the best hyperparameters. We also take 80% of our processed data for training our models, and the rest is used for testing. Note that we standardize (also known as Z-score [3]) our data for iterative techniques in order to avoid computational

problems. The performance of each technique is summarized in table II.

Table II
SUMMARY OF THE DIFFERENT REGRESSION TECHNIQUES TRIED AND THEIR PERFORMANCE

Techniques	Normalized	Accuracy
Least squares GD	x	0.7448
Least squares SGD	x	0.7457
Least squares		0.8132
Ridge regression		0.827
Logistic regression	x	0.7435
Reg. Logistic regression	x	0.7534

We decide to continue with ridge regression for two reasons: this method obtains very good results and the tuning of hyperparameters is relatively fast compared to other techniques because it computes the solution analytically.

D. Hyperparameters

Ridge regression uses an hyperparameter λ which we have to tune. Moreover, in subsection II-B, we increase the number of features, especially using polynomial expansion. There too, we need another hyperparameter which is the highest degree of the polynomial.

Finally, as explained in subsection II-A, we split our data into 4 datasets and consequently compute 4 different models. Therefore, for each of these models we have different hyperparameters.

It leads to a total number of 8 distinct hyperparameters. We perform a grid search on a subset of values for each hyperparameter using 10-fold cross validation (10 is a trade-off between accuracy and computational resources). We then select the combination that gives the largest accuracy, which we call the optimal values. A visualization of how the accuracies vary can be seen in figure 1. The optimal values are summarized in table III.

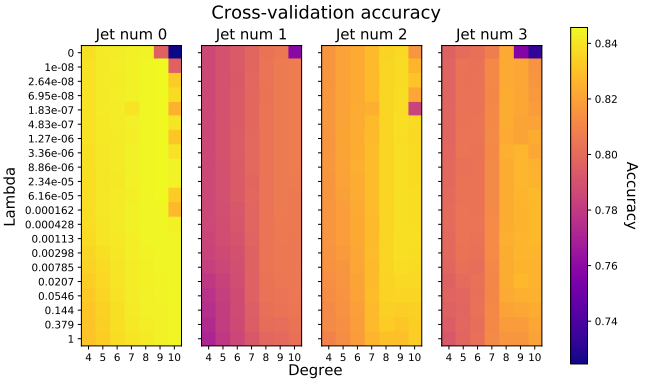

Figure 1. The accuracies obtained during cross-validation, with varying lambda and degree. For each jet number value, the combination yielding the largest accuracy was selected as the optimal hyperparameters.

Table III
VALUES OF THE DISTINCT HYPERPARAMETERS AND THEIR CORRESPONDING ACCURACY ON THE TEST SET

Jet number	λ	Degree	Accuracy
0	1.83e-7	9	0.8469
1	3.36e-6	10	0.8102
2	1.62e-4	9	0.8453
3	1.62e-4	10	0.8445

III. RESULTS

Finally, after applying all the stated steps, we have a fully trained model. When testing our model on the AIcrowd platform, we obtain an accuracy of 0.832.

IV. DISCUSSION

We are satisfied by this accuracy of prediction, but we recognize that it is possible to do better by doing more data pre-processing and by expanding more the number of features. Regarding pre-processing, we choose not to deal with outliers, but they could be either removed or clamped. To expand our feature vector even more, one could also take the logarithm or the square root of the non negative features and add them to the new augmented feature vector.

Another way of improving our accuracy that we do not try (due to time constrains) is using different regression techniques for each set of `PRI_jet_num` value. By finding the best technique for each set, the final classifier would be theoretically better.

V. SUMMARY

The biggest lesson we take from this project is that we cannot create a good machine learning model without understanding the data beforehand. Good background knowledge on the domain enables to identify useful features and produces a more efficient data pre-processing. We realise that there are many ways to tackle a machine learning problem, going from data processing methods to hyperparameters tweaking.

Finally, computational power was the bottleneck of this work, especially during hyperparameters tuning. We are convinced that our work is not perfect but we are satisfied with the result and we definitely learned a lot!

ACKNOWLEDGEMENTS

The authors would like to thank all the staff from the Machine Learning Course to have made this project possible.

REFERENCES

[1] M. Jaggi and R. Urbanke, "Description Project 1," 2019. [Online]. Available: https://github.com/epfml/ML_course/blob/master/projects/project1/project1_description.pdf

[2] 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

[3] "Normalization." [Online]. Available: https://developers.google.com/machine-learning/data-prep/transform/normalization