CS-433 Machine Learning project 1

Camillo Nicolò De Sabbata, Gianluca Radi, Thomas Berkane
Department of Computer Science, EPFL Lausanne, Switzerland

***Abstract*—This report describes our approach to the Higgs boson machine learning challenge, based on the ATLAS and CMS experiment that led to the discovery of the Higgs boson. The goal of this project was to find the machine learning classification model that would best predict the Higgs boson decay signatures from background noise.**

I. Introduction

The Higgs boson is an elementary particle generated by quantum excitation of the Higgs field, and it was discovered at CERN in 2013. Rarely, a collision between protons can generate a Higgs boson, that then decays rapidly, leaving a signature that can be measured. The objective of this project was to develop a binary classifier that would allow the identification of the Higgs boson signatures from background noise. This was achieved by a first step of data cleaning and feature processing, then proceeding by training different machine learning models of both regression and classification, in order to find the one with the highest possible accuracy.

II. Models and Methods

A. Implementation of Machine Learning models

We implemented six models, as per the project description:

- Linear regression using gradient descent
- Linear regression using stochastic gradient descent
- Least squares regression using normal equations
- Ridge regression using normal equations
- Logistic regression using gradient descent
- Regularized logistic regression using gradient descent

B. Data

The train and test sets are characterized by 30 features and contain $250,000$ and $568,238$ events respectively. Feature types varied from floating point numbers to integer values and missing values are represented by -999.0. Labels provided were 'b' for background event (no signature left by the decay of the boson), and 's' for signal event (the signature left by the decay of the boson). The categories are reasonably well balanced, as their proportion is of 34.3% and 65.7% each.

C. Categorization

According to the challenge description, some of the missing values depend on the number of jets of the event (i.e. the PRI_jet_num column). The number of jets is an integer between 0 and 3:

- If it is 0, a specific set S of the features presents missing values.
- If it is 1, only a specific subset $S^* \subset S$ of the features presents missing values.

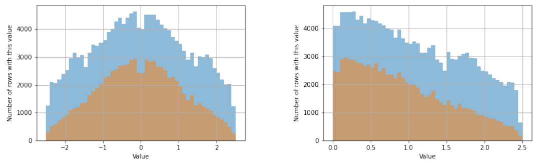

Fig. 1. Original vs. absolute value of PRI_tau_eta

- If it is either 2 or 3, there are no missing values.

To deal with these different categories we split the dataset into three subsets (one with the events with jet number 0, one with jet number 1 and one with jet numbers 2 or 3) and then trained three models, one for each subset, since the three subsets have a different set of features. Then, labels are predicted by using the three obtained models with the corresponding subset of the test dataset.

D. Missing values

We noticed that some values were missing, in particular in the feature DER_mass_MMC. According to the challenge documentation, the mass is labeled -999.0 if the topology of the event is too far from the expected topology. Since invalid values were present in significant numbers (around 15% of events for some features), removing those samples would have reduced our data too significantly. Thus, to deal with these invalid samples, we replaced them with the median of the feature, since it is more robust to outliers than the mean.

E. Absolute value of symmetric features

For features whose distribution is symmetric around 0 and for which the proportion of categories is not too even, we calculated and replaced with the absolute value. This increases the gap between the two categories, and makes them better distinguishable by the model. An example of such a feature can be seen in Fig.1.

F. Logarithmic transformation

Noticing that many of the features were heavy-tailed, we applied, exclusively on these features, the formula $n_k^* = \ln{(1 + n_k)}$, where k is the feature column. For each of those, we decided to keep both the original feature and the one obtained by shrinking the support of its distribution through the application of the log function stated above.

G. Outliers

We searched for outliers in the data. In a first attempt, we considered as outliers all samples $\leq Q_\gamma$ or $\geq Q_{100-\gamma}$

(where Q_γ is the γ percentile) and replaced them with the median of the feature. Later, we changed the median with the new min/max of the feature (i.e. the $Q_\gamma/Q_{100-\gamma}$ values respectively, according to the value being higher or lower end) instead of the median, in order to maintain a distribution similar to the original one. We repeated the same process on the test set, using the percentiles of the train set. We also tried using asymmetrical values for percentiles, taking into account that some features were positively skewed, but this did not lead to any improvement.

H. Feature selection

At this point, we used common feature selection methods to try and find a possibly better subset of features:

- Correlation: we found a high correlation between some of the features. The removal of such columns slightly improved the score.
- Low variance: no features with too low variance were found, thus no feature was removed at this point.

I. Standardization

The data was standardized using Z-score standardisation: $Z = \frac{X-\mu}{\sigma}$ where μ is the mean and σ the standard deviation of our expanded data. The same process was applied to the test set, the exception being that the test set was standardized using the mean and standard deviation of the training set.

J. Polynomial expansion

We tried to improve our predictions by transforming and expanding the features; in particular, we augmented features by adding a bias column [1] and a polynomial expansion of the form $\phi : [X] \to [X, X^2, ..., X^k]$ for each column. Finally, also added the mixed products of the columns (obtained by multiplying each original feature with each of the others).

K. Testing and Tuning

Given that the project is of binary classification, we first focused on tuning the Regularized Logistic Regression with gradient descent. However, we also performed multiple tests with Ridge Regression. Surprisingly, the latter outperformed the former. We then decided to tune Ridge Regression hyperparameters; specifically, the regularizer λ, the percentile to identify outliers and the degree of the polynomial expansion. We did it by using grid-search methods (λ from 10^{-7} to 10^{-2}, γ from 0 to 10 and degree from 2 to 12) and 3-fold cross-validation, looking for the best tradeoff between underfitting and overfitting. We tuned the hyperparameters for each of the three subsets. The best result was obtained for $\lambda = \{10^{-5}, 10^{-5}, 10^{-4}\}$, $\gamma = \{3, 6, 8\}$ and degrees $\{7, 7, 5\}$.

III. Results

The final model is trained as follows:

- Three different subsets of events are used (as in II-C), and then three different models are trained.
- The mass missing values are treated as described in II-D.
- The absolute value of features symmetric around 0 is calculated as in II-E.
- Extreme values are replaced, as described in II-G.
- Highly correlated columns are discarded (as in II-H).
- The data is standardized (as in II-I).
- The logarithmic transform and polynomial expansion of the features were added (as in II-F and J).
- The training model is Ridge Regression with hyperparameters λ, γ and d, which is optimized using cross-validation with grid search.

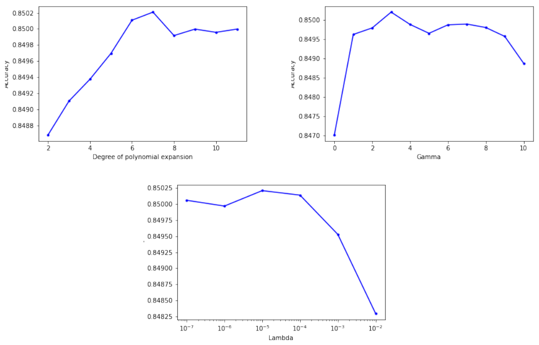

Fig. 2. Results of running grid search to optimize the hyperparameters for ridge regression for jet group 0

As a final result, we obtained a model that achieved an accuracy of 0.840 on both AIcrowd and on local cross-validation test set.

TABLE I
Subsets Accuracies at different stages of the processing

Stage	Subset 1	Subset 2	Subset 3
Raw data	74.4%	/	/
Missing values	74.4%	/	/
Absolute value	75.2%	/	/
Categorization	81.4%	72.4%	73.2%
Logarithmic Transformation	82.5%	74.5%	76.8%
Outliers	82.4%	76.3%	78.8%
Polynomial expansion	85.0%	81.7%	84.9%

IV. Discussion

Surprisingly, the Logistic Regression did not produce better results than Ridge Regression neither locally nor AIcrowd validation. This could be due to the fact that Logistic Regression is the result of an iterative method, while Ridge Regression simply solves a linear system, and does not have to solve an optimization problem. Moreover, logistic regression relies on the assumption that there is a distribution D which generates the data. This, in practice, is not always true.

V. Conclusion

In conclusion, the best performance on the Higgs boson experiment data-set was obtained with a model of ridge regression. The model's accuracy was enhanced by data cleaning and feature engineering, as well as exploiting specific peculiarities of the given data.