CS-433 Machine Learning - The Higgs Boson Machine Learning Challenge

Hilda Abigél Horváth, Tim Poštuvan, and Julian P. Schnitzler
School of Computer and Communication Sciences, EPFL

***Abstract*—Machine learning approaches have become an indispensable part of numerous research fields, providing predictions and explanations of complex multidimensional datasets. In this paper, we focus on developing an approach for detecting whether an event corresponds to the decay of the Higgs boson or not, based on features extracted from the ATLAS experiment performed at CERN. Our best approach uses regularized logistic regression with splitting on the number of jets, feature expansion, median imputing, and outlier bounding. It achieves 0.831% accuracy on the Higgs boson machine learning challenge at AIcrowd platform.[1] The source code of this project is available on GitHub [2].**

I. Introduction

Higgs bosons are elementary particles first observed in CERN LHC to prove the theory of Higgs field. They are not observed directly but are rather recognized based on the measure of the signatures and the resulting products of decay processes. In this paper, we are trying to predict whether an event corresponds to the decay of the Higgs boson or not, based on features extracted from the ATLAS experiment performed at CERN.
In order to make accurate predictions, we employ basic machine learning algorithms, such as least squares, logistic regression, and regularized logistic regression. The crucial role to achieving good performance of the mentioned algorithms play pre-processing steps we propose based on prior data exploration.
The paper is structured as follows. In Section II, data exploration is performed. Additionally, we propose data pre-processing steps there to clean the dataset. Section III introduces employed models and describes regularized logistic regression in more detail. In Section IV, a comparison of model and ablation study of pre-processing steps is presented. Finally, we conclude the paper in Section V.

II. Data Pre-processing

In this section, we present some characteristics of Higgs boson dataset. Then, we showcase its downsides and propose pre-processing steps to improve the performance of machine learning approaches.

A. Dataset

To better understand the dataset, we made some exploratory data analysis on the $250'000$ events. Firstly, we looked at the distributions of each of the 30 features, which are all continuous except the `PRI_jet_num`. The latter is discrete and contains only values in $\{0, 1, 2, 3\}$. Additionally, we observed that NaN values are denoted by -999. By plotting heatmaps of feature correlations, we found out that some features are highly correlated. Unfortunately, removing one or more of them does not show any significant improvement in terms of accuracy. Moreover, we observed differences in the distributions of features of Higgs bosons and other particles.[3]

B. Splitting on Number of Jets (JET)

According to Boudarious et al. [1], depending on the number of jets (`PRI_jet_num`), some features can be entirely undefined (e.g. `PRI_jet_leading_phi` is undefined for 0 jets, while present for 1 jet or more). Since the feature contains only four different values, it makes sense to split data based on its value and remove undefined features for each part separately.

C. Feature Expansion (FE)

Since predictions probably depend also on higher powers of initial features, we expand the feature set with polynomial features up to degree d. After that by similar reasoning, the feature set is additionally extended by $\sin$ and $\cos$ of initial features. We tried applying polynomial expansion to $\sin$ and $\cos$ features as well, however, it resulted in inferior performance. We also considered adding cross-terms, nevertheless, that would result in a quadratic number of extra terms, which is not feasible to train due to our computational resources. On the other hand, we do not have enough domain knowledge to select only some informative combinations.

D. Imputing the Median (IM)

As a result of our data exploration, we found out that there are a lot of missing values. If we would remove all instances with at least one missing value, only around 20% of data would remain. Therefore, we decided to replace missing values with the median of the non-missing values of that feature. We chose the median because it is robust to the outliers that are still present in the dataset (in contrast to the mean for example).

E. Bounding Outliers (BO)

Since we noticed that the data is still very noisy, we use Inter-quartile-range (IQR) criterion to decide whether a value is an outlier or not. As can be seen in Equation (1), this criterion defines a value as an outlier if it lies further than 1.5-times interquartile range (the difference between the 25% and the 75% quantile) from the closest quartile. Having detected those outliers, we then set their value to the value of the closest quartile.

$$min = Q_1 - 1.5 \cdot IQR, \qquad max = Q_3 + 1.5 \cdot IQR \quad (1)$$

It would also be possible to remove instances that contain one outlier value in any column, however, with this approach we kept only about one fifth of the initial dataset.

[1] https://www.aicrowd.com/challenges/epfl-machine-learning-higgs/leaderboards
[2] https://github.com/hhildaa/EPFL_ML_project

[3] The plots can be found in the *visualizations/visuals.ipynb*.

III. Models

To tackle our classification problem we use various combinations of algorithms and optimization methods: least squares with gradient descent (GD), least squares with stochastic gradient descent (SGD), logistic regression, and regularized logistic regression. We implemented least squares and ridge regression which are optimized using normal equations as well, nonetheless, the size of the dataset prevented us to include them in our analysis. As a simple baseline, we employ naive random guessing.

Out of the mentioned approaches, we select regularized logistic regression for our final model. It is more appropriate for binary classification problems than least squares algorithms, while the regularization term λ allows more flexibility than vanilla logistic regression. Regularized logistic regression is trained by optimizing the following loss function:

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{2}$$

$$\mathcal{L}(w) = -\sum_{i=1}^{N} \big[y_i \cdot \log(\sigma(w^T x_i)) \tag{3}$$

$$+(1-y_i)\cdot \log(1-\sigma(w^T x_i))\big] \tag{4}$$

$$+\,\lambda \cdot \sum_{i=1}^{D} w_i^2, \tag{5}$$

where σ is sigmoid function, N is the number of instances in the dataset, x_i are features of the i-th instance, y_i is the label of the i-th instance, D is the number of features, and w are weights of the regularized logistic regression.

IV. Experiments

In this section, we compare approaches from Section III and explore which of the data pre-processing techniques from Section II are the most pertinent.

Experimental set-up: To perform experiments, we first pre-process data where pre-processing steps are specific to each experiment. Then, we fine-tune the models by grid search and k-fold cross-validation with 10 folds on 2% of the randomly-sampled instances. When data is split according to the number of jets feature, cross-validation is performed on each of the datasets separately. If features are expanded, the following set of polynomial degrees d is considered: $d \in \{8, 9, 10, 11\}$. For logistic regression regularization term λ is fine-tuned as well: $\lambda \in \{10^{-10}, 10^{-9}, \ldots, 10^{-2}\}$. All models with exception of least squares with SGD are trained for 100 iterations over the entire dataset since training for twice as many iterations barely alters the scores. We defined iterations over the whole dataset to make results across different algorithms computationally equivalent. However, least squares with SGD works better if it is trained only on one datapoint per each iteration, so we go with that training regime in this case. Both logistic regressions are trained using batches of size 100. Furthermore, all models are trained using the learning rate $\gamma = 10^{-4}$. Once hyperparameters are chosen appropriately, we report the average of 10-fold cross-validation accuracy on the whole dataset.

A. Comparison of Methods

Comparison of models from Section III is performed on the dataset with all preprocessing steps from Section II.

[4]Theoretical result

Table I
Average cross validation accuracy of different models.

Method	Result
Random guess baseline[4]	0.5000 ± 0.0000
Least squares with stochastic gradient descent	0.6447 ± 0.0185
Least squares with gradient descent	0.6811 ± 0.0041
Logistic regression	0.8293 ± 0.0020
Regularized logistic regression	0.8293 ± 0.0020

Table II
Average cross validation accuracy when certain pre-processing steps are skipped (denoted ✗).

Data pre-processing methods				Result
IM	JET	BO	FE	
✓	✓	✓	✓	0.8293 ± 0.0020
✗	✓	✓	✓	0.8267 ± 0.0018
✓	✗	✓	✓	0.8138 ± 0.0019
✓	✓	✗	✓	0.8102 ± 0.0019
✓	✓	✓	✗	0.7732 ± 0.0020

Results and discussion: In Table I we can observe that all considered models perform considerably better than the random guess baseline. By comparing least squares with GD and least squares with SGD, it is evident that stochastic gradient is inferior on this dataset. We hypothesize this occurs due to noisy dataset and the fact that stochastic gradient descent with the batch size of only one is very sensitive to noisy instances. Both logistic regressions outperform other approaches by a large margin, which is not surprising since other approaches are not specialized for classification. Interestingly, the results of logistic regression and regularized logistic are identical. This can be explained by the optimal choice of regularization term λ of regularized logistic regression. No matter the number of jets, it is always smaller than 10^{-7}.

B. Ablation Study of Data Pre-processing

To conduct an ablation study of data pre-processing steps, we employ the best-performing model from the previous section, namely the regularized logistic regression. The importance of each pre-processing step from Section II is evaluated based on the decline in performance when that step is skipped.

Results and discussion: In Table II are shown results when certain pre-processing steps are skipped. Feature expansion is extremely important since without it, performance drops for almost 6%. Bounding outlier also results in a significant 2% gain. It turns out splitting on the number of jets still makes a difference, although we expected it to be bigger. Finally, imputing the median does not help a lot, probably because bounding outlier achieves a similar effect.

V. Conclusion

In this paper, the performance of various algorithms is studied on Higgs boson dataset. To improve the accuracy of algorithms, we propose different pre-processing steps, which are selected based on prior exploratory data analysis. Then, we experimentally compare algorithms and evaluate the importance of each pre-processing step. As expected, there are huge differences in performance among baseline, both least squares, and both logistic regressions. We find out that some pre-processing steps are more crucial than others, however, each of them contributes to higher final accuracy.

REFERENCES

[1] Claire Adam-Bourdarios, Glen Cowan, Cécile Germain, Isabelle Guyon, Balázs Kégl, and David Rousseau. The higgs boson machine learning challenge. In *Proceedings of the 2014 International Conference on High-Energy Physics and Machine Learning - Volume 42*, HEPML'14, page 19–55. JMLR.org, 2014.