Finding the Higgs boson using machine learning

William Cappelletti, *MA*, Charles Dufour, *MA*, Marie Sadler, *CGC*

ABSTRACT

We train and discuss different machine learning models and data processings to predict the Higgs boson event on a data set of 250'000 samples provided by CERN. In our best model missing values were imputed with the median, the data was normalized to zero mean and a standard deviation of one and features ending in _eta and _phi were removed. Logarithmic transformations of selected features, categorical interactions and polynomial feature expansion to the 4th degree in a ridge regression model achieved a test loss of 0.206 in a cross validation and a score of 0.794 in a *Kaggle* competition. We present a computationally efficient model which could serve as a starting point for more complex machine learning algorithms.

I. INTRODUCTION

The Higgs boson discovery, which was awarded the 2013 Nobel prize in physics, was finally claimed in 2012 in the ATLAS and the CMS experiment [1], [2], [3]. The Higgs boson can be a by-product of a proton-proton collision, however, since it decays very rapidly, it is not observed directly, but only measured through the decay signature of a collision event.

Here we are given a large training dataset with the decay signature of 250'000 events from the CERN particle accelerator which includes 16 "raw" (primitive) features as measured by the detector and 15 "derived" features, which are quantities computed from these primitive features. Each event is labelled by a binary variable corresponding to either a Higgs boson or a background signal. To correctly attribute the presence or the absence of a Higgs boson event to an unlabelled collision event given the decay signature, we develop a classification model using different machine learning regression models. We evaluate the performance of our algorithms both locally by estimating the test error using cross-validation and globally by getting the prediction score on a large unlabelled test set of 568'238 events in a *Kaggle InClass Prediction Competition*. Besides regression model selection, we improve our prediction model by optimizing various hyper-parameters and by performing data cleaning, feature engineering, while focusing on parsimony and computational efficiency.

II. MODELS AND METHODS

A. Regression models and implemented algorithms

Preprocessing includes the following steps: *Feature 22*, a categorical variable with 4 different values is replaced by 4 new features, each being a binary variable representative for the 4 categories. Since the sum of these newly formed binary features across all the data n points yields a vector of ones (size n), the offset term is omitted. Imputation of missing values (entries of -999) is done with the feature's mean or median. Exception done for the binary variables, all the features are normalized to have zero mean and a standard deviation of one.

An optional data cleaning step to detect outliers is implemented to exclude data measurements with at least one feature value being outside the normal range given a threshold m:

$$x_{i,j} - mean(x_{:,d}) > m \cdot std(x_{:,d}),\ x_{i,j} \in \mathbb{R}^{nxd}$$

where n is the number of samples, and d the number of model features. Note that this cleaning step is done before data imputation and normalization, and after logarithmic feature transformation (see section II-B).

B. Feature space and feature transformations

The feature space is enlarged by adding the interaction terms between the newly created binary variables (result of splitting *Feature 22*) and the remaining features. In addition, the feature space can be extended by adding a polynomial basis defined by an arbitrary degree M (where all the features are subjected to the same degree M, the binary features excluded). Logarithmic transformation of the raw data is done on features with an empirically identified heavy-tailed distribution. Some of the features are translated ($x \leftarrow x + k,\ k \in \mathbb{N}$) before applying the transformation to avoid taking the logarithm of negative values.

C. Research protocol

The model and parameter evaluations are, besides the *Kaggle* score, based on the minimal misclassification ratio (test loss) in the prediction, estimated though a 5-fold cross validation (CV). The weight vector $\mathbf{w}$ is then trained on the whole training dataset with the optimal hyper-parameters.

D. Bases for model and parameter selection

The following training sets are used in the result section:

- *Neutral* training data (T.D.) (solely imputation (mean or median) and normalization)
- *Logarithmic* T.D. (logarithmic feature (f.) transformation prior to imputation and normalization)
- *Interaction* T.D. (neutral training data with additional features for interaction of categorical and continuous f. (see II-B)

Combination of these processing steps are further tested on the most promising regression models and subsets of features (selected by Lasso).

III. RESULTS

A. Regression model comparison

The minimal test loss from different feature engineering processes (see II-D) is reported in Table I. We see that logistic regression performs best followed by ridge regression and least squares (normal equations). However, the least squares normal equation method is not applicable on linear dependent datasets and the method fails on the mean imputed interaction

TABLE I: Test loss from cross validation

Training data Imputation method	**Neutral** Mean/Median	**Logarithmic** Mean/Median	**Interaction** Mean/Median
Least squares (GD)	0.25	0.24	0.241
Least squares (SGD)	0.434 /0.547	0.489/ 0.506	0.546/ 0.449
Least squares	0.254	0.237	- /0.241
Ridge reg.	0.254	0.237	0.241
Logistic reg. (GD)	0.248	0.233	0.236
Regul. logistic reg.	0.314/ 0.317	0.324/ 0.325	0.325/ 0.319

training data. Mean and median imputed results do not differ on the shown significant numbers, except for the regularized logistic regression (stochastic output naturally fluctuates). Even though logistic regression performs better in the CV on this training data, computational instability came quickly with polynomially augmented datasets, for which ridge gives better results than logistic regression on the neutral data set (see Table III). The *logarithmic* and the *interaction* training data predicts better than the *neutral* one. Therefore, for further model improvements, we use ridge regression on a range of lambdas and we choose the best one for prediction (by CV) with the additional interaction terms and the logarithmic transformation of some features.

B. Feature selection using Lasso

We perform Lasso regressions (in the same fashion as ridge) on the *logarithmic* training data set and, analyzing the values of the resulting weighted vector **w**, we see that the primitive features ending in _phi and _eta have the lowest weights. Removing them results in a lower test loss (0.228 vs 0.232). Features 0 and 21 also have low weights, however their removal slightly increases the test loss (0.229 vs 0.228 in the same experiment, and 0.222 vs 0.206 when adding a polynomial degree of 3). Therefore, in the following, we exclude the primitive features ending in _phi and _eta.

C. Data cleaning

TABLE II: Progressive outlier removal

Cleaning parameter m	Training samples n	Test loss	Kaggle score
/	250'000	0.229	0.771
3	222'135	0.221	0.767
2.5	200'537	0.217	0.702

While outliers can compromise the accuracy of the model, the removal of several data points, due to one feature value being outside the defined normal range, also drastically reduces the sample size. We train ridge models with progressive outlier removal obtaining the results in Table II. We observe that, as the accepted value range is narrowed, the test loss computed in CV decreases, while the Kaggle score worsens. However, this trend does not appear in a consistent way: the Kaggle score increases with m = 3 in an experiment using ridge regression and removing all the features with *phi* in their name (score of 0.756 compared to 0.749). Thus, we do not remove any outliers.

D. Polynomial regression

The feature space is extended by adding the polynomial terms up to a degree M of the continuous variables (see Table III). Our best *Kaggle* score of 0.794 is highlighted in the table and it is a result of the following transformations of the dataset: missing values imputed median, logarithmic transformations on selected features, _eta and _phi feature exclusion, polynomial feature space expansion up to a degree of 4 and categorical interactions (*Feature 22* splitting result) with the linear term of the remaining continuous variables.

TABLE III: Polynomial feature space extension

Polynomial degree	Feature space size	Test loss	Kaggle score
1	104	0.229	0.771
3	144	0.214	0.786
4	**164**	**0.206**	**0.794**
5	184	0.204	/

IV. Discussion

When comparing the different regression models, we see that the least squares SGD performs significantly worse than the other models. In addition, regularized logistic regression has a higher test loss than simple logistic regression. `Logistic regression` becomes unstable when the dataset is augmented using polynomials, hence ridge produces better results.

An important observation for model improvement is that the different preprocessing steps (data cleaning, features selection and features expansion) are strongly related (e.g. cleaning improved the Kaggle score when removing all _phi features, but deteriorate it when some other feature expansions were applied). We believe that this is a consequence of overfitting the training data and, in order to obtain more conclusive trends, a more thorough testing approach is needed, for instance with progressive feature removal and/or different feature spaces (e.g. interactions, polynomial terms).

In our testing approach, we apply cross validation on the complete available training dataset. However, in addition to the Kaggle score, a feedback obtained from another independent test set (obtained from a training set split) would be useful to further assess opposite trends of the CV test loss and the *Kaggle* score. Though, it should be noted that given the small size of the training set compared to the *Kaggle* test set, an even smaller training set might decrease drastically the accuracy.

The most important model improvement is achieved with polynomial expansion. We suppose that the inclusion of interaction terms between the continuous variable might lead to an increased prediction accuracy, while potentially also leading to excessive computation times.

V. Summary

Starting from an initial test loss of 0.25 obtained by applying various regression models on median (or mean) imputed training data with normalized features in a cross validation experiment, we managed to reduce this loss down to 0.20 with a ridge regression model eliminating features, applying logarithmic transformations and extending the features space with categorical interactions and polynomial terms. On an independent large test set, this improved our model from an accuracy of 0.712 in the aforementioned experiment with basic linear regression to 0.794.

References

[1] G. Aad et al., *Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC*, Phys.Lett., vol. B716, pp. 129, 2012.

[2] S. Chatrchyan et al., *Observation of a new boson at a mass of 125 GeV with the CMS exper- iment at the LHC*, Phys.Lett., vol. B716, pp. 3061, 2012.

[3] The Nobel Prize in Physics 2013. NobelPrize.org. Nobel Media AB 2018. [Online]. Available: https://www.nobelprize.org/prizes/physics/2013/summary. [Accessed: Fri. 26 Oct 2018.]

[4] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning. New York, NY, USA: Springer New York Inc., 2001.

Appendix

TABLE IV: Feature names and attributed numbers

Feature Name	**Number**
DER_mass_MMC	0
DER_mass_transverse_met_lep	1
DER_mass_vis	2
DER_pt_h	3
DER_deltaeta_jet_jet	4
DER_mass_jet_jet	5
DER_prodeta_jet_jet	6
DER_deltar_tau_lep	7
DER_pt_tot	8
DER_sum_pt	9
DER_pt_ratio_lep_tau	10
DER_met_phi_centrality	11
DER_lep_eta_centrality	12
PRI_tau_pt	13
PRI_tau_eta	14
PRI_tau_phi	15
PRI_lep_pt	16
PRI_lep_eta	17
PRI_lep_phi	18
PRI_met	19
PRI_met_phi	20
PRI_met_sumet	21
PRI_jet_num (categorical)	22
PRI_jet_leading_pt	23
PRI_jet_leading_eta	24
PRI_jet_leading_phi	25
PRI_jet_subleading_pt	26
PRI_jet_subleading_eta	27
PRI_jet_subleading_phi	28
PRI_jet_all_pt	29