CS-433 Machine Learning - Detection of the Higgs Boson

Department of Computer Science, EPFL, Switzerland
Arman Ehsasi, Aleksandar Hrusanov, Thomas Stegmüller

***Abstract*—Machine Learning has become increasingly important in the last decades and has revolutionized many industrial and scientific sectors such as particle physics. This paper proposes our approach at reproducing the process that allowed physicists to claim the discovery of the Higgs boson at CERN's Large Hadron Collider in March 2013. Our final approach achieved an accuracy of over 83% on the online test set and is based on the method of Ridge Regression.**

I. Introduction

In March 2013, scientists from CERN announced the discovery of the Higgs boson. Their discovery was based on the observation of high velocity collisions of protons. Such collisions generate various subatomic particles including the infamous Higgs boson. Unfortunately, a Higgs boson cannot be directly observed, therefore its presence must be inferred from its decay signature, which consists of a collection of observable quantities treated here as features. While trying to discriminate the presence of a Higgs boson (signal) from other processes or particles (background) with a mathematical model we tried to constrain the latter such that it could be coherent with a possible underlying physical model. This constraint influenced us in the feature pre-processing and engineering. The data at our hands consisted of two files, `train.csv` and `test.csv`, containing respectively $250'000$ and $568'238$ decay signatures, each consisting of 29 features. The two files differ as each decay signature in `train.csv` is labeled as either signal (s) or background (b). The first file can therefore be used to train the models and the second to assess their ability to correctly classify unseen data. Before applying advanced Machine Learning algorithms on the data we must perform preliminary data analysis to receive a better insight about the features at hand.

II. Exploratory Data Analysis and Pre-Processing

Using the pandas library we visualized the raw data sets and discovered that many features contained a multitude of out of bound values (-999) which we decided to treat as *"Not-A-Number" (NaN)* values for more convenient further processing. We noticed the existence of a single categorical feature, *PRI_jet_num* $\in \{0, 1, 2, 3\}$. This pointed us at the high probability of the data being the result of 4 disjoint subsets of collision outcomes from the experiment, depending on the number of jets produced. Looking further into those subsets, we observed that the number of NaN values is highly correlated with each subset - e.g. some features might not make sense in the case of *PRI_jet_num* being 0, 1, 2, or 3, just like the NaN values in the corresponding rows of the given feature column. Upon this realization, we choose three different approaches to handle the missing values.

- Method A: Replace NaN values with the per-class feature's mean.
- Method B: Replace NaN values with the per-class feature's median
- Method C: Split the data based on the features *PRI_num_jet* and *DER_mass_MMC*

For method C, based on our observations in III, we decided that it would be more appropriate to further split the four resulting split sets rather than infer a new value on a missing value. This decision is increasingly important with the percentage of values that would need replacement, which is 25% in our case.

Subsequently, we moved on to assessing each of the above three pre-processing methods as well as various Machine Learning algorithms in order to find which are the most suitable for classifying collisions' decay signatures.

III. Methods and Models

In this section we review the performance of some algorithms on each of the three different pre-processing methods. Only *Ridge Regression* (*RR*) and *Regularized Logistic Regression* (*RLR*) are presented here. The remaining algorithms are sub-cases of these two. For instance, *Least Squares* (*LS*) can be obtained from *RR* by setting $\lambda = 0$.

A. Methods

The performance of each algorithm is measured with a 10-fold cross validation as it returns an unbiased estimate of the generalization error. The results of the methods with respect to approaches A, B, and C are summarized in Tables I, II, and III. In particular, the results for the method C correspond to the weighted average of the result of each of the eight classifiers. The tables further indicates the value of the hyper-parameters γ and λ that yielded the displayed accuracy according to the individual methods.

B. Method Selection

Function	γ	λ	Training A	Validation A
Ridge Regression	-	10^{-1}	0.8802	0.8802
Regularized LR	10^{-8}	10^{-1}	0.8520	0.8520

Table I
Accuracy on data set A (Mean Replacement)

Function	γ	λ	Training B	Validation B
Ridge Regression	-	10^{-10}	0.902	0.9042
Regularized LR	10^{-8}	10^{-1}	0.8781	0.8781

Table II
ACCURACY ON DATA SET B (MEDIAN REPLACEMENT)

Function	γ	λ	Training C	Validation C
Ridge Regression	-	10^{-10}	0.7652	0.7649
Regularized LR	10^{-5}	10^{-10}	0.7639	0.7632

Table III
ACCURACY ON DATA SET C (SPLIT BY EXPERIMENT NO.)

From those tables alone we would be tempted to conclude that the pre-processing methods A and B are the most suitable for this task. Nonetheless, once compared with the accuracy obtained on the online test set we observed a discrepancy of up to 20% between the validation and test accuracy. This result does not contradict with the fact that the cross-validation returns an unbiased estimate of the generalization error. This statement assumes that the train and test sets are i.i.d. sampled from the same underlying distribution. By replacing some of the values with the feature's median for a high percentage of samples, there is a high probability that we are violating the underlying distribution. Furthermore, it is even possible that we are artificially creating values which do not have any physical meaning. For instance it is very likely that the feature *PRI_jet_leading_eta* is not supposed to have any value when the feature *PRI_jet_num* is zero. Finally, we decided to use method C since it was the most physically meaningful and also did not suffer from the problem observed with the two other methods. In terms of the choice of the algorithm, *RR* and *RLR* offered very similar results. Even though, *RLR* is specially tailored for classification tasks, we chose to use *RR* as we observed a greater improvement in accuracy when augmenting the capacity of the model. For comparison, *RLR* displayed an improvement in validation accuracy of 2%, when applying polynomial expansion up to degree 10. On the other hand *RR*'s validation accuracy improved by 6%.

IV. FEATURE ENGINEERING

After taking the appropriate pre-processing steps and determining the best machine learning method, we perform further feature processing to increase the accuracy of our model.

Feature Reduction - Dimensionality reduction is usually a good method to reduce computational time and the possibility of overfitting. The ill-condition number can be used to diagnose systems which are at the limit of singularity. It is defined as follows:

$$\kappa(X^TX) = \frac{\lambda_{max}(X^TX)}{\lambda_{min}(X^TX)}$$

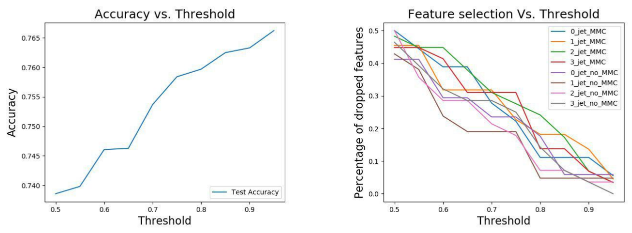

Figure 1. Change in accuracy based on a chosen threshold (Correlation Coefficient) for feature selection (LR)

When applied to our training subsets the ill-condition number were on the order of 10^{20}. This problem can be mitigated by selecting a subset of features which have no correlation coefficient over a given threshold. The relation between this threshold and the accuracy can be observed in Figure 1. This method could typically be applied before polynomial expansion to prevent the model from overfitting. Nonetheless as *RR* already takes advantages of the L2-regularization to penalize complex models, we chose not to use this implementation.

Polynomial Expansion - Since a linear model may not always fit the data to the fullest extent, we add a polynomial basis to enhance its representational power. Figure 2. exhibits a major increase in accuracy until the maximum polynomial degree 10. We do not choose a higher degree since the change in accuracy would only increase marginally at the cost of over-complicating the model and miss-representing the physical nature of the data. A similar approach was used to determine the best fractional basis expansion.

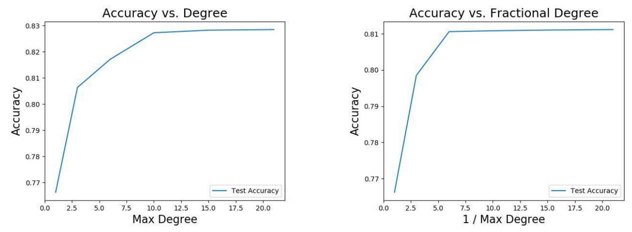

Figure 2. Change in accuracy with change in polynomial and polynomial fractional degree (RR)

V. OUTLOOK

We have some ideas for future improvement of the model accuracy. During our data exploration we noticed some of the features have values bounded by the range $[-3.142,\ 3.142]$ which led us to suspect that these features correspond to angle values (range equivalent to $[-\pi,\ \pi]$, in radians). In order to enhance the model fit with respect to these features we could add new columns with the *cosine* of the corresponding values. Similarly, we could add columns with the *log* of the values of other features to improve capturing possible logarithmic behavior.