# Project Report: Higgs Boson Challenge

Castagna Léandre, Epple Pascal, Jaoua Salima
*Department of Computer Science, EPFL Lausanne, Switzerland*

***Abstract*—The Higgs Boson Challenge is a classification problem based on the ATLAS and CMS experiment. The goal of the Higgs Boson Machine Learning Challenge is to explore the potential of machine learning methods. In this report, we present our approach to solve this problem.**

## I. Introduction

The aim of the project is to predict whether simulated particle collision can generate a Higgs boson or not. To do so, we first performed a data analysis step where we transformed our data so that we can implement our algorithms on so called "clean data". In Section II, we present how we proceed for the analysis. Then we train different machine learning methods based on regression and classification. We present in Section III the models that we implemented and how we found the one which gave the best accuracy.

## II. Data Analysis

The training set consists of 250'000 data points with 30 features and one column with the labels that we need to predict (-1 or 1). Label 1 stands for background event and label -1 stands for signal event. The test set has a total of 568'238 data points. We first work with the training set since we don't have the labels for the test set.

### A. Balancing :

The train set is a bit unbalanced (66%-34%). We did not consider this fact as critical and decided not to tackle it.

### B. Categorization :

All features are floating point, except PRI_jet_num which is integer between 0 and 3. After reading the challenge documentation, we understood that this feature is categorical and that some features only made sense depending on the number of jets. Therefore, we decided to split our data into four distinct classes (0,1,2,3). We noticed that classes 2 and 3 had the same set of meaningful features. This led us to consider them sometimes as one single class during our search for the best model.

### C. Missing values :

The meaningless values of the train and test sets are represented by the indicator number -999 for each feature. Categorization helped us dealing with missing values for each class separately. Our first idea was to delete an event if there is a missing value in its features. The problem is that there are many missing values (in roughly 72% of the events over all classes) and we would loose way too much of data, which can imply underfitting. Rather than deleting the event completely, we decided to remove the features which only contained missing values. After this process, the unique feature left with missing values was Der_Mass_MMC, with proportions depending on the class (26% at maximum). Thus, we decided to replace the missing values of this feature. To do so, we used the median of the entries which were given. We also thought of using the mean but it is vulnerable to outliers and we preferred to use a more robust value.

### D. Standardization :

We have seen that standardization can be useful in Machine Learning, especially when there are features with different scales. This can also prevent the feature matrix to be ill-conditioned. So we created a function that standardizes our train set. We also had to create another one that standardizes the test set. Indeed, it is important to use the mean and standard deviation of the train set in order to standardize the test set.

### E. Feature Engineering :

For each feature, we plotted the empirical distributions with respect to the label. This procedure allowed us to fine-tune the training set. We discovered the presence of outliers for some features and we detected that several empirical distributions were heavy tailed. Removing the outliers did not help improving our general accuracy. Regarding heavy tailed distributions for positive features, a $\log(1 + x)$-transformation was computed. This method allowed us to make slightly better for some models.

### F. Data Expansion :

To quickly improve the accuracy of our models, we used polynomial expansion to expand our features. Its maximal degree was carefully chosen using cross validation and being aware of possible over-fitting. Moreover, we added an intercept (a column of ones in the feature matrix). Lastly, for each couple of features, we expanded the feature matrix with its product. This method must take correlation into account, which we will now discuss.

### G. Correlation :

With the use of a correlation matrix, we noticed that some features were noticeably correlated. So we decided to not

| Method | Test Accuracy | Train Accuracy | Remarks |
|---|---|---|---|
| Least Squares GD | 0.757 | 0.7528 | done with 1000 iterations. |
| Least Squares SGD | 0.727 | 0.7256 | done with 3000 iterations. |
| Least Squares | 0.759 | 0.7591 | |
| Ridge Regression | 0.7888 | 0.7884 | done with 4-fold cross validation |
| Logistic Regression GD | 0.815 | 0.8186 | done with 10 000 iterations and expand our matrix to degree 2 |
| Reg Logistic Regression GD | 0.825 | 0.8213 | done with 15 000 iterations and use method II-G. |

Table I
RESULTS OF THE 6 IMPLEMENTED ALGORITHMS

multiply pairs of features with high correlation. Depending on the method, taking this into account did not help us improve the score of our predictions.

## III. MODELS AND METHODS :

Now that we have analysed our data, we can implement models to solve the problem. The difficulty here is to figure out which method we are going to focus on. We first focused on logistic regression using Gradient Descent. In fact, we choose to work with Gradient Descent and not Stochastic Gradient Descent because as you see in Table I, SGD is worse than GD with three times more iterations. Note that this is a classification problem, and we have seen that classification is not just a special case of regression. And so we preferred to focus more on logistic than ridge regression. However, we also performed multiple tests with Ridge Regression. Since the results were surprisingly good, we decided to implement 4- fold cross validation to find the best parameter $\lambda$ and the best degree of the polynomial augmentation. To minimize the over-fitting on the train set, we tried to use regularized logistic regression.

## IV. RESULTS

### A. Applying all methods :

From the Table I, we can see why we chose to work with... depends des resultats

### B. Choosing parameters for the methods :

- Ridge Regression : After running a 4-fold cross validation on each class of our training set, we obtain that the optimal value of $\lambda$ is $10^{-3}$ and the best degree of polynomial expansion equals to $d = 2$.
- Logistic Regression : We didn't want to implement cross validation for logistic because it is very time consuming. Instead, we chose to expand to the degree 2 since it was the optimal degree for Ridge. And we tried different values of $\gamma$ to optimise our predictions. We know that the $\gamma$ depends if the matrix is well-conditioned or not [1]. Depending on the class of the data, we get different values of $\gamma$, but it is of the same order: $10^{-6}$.
- Reg Logistic Regression : We repeat the procedure here and we also find different values of gammas depending on the Jet class. After getting better result, we decided to add the multiplications of columns in the data. We didn't take into account the pairs of columns which are highly correlated. This leads us to the next subsection.

### C. Best Model :

We obtained the best model by running Regularized Logistic Regression. We modified our train and test sets as follows:

- After splitting, cleaning and standardizing our train set, we expand our matrix. We check the correlation for each pair of features. If it's smaller than a given threshold (here $0.5$ ), we add the multiplication of the features. We also add a set of second degree features consisting by multiplying each feature by it self.
- We run the Regularized Logistic Regression on the modified train set.
- We reproduce modifications such as missing features deletion or feature expansion exactly as we did for our train set. To standardize the test, we use the mean and the standard value of the train set.
- We obtain a model that achieves an accuracy of $0.825$ on AIcrowd.

We must point out that Logistic Regression has one big issue with respect to Ridge Regression: there is no closed form for the minimizer. For each new model designed, a new optimal $\lambda$ is needed. We also noticed some weird behavior by looking at the norm of the gradient at each iteration: sometimes it would go up again. This fact prevented us from perfectly minimizing our loss function and we did not find a proper method (for example Line-search[1]) to overcome this issue. However, the results were still very convincing even when we knew that we were not at optimal weights.

## V. SUMMARY

The Higgs Boson Challenge underlines how important data analysis and data pre-processing is for Machine Learning. Using cross-validation and different optimization techniques, we managed to build several models with very good results. In the end, Regularized Ridge Regression gave us the best final accuracy.

## REFERENCES

[1] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.