Higgs Boson Machine Learning Challenge

Dalil Koheeallee, Hugo May, Emilien Guandalino
École Polytechnique Fédérale de Lausanne

***Abstract*—The purpose of this homework is to investigate different machine learning methods for classification over a dataset provided by the Large Hadron Collider (LHC) at CERN. As computer scientists, we attempt to solve this problem with a purely data driven approach. We will explain how our dataset was processed and expanded in order to run different predictive models which were optimized with cross validation.**

I. Introduction

Inside the LHC, particles are accelerated at high speed and collided against each other, which causes some of them to decay into new particles. Given a large training dataset (250,000 samples) containing information about those collisions, we are tasked to classify each event as either a 'signal', i.e. a new particle, or as 'background', i.e. noise. There are 30 features with different value ranges, and a substantial proportion of undefined variables, assigned the value -999.

II. Data cleaning and processing

After some analysis of out dataset, we noticed that one column, *PRI_jet_number*, took only discrete values in the $[0, 3]$ range and that the proportion of missing features (entire column is -999) could directly be inferred from this number. From a physical perspective, collisions decay into different particules for which many features are irrelevant or cannot be computed. It seems that this parameter is a good indicator of such a phenomenon. We therefore decided to split our dataset into 4 different subsets based of this value. This enables us firstly to drop the irrelevant columns, and secondly to run our predictive models with different parameters on each subset, as they may expose different physical properties.

Still, there remained about 10% of the rows with missing values scattered accross their columns. Since this is a substantial amount amount of data, we decided not to drop those rows but to replace those values with the median of that column. We also dropped columns which had standard deviation of 0, meaning that they only had a single value and therefore did not carry any information.

Finally, we standardized our dataset to ensure uniform contribution from each feature in our learning models.

III. Feature Expansion

When working with such a large dataset with a small number of features in comparison, it can be a good idea to expand those variables and try to generate a new feature space on which we can better classify. Considering the fact that there are many features which physically correspond to angles, our first try was to expand our features using sine, cosine and exponential functions. This gave us a good increase in accuracy but we also tried polynomial feature expansion, which gave better results. For each feature, we added its polynomial expansion and tested the accuracy. This worked very well up to degrees 7 or 8, after which the accuracy quickly dropped again. Our guess is that beyond a certain point, all the new information has already been captured in lower dimension and adding new features only increases the noise. We will explain later how we arrived to this in-between spot.

IV. Model Selection

During the first part of the homework, we were tasked to implement different regression and classification methods. We therefore tried out Regular and Stochastic Gradient Descent, Least Squares method, Ridge Regression and Logistic Regression. Unsuprisingly, we obtained the highest accuracy score with the Least Squares method, as it provides a closed form solution to our optimization. However the risk here is to overfit our training data and perform less on actual test data. This is why we decided to use Ridge Regression which adds a regularization term to avoid this. Yet remains to carefully pick this term to also avoid underfitting our data.

V. Cross Validation

Since we are using multiple parameters in our model, namely polynomial expansion degree and regularization term, we need to ensure that we pick the best combination of those values. This is why we used k-fold Cross Validation, which serves the purpose mentioned before, but also lets us use all of our dataset for both training and testing. To limit complexity, and since it is a multiple of our datset size, we used a k-fold number of 4 to pick a combination of lambda and expansion degree. We also decided to use testing accuracy as a metric to differentiate on combinations, which we try to maximize across all sets. We found that our results were the highest when using the following expansion degrees respectively for our sets : $d = [8, 8, 9, 9]$, and a regularizer $\lambda = [0.008, 0.017, 0.008, 0.008]$.

VI. Results

We now present the results from our training and cross validating our model. Figure 1 shows the accuracy of our model depending on the degree of the polynomial feature expansion. As mentioned previously, there seems to be a sweet spot at degree 8, after which the accuracy of our model crashes down. Since we have 4 different training sets, we are presenting an average accuracy over all sets per degree. We can notice that the test and train accuracies are close to each other which comes from the k-fold cross validation ensuring they have similar distribution.

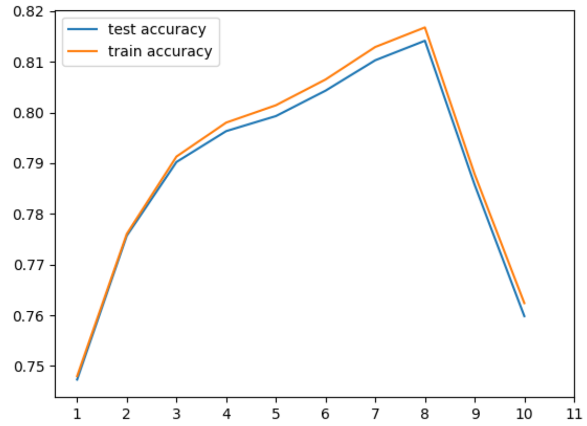

Fig. 1. Test and train accuracies depending on polynomial expansion degree

Figure 2 shows the training error of our model, also based on expansion degree, during our training. There again, we can see a minimum around degree 8.

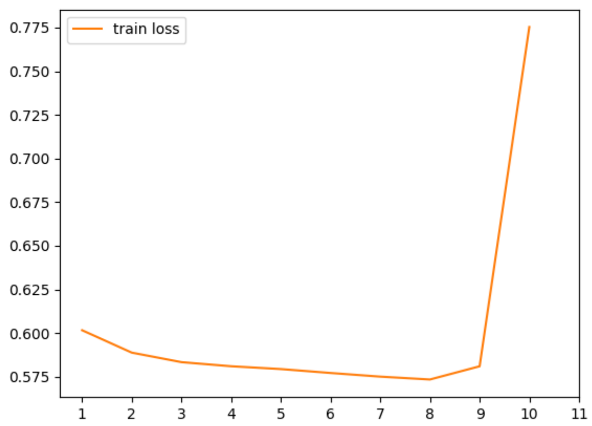

Fig. 2. Training error depending on polynomial expansion degree

During this analysis, we also cross validated on the λ parameter which was optimal for the given expansion degree.

VII. Conclusion

During the first part of the project, we have successfully implemented the different regression methods, which gave us a toolbox of methods for the second part. When working on the CERN dataset, we have applied an analytical approach to try to understand our data and to expose new feature spaces which will simplify classification. We achieved this with processing our data, training our models, and cross validating our parameters. We finally generated our predictions on the test sample and submitted it on the AIcrowd challenge website. We were able to correctly predict a signal in 81.8% of the cases.