Master Semester 1 - Fall 2021
Machine Learning

Project I

Students: PIENING Carlo
GHENASSIA Noam
NUSSBAUMER Arthur

Date: *Monday November 1, 2021*

EPFL

1.1 Introduction

The aim of this project is to classify decay signatures of collisions performed at the LHC to know if they correspond to a Higgs boson (class signal) or to some other particle/process (class background). For that, we run different regression algorithms on a training set of 250000 data points, each of which comes with 30 measurements that we use as features.

1.2 Models and Methods

1.2.1 preprocessing and feature engineering

Our first task is to preprocess the data so that the input of our models is as "easily classifiable" as possible. By looking at the project description, we learned that Nan values were replaced by -999. Hence, our first naive approach was to delete every datapoint that contained Nan values. This approach ended up yielding a "valid" dataset of around 68000 datapoints. Though we expected that number to be enough for the fine-tuning of the 30 parameters of a linear model, it still posed several problems. In fact, a model trained on this subset of observations only would not be able to classify datapoints with Nan values. This issue could be solved by replacing every Nan value with the mean of the measurements for this feature. However, this would introduce a bias in the classifier. Indeed, if it turned out that the absence of measurement for a given feature was related to some other feature, then it is likely that our model would only be fine-tuned to tackle the subset of datapoints with no Nan measurement at all, which only represents one-fifth of the datapoints.

Another alternative approach was to create custom models for each group of datapoints that shared the same defined variables, and classify a datapoint according to the prediction of its corresponding model. However, with 30 variables, such an approach would yield 2^{30} models, which is several orders of magnitude bigger than the number of datapoints. In particular, if there isn't any pattern governing whether a variable is defined, it is likely that each model would be trained on a single datapoint (if any), which would result in overfitting. Therefore, we looked for more information relative to the presence of the Nan values in the documentation of the dataset (Adam-Bourdarios et al. 2015). Notably, we found that a significant number of variables are only defined under some conditions on the PRI Jet Num feature, which is a discrete variable indicating the number of jets. Based on this feature, we were able to partition the dataset into 3 groups, namely, the datapoints with value $0, 1$ or > 1 for this variable. Furthermore, the variable DER Mass MMC isn't always defined, regardless of the other variables. Therefore, we further divided each group into 2 subgroups, based on whether the DER Mass MMC variable is defined. This procedure yielded 6 groups, each of which had the same values defined for each of its datapoints. The workflow was then consisting in training one model per group, and then classifying the test datapoints using the relevant model.

After dealing with the existence of Nan values in certain datapoints, we also needed to check for the relevance of each feature (within a given group). The most obvious step was to delete 0-variance features (that is, features that remained constant throughout an entire group have no predictive value, and can be discarded). We then looked for highly correlated features and kept only one feature per "cluster" of features with a pairwise correlation greater than 0.9 (this threshold was chosen arbitrarily). However, we got poorer accuracy using this function, so we abandoned this idea. Another test we made was to standardize our data before training the model. However, this gave us poor results for accuracy.

Finally, we added an offset (a constant feature with value 1 that allows us to shift the separating hyperplane from the origin) and performed feature augmentation by applying a degree n polynomial embedding of the data, but without considering interactions between variables (we

considered variables of the form $x^2, x^3, \ldots, x^n$, but not variables of the form xy). In fact, considering the interactions would have led us to consider a lot of variables, thereby vastly increasing the complexity of our models.

1.2.2 classification

For the classification, we restricted ourselves to linear models. For that, we implemented linear regression using the normal equations, as well as gradient descent. We also implemented logistic regression and ridge regression. Our attempts to classify the data then consisted of a series of combinations of the implemented methods, that were then applied separately to all six subsets of the data. The quality of our models was then assessed based on the test accuracy obtained on *AIcrowd*.
In our initial trial, we applied gradient descent (step size = 0.01, 1000 training epochs) to fit a linear regression model to the original features alone, using the mean squared error as our loss; predictions were then made based on the sign of the output of our model. We also trained the linear regression model using the normal equations. Comparing these two models, we found that the last one was better. This is due to the fact that without multicollinearity, the normal equations give the unique minimal solution to the least-squares problem and not an approximate one as the gradient descent does. We then used gradient descent with logistic regression, but surprisingly our results were worse than those obtained with linear regression. After looking at our data, we found that this was due to the fact that the gradient was increasing extremely quickly, and as such, we needed to put a very small step size ($1e-15$) to get convergence. However, the stopping condition for our training algorithm was the absolute difference between two subsequent loss values being less than a threshold of 0.01. For that reason, the algorithm stopped too early each time, and this lead to approximated weights for the regression and bad accuracy.
Observing that the accuracy was not so good for either model, we decided to use ridge regularization, which consists in penalizing high-valued weights in order to avoid overfitting - that is, we add an l_2 term to the loss function, multiplied with a regularization parameter λ (the choice of an l_2 term over an l_1 one was made based on ease of implementation, as only the ridge regression has a closed-form solution). The optimal value of the regularization parameter λ was found by performing 4-fold cross-validation.
Finally, we performed feature augmentation. The choice of the degree of our polynomial embedding of the data was also made with cross-validation that was splitting the data into two sets of the same size and computing the test error depending on different degrees of polynomial embedding. We took then the degree that was minimizing the test error. We also tried to develop a function to make cross-validation on a 2-dimensional grid (as both the degree of the polynomial embedding and the regularization parameter λ need cross-validation). However, this gives poorer results because it forced to split the data in the same way for the two cross-validations. That's why we decided finally to perform the first one-dimensional cross-validation for the regularization parameter λ using a polynomial embedding of degree 3, and then apply cross-validation for the degree of our polynomial embedding using the λ we found before. This method yielded our best results, as we obtained a test accuracy of 0.815.

1.3 Conclusion

To conclude, we designed different models to classify decay signatures of collisions performed at the LHC. To do so, we implemented several functions discussed during the course and preprocessed our data thanks to the definition of our features, polynomial embedding, and correlation. Our best performance consisted in decomposing our data into 6 different groups, adding an offset, augmenting the features with a polynomial embedding with a cross-validated degree, and finally running multiple ridge regressions for each group with a parameter lambda found thanks to 4-fold cross-validation.

Bibliography

[Ada+15] Claire Adam-Bourdarios et al. "The Higgs boson machine learning challenge". In: *Proceedings of the NIPS 2014 Workshop on High-energy Physics and Machine Learning*. Ed. by Glen Cowan et al. Vol. 42. Proceedings of Machine Learning Research. Montreal, Canada: PMLR, 13 Dec 2015, pp. 19–55. URL: https://proceedings.mlr.press/v42/cowa14.html.