Detecting the Higgs Boson: A ML Challenge

CS-433 - Machine Learning: Project 1

Ecole Polytechnique Fédérale de Lausanne

Antoine Bonnet antoine.bonnet@epfl.ch

Melanie Brtan melanie.brtan@epfl.ch

Camille Cathala camille.cathala@epfl.ch

Abstract

In this project, we investigate the performance of various supervised machine learning models on the binary classification task of detecting the Higgs boson through simulated particle collisions data from the Large Hadron Collider at CERN.

1 Introduction

The main objective of this project was to predict the presence of the Higgs Boson based on a labelled dataset. We performed hyperparameter tuning with 5-fold cross-validation on both our pre-processing procedure as well as model-related hyperparameters to obtain the optimal model with maximal validation accuracy. Our most important finding was that the Ridge regression model achieves the highest cross-validation accuracy by preventing overfitting and yields a test accuracy of 78.6%. We explored the reasons for this success and found that regularization indeed improves performance by preventing overfitting.

2 Dataset

We trained our models on the dataset provided by the Higgs Boson Machine Learning Challenge, which uses data extracted from the ATLAS full-detector simulation by CERN. It consists of a training set containing 250,000 events and a test set of 568,238 events. Each event represents a particular observation from the ATLAS machine, and consists of 30 simulated features, which contain raw measurements from the detector as well as some derived quantities [1]. The training set also contains the labels with which we trained our models; variable *b* represents background noise while *s* represents a signal indicating the presence of the Higgs boson.

3 Models and Methods

3.1 Feature pre-processing

Before training each model, we applied various pre-processing methods on the data. This step is a crucial part in data handling, because it can lead to significant improvements in model accuracy. To allow for binary classification, we first converted the labels *b* and *s* to -1 and 1 respectively. Inspecting the dataset, we observed that some entries contain the value -999, which is outside of the range of possible values, indicating that the measurement is missing. Those values have to be either removed or replaced. To do so, we removed all columns holding a rate of missing values higher than α, and replaced every remaining missing feature with their column-wise median. Outlier datapoints also needed to be dealt with, as they can bias the training of the model and lead to false predictions. Therefore, we removed from the training set every datapoint containing at least one feature that was β times the standard deviation away from its mean value. To increase the expressiveness of our models, we performed polynomial expansion of degree d on each event. This has the effect of expanding every feature value by every degree up to d. To add a bias term, we added a column of ones to the data matrix. Finally, we standardized the values of each feature across all events to follow a standard normal distribution $\mathcal{N}(0, 1)$.

3.2 Models

For this binary classification task, we trained the following supervised machine learning models: Least Squares (LS), Ridge Regression (RR), Gradient Descent (GD), Stochastic Gradient Descent (SGD), Logistic Regression (LR) and Regularized Logistic Regression (RLR).

3.3 Hyperparameter tuning and training

In order to select the best model, we trained each of them on 200,000 randomly chosen datapoints from the training set, then used the remaining 50,000 datapoints to use as a fake test set. This allowed us to estimate which model was overfitting, underfitting or generalized well. During the training phase, we performed hyperparameter tuning through grid search. As each model responded differently to various variants of our pre-processing procedure, we decided to experiment with both pre-processing hyperparameters (α, β and d) and model-specific hyperparameters. For every set of pre-processing hyperparameters, we accordingly processed the training set. Then, the average 5-fold cross-validation training and validation accuracies were computed for every set of model-specific hyperparameters. We then chose the optimal hyperparameters as those yielding the highest average validation accuracy over all 5 splits. Once we obtained the optimal hyperparameters, we then trained our model on the whole 250,000 points training set, then applied it on the reserved test set

(this time without removing any outliers since otherwise we would be missing some labels) to obtain the final test accuracy.

4 Results

The 5-fold cross-validation training and validation accuracies, along with the final test accuracy corresponding to the optimal hyperparameters of each model are documented in Figure 1.

Model	Hyperparameters	Training (%)	Validation (%)	Test (%)
LS	$d = 5, \alpha = 1, \beta = 10$	79.5	76.3	70.7
RR	$d = 6, \alpha = 1, \lambda = 10^{-6}$	79.8	79.7	78.6
GD	$d = 4, \alpha = 1,$ $\gamma = 0.1, N = 500$	77.7	77.2	77.6
SGD	$d = 5, \alpha = 1,$ $\beta = 10, \gamma = 0.01,$ $N = 500, B = 100$	74.9	74.8	75.3
LR	$d = 4, \alpha = 1,$ $\gamma = 0.1, N = 250$	73.3	73.3	73.4
RLR	$d = 5, \alpha = 1, \gamma = 0.01,$ $\lambda = 0.01, N = 500$	73.2	73.2	73.3

Figure 1: Optimal hyperparameters and 5-fold cross-validation accuracies for each model

Figure 2 compares the training, validation and test accuracies of the RR and LS models with the pre-processing hyperparameters $d = 4, \alpha = 1, \beta = None$ that led to optimal RR validation accuracy for several values of the regularization parameter λ.

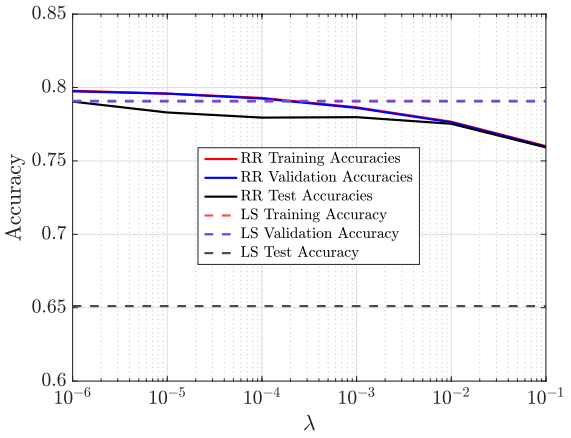

Figure 2: Cross-validation accuracies of LS and RR for different values of λ, with $d = 6$, $\alpha = 1$, $\beta = None$

5 Discussion

Different hyperparameter ranges were optimized over for each model. Interestingly, the missing value parameter $\alpha = 1$ outperformed all other values for both LS and RR, meaning that no columns were removed and all missing values were replaced by their median. For this reason, we kept $\alpha = 1$ throughout all the experiments the same. Similarly, we found that degrees between 4 and 7 yielded good overall accuracy. Other pre-processing parameters were therefore optimized over the values $4 \leq d \leq 7$ and $\beta \in \{None, 5, 10\}$. We saw a decrease in accuracy for $\beta = 5$, meaning that this was too low a threshold, while $\beta = None$ or 10 yielded good results. As seen in Figure 1, the parameter β is only listed in the LS and SGD models. This indicates including outliers improves performance (except for LS and SGD).

In Figure 1, the optimal pre-processing and model-specific parameters are shown for every model. We observe that Ridge regression yields the highest validation and test accuracy. Interestingly, the training accuracy of RR and LS are the highest overall, while the test accuracy of LS is the lowest of all models. A possible explanation for this phenomenon is that the LS model is overfitting, as indicated by high training accuracy but low test accuracy.

As the regularization parameter λ in RR grows smaller, the regularization term in the loss is neglected and the model gets closer to the LS model (as demonstrated by the similar training accuracy of LS and RR in Figure 2). However, as shown in Figure 2, smaller values of λ yielded higher overall accuracy. From Figure 1, we see that the highest test accuracy was obtained by setting $\lambda = 10^{-6}$, the lowest value over which we optimized. This was a surprise, as we expected the test accuracy of RR to shrink to the level of the LS test accuracy when reducing the value of λ. However, the test accuracy turned out to improve drastically for the same pre-processing hyperparameters $d = 4, \alpha = 1, \beta = None$, with 78.6% for RR and 65.1% for LS. This implies that the regularization term of RR does indeed penalize overfitting and that the smallest λ value led to a successful trade-off between underfitting and overfitting.

GD led to very consistent results and achieved the second highest test accuracy, while SGD showed weaker performance. During experiments, we observed that $\gamma = 0.5$ significantly maximized the LR and RLR validation accuracy, while pre-processing hyperparameters only led to low variance in the resulting accuracies. Both LR and RLR exhibited weak performance, which was unexpected as they are widely used for binary classification tasks.

It seems like the data pre-processing is the main factor affecting the test accuracy bottleneck of all models. For future experiments, it would be worth exploring different pre-processing techniques, such as weighting features by their correlation with the label, or splitting the dataset between classes depending on the jet number, as in [2].

6 Summary

Through our experiments, we conclude that regularization leads to beneficial effects on model performance, and that the Ridge regression model yielded the optimal performance in detecting the Higgs boson from CERN data, with a final test accuracy of 78.6%.

References

[1] Muhammad Abbas, Asifullah Khan, Aqsa Saeed Qureshi, and Muhammad Waleed Khan. Extracting signals of higgs boson from background noise using deep neural networks. *arXiv preprint arXiv:2010.08201*, 2020.

[2] Claire Adam-Bourdarios, Glen Cowan, Cécile Germain, Isabelle Guyon, Balázs Kégl, and David Rousseau. The higgs boson machine learning challenge. In *NIPS 2014 Workshop on High-Energy Physics and Machine Learning*, volume 42, page 37, 2014.