Class Project 1 of Machine Learning Course
Learn recognizing the Higgs boson

Florian Grötschla[1], Adrien Bertaud[2], and Maximilian Wessendorf[1]

[1]IN-H, EPFL
[2]AUDIT-H, EPFL

Abstract—**The goal of this project is to train a binary classifier to identify the appearance of the Higgs boson on a dataset of decay signatures from CERN. We compare various machine learning methods and apply different preprocessing steps that were introduced in class. Finally, we fit a classification model with the least squares method that scores an accuracy of 0.82 and a F1-score of 0.722.**

I. Introduction

The Higgs boson is an elementary particle in the Standard Model of physics. Since the Higgs boson decays rapidly into other particles, scientists don't observe it directly, but rather measure its "decay signature", or the products that result from its decay process. Deciding whether a given decay signature is actually from a Higgs bosom or just background noise is topic of a machine learning challenge that we are doing as part of this class. Our approach for the project is to apply the different methods and preprocessing steps introduced in class to train classifiers for the given task and compare their performance.

II. Dataset

The given dataset of decay signatures consists of 30 features each that are labeled with the information whether the decay process is the result of a Higgs boson or not. If values come from a faulty sensor or cannot be computed they are set to -999 to indicate that it is an error value.

III. Preprocessing

We apply four different preprocessing steps to the input dataset. We standardize, handle error values, add polynomial expansion and add an extra column only containing the value one. The individual steps are described below.

A. Standardization

We do the standardization by subtracting the mean and dividing by the standard derivation for each dimension to get features which have a mean of 0 and a standard derivation of 1. We use this standardized data for all of the methods.

B. Handling error values

Some values of features in the dataset are set to -999 to indicate an error in the measurement. To handle the error values we tried to

- remove all rows with error values, which leaves us with 68114 out of 250000 datapoints
- remove all columns with error values, which leaves us with 19 out of 30 columns
- compute standardization means and variances without taking into account those error values and then replace error values by zero when standardizing.

The last option gave us the best results so we used it for our comparison of methods.

C. Polynomial expansion

For our models to fit the data better, we used polynomial expansion on the features. We tested the methods for different degrees of the expansion where we expanded every feature up to this degree. This considerably increased the number of features used to train our models.

D. Equalization of number of datapoints for both labels

As the dataset contains more datapoints labeled 1 than -1, we tested to remove some datapoints with label -1 to have the same amount of datapoints for both labels as the input for our methods. The results were worse when doing this, so we dropped this approach.

IV. Methods

We implemented the following methods to compare their performance on the classification task:

- linear regression using gradient descent
- linear regression using stochastic gradient descent
- least squares regression using normal equations
- ridge regression using normal equations
- logistic regression using gradient descent
- regularized logistic regression using gradient descent

To evaluate them we used a ten-fold cross validation to calculate the mean accuracy and F1-score. We then tested the different methods by grid searching over their specific parameters, like number of iterations, gamma and lambda. We also tested polynomial expansion, varying the degree by grid searching.

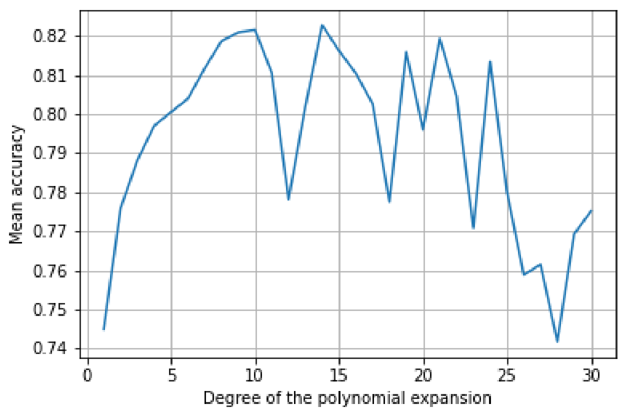

Figure 1. Mean accuracies of the ten-fold cross-validation using the least squares method

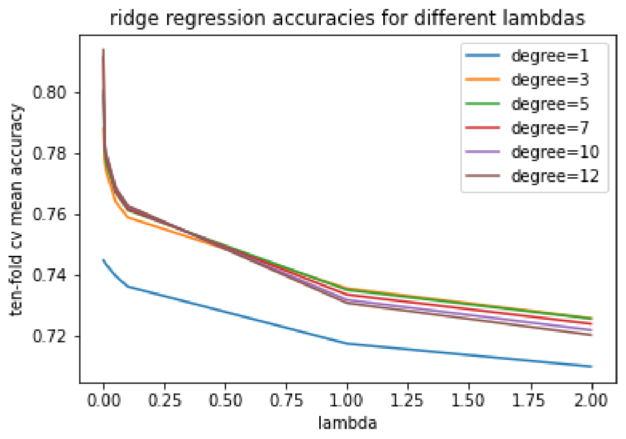

Figure 2. Mean accuracies of the ten-fold cross-validation using the ridge regression method for different degrees of polynomial expansion and different values for lambda.

V. Results

A. *Results for linear regression using gradient descent and stochastic gradient descent*

With linear regression using gradient descent we got our best mean accuracy at 0.76 (for gamma at 0.053, iterations at 1000, expansion degree at 10). The mean accuracy obtained by stochastic gradient descent was similar. Those results could have been improved by more extensive testing, but we focused on least squares method.

B. *Results for least squares*

In our preliminary testing, the least squares method worked better than the linear regression using gradient descent and stochastic gradient descent. We can explain it by the fact that the optimization problem we try to solve is the same for the least squares method, which outperforms the gradient descent approaches in the sense that it finds the exact solution to the optimization problem due to the relatively simple linear model and the convexity of the function, we find the best fit using least squares.

So, we did more testing with the least squares method. Another advantage is that it runs much faster and allowed more extensive testing.

The results can be seen in Figure 1. In the beginning, the model profits a lot by the increased complexity whereas the value decreases for expansions of very big size. In between, the values fluctuate quite a lot, reaching its maximum for a degree of 14 with an accuracy of 0.82 and a F1-score of 0.722. This is why this value was also used for creating the submission, resulting in a similar accuracy to what we observe in the cross validation.

C. *Results for Ridge regression*

We also tried ridge regression by grid searching over different values for the degree of polynomial expansion and different values for the parameter lambda. The results can be seen in Figure 2. We observe that for an increasing lambda the accuracy gets worse which might be due to the fact that we are not overfitting too much and have enough datapoints to train our model, so penalizing to get less complex models does not help.

D. *Results for logistic regression with gradient descent*

We tested various gamma, number of iteration and degree expansion for for logistic regression with gradient descent, and we got our best mean accuracy at 0.73 (for gamma at 0.01, iterations at 1000, expansion degree at 10). This results could have been improved by more extensive testing, but again, we focused on least squares method.

E. *Results for regularized logistic regression with gradient descent*

Keeping the same best values as logistic regression, we tried several lambdas for regularized logistic regression and see a similar behaviour to the one observed for ridge regression: the penalization does not help to improve the model.

VI. Summary

We used our knowledge from the first part of the Machine Learning class to find a prediction model with an accuracy of 0.82 on a real-world dataset, what is a promising result. We faced problems like error handling of values in the input by replacing error values with means computed without them. We evaluated several regression based methods using cross validation to get a confident estimate for the performance of our model. To find the best set of hyperparameters, we applied a grid search approach and found that the polynomial expansion works best with a degree up to 14. The least squares method quickly gave us better results, and we mainly focused our tests with this method.