Higgs Boson Machine Learning Challenge

Sebastien Savidan, Jean Gschwind, Tristan Besson
Machine Learning Course, EPFL, Switzerland

***Abstract*—A critical part of scientific discovery is the communication of research findings to peers or the general public. Mastery of the process of scientific communication improves the visibility and impact of research. While this guide is a necessary tool for learning how to write in a manner suitable for publication at a scientific venue, it is by no means sufficient, on its own, to make its reader an accomplished writer. This guide should be a starting point for further development of writing skills.**

I. Introduction

In this project, a full implementation of machine learning methods is made from scratch. The dataset is first being observed, in order to construct a coherent feature processing. From there, some algorithms are tested in order to produce the best results. With our current setup, we were able to achieve a percentage of 82,057% of correct predictions on the Kaggle competition.

II. Exploratory Analysis

The dataset coming from the ATLAS full detector simulator contains both information about the Higgs boson decay and background noise. Before implementing any feature processing and machine learning methods, we need to observe the dataset at our disposal. The training set is composed of 30 features with around 60'000 measurements. Among the features, we can first observe that 7 of them contain more than 70% of the value -999. Nonetheless, we will keep them in our processing part because of the small number of features of the dataset, but for the moment we will classify them as NaN values.

By looking at the table 1, we observe that they need to be standardize as their mean and standard deviation can vary from many different order of magnitudes. For example, the variable PRI_jet_num takes only discrete values between 0 and 3.

Index	Nb Nan value	Mean	Std	Value max	Value min
Feature 14	0	-0.010973	1.214076	2.497	-2.499
Feature 18	0	0.043543	1.816608	3.142	-3.142
Feature 2	0	81.181982	40.828609	1349.351	6.329
Feature 21	0	209.797178	126.499253	2003.976	13.678
Feature 26	177457	57.679474	31.985561	721.456	30.0

Figure 1. Features Example

In order to better understand the interaction and dependencies between the features, it is interesting to output their scatter plot matrix, as shown in Figure 2. The first observation from there is that a few of them are indeed correlated with clear shapes appearing on the plots. Then, their distribution can vary a lot, from being normally distributed to equally.

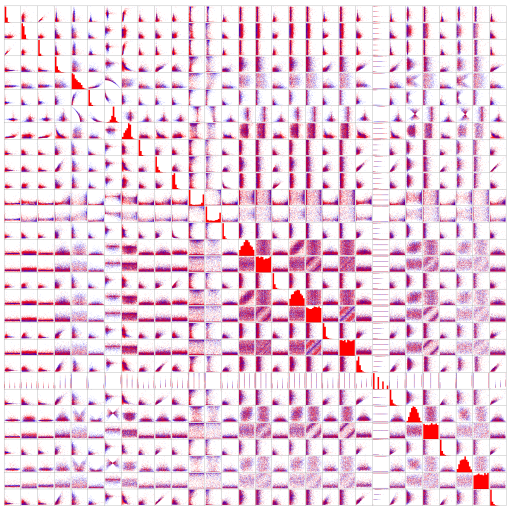

Figure 2. Scatter Plot Matrix

III. Feature Processing

Our first results with the simple dataset were not sufficient. It was therefore decided to implement some processing in order to enhance our percentage of correct predictions.

The first process was concerning the NaN values. A function nan_handling can replace all the NaN values by our desired value, here a low negative value (A VERIFIER). Going through all the features, we try to apply different functions to the features: exponantial, cosinus, square root, logarithm and the power function. If the resulting feature present a better correlation, we keep it.

Then, from the scatterplot matrix, we observe features that seem highly correlated; for example features 2 and 7. After that, we multiply them together and keep the resulting feature if the result is satisfying.

IV. Machine Learning Methods

A. Least Squares

Linear regression using a mean-squared error is one of the most popular method for data fitting. In this case, the problem consists in solving the normal equations of the system. This method is the most simple, without any explicit interation loop. To obtain better results, it is necessary to add a column of ones as a new feature. This will help the system to find an affine function solving the system (compared to a purely linear one). As a result, it yields the worst results in our system.

B. Linear Regression using Gradient Descent

In this case, the weight is not calculated through a simple equation solving, but rather updated over time. Given an initial weight vector and a maximum number of iterations, the gradient of the cost function is calculated, and the weight updated to go in the opposite direction. As seen in the class, this system won't perform well with a dataset with outliers.

C. Linear Regression using Stochastic Gradient Descent

In order to be robust to the previously mentioned outliers, one method consist in calculating the gradient for only a batch of the complete dataset, changing at each iteration. Otherwise, the principle of this method stay the same.

D. Ridge Regression

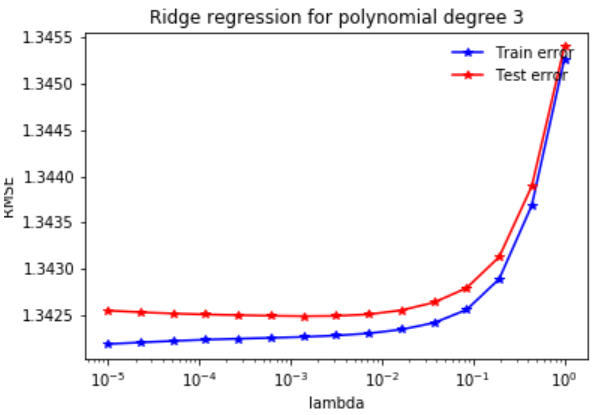

Figure 3. Ridge Regression Error for a polynomial degree of 3

E. Logistic Regression

F. Regularized Logistic Regression

V. Results

References